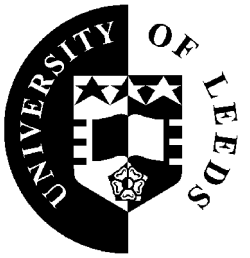
The Numerical Solution of Free-Surface Problems for Incompressible, Newtonian Fluids

by

R.C. Peterson

Submitted in accordance with the requirements
for the degree of Doctor of Philosophy.



The University of Leeds
School of Computer Studies
and
Department of Applied Mathematics

September 1999

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

Acknowledgements

I'd like to thank my supervisors Dr. Peter Jimack and Dr. Mark Kelmanson for their advice and guidance throughout this work. I'd also like to thank Prof. Berzins and Prof. Savage for their comments on issues arising in this work, Mark Walkley for providing me with codes that formed the basis of the visualisation tools developed as part of this project and for many helpful discussions, Dr. J.R. Shewchuk for making his code Triangle available, the Leeds coating group for allowing me to attend their seminars and Prof. R.M.M. Mattheij, Prof. O.A. Basaran, Prof. P.H. Gaskell, Dr. L.A. Freitag and Prof. P.R. Dawson for their assistance.

The support of both the School of Computer Studies and the Department of Applied Mathematical Studies in making this work possible through their funding of a university scholarship is gratefully acknowledged.

Finally, I'd like to thank my family and Barbara in particular, for their continuing support that has made my studies possible.

Abstract

This thesis describes a new approach for the solution of two-dimensional, time-dependent, surface-tension-driven free-surface flows involving domains of arbitrary shape that may undergo large changes in shape during the course of a problem. Both Stokes and Navier-Stokes problems are considered, a mixed Lagrangian-Eulerian finite element formulation being employed for the latter. All meshes are generated automatically using a Delaunay mesh generator, the only user input required being the specification of the initial free-surface shape. Very few constraints are placed on the shape of the initial domain and arbitrarily large deformations of the domain are permitted. A key feature of the new method is its ability to dynamically refine and de-refine the free-surface discretisation as and when necessary to maintain an accurate representation of the free surface, as is essential for surface-tension-driven problems. Full implementation details are included.

Semi-implicit time integration schemes are employed for both Stokes and Navier-Stokes problems, the resulting systems of linear equations being solved by the conjugate residual method preconditioned using high-quality, thresholded, incomplete LU factorisations. A novel scheme for the automatic selection of the maximum time step size that ensures free-surface stability is described.

A number of challenging problems are considered. First a Stokes-flow problem with a known analytic solution is employed to confirm that the expected rates of convergence in the solution are obtained. Next the Stokes-flow evolution of a film of viscous fluid on a rotating cylinder is investigated, the time-dependent case being modelled for the first time. Illustrations of the large free-surface deformations leading up to load shedding are presented. In addition, the unexpected existence of apparently stable oscillatory solutions is reported for certain configurations. Finally the axisymmetric oscillations of droplets at low Reynolds numbers ($Re \leq 100$) are considered.

Contents

1 Introduction **1**

1.1 Free-surface flow modelling 1
1.2 The Navier-Stokes equations 5
1.3 The Stokes equations 7
1.4 Overview of the difficulties involved in free-surface modelling 9
1.5 Methods for free-surface flows 11
1.5.1 Boundary-element methods 12
1.5.2 Finite-difference methods 12
1.5.2.1 Marker-and-cell methods 13
1.5.2.2 Volume-of-fluid methods 14
1.5.2.3 Phase field methods 16
1.5.3 Finite element methods 16
1.5.3.1 Fixed-connectivity meshes 17
1.5.3.2 Unstructured meshes 19
1.6 Conclusions 20

2 The finite element method **21**

2.1 Elements for incompressible flow 21
2.1.1 LBB stability 22
2.1.2 The isoparametric mapping 23
2.1.3 Finite element basis functions 25
2.1.4 A Stokes-flow test problem 26
2.2 Boundary discretisation 30
2.2.1 Tangent continuity 32
2.2.2 Computation of boundary conditions 33
2.2.3 Equidistribution of curvature 36
2.2.4 Initial boundary discretisation 40

2.2.5 Boundary discretisation constraints 40
2.3 Interior mesh generation 42
2.4 Remeshing 45
2.4.1 Boundary refinement 47
2.4.2 Boundary derefinement 50
2.4.3 Boundary edge adjustment 51
2.4.4 Selection of parameters 53
2.5 Continuous mesh update 54
2.5.1 Linear-elasticity model 55
2.5.2 Laplacian smoothing 57
2.6 Conclusions 58

3 Solvers for incompressible flows **60**
3.1 The Galerkin method 60
3.1.1 The stress boundary condition 63
3.1.2 The kinematic boundary condition 64
3.2 Moving-mesh corrections 67
3.3 Matrix formulation 70
3.4 Time-discretisation schemes 73
3.5 A semi-implicit scheme for free-surface Navier-Stokes problems 76
3.6 An implicit scheme for free-surface Navier-Stokes problems 78
3.7 A semi-implicit scheme for free-surface Stokes problems 79
3.8 A fully implicit scheme for free-surface Stokes problems 80
3.9 Notes on alternative nonlinear solution methods 81
3.10 The conjugate residual method 83
3.11 Node re-ordering 86
3.12 Preconditioning 89
3.12.1 Diagonal preconditioning 90
3.12.2 ILUT preconditioning 94
3.12.3 Graded meshes 98
3.13 Predictors 102
3.14 Interpolation 103
3.14.1 Linear and quadratic interpolation 104
3.15 Time-step selection 106
3.16 Conclusions 108

4 The coalescence of two cylinders **110**
4.1 Background . . . 110
4.2 Analytical solution . . . 113
4.3 Method . . . 115
4.4 Results . . . 121
4.4.1 Accuracy . . . 128
4.4.2 Tangential stress . . . 129
4.4.3 Conservation of mass . . . 133
4.4.4 Efficiency . . . 136
4.4.5 Shapes with corners . . . 138
4.5 Conclusions . . . 140

5 The supported-load problem **142**
5.1 Background . . . 142
5.2 The Stokes flow model . . . 147
5.3 Method . . . 149
5.4 Results . . . 150
5.4.1 Accuracy . . . 159
5.4.2 Further investigation of oscillatory solutions . . . 161
5.4.3 Large supported loads . . . 162
5.4.4 Load shedding . . . 165
5.4.5 A second load-shedding problem . . . 170
5.5 Conclusions . . . 177

6 Navier-Stokes problems **180**
6.1 Axisymmetric oscillations of droplets . . . 180
6.2 Axisymmetric problem formulation . . . 182
6.3 Boundary conditions . . . 186
6.4 Mesh update procedures . . . 188
6.5 Results . . . 190
6.5.1 Small-amplitude oscillations . . . 191
6.5.2 Large-amplitude oscillations of ellipsoidal droplets . . . 196
6.5.3 A large-amplitude second-spherical-harmonic problem . . . 204
6.5.4 Interior-mesh regeneration . . . 208
6.6 Conclusions . . . 212

7 Conclusions **213**
7.1 The unstructured mesh approach 213
7.2 Comparison with alternative approaches 215
7.3 Suggestions for further work 216

A Some useful identities **219**

B Operators in axisymmetric form **220**
B.1 The continuity equation 222
B.2 The pressure gradient 222
B.3 The convective terms 223
B.4 The viscous terms 223

Bibliography **225**

List of Figures

1.1 Extrusion coater. 2
1.2 Curtain coater. 3
1.3 Dip coating. 3
1.4 Reverse roll coating. 4
1.5 Sintering a bundle of cylindrical fibres. 4
1.6 Free-surface instability. 10
1.7 The marker-and-cell (MAC) method. 14
1.8 First-order accurate VOF method 15
1.9 Second-order accurate VOF method 16
1.10 A fixed-connectivity mesh for the finite element method. 17
1.11 The method of spines 18
1.12 Extrusion coater mesh: detail. 19

2.1 Common elements for incompressible flows 22
2.2 The isoparametric mapping 23
2.3 Free-surface boundary representation 24
2.4 Unstructured mesh test problem: circular mesh 28
2.5 Unstructured mesh test problem: velocity field 29
2.6 Unstructured mesh test problem: pressure field 30
2.7 Isoparametric element: edge displacement 32
2.8 Tangents and normals at a vertex 33
2.9 Mesh failure at a free-surface corner. 41
2.10 Meshes for an ellipse. 43
2.11 Distribution of internal angles in a graded mesh 44
2.12 Unnecessary elements 45
2.13 Splitting an edge. 49
2.14 Merging two edges. 50
2.15 Permissible locations for an edge node. 52

2.16 Adjusting an edge. 53
2.17 Stretching of elements during motion of a free surface 55
2.18 Uniform growth of a circular domain 56
2.19 Laplacian smoothing: a mode of failure. 58

3.1 A semi-implicit algorithm for free-surface Navier-Stokes problems . . 77
3.2 An implicit algorithm for free-surface problems 79
3.3 A semi-implicit scheme for free-surface Stokes-flow problems 80
3.4 A fully implicit algorithm for free-surface Stokes-flow problems 81
3.5 The preconditioned conjugate residual algorithm 84
3.6 Reverse Cuthill-McKee node re-ordering algorithm 87
3.7 Stiffness-matrix sparsity pattern: original and RCM orderings 88
3.8 PCR solver: convergence histories with diagonal preconditioning . . . 93
3.9 PCR solver: convergence history with ILUT preconditioning 95
3.10 PCR run times 98
3.11 Graded meshes for preconditioning study 99
3.12 Preconditioning: graded vs. ungraded meshes 100
3.13 ILUT preconditioning: graded vs. ungraded meshes 101
3.14 Interpolation: resulting pressure field 105

4.1 The coalescence of two infinite cylinders: initial geometry 111
4.2 The coalescence of two infinite cylinders: free-surface evolution 113
4.3 Selected initial meshes 116
4.4 Mesh detail in the neck region at $t = 0.304$ 118
4.5 The coalescence of two infinite cylinders: mesh evolution 120
4.6 Neck radius as a function of time 121
4.7 Global free-surface error 122
4.8 Neck velocity as a function of time 123
4.9 Velocity field at $t = 0.304$ 124
4.10 Pressure field at $t = 0.304$ 125
4.11 Velocity fields at $t = 1.00$ and $t = 2.00$ 126
4.12 Pressure fields at $t = 1.00$ and $t = 2.00$ 127
4.13 Initial stage velocities 129
4.14 Rate of convergence of neck velocity 130
4.15 Normal and tangential stress in the neck region 131
4.16 Tangential-stress error: effect of refinement 132
4.17 Mass as a function of time 133

4.18 Effect of spatial refinement on mass conservation 134
4.19 Effect of time-step size on mass conservation 135
4.20 Unknowns as a function of time 137
4.21 PCR iterations per time step as a function of time 137
4.22 Time-step size as a function of time 138
4.23 Stokes-flow evolution of a cross with rounded corners 139

5.1 Domain for supported-load problems 143
5.2 Maximum supportable load: various models 145
5.3 Initial mesh for preliminary investigations 149
5.4 Free-surface evolution $\Lambda = 1.3$, $\gamma = 12.5$ 151
5.5 Norm of the residual: asymptotically steady load 152
5.6 Free-surface symmetry 153
5.7 Pressure field near steady-state 154
5.8 Free-surface evolution $\Lambda = 1.2$, $\gamma = 12.5$ 155
5.9 Free-surface evolution $\Lambda = 0.75$, $\gamma = 12.5$ 155
5.10 Problem parameter space 156
5.11 Regions of the parameter space 157
5.12 Rate of decay of oscillations in film thickness 158
5.13 Effect of refining the mesh in the circumferential direction 159
5.14 Effect of refining the mesh in the radial direction 160
5.15 Effect of reducing the time-step size 161
5.16 Effect of reducing the time-step size (detail) 161
5.17 Potential, surface and kinetic energy $\Lambda = 1.3$, $\gamma = 12.5$ 163
5.18 Conservation of mass $\Lambda = 1.3$, $\gamma = 12.5$ 163
5.19 Potential, surface and kinetic energy $\Lambda = 0.75$, $\gamma = 12.5$ 164
5.20 Conservation of mass $\Lambda = 0.75$, $\gamma = 12.5$ 164
5.21 Large load film thicknesses $\Lambda = 5.7$, $\gamma = 1.3$ 165
5.22 Large load problem: velocity 166
5.23 Large load problem: pressure 166
5.24 Load-shedding problem 1: free-surface evolution 167
5.25 Load-shedding problem 1: pressure 168
5.26 Load-shedding problem 1: pressure detail 169
5.27 Load-shedding problem 1: velocity detail 169
5.28 Load-shedding problem 2: initial mesh 171
5.29 Load-shedding problem 2: free-surface evolution 172

5.30 Load-shedding problem 2: break down of rigid-body flow (a) 173
5.31 Load-shedding problem 2: break down of rigid-body flow (b) 174
5.32 Load-shedding problem 2: droplet formation (a) 175
5.33 Load-shedding problem 2: droplet formation (b) 176
5.34 Load-shedding problem 2: final velocity 177
5.35 Load-shedding problem 2: final mesh 178

6.1 Domain for axisymmetric problems. 181
6.2 Updating of the axial nodes 189
6.3 Initial meshes for small-amplitude problems 192
6.4 Velocity field for mesh 1 at $t = 0.1$ 193
6.5 Pressure field for mesh 1 at $t = 0.1$ 193
6.6 Change in droplet mass as a function of time using mesh 1 194
6.7 Variation of e with time for small-amplitude oscillations 195
6.8 Large-amplitude ellipsoidal oscillations: mesh evolution 197
6.9 Large-amplitude ellipsoidal oscillations: velocity (a) 198
6.10 Large-amplitude ellipsoidal oscillations: velocity (b) 199
6.11 Large-amplitude ellipsoidal oscillations: pressure (a) 200
6.12 Large-amplitude ellipsoidal oscillations: pressure (b) 201
6.13 Large-amplitude ellipsoidal oscillations: s 202
6.14 Large-amplitude ellipsoidal oscillations: mass change 203
6.15 Large-amplitude ellipsoidal oscillations: velocity field at $t = 2.4$ . . . 204
6.16 Initial mesh for spherical-harmonic problem with $f_2 = 0.9$ 205
6.17 Second-spherical-harmonic problem, $f_2 = 0.9$: e 206
6.18 Free-surface evolution, $f_2 = 0.9$ (1) 207
6.19 Second-spherical-harmonic problem, $f_2 = 0.9$: free surface at $t = 1.3$. 208
6.20 Free-surface evolution, $f_2 = 0.9$ (2) 209
6.21 Meshes for a second-spherical harmonic problem $f_2 = 0.9$ 210
6.22 Free-surface evolution, $f_2 = 0.9$ (3) 211

B.1 Geometry of the cylindrical coordinate system 220

List of Tables

2.1 Isoparametric V6-P3 element convergence-rate data. 28
2.2 Elliptical cylinder problem: mesh statistics. 44
2.3 Constants employed in the adaptive mesh generator. 54

3.1 Meshes for preconditioning studies (ungraded) 92
3.2 Diagonal scaling: PCR iterations 92
3.3 ILUT preconditioning: problem and preconditioner statistics 96
3.4 ILUT preconditioning: iterations and timings 97
3.5 Graded mesh statistics: diagonal preconditioning 100
3.6 Graded mesh statistics: ILUT preconditioning 101

4.1 Initial mesh statistics 117
4.2 Run statistics 128
4.3 Averaged neck velocities 130

5.1 Initial mesh statistics 150
5.2 Asymptotic film thicknesses 152

6.1 Small-amplitude axisymmetric droplets: mesh data 191
6.2 Small-amplitude axisymmetric droplets: results 196

Chapter 1

Introduction

In this chapter the subject of free-surface modelling is introduced, and a number of industrially important free-surface problems are described. Two typical non-dimensional forms of the Navier-Stokes equations are next introduced, the first corresponding to the standard formulation for advection-dominated problems, the second to a formulation appropriate for Stokesian surface-tension-driven problems, illustrating the radically different nature of free-surface problems in these two regimes. Finally a number of strategies commonly employed for the solution of free-surface problems are described and their limitations discussed.

1.1 Free-surface flow modelling

The study of incompressible Newtonian fluids is of fundamental importance in modern engineering. While the underlying physics, at the macroscopic scale at least, is well understood, the mathematical problems that arise from all but the simplest of problem geometries cannot be solved analytically, and thus some form of numerical method must be employed. Free-surface problems by definition involve the further complication that the boundary of the problem domain is not known in advance. Thus the solution of such a problem requires us to find both the location of the free surface and the flow field bounded by it. At the current time no entirely satisfactory general-purpose methods exist for dealing with time-dependent free-surface problems.

While a considerable number of papers have been published describing methods for obtaining solutions to steady-state free-surface problems, relatively few have addressed the subject of time-dependent problems. Free-surface problems can be divided into two important classes: to the first belong problems in which the solution depends only weakly on the shape of the free surface e.g. most problems involving fluids with negligible surface tension; to the second class belong those problems in which surface tension is a dominant influence. It is this latter class of problems to which this work is addressed.

To place the current work in context, it is useful at this point to describe briefly a number of practical applications that arise in the study of free surfaces. One area in which free surfaces are frequently encountered is the application of a thin coating of fluid to a surface. A number of typical *coating flow* geometries are described in the article by Kistler and Scriven [59].

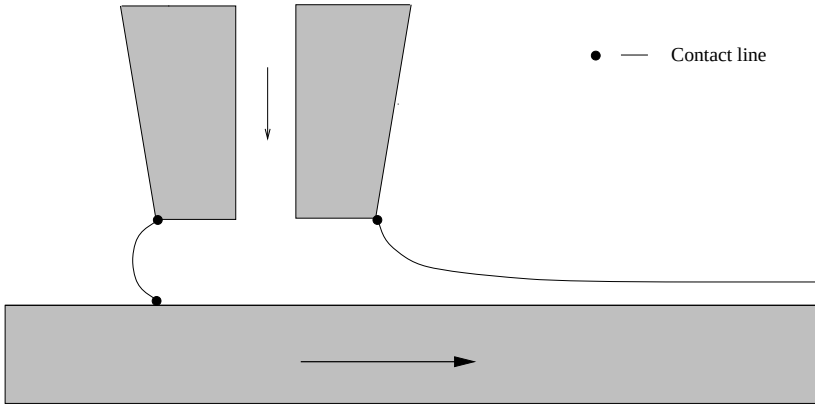


Figure 1.1: Extrusion coater.

Fig. 1.1 shows an idealised coating apparatus — the *extrusion coater*. In such an apparatus a fluid is applied under pressure via a nozzle to a moving substrate. The final thickness of the resulting layer depends only on the flow rate in the nozzle. The uniformity of the coating depends critically on the flow rate and the velocity of the substrate. If the velocity of the substrate is too large then typically some form of dynamic instability will arise — resulting in an uneven application of the coating fluid or ultimately to a complete breakdown of the transfer mechanism. As Fig. 1.1 shows, the problem geometry involves a number of *contact lines* (marked •) each of which requires careful treatment with regard to boundary conditions if a well posed problem is to be specified.

The *curtain coater* geometry illustrated in Fig. 1.2 has many features in common with the extrusion coater. The artificial truncation of the domain implicit in both

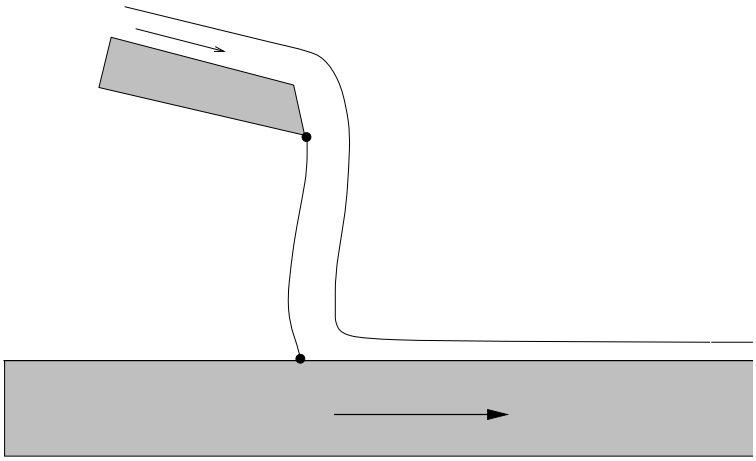


Figure 1.2: Curtain coater.

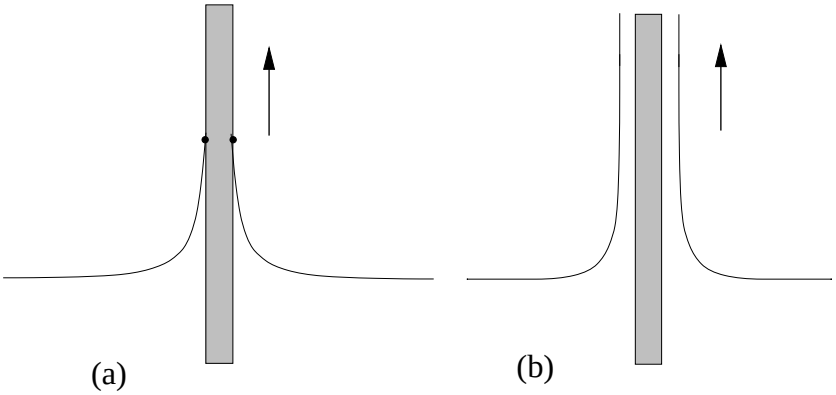


Figure 1.3: Dip coating.

models requires decisions to be made at the modelling stage about the boundary conditions to be applied at the inflow and outflow boundaries. Typically some form of steady fully-developed inflow boundary condition is selected, such as an open-channel (Couette) flow for the curtain coater or a Poiseuille flow for the extrusion coater. In the case of the curtain coater the presence of a free surface at the inflow boundary requires the specification of a boundary condition on the free-surface location. The specification of outflow boundary conditions is also frequently attempted in a similar fashion, though care must be taken that the outflow boundary condition imposed does not interfere with the development of phenomena of interest e.g. oscillations in the free-surface elevation near the outflow.

Fig. 1.3 shows a somewhat simpler configuration resulting from models of coating in which a sheet of material is drawn upwards against gravity, through the free surface of a reservoir of fluid, at a fixed velocity. Note the presence of initial start-up contact lines in Fig. 1.3(a). The problem can be further simplified by ignoring

the complications introduced by the initial contact lines, that is by assuming the free surface has reached a stable steady-state configuration and is thus parallel to the sheet sufficiently far from the reservoir, as shown in Fig. 1.3(b). If in place of a sheet of material, a wire with a circular cross section is considered, then one can take advantage of the rotational symmetry of the problem to allow the modelling of a fully three-dimensional problem (see for example [87]).

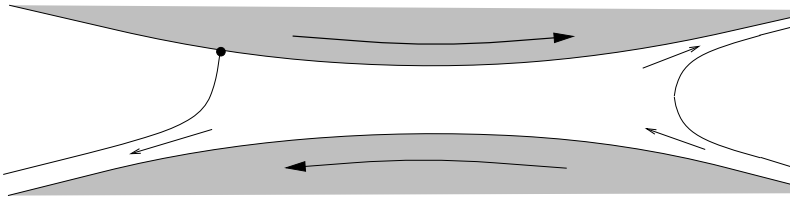


Figure 1.4: Reverse roll coating.

One family of coating problems that is of particular commercial interest results from the study of the various types of roll coating process. Fig. 1.4 illustrates the industrial process known as *reverse roll coating* [29]. In this, two counter-rotating rollers are employed to transfer fluid onto a moving substrate (known as the *web*) that passes around the upper roller.

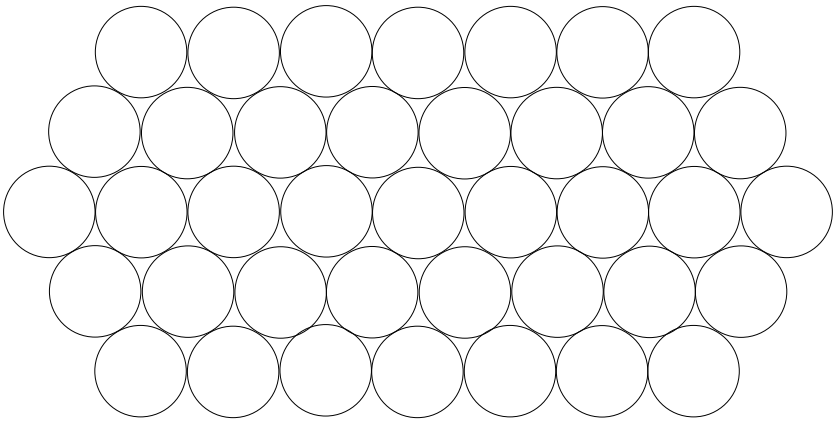


Figure 1.5: Sintering a bundle of cylindrical fibres.

Finally, as an example of a free-surface problem of commercial importance that is not related to coating flows, Fig. 1.5 illustrates the process of the *viscous sintering* of a bundle of cylindrical fibres [113, 69]. In this process, the bundle is heated uniformly, over a long period of time, until it melts, at which point the fibres coalesce under the influence of surface tension. In the industrial processes that motivate this type of study, accurate knowledge of the rate of change of the density of the resulting

material is of importance if the properties of the resulting composite material are to be accurately predicted.

1.2 The Navier-Stokes equations

The flows considered here are all assumed to obey the Navier-Stokes equations subject to the following further assumptions:

1. The fluids involved are incompressible i.e. have constant density ρ;
2. The surface energy density (surface tension) associated with any free surface is a constant γ;
3. The fluids are Newtonian and isothermal and thus have constant, isotropic, (dynamic) viscosity μ.

A number of important simplifications follow from these assumptions. Since all the material properties of the fluids are taken to be constant, there is no need to consider the effects of temperature when solving for the velocity field. Consequently there is no need to solve a separate energy equation. The assumption of constant density has the effect of removing any time-dependency from the continuity equation, and so the continuity equation becomes an algebraic constraint on the velocity field.

The Eulerian form of the Navier-Stokes equations for a fluid satisfying the above assumptions is

$$\rho\left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u}\right] = \mu\nabla^2\mathbf{u} - \nabla p - \rho g\mathbf{j}, \tag{1.1}$$

$$\nabla.\mathbf{u} = 0, \tag{1.2}$$

where g is the acceleration due to gravity, and $\mathbf{j}$ a unit vector in the upward vertical direction. The *momentum equation* (1.1) relates the velocity $\mathbf{u}$ at each point of the fluid to the pressure p, while (1.2), the *continuity equation*, imposes the constraint that the velocity field must be incompressible.

In addition to the effects of gravity, viscosity and momentum, surface tension must also be incorporated into the formulation. The effects of surface tension enter into (1.1) as boundary conditions. At points on a free surface the most convenient boundary condition is continuity of stress. For a gas/liquid interface involving a gas

of negligible viscosity and density, the stress on the free surface takes the form

$$\boldsymbol{\sigma} = -\left(p_{ext} + 2\gamma H\right)\mathbf{n}, \tag{1.3}$$

where p_{ext} is the pressure of the surrounding gas, $\mathbf{n}$ is the (outward) free-surface normal, H is the mean curvature of the surface and γ is the surface tension. In the present work, for simplicity, the assumption is made that the pressure of the surrounding gas is zero. It should be noted that, for problems involving free-surface stress boundary conditions there is no need to specify a hydrostatic pressure datum [24] as is necessary when the boundary conditions involve only the velocity.

For a time-dependent free-surface problem, the boundary-value problem (1.1-1.2) must be augmented by the *kinematic boundary condition* i.e.

$$\left(\frac{\partial \mathbf{s}}{\partial t} - \mathbf{u}\right) \cdot \mathbf{n} = 0, \tag{1.4}$$

where $\mathbf{s}$ corresponds to the position of a material particle on the free surface. Thus, *a material particle on a free surface must remain on the free surface.* Note that (1.4) places a constraint only on the normal component of the velocity, the tangential component being unconstrained.

In the limit as $\mu \to 0$ equations (1.1) and (1.2) reduce to the *incompressible Euler equations* which are commonly used to model fluids, such as water and air, when viscous effects are negligible. It should be noted that the boundary conditions for the Euler equations are somewhat different to those in the viscous case. In particular, at a free surface only a normal stress boundary condition need be specified; a tangential stress boundary condition being inappropriate since an Eulerian fluid is incapable of supporting shear stresses.

The first step in the non-dimensionalisation of a Navier-Stokes problem with a given geometry is the choice of characteristic scales for length L_0 and velocity U_0. These will of course be model specific. Once U_0 and L_0 have been chosen, for advection-dominated problems (i.e. $Re \gg 1$) one typically defines a characteristic time interval $T_0 = \frac{L_0}{U_0}$ and a characteristic pressure $P_0 = \rho {U_0}^2$ [50]. The above scales may now be used to rewrite the problem in terms of non-dimensional variables, here indicated by the superscripts $(^*)$. Thus length x is related to dimensionless length x^* by $L_0\, x^* = x$ etc. In a Cartesian coordinate system the Navier-Stokes equations thus become

$$\frac{\partial \mathbf{u}^*}{\partial t^*} + (\mathbf{u}^*.\nabla)\mathbf{u}^* = \frac{1}{Re}\left[\nabla^2 \mathbf{u}^*\right] - \nabla p^* - \frac{1}{Fr}\mathbf{j}, \tag{1.5}$$

$$\nabla \cdot \mathbf{u}^* = 0, \tag{1.6}$$

where

$$Re = \frac{\rho L_0 U_0}{\mu} \tag{1.7}$$

is the *Reynolds number* appropriate for an advection dominated flow [34], and

$$Fr = \frac{{U_0}^2}{g L_0} \tag{1.8}$$

is the *Froude number*.

The free-surface boundary condition (1.3) must also be non-dimensionalised. Setting $p_{ext} = 0$, and restricting attention the two-dimensional case, we have

$$\boldsymbol{\sigma} = -2\gamma H \mathbf{n} = -\frac{\gamma}{R_c}\mathbf{n}, \tag{1.9}$$

where R_c is the radius of curvature of the free surface. Thus, the dimensionless stress is given by

$$\boldsymbol{\sigma}^* = -\frac{1}{We}\frac{1}{{R_c}^*}\mathbf{n}, \tag{1.10}$$

where

$$We = \frac{\rho {U_0}^2 L_0}{\gamma} \tag{1.11}$$

is the *Weber number*, and ${R_c}^*$ is the dimensionless radius of curvature. For problems such as those involving the oscillation of viscous droplets released initially from rest [7], for which no obvious *a priori* choice of characteristic velocity is apparent, U_0 is commonly chosen so that $We = 1$. From now on the superscripts will be dropped and dimensionless variables together with the appropriate non-dimensional parameters employed unless otherwise stated.

1.3 The Stokes equations

An important simplification of the Navier-Stokes equations results when the effects of momentum are negligible in comparison to those of viscosity. In such situations they reduce to the Stokes (or slow flow) equations of viscous flow:

$$\nabla^2 \mathbf{u} - \nabla p - g\mathbf{j} = 0, \tag{1.12}$$

$$\nabla.\mathbf{u} = 0. \tag{1.13}$$

The absence of a material derivative in the Stokes equations allows the solution of a transient Stokes flow to be obtained by solving a sequence of quasi-steady-state boundary-value problems. Thus there are no initial conditions in the conventional sense. The kinematic boundary condition (1.4) must still be satisfied and now contains the only time-dependent terms in the system. Note, however, that in the absence of boundary conditions constraining the total velocity and angular velocity of the domain (1.12) and (1.13) form a singular system. Thus additional constraints may be necessary to render a problem non-singular [114].

Using an appropriate non-dimensionalisation procedure, (1.12) and (1.13) may be derived as a special case of the Navier-Stokes equations. One approach[67] involves choosing a length scale L_0 and defining characteristic scales for velocity U_0, time T_0 and stress σ_0, using:

$$U_0 = \frac{\gamma}{\mu}, \tag{1.14}$$

$$T_0 = \frac{\mu L_0}{\gamma}, \tag{1.15}$$

$$\sigma_0 = \frac{\gamma}{L_0}. \tag{1.16}$$

The dimensionless equations are thus

$$Su\left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\,\mathbf{u}\right] = \nabla^2\mathbf{u} - \nabla p - Bo\,\mathbf{j}, \tag{1.17}$$

$$\nabla.\mathbf{u} = 0, \tag{1.18}$$

where Su is the dimensionless *Suratman number* defined by

$$Su = \frac{\rho\gamma L_0}{\mu^2}, \tag{1.19}$$

and Bo is the dimensionless *Bond number* defined by

$$Bo = \frac{\rho g {L_0}^2}{\gamma}. \tag{1.20}$$

When the Suratman number is vanishingly small the Stokes approximation is appli-

cable and the momentum equation reduces to the form

$$\nabla^2 \mathbf{u} - \nabla p - Bo\,\mathbf{j} = 0. \tag{1.21}$$

In this case the non-dimensional form of the stress boundary condition is simply

$$\boldsymbol{\sigma}^* = -\frac{1}{R_c^*}\mathbf{n}. \tag{1.22}$$

Thus, there is no additional non-dimensional group associated with the free-surface boundary conditions.

1.4 Overview of the difficulties involved in free-surface modelling

When evaluating numerical schemes for the solution of free-surface problems the twin, interrelated, issues of efficiency and accuracy must always be borne in mind. The primary requirement for any practical numerical method is that it be capable of producing accurate results, employing affordable computational resources, within an acceptable period of time. The simulation of a two- or three-dimensional transient free-surface problem typically requires the period of integration to be split into many hundreds if not thousands of time steps. At each time step a linear or nonlinear algebraic system, involving typically many thousands of variables, must be solved to a high accuracy if the results are to be useful. Furthermore, in practice, a model may have to be run many times with different parameter values e.g. as part of an optimisation study.

One way the computational resources required to solve a particular problem may be minimised is by reducing the number of unknown variables to a minimum. Offset against this must be the need to employ a sufficiently fine discretisation of the domain to ensure adequate accuracy. By employing adaptive methods it is possible, in principle at least, to locate nodes in regions of the flow where they are most needed — allowing the solution of problems which would be prohibitively expensive if the nodes where uniformly distributed.

Another important consideration is the need to find a suitable balance between the accuracy of the time-integration scheme and the accuracy of the computed flow solutions. In practice one would like to employ as large a time step as possible,

since for many free-surface problems even simple time-integration schemes give rise to errors that are small in comparison to those arising from the spatial discretisation. Unfortunately, however, it is often necessary to restrict the size of time step in order to ensure stability of the solution.

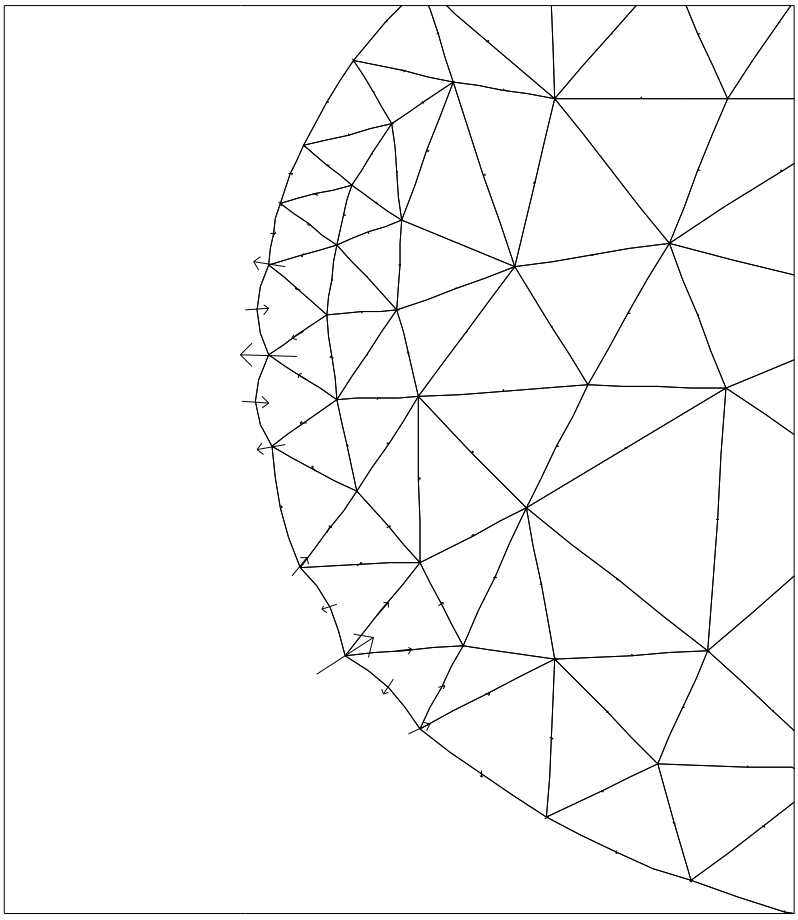


Figure 1.6: Free-surface instability: arrows show velocity at nodes.

Fig. 1.6 illustrates a form of instability that is frequently observed when too large a time step is employed for a scheme involving an explicit time discretisation of the kinematic boundary condition. Fig. 1.6 shows part of the free surface of a cylinder of viscous fluid, with elliptical cross-section, evolving under the influence of surface tension, the governing equations being those of Stokes flow. Note that here, the flow equations are being solved implicitly. The large velocities visible at a number of free-surface nodes typically reverse sign at each time step. Since the amplitude of these oscillations normally grows rapidly, the onset of this 'saw-tooth' instability almost invariably signals the imminent failure of a simulation.

Similar instabilities have also been observed when solving viscous free-surface

problems using the boundary-element method (BEM) [60], and also with inviscid free-surface problems [75]. Fully implicit methods (involving simultaneous solution for the velocity, pressure and free-surface position) are generally believed to be free from such stability restrictions on time-step size. Although the nonlinear nature of free-surface problems makes theoretical results difficult to obtain, practical experience appears to support the view that fully implicit schemes allow much larger time steps to be employed [61]. Fully implicit methods are however considerably more complicated to implement, particularly when the nodes of the computational mesh are in motion, as is generally necessary for free-surface problems. Fully implicit schemes are further discussed in Chapter 3.

When surface tension is large enough to influence a flow appreciably, the accurate representation of the free-surface boundary becomes particularly important. Unlike the majority of computational fluids problems, in a free-surface problem one does not know the boundary location *a priori*. For surface-tension-driven flows the boundary conditions depend primarily on the curvature of the boundary. Since the accuracy of the computed boundary conditions depends on the accuracy of the free-surface representation, *for a given free-surface shape, if the computational mesh is refined, the imposed boundary conditions will change, even at points that are common to both meshes*. The overall rate of convergence of the error in the solution is potentially compromised by this effect.

A related issue is the need to ensure that the accuracy of the free-surface representation and the accuracy of the velocity field used to update the free surface are compatible. Thus, ideally, the velocity solution will have the same order of spatial accuracy as the free-surface representation it is used to update.

1.5 Methods for free-surface flows

With the exception of approaches employing a global basis, such as Fourier methods, numerical schemes typically require, as a first step, the division of the computational domain into a finite number of discrete regions or *elements*. By defining a local basis on each individual element, an approximation to the desired global solution can then conveniently be represented. The process of spatial discretisation results in a set of simultaneous, initial-value ordinary differential equations, involving a set of unknowns corresponding to the values of the dependent variables at each of the nodal points. Free-surface methods may be classified according to the schemes they employ in forming a discretisation of the spatial domain and its boundary. The

main families of methods will now briefly be described.

1.5.1 Boundary-element methods

Boundary-element methods have the important advantage that they require the spatial discretisation only of the boundary of the domain, thereby resulting in much smaller systems of equations than those arising from methods that discretise the entire domain. A further practical advantage of boundary-element methods over finite element methods is that the implementation is considerably simplified, since an interior mesh is unnecessary. An important limitation of this type of approach is that the solution is computed only on the boundary. Computation of the interior flow is only possible at significant extra cost. Furthermore, boundary-element methods are applicable only to a limited range of CFD problems; in particular they cannot be employed for Navier-Stokes problems.

Where only the free-surface evolution is required, the BEM would appear to have a great advantage in that it considerably reduces the number of unknowns that must be found at each time step. However, since the resulting system of equations is normally dense, whereas a finite-element or finite-difference method would typically result in a sparse system, the advantages of the BEM are perhaps not as clear cut as they might at first seem.

Boundary-element methods have been used successfully for the solution of two-dimensional Stokes-flow problems [65, 56, 51, 37, 60, 115, 114, 113, 38, 62], and also for two- and three-dimensional potential flow problems [122, 110, 116, 117].

1.5.2 Finite-difference methods

One of the most important advantages of employing a finite-difference method is the simplicity of the approach. A regular Cartesian grid is defined which is large enough to encompass the entire region the fluid is likely to occupy. At any given time the fluid actually occupies only a portion of this grid, the location of the free surfaces being represented by auxiliary data structures. The regularity of the grid allows the economical assembly of the systems of equations involved. Regular grids also have the important property that the solutions obtained on them frequently exhibit the property of *superconvergence* [103] i.e. that the solutions obtained at the nodes are of a higher degree of accuracy than would be obtained on unstructured meshes. The regularity of the grid can also have important advantages when it comes to constructing preconditioning methods to speed the solution of the associated systems

of equations by iterative methods.

One disadvantage lies in the potentially large storage requirements of the approach, if implemented naively, since storage must be reserved for the degrees of freedom at each grid point. The greatest difficulties, however, result from the need to reconstruct a smooth free-surface boundary at each time step in order that the boundary conditions can be computed. For surface-tension-dominated flows this problem becomes particularly troublesome. Furthermore, once the boundary conditions have been computed, there remains the difficulty of having to impose them at a set of discrete nodes which will not in general lie on the reconstructed free surface. While boundary fitted grids have been suggested [109, 122] as a remedy for this difficulty, it is not clear that they have any advantages over finite element methods. A further difficulty arises from the requirement that any free-surface scheme must conserve mass (or fluid volume) to considerable accuracy, if it is to be of any practical use for time-dependent problems. This can be difficult to achieve using finite-difference methods.

While the above problems make finite-difference methods difficult to apply to surface-tension-driven flows, the great flexibility of finite-difference schemes for dealing with complex fluid geometries has led to their continued use for problems in which the flows are dominated primarily by momentum and gravity, e.g. sloshing in fuel tanks [106, 116, 100].

1.5.2.1 Marker-and-cell methods

One of the earliest finite-difference schemes for free-surface flows, the marker-and-cell or MAC method [39, 88, 71] involves the use of a large number of massless marker particles to track the motion of the free surface. Marker particles are distributed throughout the fluid, with greatest density near free surfaces, and move passively with the fluid. At the end of a time step, their new locations are used to reconstruct the position of the free surface, as illustrated by Fig. 1.7. This necessarily involves some form of numerical smoothing if the curvature of the resulting free surface is to be computed accurately.

One practical consideration is that a scheme must be decided upon to control the insertion and removal of marker particles so as to allow an accurate representation of the free surface to be maintained. The large number of marker cells that may be required can lead to considerable overheads. While variants of the MAC method are still in use for inviscid flows with negligible surface tension, for the reasons discussed above they are rarely employed for surface-tension-driven flows, though

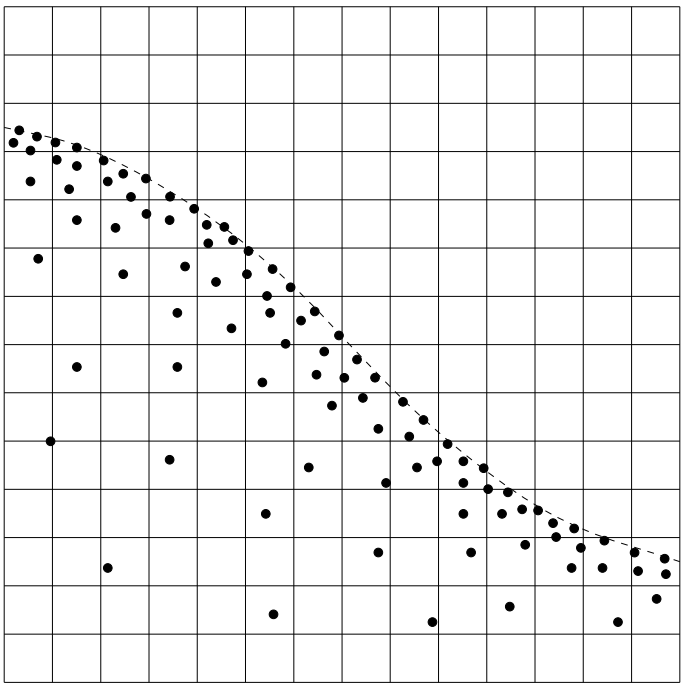


Figure 1.7: The marker-and-cell (MAC) method: • marker particle; - - - - reconstructed free surface.

a related approach — the immersed interface method [112, 61] — appears to hold promise for surface-tension-driven flows.

1.5.2.2 Volume-of-fluid methods

As an alternative to methods that attempt to model the motion of the free surface by direct application of the kinematic boundary condition, a number of so-called volume-of-fluid (VOF) methods have been developed [42, 68, 58] that use considerations of mass conservation in updating the free surface's location. These methods operate by integrating the flux into regions adjacent to the free surface. The most important advantage of such schemes is that in principle they conserve mass exactly, since each flux integral is repeated twice: once for material flowing into a region and once for material flowing out of it.

The need to know the regions over which the integrals are to be evaluated in advance has hitherto restricted the application of such methods to problems that can be tackled with regular meshes. There is, however, no reason why such an approach could not be generalised to unstructured meshes. One potential theoretical concern stems from the fact that, since these methods discard the boundary velocities computed by the flow solver in favour of their own estimates of what the velocities

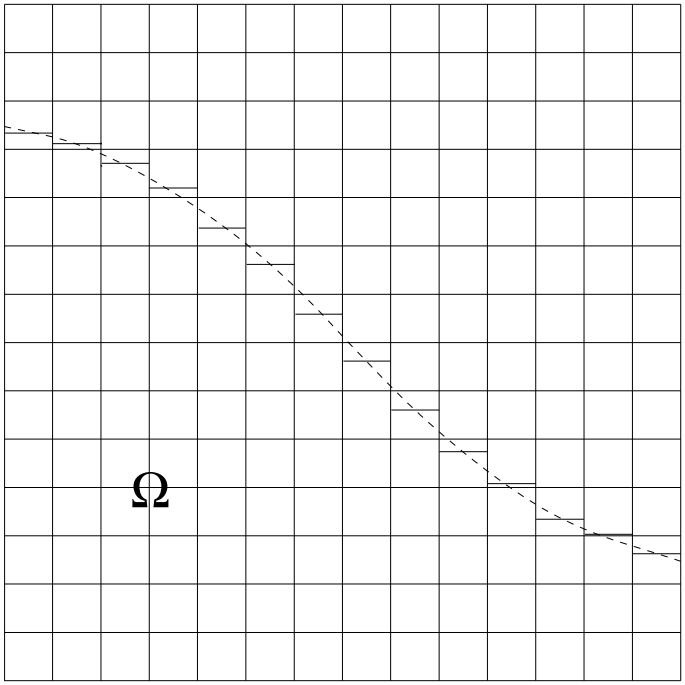


Figure 1.8: First-order accurate VOF method: - - - - exact free surface; — reconstructed free surface.

should be, there is a potential for loss of accuracy since the kinematic boundary condition is not satisfied exactly. Whether this is ever a serious problem in practice remains to be demonstrated.

Fig. 1.8 illustrates a simple first-order-accurate VOF free-surface reconstruction scheme applied to a domain Ω. The free surface is updated after each time step by integrating the flux into each column of the mesh. The new column volume then directly gives the fraction of fluid in the cell adjacent to the free surface. For the purposes of computing free-surface boundary conditions a smooth curve must now be fitted in some fashion to the resulting free-surface data. A possible reconstruction of the free surface using cubic-splines is shown in Fig. 1.8.

Fig. 1.9 depicts the reconstruction of the same free surface using a second-order-accurate boundary representation. This time, once the new column volumes have been computed, a set of simultaneous equations is solved [68] to give the angles and positions of the segments of the piecewise-linear free-surface representation. As may be seen from Fig. 1.9 provided the free-surface curvature is not too great an apparently accurate representation of the free-surface may be obtained using such a piecewise linear interpolant. VOF methods can also be applied in a finite element setting [68], with the advantage that many of the difficulties of applying natural boundary conditions to a finite-difference discretisation do not arise.

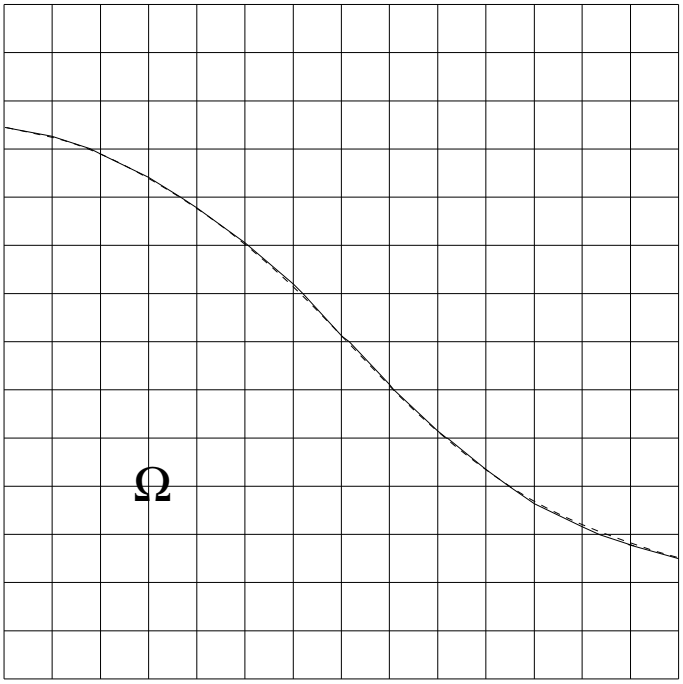


Figure 1.9: Second-order accurate VOF method: - - - - exact free surface; — reconstructed free surface.

1.5.2.3 Phase field methods

Phase field methods avoid the need to maintain a discrete boundary representation by introducing a new continuous variable ϕ, a level set of which, typically $\phi(\mathbf{x}) = 0$, is taken to represent the free surface. Potentially such methods combine great geometric flexibility with the advantages of employing regular grids and maintaining a continuous representation of the free surface. The main difficulties come from the need to solve an additional system of equations for ϕ at each time step and the need to incorporate surface-tension effects through a discontinuous forcing function. While the suitability of phase-field methods for surface-tension-driven incompressible-flow problems has only recently been demonstrated [104], they appear to have great potential.

1.5.3 Finite element methods

Finite element methods have a considerable advantage over finite-difference methods in the way in which boundaries and interfaces can be directly represented by the edges of elements. Furthermore, where such an edge forms part of the boundary of the computational domain, the correct free-surface boundary conditions may be applied in an elegant and direct manner by using an appropriate weak formulation

[34] of the Navier-Stokes equations.

1.5.3.1 Fixed-connectivity meshes

The simplest finite element discretisation schemes for free-surface problems make use of a mesh with fixed connectivity. The mesh illustrated in Fig. 1.10 is typical of those that might be employed in the simulation of a flow over a plane surface. To accommodate the evolution of the free surface without the mesh becoming too distorted the interior nodes are moved vertically, according to a predetermined rule, in response to the motion of the free surface. For problems involving only small changes in the shape of the free surface the fixed-connectivity-mesh approach is often satisfactory. However where more severe free-surface motions arise the method is liable to fail, e.g. if the free surface becomes vertical, such as when an overturning wave develops. In such circumstances the set of discrete equations associated with the problem becomes singular.

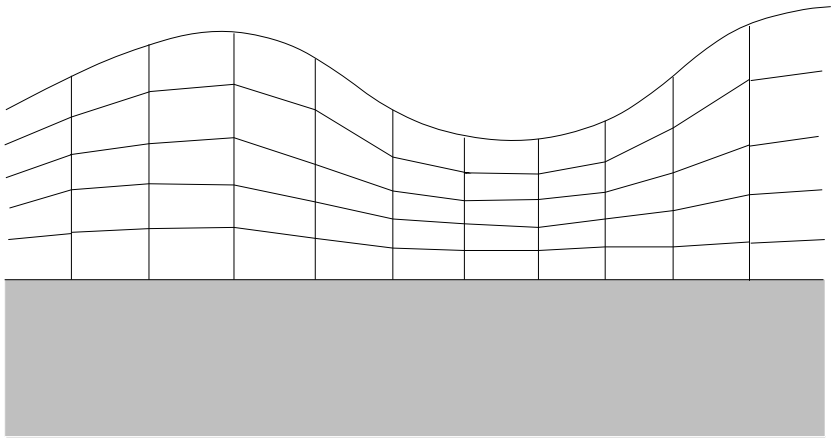


Figure 1.10: A fixed-connectivity mesh for the finite element method.

The advantages of the fixed mesh-connectivity approach, where it is applicable, are many. Maintaining a fixed mesh connectivity throughout a simulation means that the matrices associated with the discrete equations and with the flow solver will have fixed sparsity patterns. In particular it allows the reuse over many time steps of the Jacobian and any preconditioning matrices employed by iterative flow solvers.

The *method of spines* is essentially a variation on the approach described above. It involves the selection of an origin and a set of fixed *spines* through the origin that intersect the free surface. The domain is then discretised as shown in Fig. 1.11. As the free surface evolves, free-surface and interior nodes move radially along the

spines allowing the mesh to deform continuously. The mesh in Fig. 1.10 can be seen to correspond to a spine representation with an origin at $y = -\infty$.

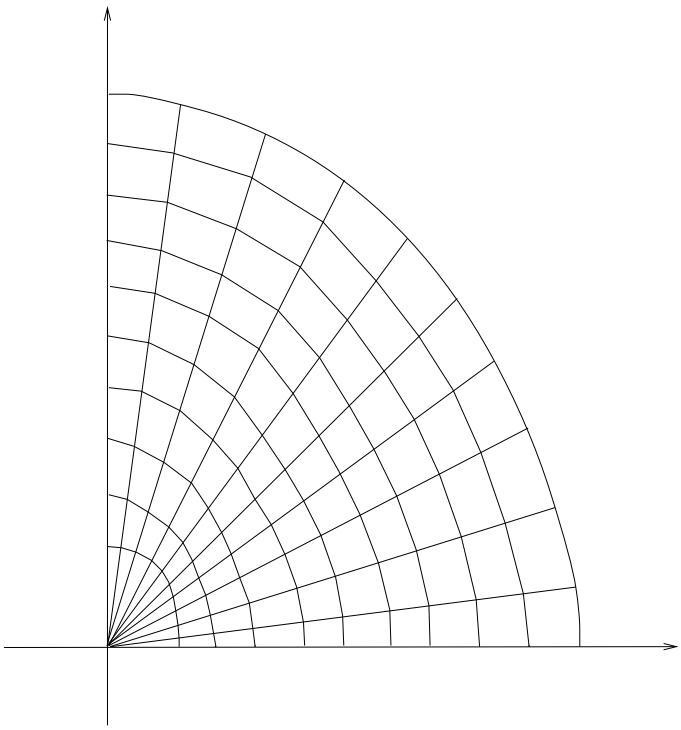


Figure 1.11: The method of spines: typical domain geometry.

To deal with more complex geometries, multiple origins may be employed for different regions of the domain, so long as the distribution of nodes along the boundary of each region is carefully chosen. This is illustrated by Fig. 1.12, which shows a detail of a mesh that might be used for the extrusion coater geometry depicted in Fig. 1.1. For liquid droplet problems involving large free-surface deformations, the use of a moving origin for the spine representation has been described [68].

The need to select the mesh connectivity and node movement scheme in advance for each new problem geometry makes fixed-mesh-connectivity methods difficult to generalise to arbitrary domains. In particular it is unclear how they might be employed in a fully automatic general-purpose code. Also, while superficially elegant, they suffer from the drawback that they cannot easily be modified to allow local mesh refinement to take place where the solution requires it. Thus, in practice, a fixed-connectivity mesh is unlikely to be optimal.

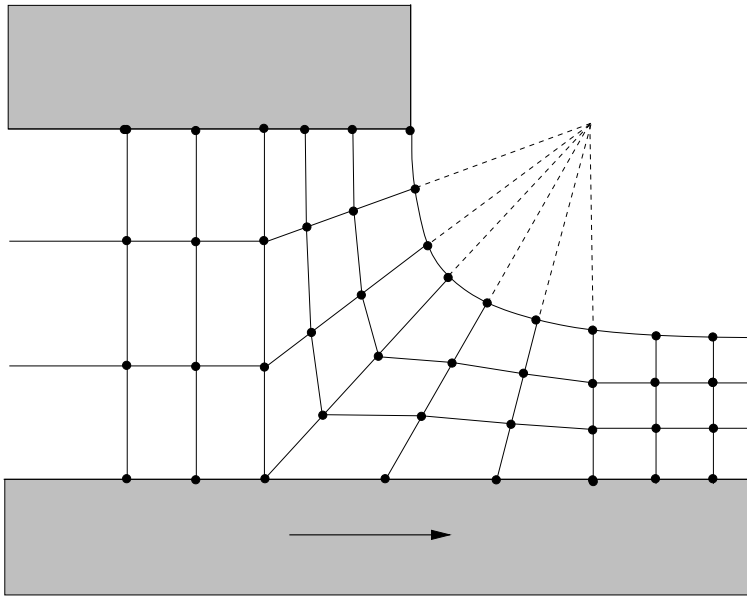


Figure 1.12: Extrusion coater mesh: detail.

1.5.3.2 Unstructured meshes

One approach that has considerable appeal as a route to developing general-purpose finite element codes for free-surface problems, involves the use of unstructured meshes, that is, meshes about which no *a priori* assumptions about connectivity or regularity are to be made. Convenient methods have in recent years been developed for generating unstructured meshes automatically for domains with arbitrary shape [96]. One of the more popular approaches involves the refinement of a crude initial mesh by inserting points until the mesh has the required nodal density in each region of the domain. By selecting insertion points using the Delaunay method [31] a mesh that is suitable for the finite element method is readily obtained.

Rather than remeshing the domain after each time step, it is often possible to reuse an existing mesh by simply displacing some or all of the interior nodes in some continuous fashion. This dual approach of employing a continuous motion of the interior nodes whenever possible, but falling back on a fully automatic remeshing algorithm when necessary, appears to be a promising method for applying the finite element method to complex free-surface geometries and is therefore the approach considered in the remainder of this thesis.

1.6 Conclusions

Many commercially important free-surface problems require inflow and outflow boundaries to be included in their models. The presence of dynamic and static contact lines is also a common feature of such problems. Nevertheless, much useful work can be done without these added complications.

There are considerable differences between the formulations required for time-dependent free-surface Stokes and Navier-Stokes problems. These have a profound effect on the solution strategies that must be employed.

The finite element method is adopted for reasons of generality, but also because of its suitability for modelling free surfaces. Conventional finite element methods are incapable of solving many interesting problems without the periodic intervention of the user, who must supply a new mesh whenever the old one becomes unsuitable.

Chapter 2

The finite element method

In this chapter methods for the automatic generation of meshes for time-dependent free-surface problems are described. Once the finite element basis has been introduced the convergence properties of unstructured meshes are investigated. Difficulties associated with the computation of free-surface stress boundary are highlighted and techniques for reducing the cost of computations in the light of the difficulties identified are described.

2.1 Elements for incompressible flow

The first step in the application of the finite element method to any problem is the division of the domain into a large number of non-overlapping polygonal regions or *elements*. In two dimensions the obvious choices for elements are quadrilaterals or triangles. In the current work triangular elements are employed since they allow the greatest flexibility when dealing with arbitrary geometries. Since the intention is to use a primitive-variable formulation [34] of the Navier-Stokes equations, an element is required that can be used to interpolate both velocity and pressure fields. The element selected is the Taylor-Hood triangular element [36], shown in Fig. 2.1(c), with three corner or vertex nodes and three midpoint or edge nodes. All six nodes have a pair of velocity variables associated with them, corresponding to the u and v velocity components. In addition, the three corner nodes have pressure variables associated with them, hence the designation V6-P3 for the element. Thus, with the

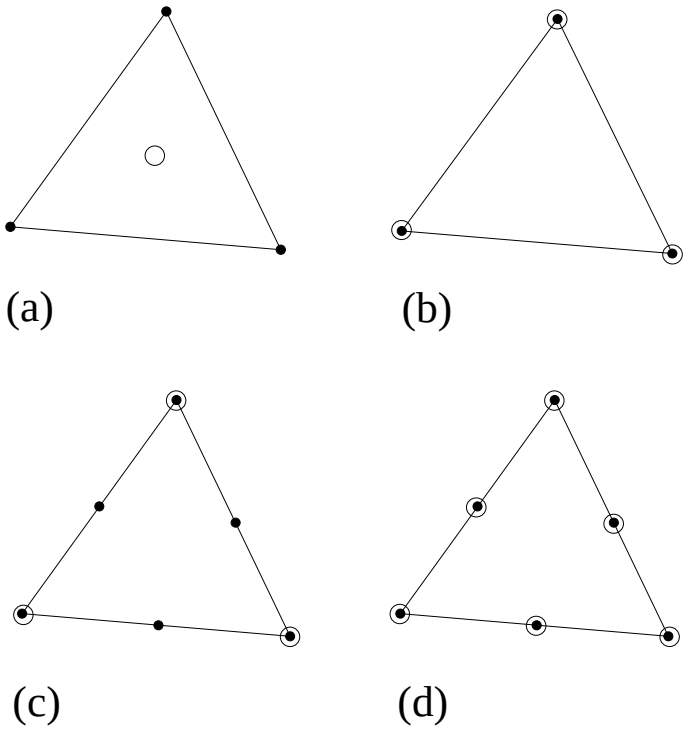


Figure 2.1: Common elements for incompressible flows: • velocity degree of freedom; ◦ pressure degree of freedom.

basis employed, the element is capable of exactly interpolating a quadratic velocity field but only a linear pressure field. The fact that the nodes making up a typical edge are common to the two elements sharing that edge means that a solution represented on a mesh of these elements will be C^0 continuous in both velocity and pressure.

2.1.1 LBB stability

The elements shown in Fig. 2.1 are all potentially useful for the modelling of incompressible flows. Their use is, however, complicated by the existence of a form of instability peculiar to incompressible-flow problems, but not restricted to finite element methods. Without the use of appropriate stabilisation techniques [119, 97, 34] the elements (a), (b) and (d) all fail to satisfy the Ladyzhenskaya-Babushka-Brezzi (LBB) stability condition [36]. The failure of a discretisation to satisfy the LBB condition often results in the occurrence of spurious mesh-scale oscillations in the pressure fields computed. Essentially, the problem is that unstable elements result in systems containing 'too many' continuity constraints [34]. One solution is to employ a lower-order approximation for the pressure than that used for the velocity,

as in (a) and (c), though even this is insufficient in the case of element (a).

While, for smooth solutions, higher-order interpolants are generally more efficient, in that fewer elements are required to obtain the same accuracy, they are considerably more complicated to implement. The Taylor-Hood element, Fig. 2.1(c), appears to be a good compromise between accuracy and simplicity and, since it is intrinsically LBB stable, it is a natural choice.

2.1.2 The isoparametric mapping

Central to the finite element method is the idea of a continuous invertible mapping **F** between a *master element* in *local-coordinate space* and a general element located in the *problem-coordinate space*. The existence of such a mapping considerably simplifies the setting-up of the finite element matrices in that it allows all the necessary integrations to be performed over a fixed region — the master element. Figure 2.2

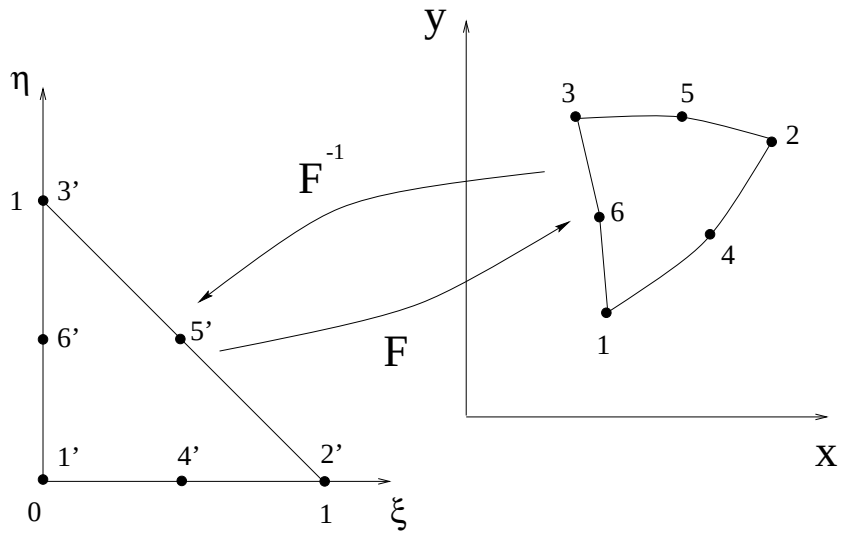


Figure 2.2: The isoparametric mapping **F** between master element and general element.

shows the isoparametric mapping **F** between the master element and a general element. Where straight-sided triangular elements are employed, the mapping **F** takes the form of an affine, subparametric transformation between the two spaces. This allows the master element to be mapped onto any non-degenerate triangular element, irrespective of its position, orientation, shape and size. The use of quadratic and higher-order elements opens up the possibility of employing an *isoparametric transformation* [34], in which the basis functions used to define the solution are also used to define the mapping onto problem space, allowing quadratic and higher-order

elements to have curved edges. This is potentially very useful when dealing with a domain with a curved boundary and allows the accurate representation of curved boundaries with far fewer elements than would be required with straight-sided elements, as illustrated by Fig. 2.3.

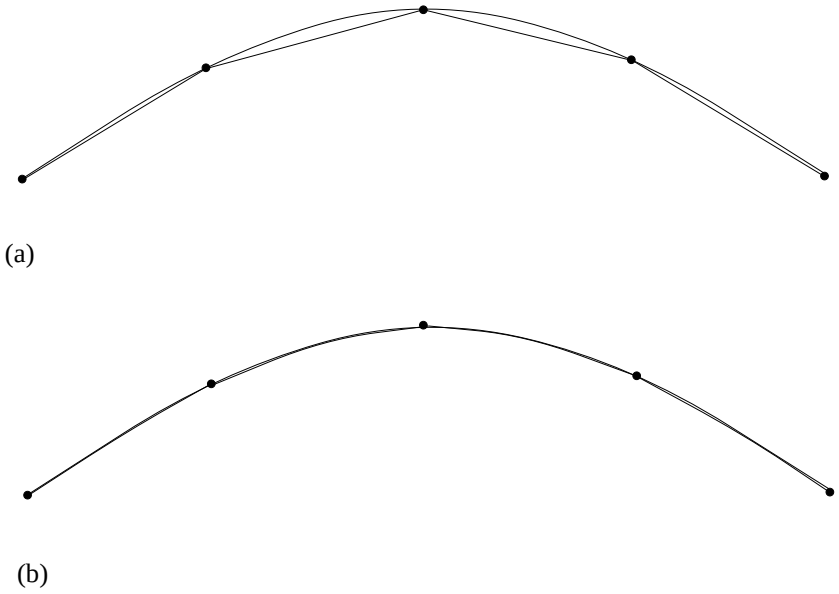


Figure 2.3: Free-surface boundary representation: (a) piecewise linear, (b) piecewise quadratic.

The shape of a curved side of a general element may be selected by perturbing the edge node from its initial mid-edge location, the resulting curved side being given by the Lagrange interpolation polynomial through the three nodes specifying that edge.

The important question arises as to whether the use of the isoparametric mapping affects the accuracy of the interpolated solution. Strang and Fix [103] analyse the accuracy of quadratic interpolants on simple quadratic triangular elements in the context of the solution of linear elasticity problems. They state that, provided edge nodes are displaced by no more than a distance of $O(h^2)$ from the midpoint of an edge of length h, the solution will have the same formal order of accuracy, $O(h^3)$, as it would have if an affine transformation were employed, though presumably with a larger constant in the leading term of the error estimate. Thus the use of quadratic isoparametric elements potentially gives an extra order of accuracy for the boundary representation without compromising the order of accuracy of the velocity interpolant. While there does not appear to be any published theoretical work generalising Strang and Fix's results to mixed velocity/pressure formulations

for incompressible-flow problems, the preliminary results recounted in Section 2.1.4 show that there is good reason to believe that the use of curved-sided isoparametric elements to represent a free surface need not affect the orders of convergence in velocity and pressure solutions observed in such circumstances. The use of isoparametric elements does however have a price: the integrals required in forming the finite element matrices are considerably more expensive for curved-sided elements than for straight-sides ones, and thus it makes sense to employ curved sides only for those elements adjacent to the free surface and other curved boundaries.

2.1.3 Finite element basis functions

An element's nodes are numbered locally in the anti-clockwise sense as shown in Fig. 2.2. Thus on each element the velocity and pressure interpolants, $\mathbf{u}$ and p, are given by

$$\mathbf{u} = \sum_{j=1}^{6} q_j \mathbf{u}_j, \tag{2.1}$$

and

$$p = \sum_{j=1}^{3} l_j p_j, \tag{2.2}$$

where the functions q_j and l_j are, respectively, the quadratic and linear Lagrange basis functions [85] associated with node j of the element, and where $\mathbf{u}_j$ and p_j are the unknowns associated with the node. The Lagrange basis functions are defined in terms of the local or element coordinates (ξ, η) as follows:

$$l_1 = 1 - \xi - \eta \tag{2.3}$$

$$l_2 = \xi \tag{2.4}$$

$$l_3 = \eta \tag{2.5}$$

$$q_1 = l_1(2l_1 - 1) = 2\xi^2 + 4\xi\eta + 2\eta^2 - 3\xi - 3\eta + 1 \tag{2.6}$$

$$q_2 = l_2(2l_2 - 1) = 2\xi^2 - \xi \tag{2.7}$$

$$q_3 = l_3(2l_3 - 1) = 2\eta^2 - \eta \tag{2.8}$$

$$q_4 = 4l_1l_2 = -4\xi^2 - 4\xi\eta + 4\xi \tag{2.9}$$

$$q_5 = 4l_2l_3 = 4\xi\eta \tag{2.10}$$

$$q_6 = 4l_3l_1 = -4\eta^2 - 4\xi\eta + 4\eta \tag{2.11}$$

It is important to note that the velocity interpolants defined using the Taylor-Hood element are not guaranteed to be divergence free in either the pointwise sense nor as an integral over the element. Interpolants that are divergence free, in the latter sense, can be obtained if an alternative element such as the augmented Taylor-Hood element [108] is employed. These add a further, piecewise-constant, pressure degree of freedom to each element, resulting in solutions for which the integral of the divergence over each element is zero, i.e. the net mass flux into the element is zero. Such elements however have the disadvantage that the pressure solution is now, in general, discontinuous at element boundaries. Furthermore, the addition of the new pressure degrees of freedom complicates the imposition of boundary conditions, due to the possible presence of additional spurious pressure modes [34].

2.1.4 A Stokes-flow test problem

Before discussing the additional difficulties faced when solving surface-tension-driven free-surface problems on finite element meshes it is appropriate to first consider the convergence properties of the spatial discretisation employed here. In particular it is important to verify that the theoretical convergence rates are attained where natural boundary conditions are employed, where unstructured meshes are used and where the boundary is represented by curved-sided isoparametric elements. In theory, when a finite element discretisation is intrinsically LBB-stable the full asymptotic rate of convergence will be obtained in both the velocity and the pressure variables [36]. Thus for the element employed here, one would expect to find that the error in the velocity components varies as $O(h^3)$ and that the error in the pressure varies as $O(h^2)$, where h is a measure of mesh resolution such as element diameter or edge length — the *mesh parameter*. Checking that these convergence rates are observed is a useful way of testing the correctness of the implementation and, in the light of the discussion in the remainder of this chapter, will be seen to be particularly important. For this purpose a standard Stokes-flow test problem [18] is considered. It is easily confirmed that the steady-state Stokes equations

$$\nabla^2 \mathbf{u} - \nabla p + \mathbf{g} = 0, \tag{2.12}$$

$$\nabla . \mathbf{u} = 0, \tag{2.13}$$

with body force $\mathbf{g} = (g_x, g_y)$, given by

$$\begin{aligned} g_x &= 2x - 4\Big(y(1-3y+2y^2)(1-6x+6x^2) + 3x^2(1-2x+x^2)(2y-1)\Big), \\ g_y &= -2y + 4\Big(x(1-3x+2x^2)(1-6y+6y^2) + 3y^2(1-2y+y^2)(2x-1)\Big), \end{aligned}$$

have the following exact solution

$$\begin{aligned} u(x,y) &= x^2(1-x)^2(2y-6y^2+4y^3), \\ v(x,y) &= y^2(1-y)^2(-2x+6x^2-4x^3), \\ p(x,y) &= x^2 - y^2. \end{aligned}$$

Such a solution cannot be represented exactly using the elements employed here and thus provides a convenient way of estimating the accuracy of solutions obtained. A family of circular meshes with radius 0.5 and centre $(0.5, 0.5)$ are employed, as detailed in Table 2.1. The second column of Table 2.1 gives the mesh parameter h for each of four meshes, while the third column gives N_v the number of vertices in each mesh. Figure 2.4 shows a typical mesh, corresponding to mesh 2 in Table 2.1. As can be seen, the meshes are unstructured ones and have curved boundary edges [1]. The mesh generation procedures used to generate the meshes are described in Section 2.3. In order to solve this problem the finite element formulation described in Section 3.7 is employed, the boundary being held fixed, and the resulting systems of linear equations solved by the methods described in Section 3.10. Natural boundary conditions corresponding to the stress $\boldsymbol{\sigma} = (\sigma_x, \sigma_y)$, computed using the formulae

$$\begin{aligned} \sigma_x &= -(x^2-y^2)n_x \\ &\quad +2n_x\Big[2x(1-x)^2(2y-6y^2+4y^3) - 2x^2(1-x)(2y-6y^2+4y^3)\Big] \\ &\quad +n_y\Big[x^2(1-x)^2(2-12y+12y^2) + y^2(1-y)^2(-2+12x-12x^2)\Big], \end{aligned}$$

$$\begin{aligned} \sigma_y &= -(x^2-y^2)n_y \\ &\quad +2n_y\Big[2y(1-y)^2(-2x+6x^2-4x^3) - 2y^2(1-y)(-2x+6x^2-4x^3)\Big] \\ &\quad +n_x\Big[x^2(1-x)^2(2-12y+12y^2) + y^2(1-y)^2(-2+12x-12x^2)\Big], \end{aligned}$$

[1] Note that the slight oscillations apparent on close inspection of the free surface are an artefact due to the limited resolution of the method of reproduction employed. In reality all the interior edges are perfectly straight. When viewed at a higher resolution the free surface appears quite smooth.

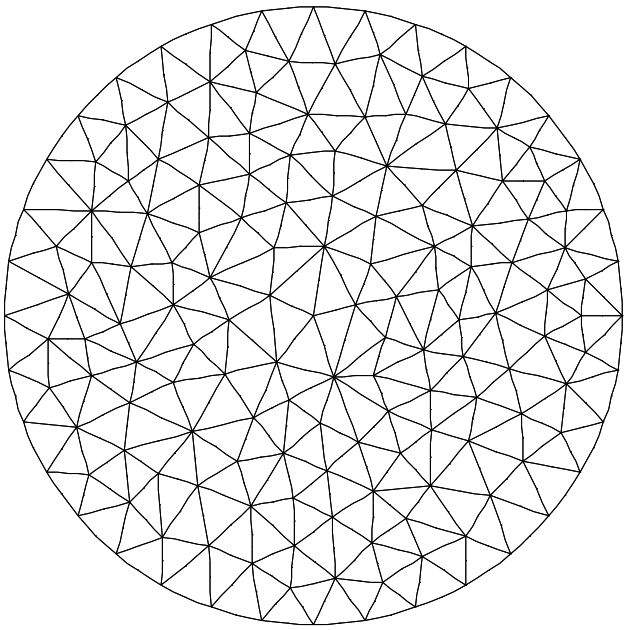


Figure 2.4: Stokes-flow test problem, unstructured circular mesh of radius 0.5 centred at $(0.5, 0.5)$: mesh 2 of Table 2.1.

were imposed at all but two boundary nodes. Dirichlet boundary conditions were imposed at the two boundary nodes lying on the line $y = 0.5$ so as to make the problem non-singular. Figures 2.5 and 2.6 show respectively the velocity and pressure fields computed on mesh 2. Note that no post-processing of the velocity or pressure fields is performed for this or any of the other figures in this thesis.

Mesh	h	N_v	Max. error u	Max. error p	Mean error u	Mean error p
1	0.5000	41	3.11E-4	1.45E-2	8.19E-5	4.51E-3
2	0.2500	161	5.16E-5	2.30E-3	1.08E-5	5.34E-4
3	0.1250	663	5.30E-6	5.48E-4	1.36E-6	1.02E-4
4	0.0625	2594	6.81E-7	1.34E-4	1.43E-7	2.46E-5

Table 2.1: Isoparametric V6-P3 element convergence-rate data.

In order to assess the accuracy of the computed solutions as the mesh was refined, the maximum absolute nodal errors in the u component of the velocity and in the pressure were recorded together with the corresponding average absolute nodal errors. These are shown as columns four through seven of Table 2.1. Note that with an unstructured mesh superconvergence [123] is not expected to occur, and thus the accuracy of the solution at nodes will be of the same order as that at arbitrary

points[2]. Least-squares analysis of the errors listed in Table 2.1 gave the following approximate relationships:

$$\begin{aligned}
\text{Maximum nodal error in u} &\approx 2.7 \times 10^{-3}\, h^{3.0}, \\
\text{Average nodal error in u} &\approx 7.1 \times 10^{-4}\, h^{3.1}, \\
\text{Maximum nodal error in p} &\approx 6.0 \times 10^{-2}\, h^{2.2}, \\
\text{Average nodal error in p} &\approx 2.1 \times 10^{-2}\, h^{2.5},
\end{aligned}$$

confirming that the error in the velocity components is $O(h^3)$, and that the error in the pressure is $O(h^2)$ — the theoretical maximum obtainable rates. These results give reason to be confident in the correctness of the implementation of the finite element codes, as well as confirming that the imposition of natural boundary conditions, on a boundary comprised of curved-sided isoparametric edges, need not compromise the asymptotic order of accuracy of the solutions obtained.

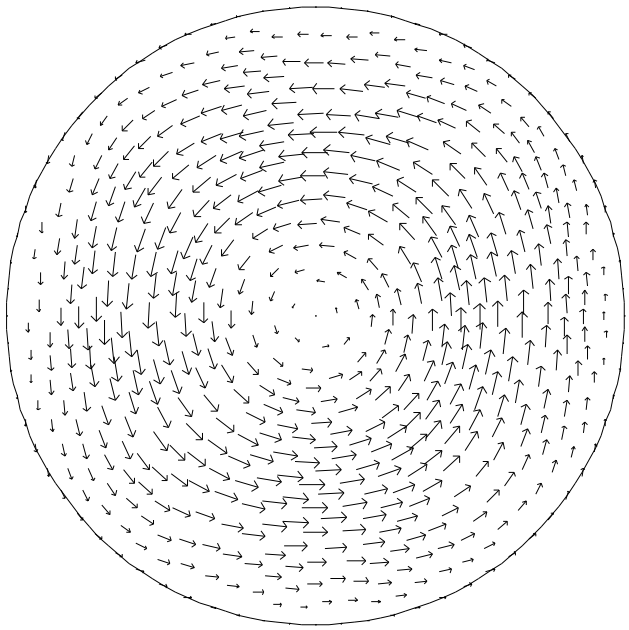


Figure 2.5: Stokes-flow test problem on an unstructured mesh, domain of radius 0.5 centred at $(0.5, 0.5)$: velocity field on mesh 2. $v \approx 0.012$ at $(0.75, 0.5)$.

[2]When regular meshes based on a uniform grid were investigated, convergence rates approaching $O(h^4)$ were observed in both u and p.

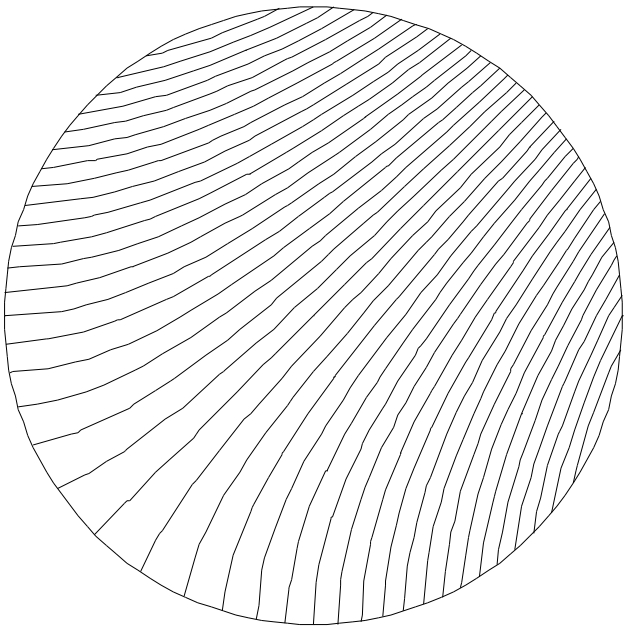


Figure 2.6: Stokes-flow test problem on an unstructured mesh, domain of radius 0.5 centred at $(0.5, 0.5)$: pressure field on mesh 2, equispaced isobars, $p = 0$ on $y = x$, $p \approx 0.75$ at $(1.0, 0.5)$.

2.2 Boundary discretisation

As mentioned in Section 1.5.3, one of the main advantages of the finite element method for free-surface problems is that the free surface can be represented directly using the edges of elements. If the edges are linear then the boundary location will be asymptotically $O(h^2)$ accurate. Allowing the use of piecewise-quadratic edges, as described above, potentially allows $O(h^3)$ accuracy to be achieved for the location of the boundary.

If one restricts the quadratic Lagrange basis (2.6–2.11) to the lower edge of the master element shown in Fig. 2.2, by setting $\eta = 0$, one obtains the following basis for the edge:

$$\begin{aligned} q_1 &= 2\xi^2 - 3\xi + 1, \\ q_2 &= 2\xi^2 - \xi, \\ q_4 &= -4\xi^2 + 4\xi. \end{aligned} \tag{2.14}$$

The edge is thus given by

$$\mathbf{s}(\xi) = \sum_{j=1,4,2} \mathbf{s}_j(\xi) q_j, \tag{2.15}$$

where $0 \leq \xi \leq 1$, and $\mathbf{s}_1$, $\mathbf{s}_4$ and $\mathbf{s}_2$ are the positions of nodes 1,4 and 2. Clearly, the curve $\mathbf{s}(\xi)$ passes through the nodes $\mathbf{s}_1$, $\mathbf{s}_4$, and $\mathbf{s}_2$. Less obvious is the fact that the tangent to $\mathbf{s}$ at $\mathbf{s}_4$ is given by $\mathbf{s}_2 - \mathbf{s}_1$, i.e. the tangent at an edge node is parallel to the chord drawn between the edge's endpoints. In principle, by defining a global arc-length parameter s, one can represent a complete free surface using a single piecewise-quadratic parameterised curve $\mathbf{s} = \mathbf{s}(s)$ and thus avoid the problems that arise when non-parametric schemes are employed. Fortunately, the construction of the global arc-length parameter is not normally required, since all the boundary integrals required by the finite element method can be evaluated using an element's local-coordinate system.

As mentioned in Section 2.1.2, the displacement, δr, of each free-surface edge node from its corresponding linear-edge midpoint, must be bounded so that

$$\delta r \leq C h^2, \tag{2.16}$$

where C is a constant and where h is the length of the chord joining the edge's endpoints. Clearly this imposes restrictions on the set of piecewise-quadratic free surfaces that are representable for a given value of C, and it suggests that difficulties might arise as a mesh is refined. In particular the question arises as to whether, for a given boundary, one can find a constant C that will bound δr uniformly as $h \to 0$. Consider a curve $\mathcal{S}$. If $\mathcal{S}$ is smooth, then on a sufficiently small scale it will have approximately constant curvature and may thus be accurately modelled by a circular arc of radius R. Let $\mathcal{S}$ initially be discretised into a number of edges such that (2.16) is satisfied by each edge for some fixed value of C. Fig. 2.7 shows a single edge together with the chord joining its ends. The distance δr is given by

$$\delta r = R - \sqrt{R^2 - \left(\frac{h}{2}\right)^2} = R - R\sqrt{1 - \left(\frac{h}{2R}\right)^2}. \tag{2.17}$$

Expanding as a Taylor series about $\frac{h}{2R} = 0$ one may obtain the following asymptotic formula

$$\delta r = R\left(\frac{1}{2}\left(\frac{h}{2R}\right)^2 + \frac{1}{8}\left(\frac{h}{2R}\right)^4 + \frac{1}{16}\left(\frac{h}{2R}\right)^6 + O(h^8)\right), \tag{2.18}$$

in which the leading term in h is $O(h^2)$. Thus, so long as (2.16) is satisfied by an

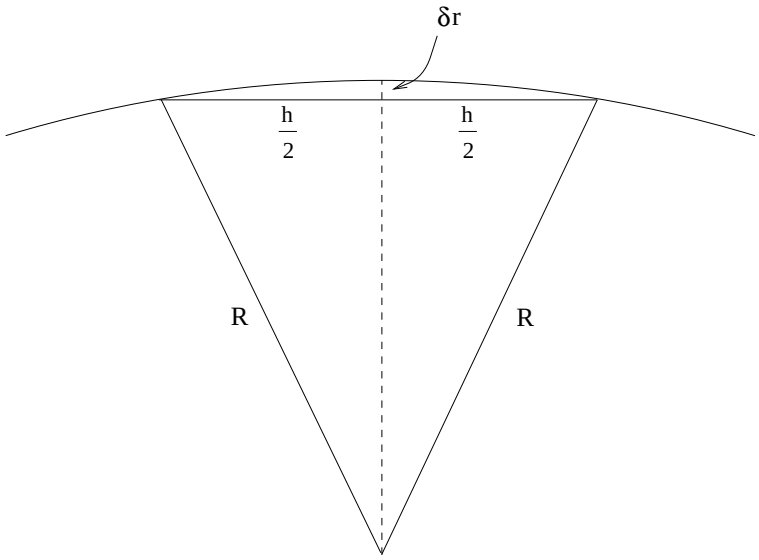


Figure 2.7: Using an isoparametric element to represent a circular boundary: computing the edge displacement δr.

initial discretisation of $\mathcal{S}$, a regular subdivision of the discretisation will also satisfy (2.16). Consequently, as $h \to 0$, the full asymptotic rate of convergence in the solution is potentially attainable.

2.2.1 Tangent continuity

The next issue to consider is that of the degree of continuity of the piecewise boundary representation at boundary vertices. Along boundary edges the interpolating curve is smooth since it is a polynomial. At free-surface vertices, however, there is no way to guarantee that the two tangents, $\mathbf{t}_1$ and $\mathbf{t}_2$, corresponding to the two adjacent elements will remain parallel, even if they are so initially. Fig. 2.8 illustrates a situation that might arise. In this case the angle between the two tangents is $\phi_1 + \phi_2$. In practice one would hope that large jumps in the tangent would not arise, and experience suggests that, when a free surface is advected using a stable scheme, they do not. If a unit circle is discretised as 32 equally-sized edges (a fairly coarse mesh) by placing all nodes on the boundary, then the initial angles between tangents at vertices are of the order of 0.05°. The angles that arise in the course of a typical simulation are generally comparable. While such discontinuities are barely visible to the naked eye, they potentially cause problems when free-surface stress boundary conditions are imposed, since at such vertices the free-surface normal is not uniquely defined. As a result tangential stress errors may arise, an issue that is discussed in Section 4.4.2.

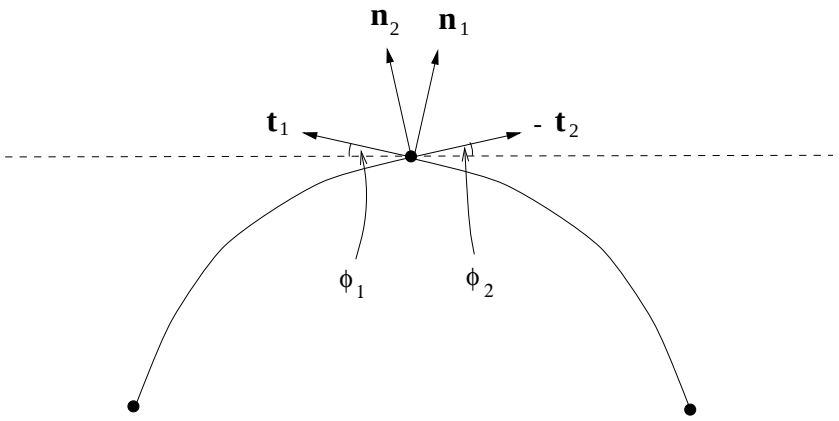


Figure 2.8: Tangents and normals at a vertex of a piecewise-quadratic boundary representation.

2.2.2 Computation of boundary conditions

The specification of stress boundary conditions, in general, requires the computation of both normal and tangential components of the stress. However, at an interface between a liquid and a gas (the latter of negligible viscosity and density) the appropriate stress boundary condition has only a normal component i.e. the tangential stress is zero. The normal component of the stress is generally made up of two contributions. The first is due to the pressure of the gas acting on the free surface. The second is due to a phenomenon that is not described by the Navier-Stokes equations, and which is associated only with interfaces between immiscible fluids. This is known as the *pressure defect* [105], the magnitude of which, $\hat{p}$, is given at a point on a free surface by

$$\hat{p} = 2\gamma H = \gamma\left(k_1 + k_2\right) = \gamma\left(\frac{1}{R_1} + \frac{1}{R_2}\right), \tag{2.19}$$

where H is the mean curvature of the surface, k_1 and k_2 are the principal curvatures and R_1 and R_2 are the corresponding radii of curvature. For a two-dimensional problem one simply sets $k_2 = 0$. The parameter γ, the surface tension or surface energy, is typically assumed to be a constant for any given pair of immiscible fluids. The computation of the curvature of the free surface is perhaps the most difficult problem faced when dealing with surface-tension-driven flows. Curvature is defined in two dimensions [74] for a free surface given in the parametric form $(x(s), y(s))$ by

$$k(s) = \frac{x_s y_{ss} - y_s x_{ss}}{\left({x_s}^2 + {y_s}^2\right)^{\frac{3}{2}}}, \tag{2.20}$$

while the outward free-surface normal is given by

$$\mathbf{n}(s) = \frac{1}{\sqrt{{x_s}^2 + {y_s}^2}}(y_s, -x_s)^T. \tag{2.21}$$

The approach to the computation of boundary conditions adopted by Mattheij and van de Vorst [113, 114] (which was investigated as an alternative to the techniques employed here) involves the estimation of x_s, x_{ss} etc. at free-surface nodes using finite-difference formulae involving the locations of neighbouring nodes. For a free surface defined by the ordered set of nodes $\{\mathbf{s}_i = (x_i, y_i) : i = 1, \ldots, N_B\}$ they employ the following estimates for the derivatives[3] at node $\mathbf{s}_i$:

$$x_s = \frac{1}{12h}(x_{i-2} - 8x_{i-1} + 8x_{i+1} - x_{i+2}) + O(h^4), \tag{2.22}$$

$$y_s = \frac{1}{12h}(y_{i-2} - 8y_{i-1} + 8y_{i+1} - y_{i+2}) + O(h^4), \tag{2.23}$$

$$x_{ss} = \frac{1}{12h^2}(-x_{i-2} + 16x_{i-1} - 30x_i + 16x_{i+1} - x_{i+2}) + O(h^4), \tag{2.24}$$

$$y_{ss} = \frac{1}{12h^2}(-y_{i-2} + 16y_{i-1} - 30y_i + 16y_{i+1} - y_{i+2}) + O(h^4). \tag{2.25}$$

The formulae (2.22–2.25) are all formally fourth-order accurate, and may be derived by fitting a fourth-degree Lagrange interpolation polynomial through a node and its four equally spaced neighbours.

It is well known that numerical differentiation in finite-precision arithmetic becomes increasingly ill-conditioned as the distance between sampling points tends towards zero [21, 40]. This results from the loss of accuracy when quantities of similar size are subtracted. Indeed it is easy to show that for a given problem there is an optimum nodal spacing h^* that minimises the total error due to the combination of truncation and rounding error. In practice experience shows that the difference formulae (2.22–2.25) become increasingly inaccurate as the distances between nodes are reduced in an attempt to make the computations more accurate. Thus, while a free surface may appear smooth to the eye, its numerically computed first derivatives may display considerable errors, and its second and higher derivatives consist entirely of numerical noise. In finite-precision arithmetic there thus appears to be a limit to the accuracy to which derivatives can be computed numerically, even when the shape of the free surface is known exactly.

While the above approach may be adequate when the free surface is smooth, locally non-oscillatory, and the nodes are not too close together; when the free surface

[3]The formulas given here are actually taken from [16].

is non-smooth, and potentially contaminated with numerical noise, as will often be the case in a discrete simulation, high-order interpolation polynomials can exhibit marked oscillations and may thus result in highly inaccurate estimates of the free surface's curvature. Mattheij and van de Vorst appear to avoid this problem by periodically redistributing free-surface nodes — a process that implicitly involves some local smoothing of the free-surface shape, removing the higher-frequency components of noise that can make higher-order difference formulae less accurate than lower-order ones.

While van de Vorst [113] suggests that the scheme (2.20—2.25) for computing k is third-order accurate, he does not attempt to demonstrate this numerically. This author's experience when investigating such methods suggests that van de Vorst's scheme may in practice be no better than first-order accurate, particularly in the limit as $h \to 0$. The use of free-surface smoothing to enhance the accuracy of computed boundary conditions, whether explicitly performed, or 'hidden' as part of a boundary node redistribution operation, may be criticised on the grounds that it effectively introduces non-physical forces into the problem. Thus, while smoothing allows the use of higher-order difference formulae with higher formal orders of accuracy, it is not clear that one actually gains any accuracy in practice. A further criticism arises from the fact that the five-point difference stencils make use of non-local information when applied to quadratic elements. That is, the boundary conditions for an element depend on information from adjoining elements, as well as the element itself, and may thus potentially introduce non-physical effects.

In the current work numerical smoothing is not employed. For surface-tension-driven flows this appears to be satisfactory since surface tension acts locally so as to rapidly smooth out any small-scale oscillations that arise in the free-surface shape. Thus, provided the time-integration schemes employed are stable, stability of the free surface should follow as a consequence of the underlying physics. For non-surface-tension-dominated flows no such mechanism is present and the use of some form of smoothing appears to be an essential consequence of the use of discrete schemes.

Boundary conditions are computed directly from the current free-surface representation using the approach to be described shortly. Since the free surface is at best $O(h^3)$ accurate, the best estimates of x_s and x_{ss} etc. that can be computed are only $O(h^2)$ and $O(h)$ accurate respectively. The approach adopted here makes use of the identity

$$k\mathbf{n} = \frac{\partial \mathbf{t}}{\partial s}, \tag{2.26}$$

derived from the Frenet formulae for a unit-speed curve [74]. As will be shown in Chapter 3, in two dimensions the finite element formulation of the free-surface stress boundary condition involves integrals of the form

$$\int_A^B \gamma q_i k \mathbf{n}\, ds = \int_A^B \gamma q_i \frac{\partial \mathbf{t}}{\partial s}\, ds, \tag{2.27}$$

where q_i is the restriction of a quadratic basis function to the free surface, and A and B are the limits of integration for a given edge. Integrating by parts in the manner suggested by Ruschak [91] one obtains

$$\int_A^B \gamma q_i \frac{\partial \mathbf{t}}{\partial s}\, ds = \gamma q_i [\mathbf{t}]_A^B - \gamma \int_A^B \mathbf{t} \frac{\partial q_i}{\partial s}\, ds, \tag{2.28}$$

where

$$\mathbf{t} = \sum_{k=1,4,2} \mathbf{s}_k \frac{\partial q_k}{\partial s}, \tag{2.29}$$

and the $\mathbf{s}_k$ are the positions of the three nodes comprising a free-surface edge. Note that the integral on the right-hand side of (2.28) involves only the first derivatives of the basis functions, q_i, which for quadratic elements are piecewise-linear functions of s. Thus, while one gains no formal accuracy by using this approach, the need to form second derivatives numerically is avoided and, at the same time, one can conveniently allow for any discontinuities in the tangent at free-surface vertices, through the jump term on the right-hand side of (2.28). Note that if piecewise linear free-surface edges were to be employed then the only contributions to the curvature would result from the discontinuities at vertices and one would expect the error in the boundary conditions to be $O(1)$, i.e. convergence would not be expected as the mesh is refined.

2.2.3 Equidistribution of curvature

In practice a typical free surface will not have constant curvature; indeed the curvature may vary by several orders of magnitude or even be of different sign on different parts of the free surface. Consequently, the free-surface normal stress and its gradient may vary considerably over the free surface. In such circumstances the use of a uniform mesh, selected so as to give a certain level of accuracy in the boundary conditions and thus the solution, will not be an efficient use of resources. The alternative, discretisation of the free surface with respect to an appropriate *error indicator*, thus appears attractive in such situations. For surface-tension-driven flows,

the driving forces are greatest in regions of high curvature and it is appropriate to have more free-surface nodes in these regions, so as to more accurately represent the free surface's shape and thus minimise the errors in the boundary conditions.

Consider the Taylor expansion for a smooth free surface $\mathcal{S}(s)$, about a point on the free surface, designated $s = 0$ for convenience, i.e.

$$\mathcal{S}(s) = \mathcal{S}(0) + \frac{\partial \mathcal{S}}{\partial s}s + \frac{\partial^2 \mathcal{S}}{\partial s^2}\frac{s^2}{2} + \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{s^3}{6} + O(s^4). \tag{2.30}$$

If one attempts to represent such a free surface using quadratic polynomials in s then it is clear that the magnitude of the coefficient in the leading term of the local-truncation error will be proportional to

$$\frac{\partial^3 \mathcal{S}}{\partial s^3}, \tag{2.31}$$

which in turn is proportional to

$$\frac{\partial k}{\partial s} \tag{2.32}$$

the rate of change of curvature with respect to arc length.

If a known free surface is discretised using piecewise-quadratic elements, at a typical point on the free surface the discrete representation will be in error by an $O(h^3)$ quantity, where h is free-surface edge length, provided that the magnitude of (2.31) is bounded on the free surface. One way of proceeding would be to position boundary vertices $\{\mathbf{s}_i : 1 = 1, \ldots, N_B\}$, so that for each element

$$\int_0^h \left| \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{s^2}{2} \right| ds \leq \epsilon, \tag{2.33}$$

for a given choice of ϵ. If one assumes that the rate of change of curvature with respect to arc length is locally approximately constant then

$$\int_0^h \left| \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{s^2}{2} \right| ds = \left| \frac{\partial^3 \mathcal{S}}{\partial s^3} \right| \int_0^h \frac{s^2}{2}\, ds = \left| \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{h^3}{6} \right| \leq \epsilon, \tag{2.34}$$

giving

$$\left| \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{s^3}{6} \right| \leq \left| \frac{\partial^3 \mathcal{S}}{\partial s^3}\frac{h^3}{6} \right| \leq \epsilon, \tag{2.35}$$

and thus bounding the local truncation error uniformly. This is however hard to achieve in practice, due to the difficulty of computing (2.31) numerically with sufficient accuracy. Thus, while it may be possible to find such an equidistribution for

an analytically defined curve, attempting to maintain such an equidistribution for a piecewise-quadratic representation would be difficult.

The alternative (less accurate) approach adopted here involves the discretisation of the free-surface boundary so as to approximately equidistribute curvature between elements. To understand why this is effective consider a Taylor expansion for the curvature $k(s)$ about a point $s = 0$ on a free surface, i.e.

$$k(s) = k(0) + \frac{\partial k}{\partial s}(0)s + O(s^2). \tag{2.36}$$

If a curve is approximated using a piecewise-quadratic interpolant then the leading term in the error estimate for the curvature will be of $O(h)$. Thus the interpolated curvature will have the same order of accuracy as would be the case if a piecewise-constant representation were employed for $k(s)$. In such circumstances only the constant term in the expression (2.36) can be represented and consequently the second term will be the leading term in the local truncation error for the curvature. It thus appears reasonable to employ the second term on the right-hand side of (2.36) as an error indicator when creating or updating a boundary mesh.

In order to estimate the accuracy of the boundary conditions that will result from a given distribution of free-surface nodes $k(s)$ must be integrated along a free-surface edge. If the finite element weighting function q_i and the normal $\mathbf{n}$ are ignored (which is reasonable since $|q_i\mathbf{n}|$ is always less than one) and $\gamma = 1$ is assumed, then the following expression for the magnitude of the discrete boundary condition that will be applied at a free-surface node may be obtained from (2.36) and (2.27)

$$\int_{\partial\Omega_i} k(s)\,ds = \int_{\partial\Omega_i} k(0)\,ds + \int_{\partial\Omega_i} \frac{\partial k}{\partial s}(0)s\;ds + O(s^2 h_i) \tag{2.37}$$

$$= k(0)h_i + \frac{\partial k}{\partial s}(0)\frac{{h_i}^2}{2} + O({h_i}^3), \tag{2.38}$$

where

$$h_i = \int_{\partial\Omega_i} d\,s \tag{2.39}$$

is the length of edge $\partial\Omega_i$ and again we assume $\frac{\partial k}{\partial s}$ is approximately constant. In the following it will be assumed, for simplicity, that $k(s) \geq 0$ on $\partial\Omega$. Given that a discretisation satisfies the equidistribution condition

$$\int_{\partial\Omega_i} k(s)\,ds = k_i h_i \leq \epsilon \tag{2.40}$$

for each edge $\partial\Omega_i$, where k_i is the mean curvature of the edge, an estimate of (2.32) at vertex i may be obtained using the following finite-difference approximation

$$\begin{aligned}\left|\frac{\partial k}{\partial s}\right|_i &\simeq \frac{2\,|k_i - k_{i-1}|}{h_i + h_{i-1}} \\ &\leq \frac{2(|k_i| + |k_{i-1}|)}{h_i + h_{i-1}} \\ &\leq \frac{2\left(\frac{\epsilon}{h_i} + \frac{\epsilon}{h_{i-1}}\right)}{h_i + h_{i-1}} \\ &= \frac{2\epsilon}{h_i h_{i-1}}. \end{aligned} \tag{2.41}$$

Thus the following bound for the second term in (2.38) is obtained

$$\left|\frac{\partial k}{\partial s}(0)\right|_i \frac{{h_i}^2}{2} \leq \epsilon \frac{h_i}{h_{i-1}}. \tag{2.42}$$

Finally, if it is assumed that

$$h_i \leq \rho h_{i-1} \tag{2.43}$$

for some constant ρ, the following bound for the magnitude of the error in the discrete boundary condition at a free-surface node is obtained

$$\left|\frac{\partial k}{\partial s}(0)\right| \frac{{h_i}^2}{2} \leq 2\rho\epsilon, \tag{2.44}$$

the additional factor of two appearing since a discrete boundary condition involves potentially two free-surface edges. One thus arrives at the following somewhat disappointing conclusion: *refining a piecewise-quadratic boundary mesh by halving ϵ, and thus doubling the number of free-surface nodes, will approximately halve the error in the discrete boundary condition imposed at any node common to both meshes, and consequently halve the error in the solution. The scheme is thus $O(h)$ accurate.*

It is interesting to note that Mattheij and van de Vorst [113, 114] employ equidistribution with respect to free-surface curvature rather than the more sophisticated schemes that would be required to truly reflect the formal accuracy of their higher-order difference schemes.

2.2.4 Initial boundary discretisation

Having discussed the theoretical issues relating to the piecewise-quadratic representations of free surfaces it is now appropriate to turn to the algorithmic details of the free-surface discretisation scheme employed. One important question that arises is whether one can reasonably expect to be able to model the evolution of a free surface that initially contains a sharp corner i.e. a large discontinuity in its tangent. Consider the corner of a square; away from the corner the edges are straight, i.e. have zero curvature, but at the actual corner itself the curvature is undefined. While the finite element method is ideally suited to dealing with domains with corners under normal circumstances, if one attempts to model a free-surface corner using a pair of elements, as depicted in Fig. 2.9(a) (being careful to 'triangulate into the corner'), one observes that after only a small number of time steps a configuration like that shown in Fig. 2.9(b) arises. The finite element method succeeds in finding a plausible evolution of the free surface, but the unfortunate bulging of the two elements adjacent to the corner, due to the localisation of the driving force near to the corner, rapidly leads to the isoparametric discretisation becoming singular.

If the mesh is automatically refined as soon as the bulging starts to occur, it may be possible to continue the simulation, but typically such refinement rapidly leads to elements that are extremely small — necessitating the use of very small time steps if an explicit time-integration scheme is employed. Thus explicit methods are unlikely to be cost effective in such circumstances and implicit methods are necessary in order to deal with the stiff systems of equations that arise. Furthermore, the considerable effort put into modelling the regions of high curvature that develop adjacent to the corners is unjustified since the shape of the free surface arises from an initially coarse mesh and is thus inaccurate. The simplest remedy, and the one employed in the current work, is to round-off the corners in such situations. This does not appear to be too unreasonable a perturbation of the original problem, given the natural tendency for corners to evolve into regions of locally high curvature. While this allows for better error control in the vicinity of corners, it does not remove the problem of the stiffness of the systems of equations involved.

2.2.5 Boundary discretisation constraints

The first step in meshing a domain Ω is the discretisation of its boundary $\partial\Omega$. For simplicity the discussion here is restricted to cases in which the free-surface boundary can be represented by a single closed curve parameterised by arc-length s. In the

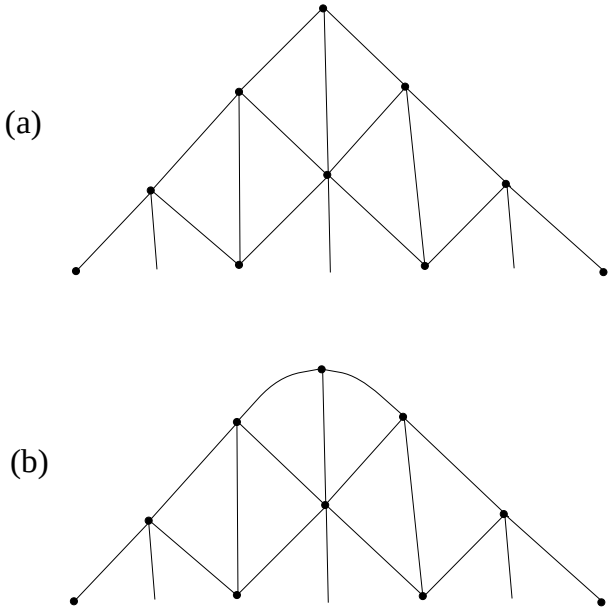


Figure 2.9: Mesh failure at a free-surface corner.

present work an initial boundary discretisation is chosen so that the curvature $k(s)$ is equidistributed, i.e.

$$\int_{\partial\Omega_i} k ds \leq k_{tol} \tag{2.45}$$

for each free-surface edge $\partial\Omega_i$, where k_{tol} is a prescribed parameter. It is further required that edge length be constrained so that

$$\int_{\partial\Omega_i} ds \leq h_{max}, \tag{2.46}$$

for all i, where h_{max} is a prescribed parameter. This constraint imposes an upper limit on the size of element generated. Finally, the following constraints on boundary edge length are imposed:

$$\int_{\partial\Omega_i} ds \leq \alpha \int_{\partial\Omega_{i+1}} ds, \tag{2.47}$$

$$\int_{\partial\Omega_i} ds \leq \alpha \int_{\partial\Omega_{i-1}} ds, \tag{2.48}$$

where α is a *mesh smoothness* parameter, chosen to prevent the ratio of lengths of adjacent boundary edges being too large. Experience suggests that a value of $\alpha = 1.5$ is satisfactory. Edge nodes are located so that they lie equidistant from their neighbouring vertices. Note the correspondence of k_{tol} and α here, with ϵ and ρ in (2.44). The above mesh-quality constraints are further reflected in the algorithms

described in Section 2.4 that are employed to maintain mesh quality as the free surface evolves.

2.3 Interior mesh generation

Once a domain's boundary has been initially discretised, and thereafter whenever it is modified, a new interior mesh must be generated. This can be achieved conveniently by the use of one of a number of widely available automatic mesh generators such as Triangle [96], GRUMMP [72] and GEOMPACK [55][4]. The package employed here is Jonathan Shewchuk's 2-D Delaunay mesh generator Triangle [96].

Based upon Ruppert's Delaunay refinement algorithm [90], Triangle will 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. Ruppert's scheme has the important property that it is guaranteed to produce a mesh with no small internal angles, and thus no elements with large aspect ratio[5]. Specifically, Triangle is guaranteed to produce a mesh with no internal angle less than approximately 20.7°. This property is however compromised if, as here, Triangle is employed to generate a boundary constrained mesh, i.e. Triangle is not allowed to split the original boundary edges. To prevent this becoming a serious problem, particularly where boundary discretisations with large variations in edge length are involved, here the interior mesh is *graded* so that edge length does not differ too greatly between neighbouring elements. This is achieved by associating with each element i of an initial coarse mesh a length l_i given by

$$l_i = \min\left(h_{max}, \min_j\left(h_j + \frac{1}{2}\left|\mathbf{m}_j - \mathbf{c}_i\right|\right)\right), \tag{2.49}$$

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

$$a_i = \frac{\sqrt{3}}{2} {l_i}^2. \tag{2.50}$$

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 purely motivated by the need to ensure

[4]See http://www.andrew.cmu.edu/user/sowen/survey/index.html for an excellent short introduction to the various methods of mesh generation commonly employed in 2- and 3-D.

[5]Longest edge divided by shortest altitude.

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 in the solution.

Fig. 2.10 illustrates the effect the parameter k_{tol} has on the mesh produced by

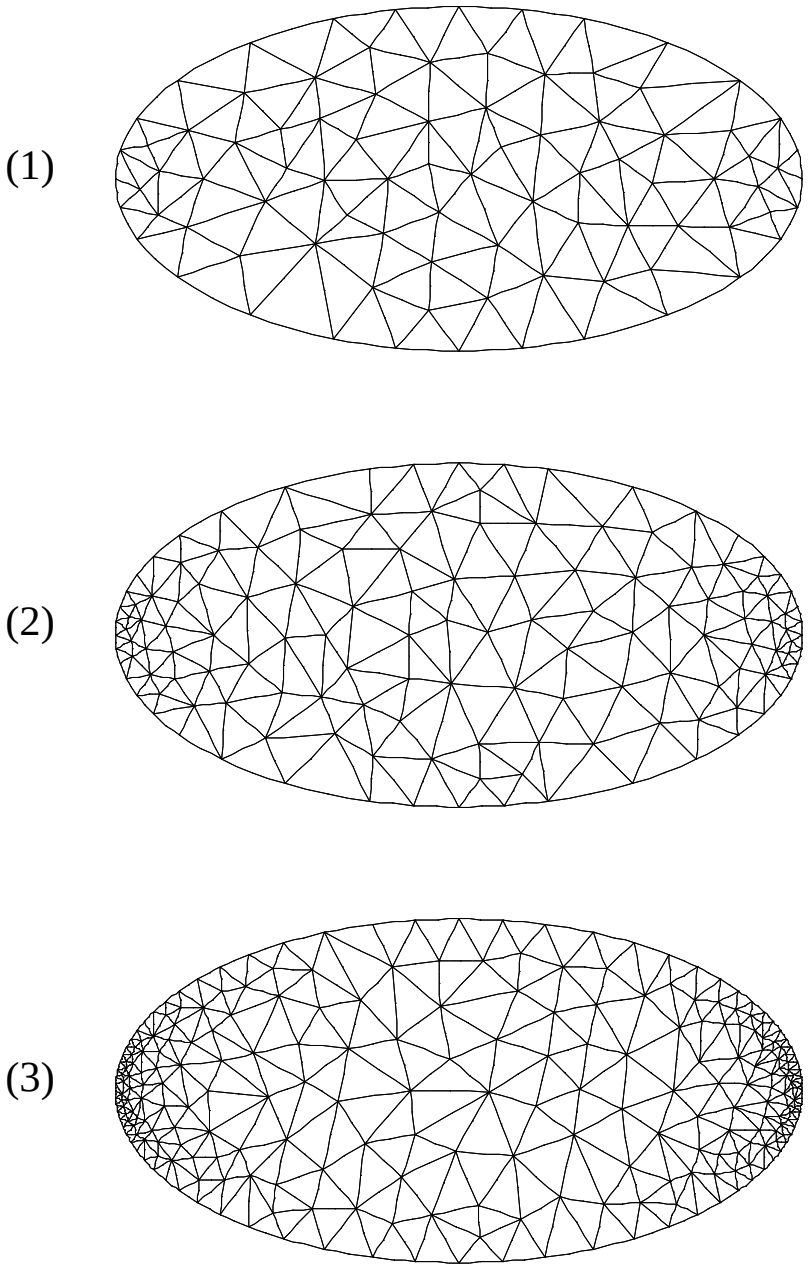


Figure 2.10: Meshes for an ellipse.

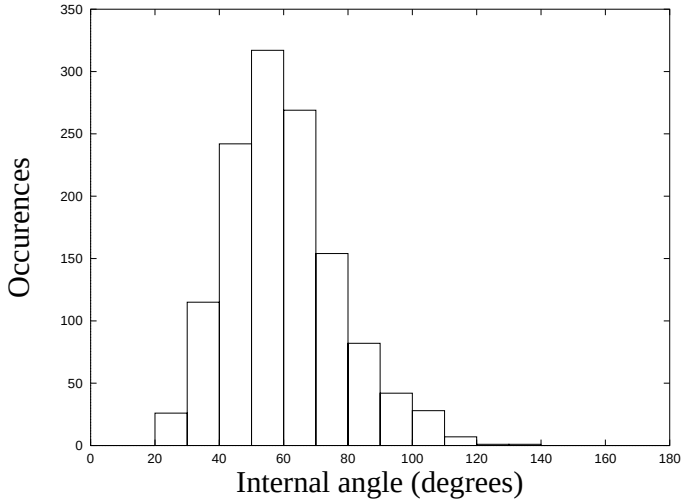


Figure 2.11: Distribution of internal angles for graded-ellipse mesh 2.

Mesh	k_{tol}	h_{max}	Elements	Unknowns
1	0.4	0.5	144	721
2	0.2	0.5	228	1139
3	0.1	0.5	428	2129

Table 2.2: Elliptical cylinder problem: mesh statistics.

Triangle when grading is performed. The figure shows three meshes for an ellipse, generated by holding h_{max} fixed while varying k_{tol}. Table 2.2 summarises the mesh statistics for the three meshes, while Fig. 2.11 shows the distribution of internal element angles for mesh 2. The minimum and the maximum internal angles in mesh 2 are 20.54° and 130.84° respectively, while the maximum element aspect ratio is 4.37. As Fig. 2.11 shows, internal angles are clustered around 60°, with the distribution being skewed towards larger angles. Thus while the theoretical Delaunay minimum angle is not quite being attained, presumably because a boundary-constrained triangulation is requested, the resulting mesh is perfectly satisfactory.

One criticism of Triangle is that it occasionally produces meshes that have unnecessary elements, as illustrated in Fig. 2.12(a). In the absence of any *a priori* knowledge of the solution expected, mesh (b) would generally be preferred, since it is likely to be just as accurate, but involves fewer nodes and thus lower computational expense. Such anomalous configurations can easily be removed, requiring a search followed by local mesh repair. In the current work this is not attempted, primarily for reasons of simplicity but also because the potential gains in efficiency are small. A second criticism of Triangle is that the meshes are 'noisy' and that

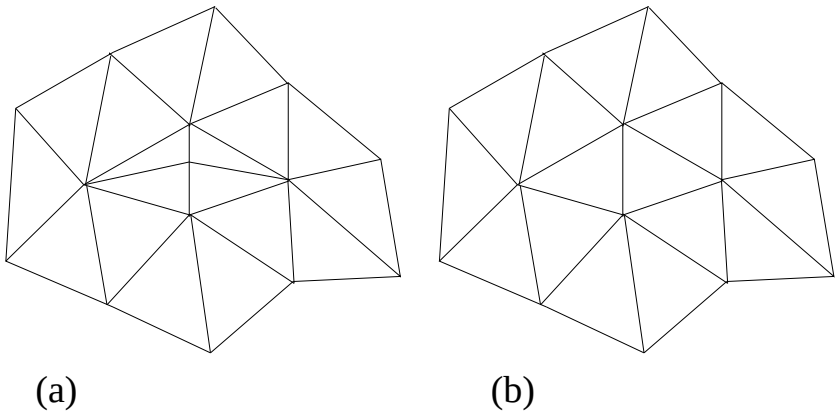


Figure 2.12: Unnecessary elements: (a) a non-optimal mesh (b) a more efficient mesh.

a greater regularity of node spacing might be expected to improve the overall accuracy of the interpolated solution. Laplacian smoothing, as described in Section 2.5.2, might usefully be employed to achieve this.

The above discussion of mesh quality does not take into account the important effects that the interaction between mesh geometry and local solution gradients have on solution accuracy. More sophisticated mesh quality indicators would take into account whatever information was available about the nature of the solution in the vicinity of an element, typically by employing the solution from previous time steps.

2.4 Remeshing

In general, as a free-surface mesh evolves with time, there will be regions of the free surface in which the curvature is increasing or decreasing. In other regions free-surface edges may be increasing or decreasing in length. Clearly, if the quality of the free-surface discretisation is to be maintained, refinement and derefinement must be performed in such regions, and will involve the insertion and removal of nodes.

Whenever nodes are inserted into or removed from a free-surface representation the regeneration of the interior mesh is necessary. For the relatively small meshes considered here, the cost of performing such a regeneration is sufficiently low that there is no need to consider more involved schemes that perform local mesh regeneration. Since remeshing, locally or globally, typically requires a full restart of any time-integration scheme involved, at considerable expense, and for the Navier-Stokes equations (but not the Stokes equations) requires interpolation of the velocity so-

lution, again at considerable expense and with potential loss of accuracy, it makes sense to aim to perform remeshing as infrequently as possible. In the interests of efficiency it is thus normally appropriate to postpone coarsening of the free-surface mesh until the next occasion a full mesh regeneration is necessary. Mesh refinement is, however, necessarily more urgent. This suggests the following strategy: *refine aggressively, coarsen cautiously.* In the scheme described here interior mesh regeneration is carried out when dictated by one or more of the following criteria:

1. a free-surface edge is too long, i.e.

$$h_i > h_{max}, \tag{2.51}$$

   where h_i is the length of edge i;

2. the integral of the modulus of the curvature along a free-surface edge is too great, i.e.

$$\int_{\partial\Omega_i} |k|\, ds > k_{tol}; \tag{2.52}$$

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

$$\theta_{min} < \varphi, \tag{2.53}$$

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

4. a free-surface edge node is located too far from the midpoint of the chord joining the ends of the edge, i.e.

$$|\mathbf{e}_i - \mathbf{s}_i| > \beta\, h_i \tag{2.54}$$

   or

$$|\mathbf{e}_i - \mathbf{s}_{i+1}| > \beta\, h_i, \tag{2.55}$$

   where $\mathbf{e}_i$ is the position of edge node i, $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$ are the vertices associated with the edge and β is a parameter chosen to bound the displacement of a free-surface edge node from its edge midpoint.

Criteria (2.53), (2.54) and (2.55) reflect constraints that must be applied if the optimum asymptotic rate of convergence of the solution is to be achieved [103]. In particular (2.53) reflects the need to bound the maximum interior angle in each

element away from 180°. In practice one instead bounds the minimum interior angle away from zero. For triangular elements (2.53) is equivalent to the bound

$$\theta_{max} < 180^\circ - 2\varphi. \tag{2.56}$$

In practice a value of $\varphi = 10^\circ$ is found to be satisfactory. The avoidance of small angles in the initial mesh is particularly important when the interior nodes of a mesh are in motion due to the application of the mesh-update methods described in Section 2.5. Since it is in practice impossible to predict which angles will increase and which will decrease in such circumstances, the only way of avoiding having to remesh too often is to employ meshes in which the minimum angle is maximised. For free-surface problems, bounding the minimum angle away from zero also helps to avoid difficulties that can arise when an element with large aspect ratio occurs with one of its long sides forming part of a curved free surface.

Criteria (2.54) and (2.55) trigger a mesh regeneration whenever a free-surface edge node is found to be displaced too far from the midpoint of the chord joining its ends. 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. Finally, since in the current scheme only the need for refinement can trigger a mesh regeneration, it is necessary to place an upper limit on the number of time steps that are attempted before a full mesh regeneration occurs, so as to allow derefinement to occur. In practice a limit of 50 time steps is often appropriate.

Once the decision has been taken to remesh the domain, the opportunity to update the free-surface representation arises. This involves three distinct stages: refinement, derefinement and adjustment. The first stage, refinement, involves the insertion of new nodes into the free-surface representation. The second stage, derefinement, involves the removal of surplus nodes. Finally, any necessary adjustments of edges are performed. These processes are described below, and involve constraints related to but not identical with those employed in Section 2.2.5.

2.4.1 Boundary refinement

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

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

$$h_i > \delta\, h_{max}; \tag{2.57}$$

where the constant $\delta = 0.9$ is included so that edges are split aggressively, minimising the risk that a forced remesh will be necessary too soon in the future.

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

$$\int_{\partial\Omega_i} |k|\, ds > \delta\, k_{tol}, \tag{2.58}$$

where again the constant $\delta = 0.9$ is included so that edges are split aggressively, so as to minimise the risk that a forced remesh will occur too soon as a result of (2.52) being violated.

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

$$h_i > \rho\, min(h_{i+1}, h_{i-1}). \tag{2.59}$$

Note that the value employed for ρ in (2.59) is considerably larger than that employed for α in (2.47) and (2.48), when the initial boundary mesh is generated. The value of the constant ρ is chosen as a compromise between maintaining acceptable grading of the mesh's boundary and allowing derefinement to occur unhindered. See Table 2.3 for suggested values of ρ and α.

Once the decision has been taken to split an edge, the problem arises of how to perform the refinement in such a way as to introduce the minimum error into the free-surface representation. Two types of error are of particular concern. First, one would like each splitting operation to preserve domain area. Second, experience shows that it important to ensure that when an edge is split the tangents at the ends of the edge do not change greatly. If the tangents at the edge endpoints are changed to a significant extent then discontinuities in the tangent may be introduced at vertices, resulting in spurious transient motions of the free surface as surface tension acts to smooth the discontinuities.

Figure 2.13 shows an edge AB that is to be split near its midpoint C. Ideally one would like to preserve the tangents at the original end points, and to have the two tangents at the newly inserted vertex parallel i.e.

$$\mathbf{t}_A{}^{(n+1)} = \mathbf{t}_A{}^{(n)} \tag{2.60}$$

$$\mathbf{t}_C{}^{+} = \mathbf{t}_C{}^{-} \tag{2.61}$$

$$\mathbf{t}_B{}^{(n+1)} = \mathbf{t}_B{}^{(n)} \tag{2.62}$$

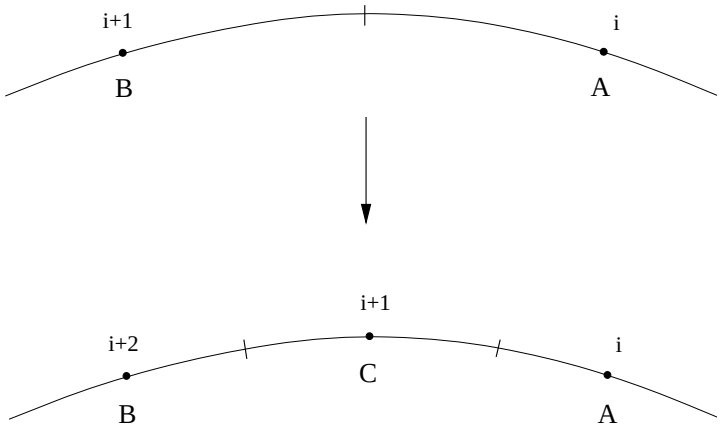


Figure 2.13: Splitting an edge.

where ${\mathbf{t}_A}^{(n)}$ is the tangent at A before refinement, ${\mathbf{t}_A}^{(n+1)}$ the tangent at A after refinement and ${\mathbf{t}_C}^-$ and ${\mathbf{t}_C}^+$ are the tangents, to the edges AC and CB respectively, at C after refinement. Thus, if area must also be conserved, there are four constraints to be satisfied. In principle there are six degrees of freedom with which to satisfy these constraints (the locations of three nodes), and even if it is insisted that the two new edge nodes lie on the bisectors of the chords AC and CB, there are still four degrees of freedom to work with. While such approaches were investigated as part of the current work, in practice the systems of simultaneous equations that arise are often ill-conditioned, and consequently difficult to solve accurately and reliably, resulting in the introduction of error in the free-surface shape. Such ill-conditioning generally becomes worse as the mesh is refined in response to these errors, and thus invariably rapidly leads to the complete breakdown of a simulation.

In the interests of robustness and simplicity it has been found to be necessary to relax the constraints on the tangents and with regard to area conservation, and to adopt the following procedure for splitting an edge:

> take the location of the original edge node to be that of new vertex, and take the two points on the original edge that lie on the perpendicular bisectors of the chords AC and CB to be the locations of the two new edge nodes.

This procedure is the one that has been found to be most satisfactory in practice, even though it is recognised that it not only fails to preserve tangent continuity, but also does not conserve mass exactly.

2.4.2 Boundary derefinement

Two adjacent edges, $\partial\Omega_i$ and $\partial\Omega_{i+1}$, are merged in either of the following circumstances:

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

$$\int_{\partial\Omega_i} |k|\, ds + \int_{\partial\Omega_{i+1}} |k|\, ds < \mu\, k_{tol}, \tag{2.63}$$

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

2. The combined length of the pair of adjacent edges lies below a given tolerance i.e.

$$\int_{\partial\Omega_i} ds + \int_{\partial\Omega_{i+1}} ds < 2\, h_{min}, \tag{2.64}$$

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

The second criterion is included as a pragmatic measure for dealing with difficult free-surface geometries such as those with sharp corners. In normal practice one would set $h_{min} = 0$. Derefinement would appear to be more difficult than refine-

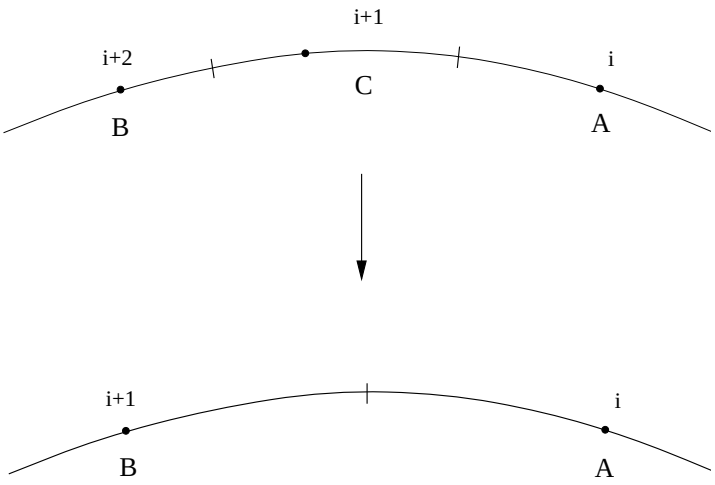


Figure 2.14: Merging two edges.

ment since the aim is to preserve the free surface's current shape but, at the same time, to employ fewer nodes in representing it — which is clearly in general impossible. Fortunately, however, the fact that derefinement is being attempted normally indicates that the current discretisation is more than adequate to represent the free surface locally, and thus that one can afford to sacrifice some accuracy; the exception

being when a minimum edge length is imposed, i.e. $h_{min} > 0$. Attempts to match the existing tangents at A and B and to simultaneously preserve area will clearly not work since there are only two degrees of freedom available. The procedure adopted for merging two edges AC and CB is as follows:

> A point is found that lies on either of the two existing edges, and which is equidistant between A and C. This is taken to be the location of the new edge node.

Figure 2.14 illustrates this process. As with refinement, this approach does not conserve mass exactly, nor does it preserve the tangents at the ends of the edge. It does however appear to work well in practice.

2.4.3 Boundary edge adjustment

Fundamental to the success of the isoparametric free-surface scheme is the requirement that the bound (2.16) be satisfied at all times. Figure 2.15 shows the result of applying an affine transformation to a general element with a single curved side, so as to map the node opposite the curved side to the origin and the two straight edges onto the x' and y' axes, two different locations for the edge node being shown. The midpoint and straight side of the master element are also shown. The need to keep the isoparametric mapping invertible [103] means that an edge node must lie in the shaded region (i.e. $x' > \frac{1}{4}$ and $y' > \frac{1}{4}$) shown in Fig. 2.15. If an edge node is placed outside this region then the Jacobian of the isoparametric transformation will be zero at some points in the element and thus the transformation will be singular. Allowing an edge node to approach the boundary of this region will result in a rapid loss of accuracy, potentially leading to the complete failure of the finite element method.

The motion of an edge node along the perpendicular bisector of the chord defined by the end points of the edge has already been allowed for in that if, for example, it results in the curvature of the edge becoming too large, the edge will automatically be split. Singularity of the isoparametric transformation may thus be avoided by selecting a sufficiently small values of k_{tol}. The possibility of the tangential motion of edge nodes along the free surface must also be allowed for. While the displacement of an edge node in a direction parallel to the chord joining the edge's end points changes the edge's curvature locally, as may be seen in Fig. 2.15, it need not change the total curvature of the edge by very much. Thus such displacements will not, in general, be detected by the refinement criterion (2.58). Since the kinetic bound-

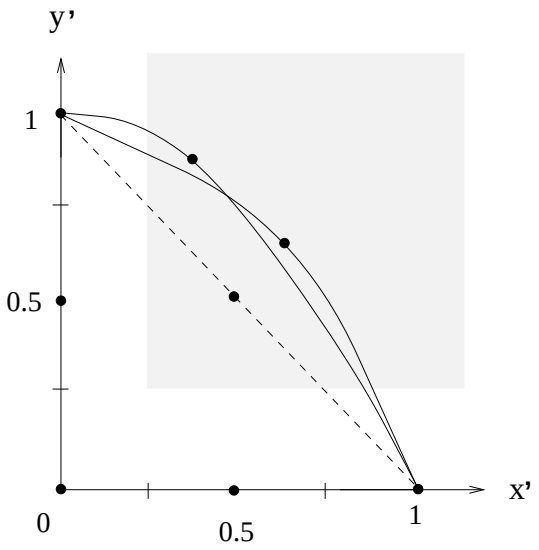


Figure 2.15: Permissible locations for an edge node of an isoparametric element: shaded area. See text for explanation of coordinate system.

ary condition specifies that free-surface nodes need only be moved in the normal direction, one would expect that any tangential motion of the nodes would be minimal and experience bears this out. While there is no reason why small tangential displacements of free-surface edge nodes should not be tolerated, there remains the possibility that the accumulation of such displacements may result in (2.16) being violated. Thus some form of intervention, termed here the *adjustment* of an edge, may occasionally be necessary. Adjustment is carried out if

$$|\mathbf{e}_i - \mathbf{s}_i| > \delta\beta\, h_i \tag{2.65}$$

or

$$|\mathbf{e}_i - \mathbf{s}_{i+1}| > \delta\beta\, h_i, \tag{2.66}$$

the constant $\delta = 0.9$ being included here so that the need to adjust edges does not trigger mesh generation too often. The procedure for adjusting an edge is as follows:

> the intersection of the current edge with the perpendicular bisector of the chord drawn between the edge's end points is selected to be the new location of the edge node.

An alternative approach would be to split an edge in two whenever the edge node's displacement becomes too large. As with free-surface refinement and derefinement no explicit attempt is made to preserve the tangents at end points or to conserve

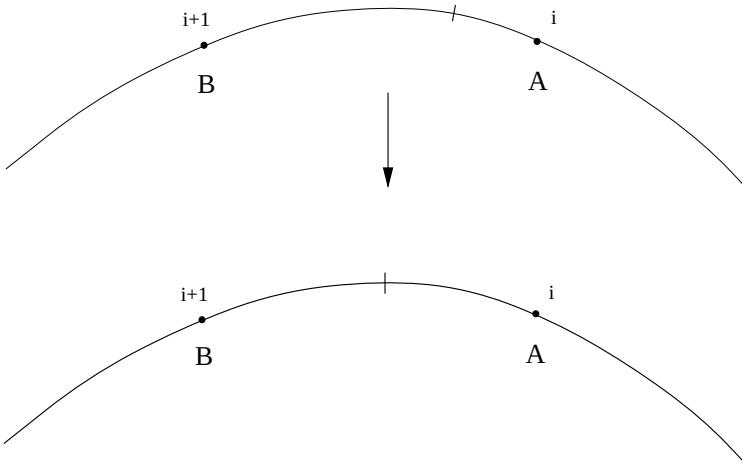


Figure 2.16: Adjusting an edge.

area. In practice, the adjustment of edges is rarely necessary and thus is not a major source of error.

2.4.4 Selection of parameters

In an investigation employing the methods described above, it is intended that the parameters k_{tol} and h_{max} will be varied by the user, so as to control the overall accuracy of the solution. For a surface-tension-driven flow k_{tol} will normally be the key parameter since it dictates the accuracy of the free-surface representation and thus that of the free-surface boundary conditions. Since, as shown in Section 2.2.3, the error in the discrete free-surface boundary conditions is proportional to k_{tol}, halving the overall error will require a doubling of the number of free-surface nodes. If the global mesh parameter h_{max} is also halved then the number of nodes in the resulting mesh will approximately quadruple. Clearly this is an undesirable state of affairs, since, even if an optimal flow-solver is employed[6], the computational cost of any calculation will increase quadratically, while the accuracy will increase only linearly.

The $O(h^3)$ accuracy of the velocity fields computed using the finite element method described in Section 2.1.4 provides a means for overcoming this difficulty. When k_{tol} is reduced by a factor of two h_{max}, which controls the mesh resolution away from the boundary, need only be reduced by a factor of $2^{\frac{1}{3}}$ to achieve a linear

[6]One for which computational cost is proportional to the number of unknowns involved.

rate of convergence in the velocity throughout the domain. Thus one may select

$$h_{max} \propto {k_{tol}}^{\frac{1}{3}} \tag{2.67}$$

and as a consequence the overall cost of such a computation, assuming an optimal solver, will be

$$O({k_{tol}}^{-1}, {h_{max}}^{-2}) = O({k_{tol}}^{-1}, {k_{tol}}^{-\frac{2}{3}}) = O({k_{tol}}^{-1}). \tag{2.68}$$

Thus, asymptotically the computational cost will be proportional to the accuracy of the velocity field obtained.

Constant	Function	Minimum	Value	Maximum
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	0.50	0.70	δ - 0.1
ρ	Boundary mesh smoothness tolerance	2.20	2.50	3.00
φ	Minimum interior angle	5.00	10.00	15.00

Table 2.3: Constants employed in the adaptive mesh generator: Actual values employed together with suggested ranges.

Table 2.3 summarises the values of the various constants employed by the automatic mesh generator described above, together with suggested minimum and maximum values. The values are not particularly critical and may be varied within the ranges shown without compromising the robustness of the method.

2.5 Continuous mesh update

Consider the section of a piecewise-linear free surface depicted in Fig. 2.17(a), with the outward free-surface normal shown. If the free-surface nodes move outward but the interior nodes are held fixed then, as illustrated in Fig. 2.17(b), the elements adjacent to the free surface will rapidly become distorted, leading to a potential loss of accuracy. If on the other hand the free surface is retreating then the situation is even worse, and there is now the danger that free-surface nodes might cross-over into the second layer of elements, causing the mesh to tangle. Clearly in any general purpose scheme the locations of the interior nodes must be updated at each time step in some fashion, if only to allow them to escape a retreating free surface.

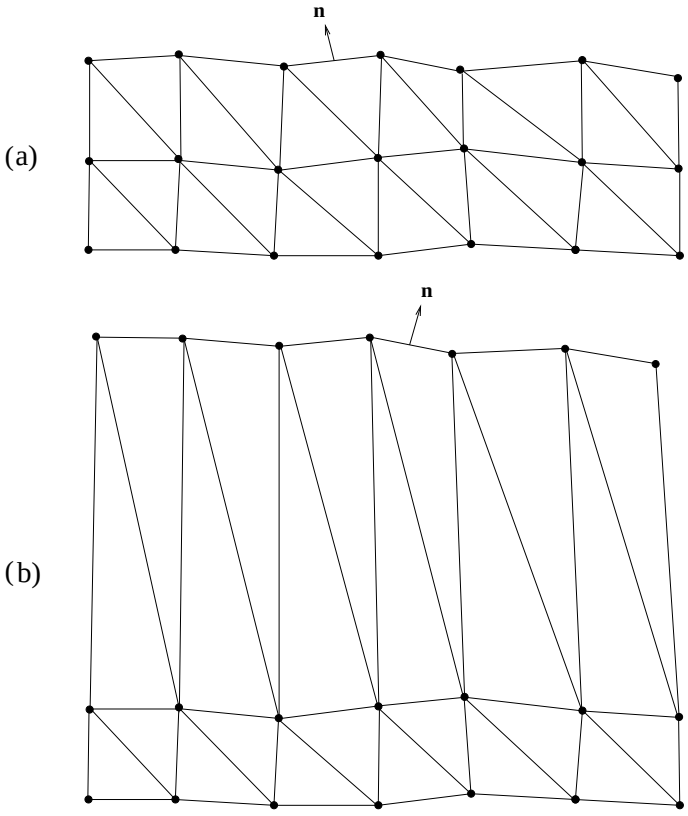


Figure 2.17: Stretching of elements during motion of a free surface, $\mathbf{n}$ = outward free-surface normal: (a) before motion of free surface; (b) after motion of free surface with interior nodes held fixed $\mathbf{n}$ = outward free-surface normal.

Central to the strategy described in Section 2.4 is the idea that full mesh regeneration is performed at the end of a time step only if it is absolutely necessary. At the end of all other time steps the mesh is updated using some form of continuous mapping. To this end two techniques were investigated: a method based upon a global linear-elasticity model and a local Laplacian smoothing method.

2.5.1 Linear-elasticity model

In the first approach considered, the interior mesh is updated at the end of a time step, using an elastic-mesh model based on that proposed by Lynch [66]. This involves solving a linear-elasticity problem for a set of interior-vertex displacements, using the most recent boundary-vertex displacements as boundary conditions. The linear elasticity model takes the form of a Poisson problem

$$\nabla \cdot (\mathbf{C} \nabla \mathbf{x}) = \mathbf{f}, \tag{2.69}$$

where $\mathbf{x}$ is a vector of interior-vertex displacements to be found, $\mathbf{f}$ a vector representing an optional body force and $\mathbf{C}$ a fourth-order elasticity tensor. For simplicity $\mathbf{f}$ was taken to be the zero vector and $\mathbf{C}$ to be the identity tensor. The apparent advantage of Lynch's method results from the understanding that continuous boundary conditions will result in a continuous deformation of the elastic sheet, and thus any mesh embedded in the sheet will itself distort in a continuous manner. Thus tangling of the mesh should not occur.

If an appropriate weak form of (2.69) is discretised using linear elements and the Galerkin method applied, a symmetric positive-definite system of linear-algebraic equations is obtained. This auxiliary system is typically much smaller than the system arising from the main problem, involving approximately 1/9 the number of unknowns. Furthermore it need only be solved approximately. This can be achieved cost-effectively using an iterative method, such as those described in Chapter 3. In practice the cost of such a solution is small in comparison to that of the main system of equations.

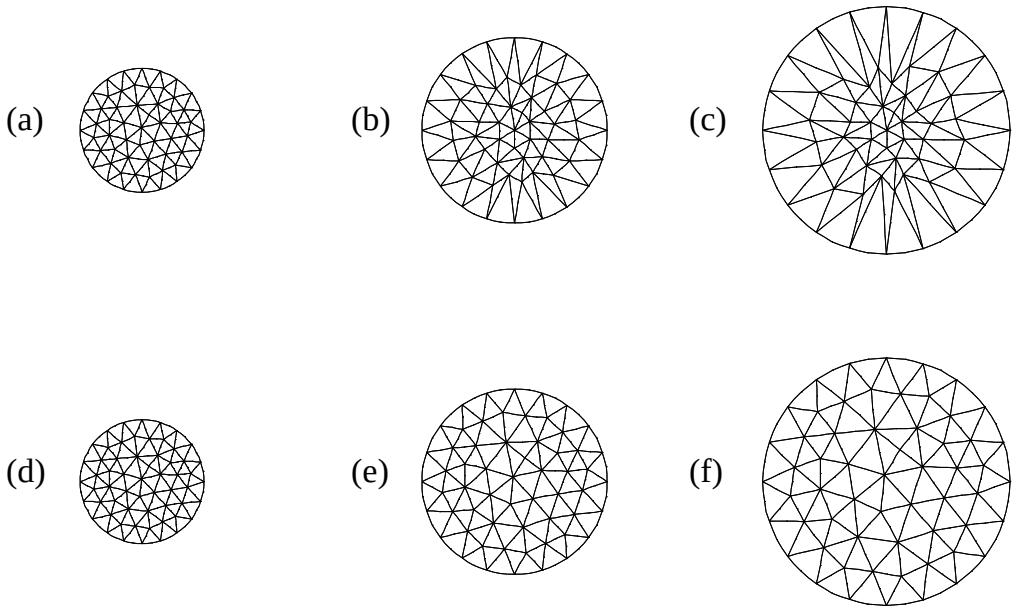


Figure 2.18: Uniform growth of a circular domain: a, b, c — elastic-mesh method; d, e, f — elastic-mesh method with Laplacian smoothing.

Practical experience has however shown that Lynch's elastic-mesh method is not, on its own, sufficient to ensure acceptable mesh quality. One failing of the method can be illustrated by considering a particularly simple free-surface problem. Figure

2.18 shows the evolution of a circular mesh in which the free-surface nodes are constrained to move radially outwards at a constant velocity. The top row (a)–(c) illustrates the distortions that arise when the interior mesh is updated using only the elastic-mesh method. As Fig. 2.18(c) shows, elements near to the free surface have undergone the largest deformations, while elements near to the centre have deformed only slightly. Similar effects are also observed for more complex geometries such as those described in Chapter 4. While this problem might be addressed by varying $\mathbf{C}$ in response to the sizes of nearby elements, such an approach would clearly require considerable additional research.

2.5.2 Laplacian smoothing

In view of the above difficulty Laplacian smoothing of the mesh [19, 64, 25, 73, 27] was also investigated. This involves the application of a weighted-Jacobi smoothing operator a number of times after each elastic-mesh solve. The operator employed amounts to updating the position $\mathbf{r}_i$ of each interior node according to the following iterative scheme

$$\mathbf{r}_i^{(n+1)} = (1-\omega)\mathbf{r}_i^{(n)} + \frac{\omega}{N_i}\sum_{j=1}^{N_i}\mathbf{r}_j^{(n)}, \tag{2.70}$$

where j sums over the N_i neighbouring vertices of vertex i, and ω is a relaxation parameter, here taken to be 0.1. The operator (2.70) is applied a small number of times (e.g. 40), rather than iterating to convergence; the Jacobi-type relaxation scheme being preferred to its Gauss-Seidel equivalent since it does not introduce any effects dependent upon node ordering. The cost of applying such a smoothing operation is negligible in comparison to that of solving the main system of equations. The bottom row of Fig. 2.18 shows the evolution of the same mesh when Laplacian smoothing is carried out in addition. Clearly the results obtained with Laplacian-smoothing are superior to those obtained using only the elastic-mesh approach, at least for this sort of problem. Indeed there does not appear to be any obvious reason why Laplacian smoothing may not be used as the sole means of performing the continuous mesh updates.

In practice Laplacian smoothing appears to be remarkably robust. There are however certain situations in which it will fail, though in the current context these cause no real difficulties since the mesh may be regenerated automatically if such configurations arise. Figure 2.19 illustrates one such pathological configuration. In this situation, the possibility exists that the node indicated may jump over the intervening node — turning two of the elements 'inside-out' in the process. Such

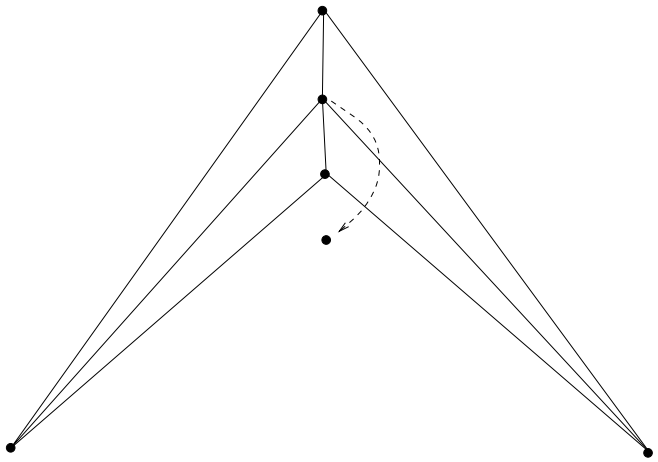


Figure 2.19: Laplacian smoothing: a mode of failure.

configurations are possible only when a mesh contains small angles. The risk of this type of failure occurring may be minimised by employing a suitably small relaxation parameter ω in (2.70), limiting the distance a node may move in any given sweep of the Laplacian-smoothing operator. It is also clear that bounding the minimum mesh angle in (2.53) helps to prevent this type of configuration arising. In practice failures of this type have not been observed to occur under normal circumstances.

2.6 Conclusions

The convergence properties of Taylor-Hood elements have been investigated for problems with natural boundary conditions on unstructured meshes and the theoretical asymptotic rates of convergence confirmed. Techniques for the automatic regeneration of unstructured meshes have been described for time-dependent surface-tension-driven free-surface problems, and constraints on initial free-surface shapes discussed. The difficulty of accurately computing the free-surface curvature and thus the stress on the boundary is pointed out. An analysis of the rate of convergence of stress boundary conditions computed using Ruschak's method has been presented for quadratic isoparametric elements. The conclusion is reached that the overall rate of convergence of the scheme will be limited by the rate of convergence of the stress boundary conditions.

The use of meshes with different resolutions for the free surface and interior meshes has been described. This technique allows the number of degrees of freedom in a problem to be substantially reduced, while at the same time allowing the full rate of convergence dictated by the free-surface boundary conditions to be obtained.

Finally, Lynch's global elastic-mesh scheme is shown to have serious deficiencies which are not shared by local Jacobi-type mesh-smoothing schemes.

Chapter 3

Solvers for incompressible flows

In this chapter the weak form of the Navier-Stokes equations is introduced and the finite element formulation presented. Next, semi and fully implicit time discretisation schemes are described for both Stokes and Navier-Stokes problems. The conjugate residual algorithm and a number of preconditioning schemes are next described. This is followed by a discussion of a number of simple interpolation schemes for the transfer of solutions between meshes when mesh regeneration is necessary. Finally the issue of the choice of time-step size is addressed, a novel approach to this issue being described.

3.1 The Galerkin method

In Eulerian form the dimensionless Navier-Stokes equations for an incompressible Newtonian fluid, as derived in Chapter 1 as (1.5) and (1.6), are

$$\left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u}\right] = \frac{1}{Re}\left[\nabla^2\mathbf{u} + \nabla(\nabla.\mathbf{u})\right] - \nabla p - \frac{1}{Fr}\mathbf{j}, \tag{3.1}$$

$$\nabla.\mathbf{u} = 0. \tag{3.2}$$

Note that here the viscous term is written in the alternative stress-divergence form [34]. While $\nabla(\nabla.\mathbf{u})$ is, by definition, zero at any point in an incompressible fluid, in general for the piecewise-continuous interpolants employed here this will not be the

case. The use of the stress-divergence form is crucial to the success of the current formulation in that, when the Galerkin finite element method is applied, it leads to a weak form with physically meaningful natural boundary conditions, i.e. the stress. Writing (3.1) and (3.2) in Cartesian-component form one obtains

$$\frac{\partial u}{\partial t}+\left(u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)-\frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)+\frac{\partial p}{\partial x}+\frac{1}{Fr}j_x=0, \quad (3.3)$$

$$\frac{\partial v}{\partial t}+\left(u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)-\frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)+\frac{\partial p}{\partial y}+\frac{1}{Fr}j_y=0, \quad (3.4)$$

$$\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0. \quad (3.5)$$

The global velocity and pressure trial solutions are given by

$$u=\sum_{j=1}^{N}u_j q_j, \quad (3.6)$$

$$v=\sum_{j=1}^{N}v_j q_j, \quad (3.7)$$

$$p=\sum_{j=1}^{M}p_j l_j, \quad (3.8)$$

where M is the number of pressure unknowns, and $2N$ is the number of velocity unknowns. The functions q_j and l_j are the quadratic and linear Lagrange basis functions described in Chapter 2, the variables u_j, v_j and p_j the unknown values of the velocities and pressures at the nodes.

In the Galerkin method [85] the basis functions used in defining the trial solution are also used as test functions when constructing a weighted-residual formulation. To obtain the Galerkin weighted-residual formulation, the momentum equations (3.3) and (3.4) are first multiplied by each of the N quadratic test functions, q_i, giving $2N$ discrete momentum equations. Similarly, the continuity equation (3.5) is multiplied by each of the linear test functions, l_i, to give M discrete continuity equations. The resulting discrete equations are then integrated over the domain, to

give the following system of equations

$$\int_\Omega q_i \left\{ \frac{\partial u}{\partial t} + \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} \right) - \frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \right) + \frac{\partial p}{\partial x} + \frac{1}{Fr} j_x \right\} d\Omega = 0, \tag{3.9}$$

$$\int_\Omega q_i \left\{ \frac{\partial v}{\partial t} + \left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} \right) - \frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) \right) + \frac{\partial p}{\partial y} + \frac{1}{Fr} j_y \right\} d\Omega = 0, \tag{3.10}$$

$$\int_\Omega l_i \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) d\Omega = 0, \tag{3.11}$$

i.e. $2N+M$ equations in $2N+M$ variables. Since each basis function is non-zero only on elements immediately adjacent to a node, the matrix of coefficients corresponding to the above system of equations is sparse. The viscous and pressure terms in (3.9) and (3.10) are next integrated by parts, using Green's first identity (A.1–A.5) to obtain the following weak form [85] of the equations

$$\int_\Omega q_i \frac{\partial u}{\partial t} d\Omega + \int_\Omega q_i \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} \right) d\Omega +$$

$$\int_\Omega \frac{2}{Re}\frac{\partial q_i}{\partial x}\frac{\partial u}{\partial x} d\Omega + \int_\Omega \frac{1}{Re}\frac{\partial q_i}{\partial y}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) d\Omega - \int_\Omega \frac{\partial q_i}{\partial x} p\, d\Omega + \int_\Omega q_i \frac{1}{Fr} j_x d\Omega$$

$$= \int_{\partial\Omega} \frac{2}{Re} q_i \frac{\partial u}{\partial x} n_x ds + \int_{\partial\Omega} \frac{1}{Re} q_i \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_y ds - \int_{\partial\Omega} q_i p n_x ds, \tag{3.12}$$

$$\int_\Omega q_i \frac{\partial v}{\partial t} d\Omega + \int_\Omega q_i \left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} \right) d\Omega +$$

$$\int_\Omega \frac{2}{Re}\frac{\partial q_i}{\partial y}\frac{\partial v}{\partial y} d\Omega + \int_\Omega \frac{1}{Re}\frac{\partial q_i}{\partial x}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) d\Omega - \int_\Omega \frac{\partial q_i}{\partial y} p\, d\Omega + \int_\Omega q_i \frac{1}{Fr} j_y d\Omega$$

$$= \int_{\partial\Omega} \frac{2}{Re} q_i \frac{\partial v}{\partial y} n_y ds + \int_{\partial\Omega} \frac{1}{Re} q_i \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_x ds - \int_{\partial\Omega} q_i p n_y ds, \tag{3.13}$$

$$- \int_{\Omega} l_i \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) d\Omega = 0, \tag{3.14}$$

where $\mathbf{n} = (n_x, n_y)$ is the outward free-surface normal. This leaves only first spatial derivatives of the velocity in the momentum equations, and eliminates the spatial derivatives of the pressure. Note that the continuity equation has been multiplied by -1 so that the matrix corresponding to the Stokes operator embedded in (3.12–3.14) is symmetric. Finally, the right-hand sides of (3.12) and (3.13) may be rearranged to give

$$\int_{\partial\Omega} q_i \left(-p n_x + \frac{2}{Re} \frac{\partial u}{\partial x} n_x + \frac{1}{Re} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_y \right) ds, \tag{3.15}$$

$$\int_{\partial\Omega} q_i \left(-p n_y + \frac{2}{Re} \frac{\partial v}{\partial y} n_y + \frac{1}{Re} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_x \right) ds. \tag{3.16}$$

As will be shown in the following section, the bracketed expressions in (3.15) and (3.16) take the same form as the stress at a point in an incompressible Newtonian fluid. Thus the homogeneous natural boundary condition for this particular weak form of the Navier-Stokes equations, i.e. that obtained by setting the boundary integrals (3.15) and (3.16) to zero in the discrete formulation, corresponds to the imposition of zero stress on the boundary.

3.1.1 The stress boundary condition

In two dimensions the components of the stress, $\boldsymbol{\sigma}$, on a surface element with unit normal $\mathbf{n}$ are given, at any point in or on the surface of a fluid, by

$$\sigma_i = \sum_{j=1}^{2} \mathbf{T}_{ij} n_j, \tag{3.17}$$

where $\mathbf{T}$ is the stress tensor [2]. For an incompressible Newtonian fluid of constant viscosity, μ, the stress tensor takes the form

$$\mathbf{T}_{ij} = -p\delta_{ij} + \mu \left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} \right). \tag{3.18}$$

By combining (3.17) and (3.18) it can be shown that, at a point on the surface of an incompressible Newtonian fluid with normal $\mathbf{n}$, the x and y components of the stress are given by

$$\sigma_x = -pn_x + 2\mu\frac{\partial u}{\partial x}n_x + \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)n_y, \tag{3.19}$$

$$\sigma_y = -pn_y + 2\mu\frac{\partial v}{\partial y}n_y + \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)n_x. \tag{3.20}$$

Since the non-dimensional forms of (3.19) and (3.20) may be obtained by simply replacing μ with $\frac{1}{Re}$ it may be seen that (3.19) and (3.20) correspond exactly with the bracketed expressions in Eqs. (3.15) and (3.16). Clearly, this particular weak formulation of the Navier-Stokes equations is ideal for the imposition of physically meaningful stress boundary-conditions at free surfaces. The ease of implementation of the stress boundary-condition in this finite element formulation contrasts sharply with the difficulties involved in attempting to impose stress boundary-conditions when finite-difference methods are employed [26].

3.1.2 The kinematic boundary condition

Engelman *et al.* [24] draw attention to the need to compute finite element boundary normals with care. In particular they point out that an analytically or geometrically computed free-surface normal will not in general give rise to a well-posed problem when an essential normal boundary condition and a natural tangential boundary condition are imposed at a node. Instead, they suggest that the *finite element mass-consistent normal* be employed. They derive the mass-consistent normal by considering the discrete global form of the continuity equation

$$\int_\Omega \sum_{i=1}^{M} (l_i \nabla.\mathbf{u})\, d\Omega = 0, \tag{3.21}$$

which is obtained by summing the M discrete continuity equations. Since at every point in the domain

$$\sum_{i=1}^{M} l_i = 1, \tag{3.22}$$

it follows from (3.21) that

$$\int_\Omega \nabla.\mathbf{u}\, d\Omega = 0. \tag{3.23}$$

Thus conservation of mass is imposed globally by the finite element formulation. Note, however, that on individual elements continuity is imposed only in a discrete weighted sense, and thus mass is not conserved locally.

Substituting the finite element trial velocity solution (3.6–3.7) into (3.23) gives

$$\sum_{i=1}^{N} \left(u_i \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega + v_i \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega \right) = 0, \tag{3.24}$$

the summation being over all nodes. Here, for simplicity, it will be assumed that the free-surface nodes are numbered before the interior nodes. Using the divergence theorem, (3.24) may be rewritten in the form

$$\sum_{i=1}^{F} \left(u_i \int_{\partial\Omega} q_i n_x\, ds + v_i \int_{\partial\Omega} q_i n_y\, ds \right) = 0. \tag{3.25}$$

the sum reducing to one over the F free-surface nodes since q_i is zero on $\partial\Omega$ for each basis function corresponding to an internal node. It follows that the contributions to (3.24) from the interior nodes must cancel one another out, and thus (3.24) may also be rewritten as a sum over the F free-surface nodes.

The Cartesian velocity components u_i and v_i at free-surface node i may be expressed in the form

$$u_i = n_{x,i} u_{n,i} - n_{y,i} u_{t,i}, \tag{3.26}$$

$$v_i = n_{y,i} u_{n,i} + n_{x,i} u_{t,i}, \tag{3.27}$$

where $(n_{x,i}, n_{y,i})$ is the outward free-surface normal at node i, and $u_{n,i}$ and $u_{t,i}$ are the normal and tangential components of the velocity at node i. Substituting these expressions for u_i and v_i into (3.24) gives

$$\sum_{i=1}^{F} \left\{ (n_{x,i} u_{n,i} - n_{y,i} u_{t,i}) \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega + (n_{y,i} u_{n,i} + n_{x,i} u_{t,i}) \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega \right\} = 0, \tag{3.28}$$

which can be rearranged to give

$$\sum_{i=1}^{F} \left\{ \left(n_{x,i} \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega + n_{y,i} \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega \right) u_{n,i} + \left(n_{x,i} \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega - n_{y,i} \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega \right) u_{t,i} \right\} = 0. \tag{3.29}$$

Since (3.29) must hold for any set of tangential velocities, $u_{t,i}$, regardless of the

values of the normal velocity components, $u_{n,i}$, it follows that

$$n_{x,i} \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega = n_{y,i} \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega \tag{3.30}$$

for all i. Thus, the mass-consistent normal at node i is defined using

$$n_{x,i} = \frac{1}{n_i} \int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega, \tag{3.31}$$

$$n_{y,i} = \frac{1}{n_i} \int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega, \tag{3.32}$$

where, since a unit normal is required, n_i is given by

$$n_i = \sqrt{\left(\int_\Omega \frac{\partial q_i}{\partial x}\, d\Omega \right)^2 + \left(\int_\Omega \frac{\partial q_i}{\partial y}\, d\Omega \right)^2}, \tag{3.33}$$

the positive root being selected so as to give the outward normal. Engelman *et al.* [24] point out that the mass-consistent normal, computed in this fashion, gives acceptable results when applied to triangular elements with no more than two vertices on the boundary. Here this is ensured by triangulating into corners, so that no element has two edges that form part of the boundary.

In the current work the mass-consistent normal is employed when updating the locations of free-surface nodes, so that the kinematic boundary condition

$$\mathbf{u}.\mathbf{n} = \dot{\mathbf{s}}.\mathbf{n} \tag{3.34}$$

may be implemented in a consistent manner. Note that the notation $\mathbf{s}$ is employed when referring to a particular free surface, and that $\mathbf{s}_i$ corresponds to the position of free-surface node i. In a first-order explicit free-surface advection scheme, once the normals at the free-surface nodes have been computed, the locations of the free-surface nodes $\mathbf{s}_i$ are updated using the rule

$$\mathbf{s}_i^{(n+1)} = \mathbf{s}_i^{(n)} + k(\mathbf{u}_i^{(n)} \cdot \mathbf{n}_i{}^{(n)})\mathbf{n}_i{}^{(n)}, \tag{3.35}$$

where k is the size of the current time step, $\mathbf{s}_i^{(n)}$ the position of node i, $\mathbf{n}_i{}^{(n)}$ the normal at node i and $\mathbf{u}_i^{(n)}$ the velocity at node i. Note, however, that while the choice of normal direction is correct at the start of a time step, in general, it will not be correct at the end of the time step. Thus conservation of mass will not be enforced exactly, although if the boundary is well resolved and moving sufficiently

slowly, and if the time-step size is sufficiently small, then the resulting errors can be very small.

3.2 Moving-mesh corrections

Both methods described in Chapter 2 for updating the mesh between regenerations give rise to motions of the interior nodes. If the Laplacian smoother described in Section 2.5.2 is applied to a newly regenerated mesh, the boundary of which is moving only slowly then for the first few time steps after the regeneration has occurred many of the interior nodes will move relatively quickly. Eventually an equilibrium will be reached and any movement of interior nodes due to the Laplacian smoother will become negligible. Once such an equilibrium has been reached, application of the smoother at the end of each time step will result in motions of interior nodes that reflect the motion of the free surface. Similar behaviour is also observed where the elastic-mesh method is employed, though in this case there is no initial transient phase after mesh regeneration.

After the initial transient phase interior nodes continue to move, now in response to the evolution of the free surface. Thus the potential for errors to arise remains. One way of dealing with these errors is simply to ignore them. This is not entirely unreasonable, since when the time-step size is small and the free surface is moving slowly the motions of the interior nodes will also be small. Thus, if the local velocity and pressure gradients are sufficiently small, then the errors that result from the movement of nodes may be negligible. Free-surface nodes, however, must always move with the same normal velocity as the fluid. Consequently it is necessary to consider ways in which such motions may be incorporated into the finite element formulation so as to remove this source of error.

Two well-known approaches already exist for dealing with deforming meshes. The Lagrangian method [13, 4], makes use of the fact that the Navier-Stokes equations may be written directly in terms of a moving coordinate system. If the interior nodes are required to move at the fluid's local velocity, then no convective derivative terms need be included in the Navier-Stokes equations. This has the major advantage that the finite element stiffness matrix that results is now symmetric, allowing more efficient solution techniques to be employed. The main disadvantage of this approach is that the nodes must always move at the same velocity as the fluid, which can lead to the degeneration of mesh quality and even tangling of the mesh. Thus, in general, periodic regeneration of the mesh will be necessary more frequently than

if an Eulerian formulation were employed.

The Arbitrary-Lagrangian-Eulerian (ALE) approach [41, 23, 49, 100] combines the Eulerian and Lagrangian methods, allowing interior nodes to have arbitrary velocities, chosen independently of the fluid's velocity. Thus the velocities of interior nodes may be chosen so as to maintain mesh quality. The ALE formulation requires the inclusion in the Navier-Stokes equations of a new term of the form $-(\dot{\mathbf{s}} \cdot \nabla)\mathbf{u}$, where $\dot{\mathbf{s}}$ corresponds to the local velocity of the mesh. Where $\dot{\mathbf{s}} = \mathbf{u}$, as in the Lagrangian method, this new term exactly cancels the convective derivative.

The approach adopted in the current work is essentially an ALE one. Here, however, the new moving-mesh terms, rather than being assembled as part of a nonlinear finite element problem, are evaluated explicitly at the start of a time step and included in the data for the linear-algebraic systems solved for that time step.

The correct form of the moving-mesh corrections for a finite element formulation is derived in [54] and employs the following argument. Let $\mathbf{m}_i$ denote the position of node i in a mesh $\mathbf{m} = \{\mathbf{m}_i : i = 1, \ldots, N\}$, composed entirely of straight-sided triangular elements. For simplicity it is assumed that vertices are numbered before edges. The velocity $\mathbf{u}$ at a point $\mathbf{x}$ in the domain is given by

$$\mathbf{u} = \sum_{j=1}^{N} \mathbf{u}_j(t) q_j(\mathbf{x}, \mathbf{m}(t)), \tag{3.36}$$

where $\mathbf{u}_j$ is the velocity of the fluid associated with node j, and q_j the corresponding basis function, now considered as a function of $\mathbf{m}$ as well as $\mathbf{x}$. Thus for straight-sided elements the rate of change of $\mathbf{u}$ will be given by

$$\frac{\partial \mathbf{u}}{\partial t} = \sum_{j=1}^{N} \frac{d\mathbf{u}_j}{dt} q_j + \sum_{j=1}^{N} \mathbf{u}_j \sum_{k=1}^{V} \left(\frac{\partial q_j}{\partial \mathbf{m}_k} \cdot \dot{\mathbf{m}}_k \right), \tag{3.37}$$

where V is the number of vertices and $\dot{\mathbf{m}}_k$ is the velocity of vertex k. The second term on the right-hand side of (3.37) may be written as

$$\sum_{k=1}^{V} \left(\frac{\partial \mathbf{u}}{\partial \mathbf{m}_k} \cdot \dot{\mathbf{m}}_k \right) \tag{3.38}$$

and must be included whenever the vertices forming a mesh are in motion relative to one another.

Since the Laplacian smoothing and elastic-mesh schemes both result in continuous deformations of the mesh, and since a Lagrange finite element basis is involved,

Jimack and Wathen's Theorem 2.4 [54] may be employed to rewrite (3.38) in the form

$$-\sum_{k=1}^{V} l_k \nabla \mathbf{u} \cdot \dot{\mathbf{m}}_k, \tag{3.39}$$

or equivalently

$$\left(\left(\sum_{k=1}^{V} -l_k \dot{\mathbf{m}}_k \right) \cdot \nabla \right) \mathbf{u}. \tag{3.40}$$

This has a similar form to the standard convective term $(\mathbf{u} \cdot \nabla)\mathbf{u}$, the expression $\sum_{k=1}^{V} -l_k \dot{\mathbf{m}}_k$ being a piecewise linear mesh-velocity field. Thus for a moving-mesh problem the material derivative has the form

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u} - (\left(\sum_{k=1}^{V} (l_k \dot{\mathbf{m}}_k) \right) \cdot \nabla)\mathbf{u}. \tag{3.41}$$

Note that if

$$\mathbf{u} = \sum_{k=1}^{V} (l_k \dot{\mathbf{m}}_k) \tag{3.42}$$

then the moving-mesh convective derivative will exactly cancel the standard convective derivative.

The inclusion of the term (3.40) in the formulation of a problem necessarily results in the problem being nonlinear, even if the original Eulerian formulation is linear. Where functional iteration is employed as a means of handling the standard nonlinear convective derivative, i.e. by repeatedly linearising the problem about the most recent estimate for the velocity field at the end of the time step, (3.40) may either be incorporated directly into the finite element matrix, giving rise to an non-symmetric linear-algebraic problem, or it may be evaluated explicitly and treated as part of the data for the problem. The latter approach is adopted here.

In the current work the expression

$$-\sum_{k=1}^{V} (l_k \dot{\mathbf{m}}_k) \tag{3.43}$$

is simply replaced with $-\dot{\mathbf{m}}_j$ when assembling entries in the finite element matrix corresponding to the velocity degrees of freedom at node j, i.e. a pointwise weighted

form of the term. This simplification is employed since it allows the assembly of the moving-mesh terms to be combined with that of the standard convective derivatives, reducing the overall computational cost considerably. At the end of each time step the nodal velocities, $\dot{\mathbf{m}}_j^{(n+1)}$, are estimated explicitly using $\mathbf{m}_j{}^{(n)}$ and $\mathbf{m}_j{}^{(n+1)}$. Consequently, such estimates are not available immediately after a mesh regeneration, though in principle there is no reason why the mesh-velocity field could not also be interpolated when the mesh is regenerated. In practice, in the experiments described in Chapter 6, the inclusion of the moving-mesh terms was not found to affect greatly either the quality or the accuracy of the solutions obtained, suggesting that, with the relatively coarse meshes employed here, spatial discretisation errors dominate the computations.

3.3 Matrix formulation

In order to solve (3.12–3.14) numerically the problem must also be discretised in time as well as in space. Temporal discretisation of finite element formulations frequently involves the use of finite-difference approximations for the temporal derivatives, the so-called method of lines. Here the additional complication arises that the mesh is required to deform as the free surface evolves, and thus the nodes will be in motion relative to one another. Consequently, when solving over a time step it is often necessary to consider two different meshes, the initial mesh at the start of the time step, and the final mesh that represents the domain at the end of the time step. The following matrices are now defined

$$[\mathbf{M}_1] = [\mathbf{M}_2] = \int_\Omega q_i q_j d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}, \tag{3.44}$$

$$[\mathbf{A}_{11}] = \int_\Omega \frac{2}{Re}\frac{\partial q_i}{\partial x}\frac{\partial q_j}{\partial x} + \frac{1}{Re}\frac{\partial q_i}{\partial y}\frac{\partial q_j}{\partial y} d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}, \tag{3.45}$$

$$[\mathbf{A}_{12}] = \int_\Omega \frac{1}{Re}\frac{\partial q_i}{\partial y}\frac{\partial q_j}{\partial x} d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}, \tag{3.46}$$

$$[\mathbf{A}_{21}] = \int_\Omega \frac{1}{Re}\frac{\partial q_i}{\partial x}\frac{\partial q_j}{\partial y} d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}, \tag{3.47}$$

$$[\mathbf{A}_{22}] = \int_\Omega \frac{2}{Re}\frac{\partial q_i}{\partial y}\frac{\partial q_j}{\partial y} + \frac{1}{Re}\frac{\partial q_i}{\partial x}\frac{\partial q_j}{\partial x} d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}, \tag{3.48}$$

$$[\mathbf{B}_1] = -\int_\Omega \frac{\partial q_i}{\partial x} l_j d\Omega \qquad \text{i} = 1, \ldots, \text{N}, \ \text{j} = 1, \ldots, \text{M}, \tag{3.49}$$

$$[\mathbf{B}_2] = -\int_\Omega \frac{\partial q_i}{\partial y} l_j d\Omega \qquad \text{i} = 1, \ldots, \text{N}, \ \text{j} = 1, \ldots, \text{M}, \tag{3.50}$$

$$[\mathbf{C}_{11}] = [\mathbf{C}_{22}] = \int_\Omega q_i \left(u^* \frac{\partial q_j}{\partial x} + v^* \frac{\partial q_j}{\partial y} \right) d\Omega \qquad \text{i,j} = 1, \ldots, \text{N}. \tag{3.51}$$

Note that $\mathbf{C}_{11}$ and $\mathbf{C}_{22}$ represent a linearisation of the convective derivative around an estimate of the velocity field $\mathbf{u}^* = (u^*, v^*)$, and is only one of the schemes possible [33, 34]. Typically $\mathbf{u}^*$ will be either the velocity field from the previous iteration of the nonlinear solver or the velocity field from the end of the previous time step.

The following vectors are also defined, representing respectively, the gravitational body force,

$$\mathbf{g}_1 = -\int_\Omega q_i \frac{1}{Fr} j_x d\Omega \qquad \text{i} = 1, \ldots, \text{N}, \tag{3.52}$$

$$\mathbf{g}_2 = -\int_\Omega q_i \frac{1}{Fr} j_y d\Omega \qquad \text{i} = 1, \ldots, \text{N}, \tag{3.53}$$

the free-surface stress boundary conditions,

$$\mathbf{d}_1 = \int_{\partial\Omega} -q_i \frac{1}{We}\frac{1}{R_c} n_x ds \qquad \text{i} = 1, \ldots, \text{N}, \tag{3.54}$$

$$\mathbf{d}_2 = \int_{\partial\Omega} -q_i \frac{1}{We}\frac{1}{R_c} n_y ds \qquad \text{i} = 1, \ldots, \text{N}, \tag{3.55}$$

and the moving mesh corrections

$$\mathbf{h}_1 = \sum_{j=1}^{N} \left(\int_\Omega q_i \dot{m}_{j,x} \frac{\partial q_j}{\partial x} u_j d\Omega + \int_\Omega q_i \dot{m}_{j,y} \frac{\partial q_j}{\partial y} u_j d\Omega \right) \quad \text{i} = 1, \ldots, \text{N}, \tag{3.56}$$

$$\mathbf{h}_2 = \sum_{j=1}^{N} \left(\int_\Omega q_i \dot{m}_{j,x} \frac{\partial q_j}{\partial x} v_j d\Omega + \int_\Omega q_i \dot{m}_{j,y} \frac{\partial q_j}{\partial y} v_j d\Omega \right) \quad \text{i} = 1, \ldots, \text{N}, \tag{3.57}$$

where $\dot{\mathbf{m}}_\mathbf{j} = (\dot{m}_{j,x}, \dot{m}_{j,y})$ is an estimate of the velocity of node j. The free-surface stress boundary conditions included in (3.54) and (3.55) are those appropriate for a Navier-Stokes problem, as discussed in Section 1.2. If the factors $\frac{1}{We}$ are deleted then the stress boundary conditions for a Stokes-flow problem result (see Section 1.3). Where general stress boundary conditions are known explicitly in the form $\boldsymbol{\sigma} = (\sigma_x, \sigma_y)$, as in the Stokes-flow test problem described in Chapter 2, (3.54) and (3.55) may be replaced by

$$\mathbf{d}_1 = \int_{\partial\Omega} q_i \sigma_x ds, \tag{3.58}$$

$$\mathbf{d}_2 = \int_{\partial\Omega} q_i \sigma_y ds. \tag{3.59}$$

All the above integrands, with the possible exceptions of the boundary conditions $\mathbf{d}_1$ and $\mathbf{d}_2$, are polynomials and may thus be evaluated exactly using Gauss-Legendre quadrature rules [22] of appropriate degree. The boundary integrals in (3.54) and (3.55) are evaluated using the approach described in Section 2.2.2. Employing the above definitions the system (3.12–3.14) may be rewritten as

$$\begin{pmatrix} \mathbf{M}_1 & 0 & 0 \\ 0 & \mathbf{M}_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{p} \end{pmatrix} + \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{B}_1 \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{B}_2 \\ {\mathbf{B}_1}^T & {\mathbf{B}_2}^T & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix}$$

$$+ \begin{pmatrix} \mathbf{C}_{11}(u^*, v^*) & 0 & 0 \\ 0 & \mathbf{C}_{22}(u^*, v^*) & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ p \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ 0 \end{pmatrix} + \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ 0 \end{pmatrix} + \begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ 0 \end{pmatrix}, \tag{3.60}$$

were u, v and p are vectors representing the velocity and pressure unknowns.

When working with a problem with essential boundary conditions at some boundary nodes, the equations corresponding to the degrees of freedom that are known *a priori* need not be assembled as part of the linear-algebraic problem. Thus, the rows and columns corresponding to these degrees of freedom may be eliminated from the problem, reducing its size considerably. When this is done, the contributions to the remaining momentum equations that result when the unwanted equations are condensed out of the system are incorporated as a vector $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)^T$, which is added to the right-hand side of (3.60).

3.4 Time-discretisation schemes

Temporal discretisation is performed using the *θ-method*, a standard generalisation of the Crank-Nicholson or trapezoidal rule [16]. Thus (3.60) becomes

$$\begin{pmatrix} \mathbf{M}_1 & 0 & 0 \\ 0 & \mathbf{M}_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n+1)} - \begin{pmatrix} \mathbf{M}_1 & 0 & 0 \\ 0 & \mathbf{M}_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}^{(n)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n)}$$

$$+k\theta \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{B}_1 \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{B}_2 \\ \mathbf{B}_1{}^T & \mathbf{B}_2{}^T & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n+1)}$$

$$+k(1-\theta) \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{B}_1 \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{B}_2 \\ \mathbf{B}_1{}^T & \mathbf{B}_2{}^T & 0 \end{pmatrix}^{(n)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n)}$$

$$+k\theta \begin{pmatrix} \mathbf{C}_{11}(u^*, v^*) & 0 & 0 \\ 0 & \mathbf{C}_{22}(u^*, v^*) & 0 \\ 0 & 0 & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n+1)}$$

$$+k(1-\theta) \begin{pmatrix} \mathbf{C}_{11}(u^{(n)}, v^{(n)}) & & 0 \\ 0 & \mathbf{C}_{22}(u^{(n)}, v^{(n)}) & 0 \\ 0 & & 0 \end{pmatrix}^{(n)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n)}$$

$$= k\theta \left(\mathbf{d}^{(n+1)} + \mathbf{e}^{(n+1)} + \mathbf{g}^{(n+1)} + \mathbf{h}^{(n+1)} \right)$$

$$+k(1-\theta) \left(\mathbf{d}^{(n)} + \mathbf{e}^{(n)} + \mathbf{g}^{(n)} + \mathbf{h}^{(n)} \right), \tag{3.61}$$

where $0 < \theta \leq 1$, k is the time-step size and the notation $^{(n)}$ denotes variables and operators corresponding to the start of the time step, $^{(n+1)}$ those at the end.

If one takes $\theta = 1.0$ then (3.61) reduces to a standard backward-Euler scheme [34], of temporal accuracy $O(k)$. A value of $\theta = 0.5$, on the other hand, results in a Crank-Nicholson or central-difference scheme of accuracy $O(k^2)$. In practice, values of θ other than 0.5 may be employed [12], despite the fact that then the scheme is only formally $O(k)$ accurate. Values of θ other than 0.5 are often preferred since the Crank-Nicholson scheme has a tendency to propagate discontinuities in the initial conditions rather than damping them out. Including a small component of the more diffusive backward-Euler scheme helps to damp out these transients though without significantly altering the scheme's accuracy. Indeed, for an appropriate choice of θ the accuracy obtained may actually be better than that achievable with the Crank-Nicholson scheme [12]. Finally note that if $\theta = 0$ the system (3.61) is singular.

The convective term in (3.61) that corresponds to the end of the time step is evaluated using the most recently obtained estimate $\mathbf{u}^*$ for the velocity field $\mathbf{u}^{(n+1)}$ at the end of the time step and incorporated as part of the data for the problem as the vector $\mathbf{f}^{n+1}$ which is defined as follows

$$\mathbf{f}^{(n+1)} = - \begin{pmatrix} \mathbf{C}_{11}(u^*, v^*) & 0 & 0 \\ 0 & \mathbf{C}_{22}(u^*, v^*) & 0 \\ 0 & 0 & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u^* \\ v^* \\ 0 \end{pmatrix}^{(n+1)} . \tag{3.62}$$

To obtain the computational formulation employed when solving a Navier-Stokes problem (3.61) is rearranged in the following form

$$\begin{pmatrix} \mathbf{M}_1 + k\theta\mathbf{A}_{11} & k\theta\mathbf{A}_{12} & k\theta\mathbf{B}_1 \\ k\theta\mathbf{A}_{21} & \mathbf{M}_2 + k\theta\mathbf{A}_{22} & k\theta\mathbf{B}_2 \\ k\theta{\mathbf{B}_1}^T & k\theta{\mathbf{B}_2}^T & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n+1)}$$

$$= \begin{pmatrix} \mathbf{M}_1 - k(1-\theta)\mathbf{A}_{11} & -k(1-\theta)\mathbf{A}_{12} & -k(1-\theta)\mathbf{B}_1 \\ -k(1-\theta)\mathbf{A}_{21} & \mathbf{M}_2 - k(1-\theta)\mathbf{A}_{22} & -k(1-\theta)\mathbf{B}_2 \\ -k(1-\theta){\mathbf{B}_1}^T & -k(1-\theta){\mathbf{B}_2}^T & 0 \end{pmatrix}^{(n)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n)}$$

$$+k\theta\left(\mathbf{d}^{(n+1)} + \mathbf{e}^{(n+1)} + \mathbf{f}^{(n+1)} + \mathbf{g}^{(n+1)} + \mathbf{h}^{(n+1)}\right)$$

$$+k(1-\theta)\left(\mathbf{d}^{(n)} + \mathbf{e}^{(n)} + \mathbf{f}^{(n)} + \mathbf{g}^{(n)} + \mathbf{h}^{(n)}\right). \tag{3.63}$$

The stiffness and mass matrix terms that correspond to the start of the time step are normally available from the previous time step and thus do not need recomputing unless the mesh is regenerated. Note that the matrix in the linear-algebraic problem (3.63) is symmetric and consequently the conjugate residual method may be applied directly. Since (3.63) is referred to frequently in the following sections, it will be convenient to write it in the concise form

$$\mathbf{K}\mathbf{x} = \mathbf{b}, \tag{3.64}$$

where $\mathbf{x} = (u, v, p)^T$.

By solving a sequence of linearised problems of the form (3.63) to obtain increasingly accurate approximations to $\mathbf{u}^* = \mathbf{u}^{(n+1)}$ the solution of the nonlinear system may be obtained, a process known as formula or Picard iteration. While this is not guaranteed to converge, in practice convergence normally does occur provided the time step is sufficiently small. If the mesh at the end of a time step is held fixed during the nonlinear solution process, then at each outer iteration of the nonlinear solver only the vectors $\mathbf{f}^{(n+1)}$ and $\mathbf{h}^{(n+1)}$ need be recomputed, it being possible to compute the others as soon as the mesh at the end of the time step is known. Where the mesh at the end of the time step is allowed to change during the nonlinear solution process, the matrix on the left-hand side of (3.63) and the vectors on the right-hand side that correspond to the end of the time step must be reassembled for each new mesh.

3.5 A semi-implicit scheme for free-surface Navier-Stokes problems

The solution of time-dependent free-surface problems is complicated by the need to update the position of the free surface at each time step in accordance with the kinematic boundary condition (1.4). In time-dependent problems the kinematic boundary condition gives rise to an additional set of nonlinear ordinary differential equations that must, ideally, be solved simultaneously with the momentum and continuity equations. This is, however, potentially rather expensive, since whenever the free surface is modified, the mesh must also be modified, giving rise to a new system of discrete equations. One way of avoiding this complication is to assume that the kinematic boundary condition may safely be decoupled from the momentum and continuity equations. Thus, the kinematic boundary condition may be applied explicitly at the start of a time step, using the velocity field computed at the end of the previous time step, so as to give the location of the free surface and thus the new mesh at the end of the time step. The Navier-Stokes equations may then be solved on a pair of fixed meshes of identical connectivity to give the velocity and pressure at the end of the time step.

This approach, when applied to the system (3.64), results in the semi-implicit algorithm shown in Fig. 3.1. The algorithm assumes the existence of an initial mesh $\mathbf{m}^{(n)}$, with free surface $\mathbf{s}^{(n)}$, on which the initial velocity field is $\mathbf{u}^{(n)}$. Note that no initial conditions are required for the pressure, though it would appear appropriate to make the first time step a backward-Euler one if the initial pressure field is not known.

As a first stage of each time step $\mathbf{u}^{(n)}$ is used in conjunction with (3.35) to explicitly compute $\mathbf{s}^{(n+1)}$, the free surface at the end of the time step. Techniques such as those described in Section 2.5 are then used to modify $\mathbf{m}^{(n)}$ in order to give $\mathbf{m}^{(n+1)}$, the mesh at the end of the time step. Both $\mathbf{m}^{(n+1)}$ and $\mathbf{s}^{(n+1)}$ are now held fixed for the remainder of the step. The matrix on the left-hand side of (3.63) is now assembled on $\mathbf{m}^{(n+1)}$ and the nonlinear Navier-Stokes problem solved using functional iteration to give the velocity field $\mathbf{u}^{(n+1)}$ at the end of the time step. The procedure may now be repeated, $\mathbf{u}^{(n+1)}$ giving the initial conditions for the next time step.

Note that once $\mathbf{m}^{(n+1)}$, and $\mathbf{s}^{(n+1)}$ have been found, $\mathbf{d}^{(n+1)}$, $\mathbf{e}^{(n+1)}$ and $\mathbf{g}^{(n+1)}$ may all be computed immediately, and remain fixed throughout the remainder of the nonlinear solve. Thus only the vectors $\mathbf{f}^{(n+1)}$ and $\mathbf{h}^{(n+1)}$ must be recomputed

whenever a new approximation for $\mathbf{u}^{(n+1)}$ becomes available. At each nonlinear iteration the solution from the previous iteration, or from the previous time step, is employed as an initial guess for the linear solver. The algorithm terminates when the L_∞ norm of the residual vector (pressure and velocity components) falls below an absolute tolerance *tol*. This typically takes between five and ten outer iterations, the precise number depending on the length of the time step and the degree of accuracy required. It is this algorithm that is employed in the investigations described in Chapter 6.

1. Generate a new free surface $\mathbf{s}^{(n+1)}$, using $\mathbf{u}^{(n)}$, $\mathbf{s}^{(n)}$ and (3.35).
2. Generate a new mesh $\mathbf{m}^{(n+1)}$ by updating $\mathbf{m}^{(n)}$ using the techniques described in Section 2.5.
3. Assemble the matrix $\mathbf{K}$ on $\mathbf{m}^{(n+1)}$.
4. Set $\mathbf{u}^* = \mathbf{u}^{(n)}$.
5. Compute preconditioner for $\mathbf{K}$.
6. Assemble $\mathbf{b}$, using $\mathbf{u}^*$ to recompute $\mathbf{f}^{(n+1)}$.
7. Solve the linear system (3.64) to give $\mathbf{x}^{(n+1)}$.
8. If $|\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}|_\infty < tol$ then proceed to next time step.
9. Set $\mathbf{u}^* = \mathbf{u}^{(n+1)}$
10. Go to step 6.

Figure 3.1: A semi-implicit algorithm for free-surface Navier-Stokes problems.

A potentially very advantageous simplification arises if the convective and moving-mesh terms are implemented using a purely explicit scheme. Thus, for example, $\mathbf{f}^{(n+1)}$ and $\mathbf{h}^{(n+1)}$ may be approximated using $\mathbf{f}^{(n)}$ and $\mathbf{h}^{(n)}$. Experience has shown that this modification is, in practice, often satisfactory for the types of problems considered here, and may be employed without seriously affecting the accuracy or compromising the stability of the method. Where this is done each time step requires the solution of only a single system of linear equations, reducing the cost per time step by up to an order of magnitude. Since, in such schemes, the viscous operator is treated implicitly one would expect that any stability constraints on the time-step size that will arise will be due entirely to the use of explicit schemes for the update of the free surface and for the convective and moving-mesh terms. Thus the maximum stable time step will be $O(h)$ rather than the more restrictive $O(h^2)$ that would apply if the viscous term were treated explicitly [34].

3.6 An implicit scheme for free-surface Navier-Stokes problems

The fully implicit scheme that is described here is similar in many ways to the semi-implicit scheme described above. This time whenever a new estimate $\mathbf{u}^{(n+1)}$ is computed it is used in conjunction with $\mathbf{u}^{(n)}$ to re-apply the kinematic boundary condition to the original free surface $\mathbf{s}^{(n)}$, thus giving a better approximation to $\mathbf{s}^{(n+1)}$. Since, generally, only the normal component of the motion of free-surface nodes is of interest, the kinematic boundary condition (3.35) may be written in the form

$$\dot{\mathbf{s}}_i = \mathbf{n}_i(\mathbf{u}_i \cdot \mathbf{n}_i), \tag{3.65}$$

where $\mathbf{s}_i$ is the position of free-surface node i and $\mathbf{n}_i$ is the mass-consistent normal at node i. If (3.65) is discretised using the θ-method, one obtains the following expression for $\mathbf{s}_i^{(n+1)}$

$$\mathbf{s}_i^{(n+1)} = \mathbf{s}_i^{(n)} + k\left\{\theta\mathbf{n}_i^{(n+1)}(\mathbf{u}_i^{(n+1)} \cdot \mathbf{n}_i^{(n+1)}) + (1-\theta)\mathbf{n}_i^{(n)}(\mathbf{u}_i^{(n)} \cdot \mathbf{n}_i^{(n)})\right\}. \tag{3.66}$$

The question must be asked as to whether (3.66) introduces any new stability constraints on the time-step size. While one might intuitively expect that such a scheme would be unconditionally stable for $\theta \geq 0.5$, the scheme is clearly highly nonlinear since $\mathbf{n}_i^{(n+1)}$ depends in a time-dependent manner on the locations of a number of free-surface nodes. The author's experience with fully implicit schemes however suggests that, in practice, the time-step size constraint associated with (3.66) is of $O(h)$.

Figure 3.2 shows the fully implicit algorithm investigated as part of the current work for the solution of problems of the form (3.63). Note that now a new matrix $\mathbf{K}$ must be assembled, and possibly a new preconditioner computed, at each iteration of the nonlinear solver. The algorithm terminates when the maximum change in any component of the pressure or velocity falls below a prescribed absolute tolerance tol_1 and the maximum free-surface nodal displacement falls below an absolute tolerance tol_2.

In practice, provided the time-step size k is small enough, the functional iteration scheme shown in Fig. 3.2 converges. If, however, an attempt is made to employ a time step much larger than that indicated by the time-step constraint (3.79) then the scheme becomes unreliable. In such circumstances convergence may be extremely slow, if it occurs at all. Consequently, the maximum time-step size that

1. Set $\mathbf{u}^* = \mathbf{u}^{(n)}$.
2. Generate a new free-surface $\mathbf{s}^{(n+1)}$, using $\mathbf{u}^{(n)}$, $\mathbf{u}^*$, $\mathbf{s}^{(n)}$ and (3.66).
3. Generate a new mesh $\mathbf{m}^{(n+1)}$ by updating $\mathbf{m}^{(n)}$ using the techniques described in Section 2.5.
4. Assemble the matrix $\mathbf{K}$ on $\mathbf{m}^{(n+1)}$.
5. Compute preconditioner for $\mathbf{K}$.
6. Assemble $\mathbf{b}$, using $\mathbf{u}^*$ to compute $\mathbf{f}^{(n+1)}$.
7. Solve the linear system (3.64) to give $\mathbf{x}^{(n+1)}$.
8. If $|\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}|_\infty < tol_1$ and $|\mathbf{s}^{(n+1)} - \mathbf{s}^{(n)}|_\infty < tol_2$ then proceed to next step.
9. Set $\mathbf{u}^* = \mathbf{u}^{(n+1)}$.
10. Go to step 2.

Figure 3.2: A fully implicit algorithm for free-surface Navier-Stokes problems.

may be employed is now constrained by the need to ensure that the nonlinear solver converges, and thus little is gained by the use of the fully implicit method. Since the additional costs associated with having to re-assemble essentially the entire problem at each iteration of the nonlinear solver are large, the scheme is clearly inefficient. Thus it appears prudent to postpone further investigation in this area until more sophisticated nonlinear solvers are available.

3.7 A semi-implicit scheme for free-surface Stokes problems

A semi-implicit algorithm for Stokes-flow problems may be derived from that described above for the Navier-Stokes equations by making a number of simplifications. As mentioned in Section 1.3, the absence of temporal derivatives in the Stokes equations means that at each time step only a quasi-steady-state problem need be solved for the velocity, with the only time-dependency resulting from the kinematic boundary condition (1.4). In this semi-implicit scheme the kinematic boundary condition is implemented explicitly using only the velocity field at the start of the time step. The interior of the mesh is then updated and the flow problem is solved at the end of the time step on the new mesh. Thus, at each time step only a single linear-algebraic

problem of the form

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{B}_1 \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{B}_2 \\ \mathbf{B}_1{}^T & \mathbf{B}_2{}^T & 0 \end{pmatrix}^{(n+1)} \begin{pmatrix} u \\ v \\ p \end{pmatrix}^{(n+1)} = \mathbf{d}^{(n+1)} + \mathbf{e}^{(n+1)} + \mathbf{g}^{(n+1)} \tag{3.67}$$

must be solved. Note the absence of the moving-mesh correction terms in this case, since now the problem is solved on a fixed mesh. Figure 3.3 shows the semi-implicit algorithm for free-surface Stokes-flow problems. This is the algorithm that was used to obtain the results described in Chapters 4 and 5.

1. Generate a new free surface $\mathbf{s}^{(n+1)}$, using $\mathbf{u}^{(n)}$, $\mathbf{s}^{(n)}$ and (3.35).
2. Generate a new mesh $\mathbf{m}^{(n+1)}$ by updating $\mathbf{m}^{(n)}$ using the techniques described in Section 2.5.
3. Assemble the matrix $\mathbf{K}$ on $\mathbf{m}^{(n+1)}$.
5. Compute preconditioner for $\mathbf{K}$.
6. Assemble $\mathbf{b}$ on $\mathbf{m}^{(n+1)}$.
7. Solve the linear system (3.64) to give $\mathbf{x}^{(n+1)}$.
8. Proceed to next time step.

Figure 3.3: A semi-implicit scheme for free-surface Stokes-flow problems.

3.8 A fully implicit scheme for free-surface Stokes problems

For a Stokes-flow problem nonlinearity is present through both the free-surface stress boundary condition and the kinematic boundary condition. The semi-implicit method described in the previous section avoids the need to solve a nonlinear problem at each time step by employing an explicit free-surface update step. Thus the linear part of the problem, the flow calculation, is solved implicitly while the more difficult nonlinear part of the problem is solved explicitly.

An implicit scheme arises if the kinematic boundary condition is implemented using the velocity fields at both the start and the end of the time step, i.e. using (3.66). Consequently the free surface, and thus the mesh at the end of the time step, must be found by an iterative process. The problem that must be solved at each time step is now nonlinear, the nonlinearity entering through the effects of the motions

of the free-surface nodes on the boundary conditions. Figure 3.4 gives the fully implicit algorithm investigated as part of this work for the solution of free-surface Stokes-flow problems. At each iteration of the nonlinear solver, the solution from

1. Set $\mathbf{u}^* = \mathbf{u}^{(n)}$.
2. Generate a new the free surface $\mathbf{s}^{(n+1)}$, using $\mathbf{u}^{(n)}$, $\mathbf{u}^*$, $\mathbf{s}^{(n)}$ and (3.66).
3. Generate a new mesh $\mathbf{m}^{(n+1)}$ by updating $\mathbf{m}^{(n)}$ using the techniques described in Section 2.5.
4. Assemble the matrix $\mathbf{K}$ on $\mathbf{m}^{(n+1)}$.
5. Compute preconditioner for $\mathbf{K}$.
6. Assemble $\mathbf{b}$.
7. Solve the linear system (3.64) to give $\mathbf{x}^{(n+1)}$.
8. If $|\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}|_\infty < tol_1$ and $|\mathbf{s}^{(n+1)} - \mathbf{s}^{(n)}|_\infty < tol_2$ then proceed to next time step.
9. Set $\mathbf{u}^* = \mathbf{u}^{(n+1)}$.
10. Go to step 2.

Figure 3.4: A fully implicit algorithm for free-surface Stokes-flow problems.

the previous iteration, or from the end of the previous time step, is used as an initial estimate. The algorithm terminates when the maximum change in any component of the pressure or velocity falls below a prescribed absolute tolerance tol_1, and the maximum free-surface nodal displacement falls below an absolute tolerance tol_2.

Note that, in principle, if one sets $\theta = 0.5$ in (3.66) then the fully implicit scheme will have $O(k^2)$ temporal accuracy for velocity, as compared to the $O(k)$ accuracy of the semi-implicit scheme described in Section 3.7. It does however appear to suffer from convergence problems similar to those observed with the fully implicit scheme for the Navier-Stokes equations described above. Consequently, little use has been made of it in the present work.

3.9 Notes on alternative nonlinear solution methods

While the semi-implicit schemes described above may be used for many problems, the time-step constraints found to be necessary result in the need to take many small time steps when a mesh is fine or the solution velocities are large. Fully implicit methods would appear to be the answer to this problem, since in principle they are

not bound by time-step constraints. However, as already mentioned, when functional iteration is employed in order to solve the nonlinear problems that result from these schemes, convergence is only reliable when relatively short time steps are employed. If large time steps are attempted then convergence may be arbitrarily slow, or indeed it may not occur at all. In such circumstances the solution computed at alternate iterations of the nonlinear solver typically cycles between two or more distinct basins of attraction. It thus appears that the fully implicit schemes described above are not globally convergent. Since, even when functional iteration is successful each implicit time step requires typically ten nonlinear iterations to reach convergence, it is unclear whether any advantage is gained in practice by the use of fully implicit schemes when they are solved using this approach.

An alternative approach to solving nonlinear systems of equations, such as those arising from free-surface Navier-Stokes or Stokes problems, involves the use of Newton's method, or some modification of it [36]. To allow Newton's method to be applied, the problem must be reformulated to include the locations of the free-surface nodes as variables. This results in a somewhat larger system of equations, for which the Jacobian is non-symmetric.

Since Newton's method is quadratically convergent when the initial estimate of the solution lies within its region of convergence, the method is potentially very efficient, particularly where the system must be solved accurately. Other iterative schemes [36] that are globally convergent, if only linearly so, are in general needed to obtain an initial estimate of the solution that lies within the region of convergence [17]. In the case of time-dependent problems it may be possible to obtain a suitable initial guess for Newton's method using explicit predictors based on solutions computed at earlier time steps, though since the motivation for employing implicit schemes is generally to allow large time steps to be employed, the accuracy of such predictors cannot necessarily be relied upon.

The main difficulty with implementing Newton's method is that the Jacobian matrix for the system must be assembled at least once, and possibly a number of times. Assembly of the Jacobian is potentially a very expensive operation, even if done numerically, since for a nonlinear moving-mesh problem every flow variable is coupled to the position of every free-surface node. Thus the Jacobian contains a dense block that is expensive to assemble. It is however noted that techniques for approximating the Jacobian at reduced cost might prove useful. For example, one approach to assembling the dense block would involve perturbing each free-surface node in turn, updating the interior mesh using a Laplacian-smoothing scheme in

each case, but only assembling entries in the Jacobian that correspond to nodes that lie close enough to the perturbed free-surface node to be significantly affected.

The inversion of the Jacobian, required by Newton's method, may be achieved by employing iterative methods such as GMRES [94], though note that now the Jacobian will be non-symmetric, and thus convergence may be problematic. In the discrete case the uniqueness of a solution of a system of nonlinear equations depends on the Jacobian being non-singular [36]. This must be verified for the particular discrete formulation employed and for the problem under consideration.

3.10 The conjugate residual method

The solutions of the discrete linear sub-problems required by the Stokes and Navier-Stokes solvers described above may be obtained using either direct or iterative methods. For large systems of equations naive implementations of direct methods such as those based on Gaussian-elimination require large amounts of memory for the storage of the fill-in generated and are thus prohibitively expensive. While the large storage requirements of direct methods can be considerably reduced by the use of bandwidth-reducing node reorderings [30] and sophisticated memory management schemes [43], iterative methods typically require considerably less storage, and for sufficiently large systems, are nowadays generally recognised to be more efficient than direct methods when applied to problems arising from finite element discretisations of partial differential equations. A further advantage that iterative methods have over direct methods is that, in a sense, direct methods always attempt to solve a problem to machine precision regardless of whether this is actually required. Iterative methods on the other hand may be terminated once the desired level of accuracy has been obtained.

Where a time-dependent flow is being modelled and short time steps are in use iterative methods are potentially highly cost-effective for two reasons. The first is that since accurate predictors for the desired solution may be computed, the iterative solver may require very few iterations in order to improve the solution sufficiently to satisfy the convergence criteria. Direct solvers cannot take advantage of predictors. The second reason is that in such circumstances the possibility of reusing preconditioners arises, and thus the cost of computing a good preconditioner may be offset against savings made over a number of time steps.

The iterative method employed here is the conjugate residual (CR) or MINRES method [3], a variant of the popular conjugate gradient (CG) method that is di-

rectly applicable to symmetric indefinite systems of linear equations. The algorithm implemented here is the more efficient ORTHOMIN form of the conjugate residual method, which uses a two-term recurrence relationship to generate a sequence of orthogonal search directions. The related algorithm GMRES [94], which is commonly employed for the solution of indefinite and non-symmetric problems, requires the storage of a much larger number (typically 20—50) of previous search directions and thus has considerably larger storage requirements and execution costs.

1. $\mathbf{x} = \mathbf{x}_0$
2. $\mathbf{r} = \mathbf{b} - \mathbf{K}\mathbf{x}$
3. $\mathbf{p} = \mathbf{M}^{-1}\mathbf{r}$
4. $\mathbf{w} = \mathbf{K}\mathbf{p}$
5. $\mathbf{y} = \mathbf{M}^{-1}\mathbf{w}$
6. $\mathbf{s} = \mathbf{p}$
7. If $|\mathbf{r}|_2 < \epsilon$ then finished.
8. $\rho = \mathbf{y} \cdot \mathbf{w}$
9. $\alpha = (\mathbf{s} \cdot \mathbf{w})/\rho$
10. $\mathbf{x} = \mathbf{x} + \alpha\mathbf{p}$
11. $\mathbf{r} = \mathbf{r} - \alpha\mathbf{w}$
12. $\mathbf{s} = \mathbf{s} - \alpha\mathbf{y}$
13. $\mathbf{z} = \mathbf{K}\mathbf{s}$
14. $\beta = -(\mathbf{z} \cdot \mathbf{y})/\rho$
15. $\mathbf{p} = \mathbf{s} + \beta\mathbf{p}$
16. $\mathbf{w} = \mathbf{z} + \beta\mathbf{w}$
17. $\mathbf{y} = \mathbf{M}^{-1}\mathbf{w}$
18. Go to step 7.

Figure 3.5: The preconditioned conjugate residual method: implementation of the ORTHOMIN algorithm for the problem $\mathbf{K}\mathbf{x} = \mathbf{b}$.

The implementation of the preconditioned conjugate residual (PCR) algorithm employed in the current work is shown in Fig. 3.5. It is based on that described by Ramage and Wathen [82]. The inputs to the solver are a matrix $\mathbf{K}$, an initial estimate of the solution $\mathbf{x}_0$, an absolute convergence tolerance ϵ, and a preconditioning operator $\mathbf{M}^{-1}$. The implementation requires a single sparse matrix-vector product (step 13), a single application of the preconditioner $\mathbf{M}^{-1}$ (step 17), and the equivalent of eight vector scalar-products per iteration. The algorithm terminates when the L_2 norm of the residual vector falls below the prescribed tolerance, and returns the solution in the vector $\mathbf{x}$.

In the simplest cases, the preconditioning operator involves multiplication of a vector by a diagonal matrix $\mathbf{M}^{-1}$. More sophisticated preconditioners may also be employed, involving, for example, incomplete Cholesky [32] or incomplete LU [94, 20] factorisations. In these cases each preconditioning step will require the solution of a linear-algebraic problem, though since the matrix is already factorised, this may be achieved efficiently. A third type of preconditioner [10, 94] involves the computation of an approximate inverse of the matrix, i.e. $\mathbf{M}^{-1} \approx \mathbf{K}^{-1}$. The latter types of preconditioner have a computational cost associated with them that is proportional to the number of non-zero entries in the approximate factorisation or inverse.

Note that, theoretically, the ORTHOMIN method is applicable only when the matrices $\mathbf{K}$ and $\mathbf{M}$ are both Hermitian positive-definite, and that the more robust ORTHODIR [3] method must be employed where $\mathbf{K}$ is indefinite. In practice the ORTHOMIN algorithm has been found to converge for the indefinite matrices considered here, and thus it has not been necessary to employ the less efficient ORTHODIR form of the algorithm as a backup, as recommended in [82]. That is to say, in the work described here, no situation has been observed in which ORTHODIR converges but ORTHOMIN does not.

The numbers of conjugate residual iterations required to solve the problems considered here are typically considerably larger than the numbers quoted in the literature for similarly sized problems. In part this is due to the fact that here the problems must be solved to a high degree of accuracy in order to conserve mass. As a convenient rule of thumb, the number of iterations required is roughly proportional to the number of bits required in the solution. Consequently, reducing the size of the convergence tolerance ϵ will increase the number of iterations required approximately logarithmically.

A considerable difficulty faced when relying on iterative solution methods is that no efficient reliable method is available for estimating the condition number of the linear problem at each time step. Thus, it is hard to arrange for the convergence tolerance ϵ to be automatically varied as a problem progresses so as to avoid unnecessary work being performed. Estimates of the condition number for some of the smaller problems have however been obtained using the NAG routine F02WEF [35]. These suggest that even for the smallest meshes considered here the condition number is of the order of 10^6. It thus appears likely that for all the meshes employed in this work the condition number lies in the range 10^6—10^8. Thus a value of $\epsilon = 10^{-10}$ appears to be the largest that guarantees at least two digits of accuracy in each component of the solution. Consequently, in the current work, a fixed absolute

tolerance of $\epsilon = 10^{-10}$ is employed as the convergence criterion for the solution of all linear systems unless otherwise stated. Practical experience, however, suggests that error estimates such as these may well be excessively pessimistic.

3.11 Node re-ordering

The use of bandwidth-reducing node reorderings as a means of improving both the efficiency and accuracy of incomplete LU factorisations is a well-known technique [30]. A good node ordering for a sparse linear-algebraic problem will result in a matrix in which all the non-zero entries lie within a narrow diagonal band. Since, in Gaussian-elimination-based methods, fill-in can only occur within the band, bandwidth minimisation reduces the number of fill-in entries that must be either stored or discarded. Thus, where an incomplete LU factorisation of a matrix is required, a bandwidth-reduced node ordering will simultaneously reduce storage costs and improve the accuracy of the resulting factorisation. For finite element problems the minimum bandwidth achievable is primarily dictated by domain geometry. One popular node-ordering scheme is the Reverse Cuthill-McKee (RCM) ordering [30]. While numerous, alternative schemes for computing node reorderings exist, none would appear to have any clear advantage over the Cuthill-McKee method where arbitrary unstructured meshes are involved [30].

The algorithm shown in Fig. 3.6 is based on that given in [30], with modifications for the quadratic elements employed here. It numbers the N nodes forming a two-dimensional mesh starting at the vertex $firstnode$. The resulting Cuthill-McKee ordering is then reversed before being used to number the degrees of freedom associated with each individual node in turn. The set of neighbours of a node is defined to be all the nodes in the elements adjacent to the node in question, with the exception of the node itself. The degree of a node is simply the number of neighbours it possesses. Note that when computing the degree and the neighbours of a node both edge and vertex nodes must be included, i.e. every node has five neighbours in each element in which it occurs.

Figure 3.7 shows the sparsity patterns associated with both the original and the RCM orderings for the Stokes-flow test problem described in Section 2.1.4, when assembled on the mesh illustrated in Fig 2.4. Note that in the original ordering the u degrees of freedom are assembled first, followed by the v degrees of freedom and finally the pressure degrees of freedom. Thus a zero block may be seen in the lower right corner of Fig. 3.7(a). In Fig. 3.7(b) the effects of the reordering can clearly be

```
 1.  do i := 1 to N
         find neighbours of node i and store as
         neighbours(i,j).
     enddo
 2.  do i := 1 to N
         find degree of node i and store as degree(i)
     enddo
 3.  Find maximum and minimum node degrees maxdeg and mindeg.
 4.  do i := 1 to N do
         visited(i) := false
     enddo
 5.  next := 1
 6.  ptr := 1
 7.  ordering(next) := firstnode
 8.  visited(firstnode) := true
 9.  cnode := ordering(ptr)
10.  ptr := ptr + 1
11.  do d := mindegree to maxdegree
         do j := 1 to degree(cnode)
              node := neighbour(cnode,j)
              if (degree(node) = d) and (not visited(node)) then
                   visited(node) := true
                   next := next + 1
                   ordering(next) = node
              endif
         enddo
     enddo
12.  if next < N then go to step 9.
13.  Reverse the ordering.
14.  Number degrees of freedom at each node (velocity first,
     then pressure) using the reverse node ordering.
15.  Finished.
```

Figure 3.6: Reverse Cuthill-McKee node re-ordering algorithm for V6-P3 elements.

seen. The non-zero entries in the matrix now all lie within a relatively narrow band. Since elimination proceeds from the top row down, the re-ordered matrix is clearly far superior, as much of the "fill-in" will now be added to entries in the matrix which are already non-zero.

While the reverse Cuthill-McKee ordering is capable of producing node orderings for which the bandwidth is close to optimal, the results are to a large extent

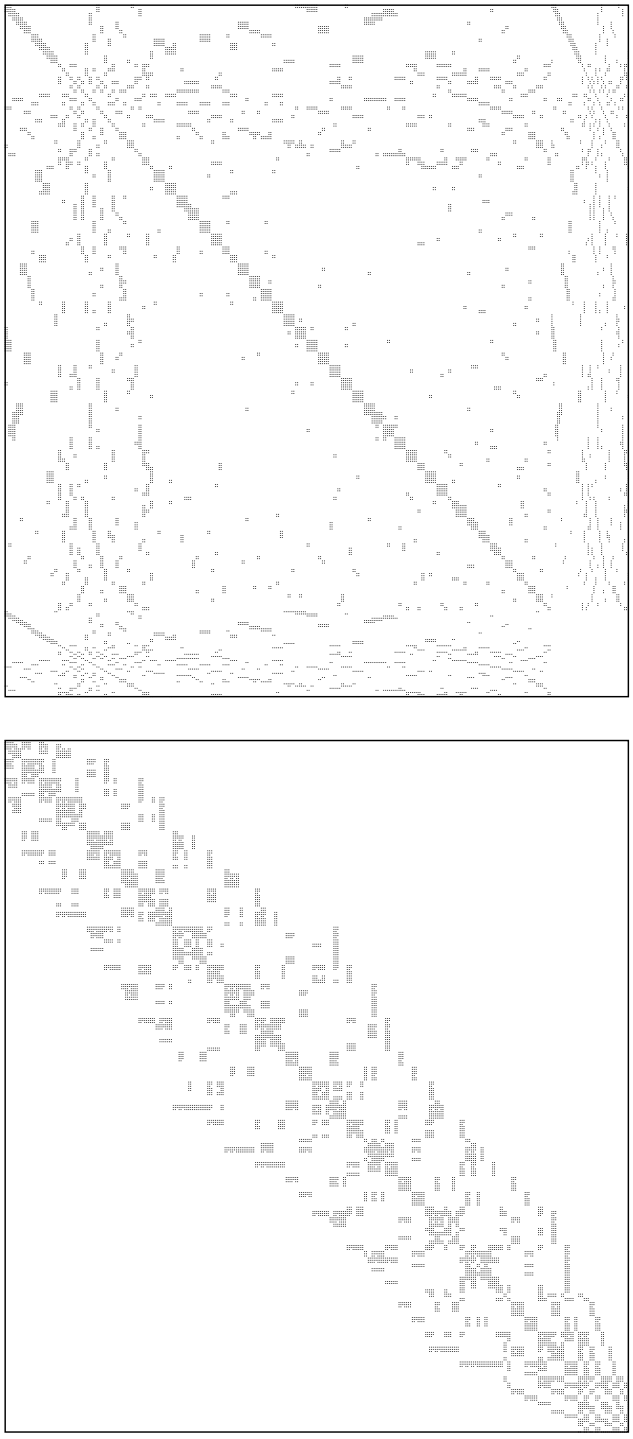


Figure 3.7: Finite element stiffness-matrix sparsity patterns for mesh 2: (a) original ordering of degrees of freedom; (b) Reverse Cuthill-McKee ordering.

dependent upon the choice of initial node and the geometry of the problem. A useful rule of thumb is that: if the reverse Cuthill-McKee ordering is used with an appro-

priate choice of initial node, then long thin domains tend to give rise to narrower bands, and thus require less storage for their preconditioners than ones that are roughly circular. While automatic methods for selecting optimal initial nodes for a given ordering scheme have been described [30], in practice a near optimal choice is often fairly easy to make by visual inspection of the mesh and by employing an understanding of the Cuthill-McKee algorithm. Typically, a reasonable choice of initial node will be one lying on the boundary, in a corner, or at one end of a long thin domain. Since the costs associated with the Cuthill-McKee algorithm are fairly small, it appears that automatic methods for the optimal choice of the initial node might well be practical for time-dependent free-surface problems.

3.12 Preconditioning

The use of a good preconditioner can markedly improve the efficiency of an iterative solver, such as the conjugate residual method, in that it considerably reduces the number of iterations required to achieve a given level of accuracy. Furthermore, where the matrix involved is ill-conditioned, or non-symmetric, the use of a good preconditioner may be essential to ensure the solver converges at all [94].

One family of solution techniques that has been extensively studied in recent years are the multigrid methods [14, 15, 120, 86, 97]. Although potentially very efficient, such methods are relatively complicated to implement, requiring a hierarchy of meshes of different resolutions, and operators for transferring residuals and corrections between the meshes. The main difficulty in implementing a multigrid solver for free-surface problems results from the requirement that a hierarchy of meshes, of differing resolutions, must be employed. While techniques have been described for automatically coarsening unstructured meshes in order to generate a hierarchy of meshes from the finest mesh [6], applying multigrid techniques in such circumstances requires boundary conditions on the coarser meshes to be estimated in some fashion, since edge nodes will not necessarily correspond to nodes in finer meshes in the hierarchy. While multigrid techniques have been described for problems where the domain is convex [95], concave domains introduce additional complications, since where a concave boundary is present some of the edge nodes forming the coarser meshes will actually lie outside the true domain (as defined by the finest mesh). Thus it is unclear as to whether the convergence properties of the multigrid methods apply when non-homogeneous natural boundary conditions are imposed, as is necessary in surface-tension driven problems. While algebraic or black-box multigrid

techniques have been developed [14] which avoid the need to maintain a hierarchy of meshes, again the published schemes have typically been for regular meshes with homogeneous boundary conditions, making them difficult to adapt to free-surface problems. Recently, however, it has been demonstrated [94] that the use of iterative methods with high-quality preconditioners, based on incomplete LU factorisations, may result in solvers that are comparable in efficiency to multigrid schemes. Since such preconditioners are relatively simple to compute it was this approach that was chosen for further investigation as part of the current work.

In the current context a preconditioner for the solution of a system of N linear-algebraic equations, of the form

$$\mathbf{K}\mathbf{x} = \mathbf{b}, \tag{3.68}$$

is a matrix or product of matrices $\mathbf{M}$, chosen to have similar spectral properties to the matrix $\mathbf{K}$ [119]. A preconditioner is normally applied at each iteration of an iterative solver and, for the implementation of the conjugate residual method shown in Fig. 3.5, requires the solution of a linear-algebraic system of the form

$$M\mathbf{y} = \mathbf{w}. \tag{3.69}$$

Clearly, it is advantageous that the system (3.69) should be easy to solve.

The time-discretisations of the Navier-Stokes equations, based on (3.63), require the solution of linear-algebraic systems in which the matrix is formed by taking a linear combination of the finite element stiffness and mass matrices. For short time steps such problems are dominated by the contribution from the mass matrix. Since the mass matrix is symmetric positive-definite when the Galerkin method is employed, it is somewhat easier to 'invert' than the corresponding indefinite stiffness matrix. Thus it appears reasonable to suppose that any preconditioning scheme that performs well for the stiffness matrix will perform at least as well when the problem is dominated by a contribution from the mass matrix.

3.12.1 Diagonal preconditioning

The simplest preconditioning technique, known as diagonal preconditioning, requires a diagonal matrix $\mathbf{M}$ to be selected, where $\mathbf{M}$ is of full rank. Since a diagonal matrix may be trivially inverted, the application of the preconditioner at each iteration of the solver requires only the product of a diagonal matrix and a vector to be computed, an operation of cost $O(N)$ where there are N degrees of freedom. While

diagonal preconditioning is cheap to implement, a large number of iterations of the conjugate residual solver are often found to be necessary to obtain an accurate solution. Thus the correct choice of preconditioner is important.

In order to allow comparison of preconditioners the steady-state Stokes-flow problem introduced in Section 2.1.4 is here revisited. Table 3.1 lists V, the number of vertices, and N, the number of unknowns, together with the values of K_{tol} and h_{max} employed for each of the ungraded meshes considered. The values of h_{max} were chosen so that the maximum edge length in the mesh decreased proportionately each time k_{tol} was reduced. Note that meshes $1, 3, 5$ and 7 correspond to meshes $1, 2, 3$ and 4 in Section 2.1.4. Table 3.2 gives the numbers of iterations required to solve the test problem on each of the meshes, together with the run time in seconds, for three different diagonal preconditioners, $\mathbf{M}_1$, $\mathbf{M}_2$ and $\mathbf{M}_3$. If no reordering of the unknowns is performed then these may be defined as follows

$$\mathbf{M}_1 = \begin{pmatrix} & \mathbf{I} & \\ & & \end{pmatrix}, \tag{3.70}$$

$$\mathbf{M}_2 = \begin{pmatrix} diag(\mathbf{A}_{11}) & & \\ & diag(\mathbf{A}_{22}) & \\ & & \mathbf{I} \end{pmatrix}, \tag{3.71}$$

$$\mathbf{M}_3 = \begin{pmatrix} diag(\mathbf{A}_{11}) & & \\ & diag(\mathbf{A}_{22}) & \\ & & diag(M_p) \end{pmatrix}. \tag{3.72}$$

The diagonal matrices $diag(\mathbf{A}_{11})$ and $diag(\mathbf{A}_{22})$ are formed from the leading diagonal of the viscous block of the Stokes operator, while $diag(\mathbf{M}_p)$ is formed from the leading diagonal of the pressure-mass matrix defined by

$$[\mathbf{M}_p] = \int_\Omega l_i l_j d\Omega \qquad \text{i,j} = 1, \ldots, \text{M}, \tag{3.73}$$

where i and j range over the M pressure basis functions. The identity matrix $\mathbf{I}$ is chosen in each case to give a matrix of full rank. The conjugate residual solver was halted when the L_2 norm of the residual vector had been reduced to below 10^{-10}. Initial residual norms were in the range 1×10^{-1} to 3×10^{-1}. The times given in Table 3.2 were obtained on a 180MHz Silicon Graphics R5000 workstation with 96Mb of main memory and 32Kb of primary cache, employing statistical processor-counter

Mesh	V	N	K_{tol}	h_{max}
1	41	327	0.5000	0.2500
2	71	581	0.3540	0.1770
3	161	1367	0.2500	0.1100
4	310	2676	0.1770	0.0780
5	663	5805	0.1250	0.0525
6	1293	11411	0.0880	0.0371
7	2594	23032	0.0625	0.0260

Table 3.1: Meshes for preconditioning studies (ungraded): statistics.

Mesh	M_1 Itns	M_1 Time	M_2 Itns	M_2 Time	M_3 Itns	M_3 Time
1	613	0.60	408	0.39	270	0.29
2	884	1.86	640	1.33	410	0.93
3	1427	9.46	1027	6.78	695	4.68
4	2214	34.60	1562	25.02	1151	19.53
5	3452	137.25	2335	95.97	2048	85.10
6	5296	462.54	3619	311.85	3289	281.00
7	7834	1445.71	5401	1013.18	5726	1054.00

Table 3.2: Diagonal scaling, PCR iterations to convergence and run time (in seconds): M_1 — no preconditioning; M_2 — preconditioning using viscous terms only; M_3 — preconditioning using viscous terms and pressure-mass matrix.

sampling. For the smaller meshes timings were obtained by solving the linear-algebra problem a number of times, so as to allow more accurate measurements to be taken. The timings must in any case be interpreted with caution, and only the first digit should be taken as significant for the smaller meshes. As Table 3.2 shows, $\mathbf{M}_2$ and $\mathbf{M}_3$ considerably reduce the number of iterations required to attain convergence. While for the smaller problems $\mathbf{M}_3$ significantly out performs $\mathbf{M}_2$, for the finest mesh the situation is reversed; $\mathbf{M}_3$ requiring more iterations than $\mathbf{M}_2$ [1]. For $\mathbf{M}_3$

[1]Interestingly, if the pressure-mass matrix $\mathbf{M}_P$ in (3.72) is replaced by its mass-lumped form

$$[\mathbf{M}_{LP}] = \int_\Omega l_i d\Omega \qquad \text{i,j} = 1, \ldots, \text{M}, \tag{3.74}$$

then the situation is reversed. For large problems $\mathbf{M}_3$ now requires fewer iterations than $\mathbf{M}_2$ (5070 for mesh 7) but for small problems requires more. For a regular mesh, using the diagonal of the lumped pressure-mass matrix results in a preconditioner in which the entries for the continuity equations are equal to the area of an element. While Wathen and Silvester [119] describe both types of preconditioner they make no reference to the poor performance of preconditioners based on the diagonal of the pressure-mass matrix for fine meshes. Further work thus appears necessary to clarify this issue. The form of $\mathbf{M}_3$ based on the diagonal of the pressure-mass matrix is preferred in the current work since it is more efficient for small meshes.

it can be seen that as the number of unknowns grows, the number of iterations required per unknown falls considerably, from around 0.8 for mesh 1 to around 0.25 for mesh 7. Analysis of the results shown suggests that the total number of iterations required scales approximately as $N^{0.75}$ for all three preconditioners, but that run time scales as approximately N^2. While the approximately $O(N^{0.75})$ iteration count was expected for $\mathbf{M}_3$ [119], the failure of the favourable exponent to translate into a hoped-for $O(N^{1.75})$ run time was not. Since the number of floating-point operations per solve is certainly proportional to $N^{1.75}$ the failure to achieve an $O(N^{1.75})$ run time must, presumably, be due to an increase in the frequency of cache misses as the program's data structures grow in size.

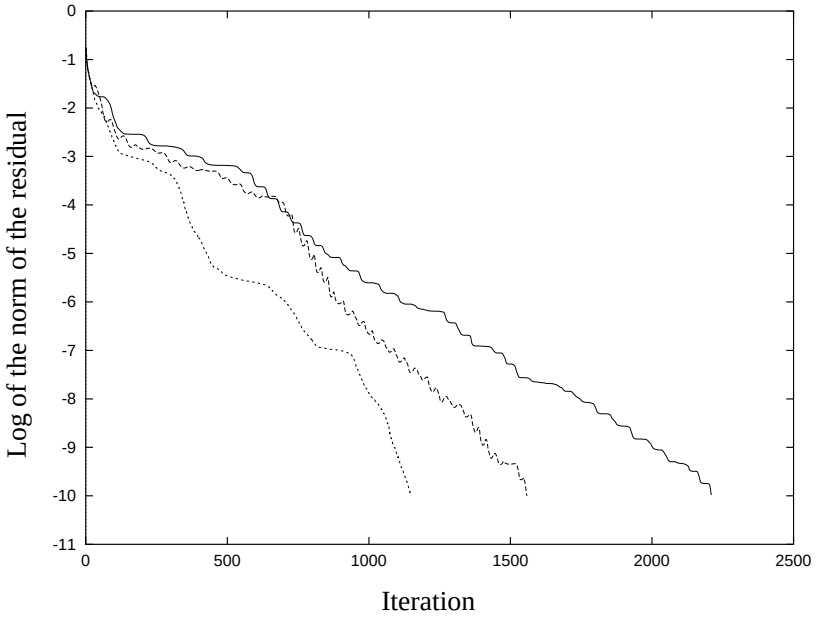


Figure 3.8: Conjugate residual solver convergence history employing diagonal preconditioning: base-10 logarithm of the L_2 norm of the residual vector at each iteration: —— M_1, - - - - M_2, $\cdots\cdots$ M_3.

For comparison, a standard sparse direct solver from the NAG library [35] (F04BRF/F01AXF) required approximately 15 seconds to solve the problem on mesh 3, the run time nominally scaling as $O(N^2)$. Thus, even for a small problem such as this, the conjugate residual method appears to be considerably more efficient than the direct solver, being approximately four times as fast.

Figure 3.8 gives the convergence histories for the three diagonal preconditioners, when applied to the problem on mesh 4; the curves showing the base-10 logarithm of the L_2 norm of the residual vector at the end of each iteration. As can be seen, the rate of reduction of the residual per iteration is not constant, and occasionally the solver appears nearly to stall. Indeed for $\mathbf{M}_2$ the residual actually increases at

times. There is, however, no real tendency for the rate of reduction in the residual to slow as the residuals become small; indeed, experience has shown that convergence to tolerances near to machine precision is attainable with no loss of performance.

3.12.2 ILUT preconditioning

Recent work by Saad and others [94] has highlighted the possibility that, by employing sufficiently accurate preconditioners, solutions to linear-algebraic systems derived from finite element problems may be computed at a cost comparable to that of multigrid methods, i.e. at cost approaching $O(N)$. Clearly such efficiency is highly desirable in a solver for large systems of equations. While here only one of the many approaches is explored, the reader's attention is drawn to the existence of numerous alternative schemes for computing preconditioners e.g. 'element-by-element' methods [107, 63], independent set orderings [93] and graph-based factorisation methods [5].

The ILUT approach developed by Saad is based on an incomplete-LU factorisation of the matrix $\mathbf{K}$ [94]. This involves finding a pair of sparse lower- and upper-triangular matrices $\mathbf{L}$ and $\mathbf{U}$ such that

$$\mathbf{M} = \mathbf{LU} \approx \mathbf{K}. \tag{3.75}$$

Given the special structure of $\mathbf{L}$ and $\mathbf{U}$ (3.69) may be solved directly by backward and forward substitution, a process having roughly the same computational cost as a sparse matrix-vector product.

The computation of an ILU preconditioner requires a considerable investment of work but potentially results in a very large decrease in the number of iterations required. Furthermore, experience suggests that it is often possible to reuse a preconditioner over many (perhaps 20 to 50) time steps, and indeed within a time step where the iterative solution of a nonlinear problem is sought. Thus, where an ILU preconditioner need only be recomputed infrequently the preconditioned conjugate residual method is potentially highly efficient.

An important advantage of Saad's approach [94] is that it may be directly applied to indefinite and non-symmetric problems, though there is in general no guarantee of the accuracy of the LU factorisation, or of the stability of the resulting lower- and upper-triangular systems in these cases [20]. Saad's method is also convenient in that it may be applied directly to problems for which additional algebraic constraints must be applied, such as those required for the free-surface location at inflow and

outflow boundaries. It should be emphasised, however, that little theory presently is available to guide investigations in this active area of research. In this work emphasis has been placed on obtaining practical experience of the use of such methods, and the gathering of empirical evidence as regards their efficiency for problems of the form (3.63).

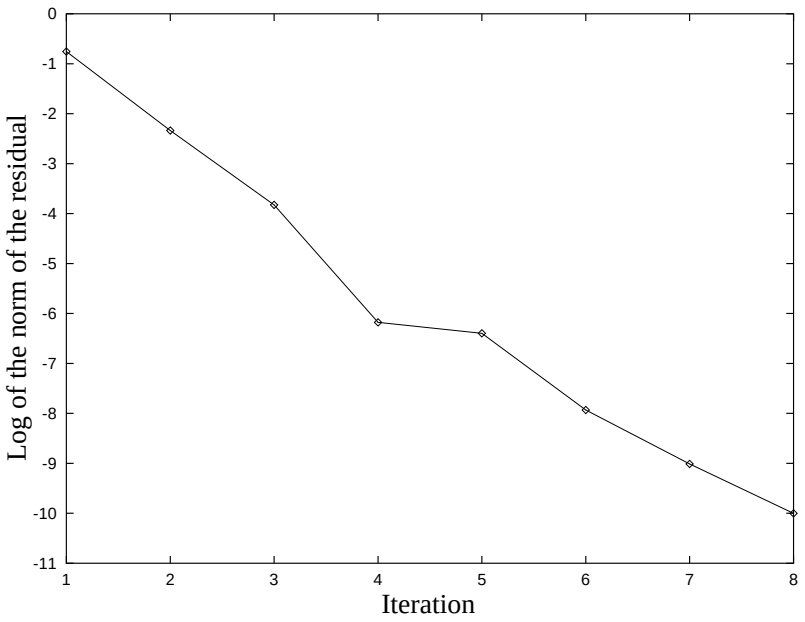


Figure 3.9: Conjugate residual solver convergence history for mesh 4: base-10 logarith of the L_2 norm of the residual vector at each iteration using an ILUT preconditioner.

The simplest form of incomplete LU factorisation is the ILU(0) factorisation [94], so-named because all fill-in generated during the process is discarded. More sophisticated approaches allow for the retention of selected components of the fill-in, e.g. all fill-in entries above a given threshold might be retained. Saad [94] terms this an ILUT preconditioner, and it is this approach which is considered here. The routine employed here, ILUT, is taken from Saad's SPARSKIT package [92]. It allows control of the amount of fill-in to be exercised through two parameters, $lfil$ and *droptol*. Two rules are used to decide whether to keep a particular component of fill-in; $lfil$ specifies the maximum number of fill-in entries to be retained in each row of $\mathbf{L}$ and $\mathbf{U}$, while *droptol* specifies a threshold. If a given fill-in entry has a smaller absolute magnitude than this threshold then it will be dropped.

In the following experiments an incomplete factorisation was computed using Saad's routine ILUT after the unknowns had been reordered using the reverse Cuthill-McKee algorithm described in Section 3.11. Note that ILUT is not guaranteed to be stable for indefinite matrices; instead normally an implementation

with pivoting would be considered necessary [20]. Despite this, in the current work Saad's non-pivoting ILUT routine has been found to be most satisfactory, and when combined with the conjugate residual solver employed here, out performs Saad's preconditioned GMRES solver (PGMRES) with a preconditioner computed using pivoting.

Mesh	$lfil$	$droptol$	nza	$nzlu$
1	60	1×10^{-4}	8520	20241
2	100	1×10^{-5}	15488	55372
3	200	5×10^{-6}	37612	163398
4	200	5×10^{-6}	74716	474501
5	200	5×10^{-6}	164000	1336960
6	300	2×10^{-6}	324472	3728512

Table 3.3: ILUT preconditioning: problem and preconditioner statistics.

Figure 3.9 shows the convergence history for the solution of the test problem on mesh 4. As may be seen, the use of the ILUT preconditioner greatly reduces the number of iterations required compared to the best diagonal preconditioner $\mathbf{M}_3$. Table 3.3 gives details of the ILUT preconditioners selected for each of the six meshes considered. The columns are as follows: $lfil$, the maximum number of entries per row in the L and U factors; $droptol$, the absolute threshold for dropping fill-in; nza, the number of non-zero entries in the original matrix; $nzlu$, the number of non-zero entries in L and U. The values of $lfil$ and $droptol$ were chosen by trial and error. Selecting a smaller value for $lfil$, or a larger one for $droptol$, will in general result in a smaller preconditioner, but will require a larger number of iterations. If an attempt is made to employ too small a value of $lfil$, or too large a value of $droptol$ then convergence will not occur.

No real attempt was made to optimise the values of $lfil$ and $droptol$ given in Table 3.3. For time-dependent problems, in which the size and difficulty of the problem changes with time, the choices of $lfil$ and $droptol$ must be made with care. More sophisticated approaches than that employed here might involve automatic adaptive selection of $lfil$ and $droptol$. Thus the number of iterations required might be monitored, and the value of $droptol$ decreased when the number of iterations grows too large, but increased whenever the number of iterations falls. Similarly $lfil$ might also be varied automatically. Note, however, that for time-dependent problems, in which the intention is to reuse the preconditioner over a number of time steps, the choice of an optimal preconditioner is more difficult, since as a

preconditioner ages the number of iterations required grows approximately linearly. Indeed, if too crude a preconditioner is employed then it will require recomputing at almost every time step. Further work on this issue is thus necessary if ILUT preconditioners are to be employed efficiently and reliably for general free-surface problems.

Mesh	Iterations	*ilut*	*lusol*	*prodmv*	Total time
1	10	0.059	0.040	0.015	0.114
2	4	0.300	0.071	0.015	0.386
3	4	1.300	0.240	0.038	1.578
4	7	5.100	0.980	0.140	6.220
5	14	24.000	4.800	0.400	29.200
6	9	110.000	9.900	0.700	120.600

Table 3.4: ILUT preconditioning: iterations and timings (times in seconds).

For mesh 7, no values of $lfil$ and *droptol* were found to result in a preconditioner that lead to convergence within the memory available. For this mesh it is estimated that the minimum storage required for a satisfactory preconditioner would be around 80Mb, i.e. $nzlu \approx 5 \times 10^6$.

Table 3.4 gives the following figures for each of the problems: the number of iterations required; *ilut*, the time spent computing the ILUT factorisation; *lusol*, the time spent in upper- and lower-triangular solves; *prodmv*, the time spent in performing the matrix-vector products required by the conjugate residual algorithm, and finally, the total solver run time. As can be seen, the number of iterations is now essentially independent of the number of unknowns in each problem. A simple analysis of column 5 of Table 3.3 shows that the number of non-zero entries in the preconditioner is roughly proportional to $N^{1.5}$. Thus, on an ideal machine, ignoring factorisation time, one would expect run time to be proportional to $N^{1.5}$, since then *lusol* will dominate the run time for sufficiently large problems. As column 6 of Table 3.4 shows, in practice overall run time is roughly $O(N^2)$, i.e. the same order as was observed for diagonal preconditioners. Figure 3.10 shows the run time for each of the three preconditioners $\mathbf{M}_1$, $\mathbf{M}_3$ and $ILUT$ when applied to the test problem on meshes 1 through 6. As may be seen, even if factorisation is performed at every time step, the ILUT preconditioner is still around 50% faster than the best diagonal preconditioner $\mathbf{M}_3$ for all the problems considered.

In practice it is often possible to reuse a preconditioner over many time steps, particularly when the mesh changes little from step to step as is normally the case

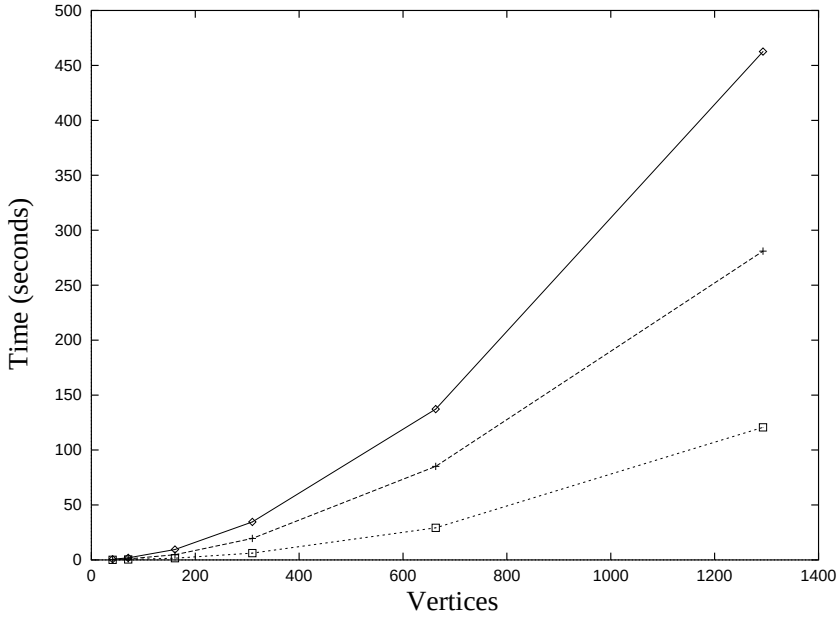


Figure 3.10: PCR run times for three different preconditioners: - - - - M_1; – – – M_3; —— ILUT.

when a semi-implicit scheme is employed. In such circumstances the preconditioner need be recomputed only when the number of conjugate residual iterations exceeds a prescribed limit. Indeed, ultimately, as a steady-state solution is approached, the same preconditioner may be employed *ad infinitum* and thus the approach becomes particularly efficient.

While hardware issues appear to prevent the hoped-for theoretical run times from being achieved, it is perhaps worth bearing in mind that such considerations no doubt apply equally when direct solvers are employed.

3.12.3 Graded meshes

The above experiments all involve meshes that are quasi-regular, i.e. edge length is approximately constant throughout the mesh. In contrast the mesh generation techniques described in Chapter 2 produce graded meshes in which edge length may vary by up to two orders of magnitude, and thus element area by up to four orders of magnitude. Consequently, one would expect the size of comparable entries in the finite element stiffness matrix to similarly vary by up to four orders of magnitude. It is clearly necessary to confirm that the iteration counts and run time estimates discussed above also apply for such graded meshes.

Table 3.5 gives statistics for five graded meshes, numbered 8 to 12, employed for this purpose, together with the iteration counts and run times observed when the

diagonal preconditioner $\mathbf{M}_3$ was employed. Figure 3.11 shows a selection of these meshes. Note that here h_{max} is held fixed while k_{tol} alone is varied. Mesh 12 has the greatest disparity in element size with a ratio of roughly 1:25 between the smallest and largest elements' areas.

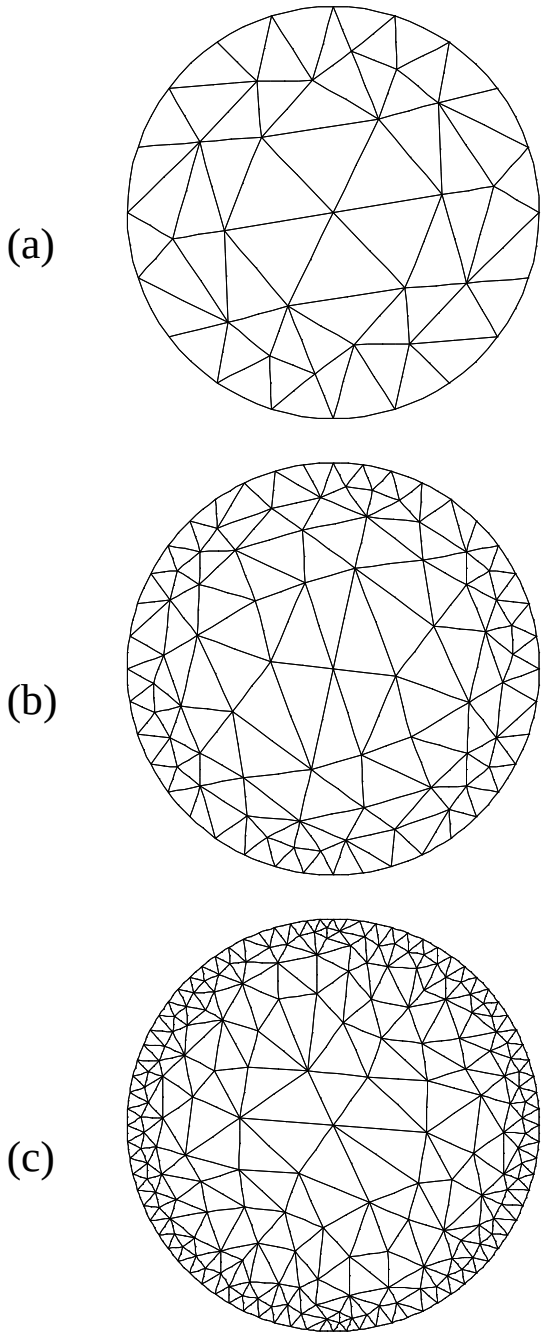


Figure 3.11: Selected graded meshes for preconditioning study: (a) mesh 8; (b) mesh 10; (c) mesh 12.

Mesh	k_{tol}	h_{max}	Vertices	Unknowns	Iterations	Time
8	0.400	0.5	41	319	274	0.284
9	0.238	0.5	60	474	369	0.679
10	0.200	0.5	99	801	558	1.979
11	0.141	0.5	165	1363	784	5.540
12	0.100	0.5	237	1963	983	11.070

Table 3.5: Graded mesh statistics, iterations and timings: diagonal preconditioning using M_3.

From Table 3.5 it can be seen that mesh 11 with 1363 unknowns requires 784 iterations. In contrast the ungraded mesh 3 with 1367 unknowns requires only 695 iterations, around 10% fewer. Figure 3.12 shows the number of iterations required for both graded and ungraded meshes as a function of the number of vertices. As can be seen a graded mesh generally takes around 10—20% more iterations than an ungraded one with the same number of vertices. Thus it may be concluded that provided an appropriate diagonal preconditioner is employed mesh grading causes no great increase in the number of iterations required.

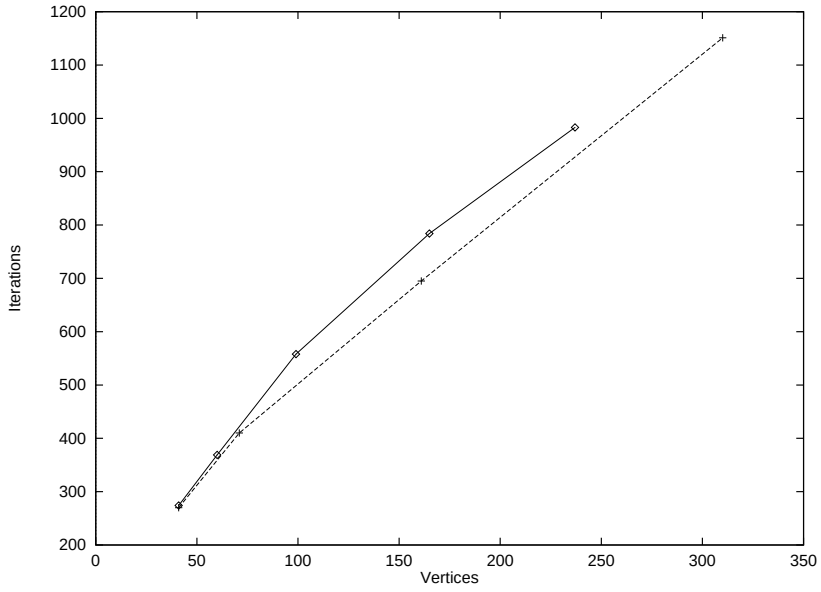


Figure 3.12: Preconditioning of graded vs. ungraded meshes, with diagonal preconditioner M_3. Iterations as a function of vertices: —— graded; – – – ungraded.

The analogous results obtained when ILUT preconditioning is employed are given in Table 3.6. Figure 3.13 shows the run times measured as a function of the number of unknowns for both graded and the ungraded meshes. As can be seen, while

Mesh	$lfil$	$droptol$	$ilut$	$lusol$	$prodmv$	Iterations	Time
8	60	10^{-4}	0.067	0.044	0.009	6	0.120
9	100	10^{-5}	0.300	0.100	0.021	5	0.421
10	100	10^{-5}	0.390	0.130	0.030	6	0.550
11	100	10^{-5}	1.300	0.290	0.057	6	1.647
12	100	10^{-5}	2.300	0.630	0.110	9	3.040

Table 3.6: Graded mesh statistics, iterations and timings: ILUT preconditioning.

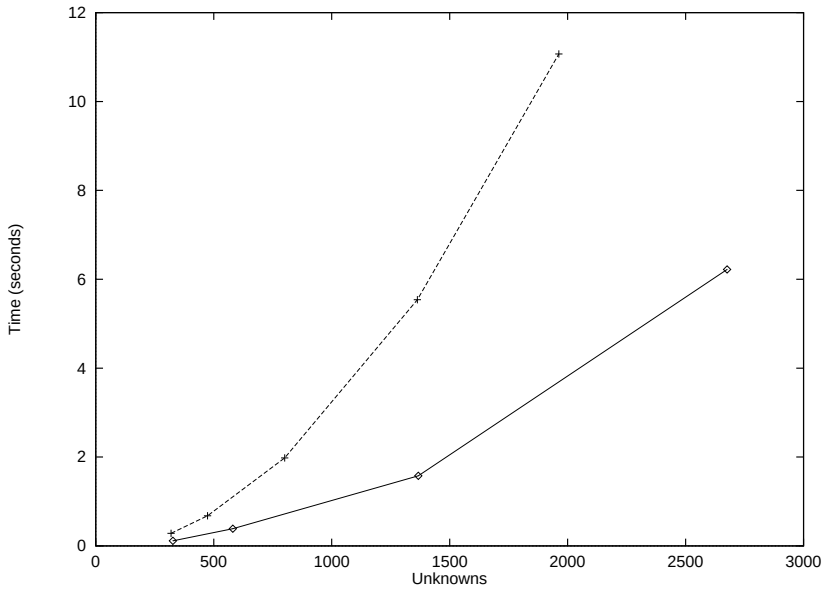


Figure 3.13: Preconditioning of graded vs. ungraded meshes using ILUT preconditioners. Time in seconds as a function of the number of unknowns: – – – graded; —— ungraded.

the number of iterations required is essentially independent of problem size for both graded and ungraded meshes, in both cases run time scales approximately as $O(N^2)$, graded-mesh problems with the same number of unknowns taking roughly four times as long to solve. Further work is necessary to fully understand the cause of this discrepancy but it is clear that it is due to the much fuller incomplete factorisations computed for graded-mesh problems.

The above experiments demonstrate that the preconditioned conjugate residual method is a viable method for solving moderately-sized free-surface finite element problems in two dimensions. Even a relatively unsophisticated implementation, employing only diagonal preconditioning, would be expected to out-perform the fastest implementations of direct methods for all but the smallest of problems. The above

experiments should however only be taken as a guide, since in practice additional factors such as mesh geometry can have a profound influence on the accuracy and efficiency of preconditioners.

3.13 Predictors

For time-dependent problems the computation of simple explicit predictors for use as initial estimates of the solution for the linear solver has been found to be advantageous. For both Stokes and Navier-Stokes problems an obvious choice of initial estimate is the solution from the previous time step. A more sophisticated approach involves using the solutions at a number of previous time steps to form a more accurate explicit predictor. In the current work explicit predictors computed using the previous two solutions are employed whenever possible. These are found using the finite-difference formula

$$\mathbf{u}^{(n+1)} = \mathbf{u}^{(n)} + \left(\mathbf{u}^{(n)} - \mathbf{u}^{(n-1)}\right) \frac{k^{(n+1)}}{k^{(n)}}, \tag{3.76}$$

which gives a predictor that is second-order accurate in time. For the time step immediately after a mesh regeneration (3.76) cannot be used. Instead for Stokes-flow problems a fixed vector (of all ones) is employed as the initial guess, while for Navier-Stokes problems the interpolated velocity field is used. Predictors for the pressure components of the solution are not necessary and in any case appear to have little effect on the number of iterations required. For the second time step after a mesh regeneration the solution at the end of the previous step is available and is used to give first-order accurate predictors for both Stokes and Navier-Stokes problems.

Both first- and second-order predictors are observed to result in considerable reductions in the number of conjugate residual iterations required to obtain a solution to a given accuracy. A good second-order accurate predictor can reduce the norm of the initial residual vector by a factor of 10^3, cutting the number of iterations required by 20—60% (see for example Fig. 4.21). Even where an ILUT preconditioner is used, the resulting saving of perhaps one or two iterations per time step can be considerable. Finally, note that attempts to compute second-order predictors for Navier-Stokes problems using interpolation of solutions from time steps immediately prior to mesh regeneration were unsuccessful, the resulting predictors proving no better than the first-order ones already mentioned.

3.14 Interpolation

When solving the Navier-Stokes equations it is necessary to be able to transfer a velocity field from one mesh to another in order to continue integration after a mesh regeneration. This reflects the fact that initial conditions are required for a time-dependent Navier-Stokes problem. Such transfers are potentially both expensive and a source of error. Fortunately they need only be performed for the velocity field, since no initial conditions are required for the pressure. While the velocity at a free-surface node that is present in both the new and the old meshes needs no modification, at nodes introduced when free-surface edges are split some means of estimating the solution at the new node is required. Here this is done using interpolation. As for the velocity field on the interior of the mesh, three methods for performing its transfer are immediately apparent. The first involves the simple interpolation of the old solution onto the new mesh, i.e. the value of the solution at a node in the new mesh is set to the value of the old solution at the corresponding point in the old mesh. Such an approach has a number of drawbacks. Firstly, it does not, in general, conserve momentum or kinetic energy. Secondly, it results in velocity fields that are not divergence free, i.e. that do not satisfy the discrete continuity equations (3.14). While conservative interpolation schemes have been described for low-order elements [83], the generalisation of such schemes to quadratic elements in two dimensions has not, to the author's knowledge, been described and would appear to be a non-trivial exercise.

The second type of transfer scheme that might be considered involves a projection method [34] requiring the solution of a system of linear equations for a correction to a velocity field obtained using interpolation. When the correction is subtracted from the original velocity field a divergence-free velocity field results. Such an approach, while considerably more expensive than simple interpolation schemes, also fails to conserve momentum and kinetic energy exactly. Since by their implicit nature the time-integration schemes employed here enforce incompressibility at the end of each time step, even if the initial conditions are not divergence free the velocity field at the end of the first time step will be. Thus little appears to be gained from the use of this approach.

The third type of scheme that might be employed makes use of the finite element formulation directly. In essence it involves the solution of a Navier-Stokes problem on the new mesh with a very small or even zero time step, the old solution's contributions to the right-hand side of the linear-algebraic problem being computed by

exact integration. Such a scheme would conserve momentum globally, and would automatically result in a divergence-free velocity field. The main difficulty would appear to be that of exactly integrating a piecewise-continuous velocity field, the discontinuous nature of the integrand making numerical quadrature a non-trivial operation in such circumstances. One solution is to first smooth the old solution by, for example, fitting a local cubic-spline basis using a least-squares procedure [11]. Thus, while some accuracy will be sacrificed (momentum will not be exactly conserved), numerical integration will be greatly facilitated.

In the present work the transfer of solutions between meshes was performed using the first of these schemes, i.e. simple interpolation. Two approaches were investigated, details of which are given in the following section.

3.14.1 Linear and quadratic interpolation

A major complication when performing interpolation between unstructured meshes is the need to be able to efficiently find the element in which a particular point is located. Where the internal mesh edges are linear, deciding whether a point lies within a given triangular element is straightforward and fairly inexpensive. Finding the element, however, requires some form of search. Since this process must be repeated for each element, it is clear that, if naively implemented, interpolation will have complexity $O(N^2)$ where N is the number of elements. By employing more sophisticated search procedures, involving additional data structures, the search cost may be reduced to $O(NlogN)$, using lists of nodes sorted by their x and y coordinates, or even $O(N)$, by employing an heuristic search that makes use of the element adjacency graph. In the current work elements were located using a simple linear search, an approach that has proved satisfactory for the relatively small meshes employed.

The simplest interpolation scheme considered here is a linear one. In this an element's edge nodes are ignored. Thus the local coordinates, (ξ, η), of a point in an element may be computed directly by solving a pair of linear equations. Once this has been done, computing the velocity components at the point is trivial.

A quadratic interpolation scheme using the full isoparametric basis is also possible though somewhat more complicated, since finding (ξ, η) now, in general, involves the solution of the following pair of simultaneous nonlinear equations

$$\sum_{i=1}^{6} x_i q_i(\xi, \eta) - x = 0, \tag{3.77}$$

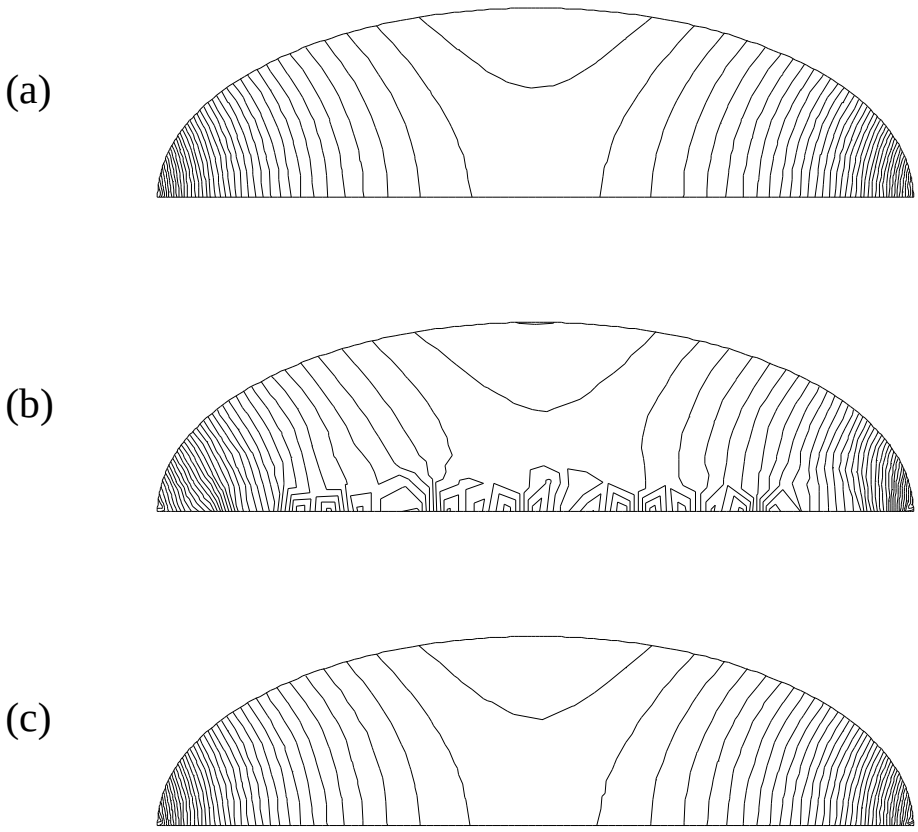


Figure 3.14: Pressure field resulting after interpolation of velocity field, $Re = 10$, axisymmetric oscillation of a droplet: (a) pressure field before interpolation; (b) pressure field after linear interpolation; (c) pressure field after quadratic interpolation.

$$\sum_{i=1}^{6} y_i q_i(\xi, \eta) - y = 0, \tag{3.78}$$

where x_i and y_i are the coordinates of the nodes forming the element. Here, equations (3.77) and (3.78) are solved for ξ and η using Newton's method, which typically takes three or four iterations to reach machine precision.

The improvement in accuracy resulting from the use of quadratic rather than linear interpolation may be seen from Fig. 3.14, which illustrates the effects of interpolation on the pressure field computed for an oscillating axisymmetric droplet. The simulation was conducted at $Re = 10$, on an unstructured mesh, using the methods described in Chapter 6. Fig. 3.14(a) shows the pressure field immediately before a mesh regeneration takes place, while (b) and (c) show the pressure fields observed at the end of the first time step after regeneration, for linear and quadratic

interpolation respectively. Note that only the velocity field is transferred onto the new mesh; the oscillations visible in the pressure in (b) are due to the velocity field resulting from linear interpolation not being divergence free. After only a single further time step these oscillations disappear. To the naked eye, the velocity fields corresponding to (b) and (c) appear identical and for this reason are not shown. It is this quadratic interpolation scheme that is employed for the problems described in Chapter 6.

3.15 Time-step selection

Since all the semi-implicit schemes employed in this work have explicit components, one would naturally expect constraints on the maximum time step to be necessary if the schemes are to be stable. For the semi-implicit Stokes-flow solver described in Section 3.7, the constraint is due to the explicit treatment of the kinematic boundary condition. For the semi-implicit Navier-Stokes solver described in Section 3.5 the explicit treatment of the convective term and the moving-mesh correction term also potentially introduce constraints on the maximum permissible step size. The author's experience is that in practice it is always the explicit free-surface advection scheme that causes the greatest problems. Figure 1.6 illustrates the form of the free-surface instabilities that are invariably observed if too large a time step is employed. The rapid motions of the free-surface nodes visible reverse in direction at alternate time steps. If automatic free-surface remeshing is performed in such circumstances, at some point the highly curved edges will be split, resulting, if time steps of fixed size are employed, in an even more unstable situation. Thus the solver typically fails shortly after the onset of such instability, due to excessive refinement of the mesh.

Although it is possible to solve a free-surface problem using a fixed time-step size, the need to ensure stability throughout a problem will inevitably mean that an unnecessarily large number of time steps will be required. Clearly it would be useful to have some means of choosing the time-step size adaptively so that it remains as large as possible, while at the same time ensuring free-surface stability.

Experience has shown that the semi-implicit schemes described in Sections 3.5 and 3.7 are satisfactory for many problems provided that the time-step size, k, is selected so that

$$k \leq C_1 \frac{h_{min}}{v_{max}}, \tag{3.79}$$

where h_{min} is the minimum edge length in the mesh, v_{max} is the maximum velocity

at a node and C_1 is a constant. In practice taking $C_1 = \frac{1}{4}$ is often sufficient to prevent instability of the free-surface advection scheme arising. For a regular mesh of equilateral elements, using this value of C_1 in (3.79) effectively restricts the distance a notional particle of fluid can travel during a time step to half the minimum distance between nodes. Thus (3.79) may be seen to be a form of Courant-Friedrichs-Lewy (CFL) condition [98]. While (3.79) limits the maximum size of time step that may be employed, it is far less restrictive than the

$$k \leq C_2 \frac{{h_{min}}^2}{v_{max}} \tag{3.80}$$

constraint on time-step size that would be necessary if the viscous term in the Navier-Stokes equations were to be treated explicitly [34].

Employing (3.79) as a sole guide to choosing the time step is however problematic, a smaller value of C_1 often being necessary to ensure stability. It is easy to see why this might be the case. In an arbitrary unstructured mesh of quadratic elements the minimum edge length h_{min} does not give a reliable guide to the minimum distance between nodes. At other times the time step constraint (3.79) is likely to be excessively pessimistic since it employs global measures of the solution's velocity and the mesh's resolution.

Note that the choice of time-step size affects not only the stability of a time-integration scheme but also its accuracy. Thus, while a given size of time step may be adequate to ensure stability, it may not give sufficient temporal accuracy. This is most likely to be a problem when a stable steady-state configuration is being approached, and for this reason a maximum time-step size (typically $0.005 \leq k \leq 0.01$) is generally imposed.

In the course of the current investigations a novel scheme for the selection of time-step size has been found to be particularly useful. Since the stability problem appears essentially to be one of ensuring free-surface stability, and, since experience has shown that such instabilities invariably manifest themselves in the form of time-step-scale temporal oscillations in the sign of the normal velocity at free-surface nodes, the instability has a clear and well-defined signature that may easily be detected. Whenever such instability is present, the expression

$$\left| \frac{\mathbf{u}_i{}^{(n+1)} - 2\mathbf{u}_i{}^{(n)} + \mathbf{u}_i{}^{(n-1)}}{k^2} \right|, \tag{3.81}$$

i.e. a central-difference approximation for

$$\left| \frac{\partial^2 \mathbf{u}}{\partial t^2} \right| , \tag{3.82}$$

will be large at the free-surface nodes effected. Since under normal circumstances (3.81) is relatively small, its rapid growth is a clear and convenient indicator that the time step is too large.

The stability method of time-step size selection operates as follows. At each time step an estimate of the maximum allowable time step, t_c, is computed using

$$t_c = \left(\frac{lte}{\delta} \right)^{\frac{1}{3}} , \tag{3.83}$$

where δ is the maximum value of (3.81) at a node, and lte is a prescribed tolerance, the form of (3.83) being justified only by the observation that it works well in practice. To prevent the possibility of oscillation in the time-step size itself, at each time step the value of t_c computed using (3.83) is combined with the current time-step size, k, using the following exponentially-weighting scheme

$$k^{(n+1)} = 0.9k^{(n)} + 0.1t_c^{(n)}. \tag{3.84}$$

This tends to smooth out rapid fluctuations in time-step size and while it makes the scheme slower to respond when, for example, the mesh is refined and a smaller time step is required, this does not appear to be a great problem in practice since oscillations due to free-surface instability typically grow in amplitude only slowly at first.

In practice a small initial value is chosen for the time-step size. This then grows steadily until the maximum stable time-step size for the configuration is approached. Again, for reasons of accuracy, an upper limit on the time step is also generally imposed. In the current work a value of $lte = 10^{-6}$ is employed, resulting in time steps of comparable size to those found necessary by trial and error. Adaptive time-step size selection using this approach has proved to be very reliable in practice.

3.16 Conclusions

In this chapter the weak form of the Navier-Stokes equations has been introduced and the finite element formulation described. The issue of choosing the appropriate

free-surface normal for use in the implementation of the kinematic boundary condition has been addressed. The question of the need to allow for the motion of nodes has also be raised, and methods for incorporating such motions into the formulation have been outlined.

A number of simple semi- and fully-implicit time-discretisation schemes have been described. The relative costs of these schemes are discussed as are techniques for their solution. Fully implicit schemes, at least when solved using functional iteration, have been found not to be cost-effective due to the small time steps necessary to ensure convergence of the functional iteration scheme.

The form of the preconditioned conjugate residual method employed here has been described and investigations into the relative efficiencies of a number of preconditioners are reported. In particular, Saad's ILUT preconditioner has been found to considerably reduce the overall run time compared to the best diagonal preconditioner considered, even when the cost of computing an LU factorisation is incurred at each linear solve. The run times measured for the various preconditioners were found to increase more rapidly with problem size than theoretical estimates would lead one to expect. In practice, all the preconditioning schemes considered were observed to result in approximately $O(N^2)$ solution times. Despite this, comparison of the conjugate residual method, as implemented here, with a standard direct solver indicates that, even for relatively small problems, the conjugate residual method considerably out-performs the direct solver. A number of suggestions have been made which should allow ILUT preconditioners to be used reliably and efficiently for a wider range of free-surface problems.

A variant of the Cuthill-McKee algorithm has been described for the ordering of the unknowns associated with quadratic mixed-variable finite elements. Methods for transferring solutions between piecewise triangular meshes have been briefly discussed, and the implementation of a simple quadratic approach is described.

Finally, the problem of automatically selecting the largest time step that is small enough to ensure stability of the free-surface advection has been considered. A novel approach to this problem is described.

Chapter 4

The coalescence of two cylinders

The material contained in this chapter is the subject of an IJNMF paper [78], a shorter form of which was presented at the 1998 ICFD conference in Oxford [76].

In this chapter a solver implemented using the methods described in Chapters 2 and 3 is validated using a standard benchmark problem: the surface-tension-driven Stokes-flow coalescence of two infinitely long, parallel cylinders of unit radius. The rate of convergence of the spatial discretisation scheme is shown to be linear, and the mass-conservation properties of the scheme are investigated. Finally, the computational costs associated with the scheme are discussed.

4.1 Background

The need to validate the numerical techniques introduced in the previous chapters provides a strong motivation for the investigation of free-surface problems for which an analytical solution is known. The work of R.W. Hopper is thus of particular interest, since his papers [44, 45, 46] describe exact analytical solutions for a number of surface-tension-driven Stokes-flow problems. These allow the evolution of the free surface to be computed for the simply-connected two-dimensional shapes involved. Thus, while the normal component of the free-surface velocity may be deduced, the solutions do not give the tangential component of the free-surface velocity or the

velocity and pressure fields in the interior of the domain. Hopper's solution for the

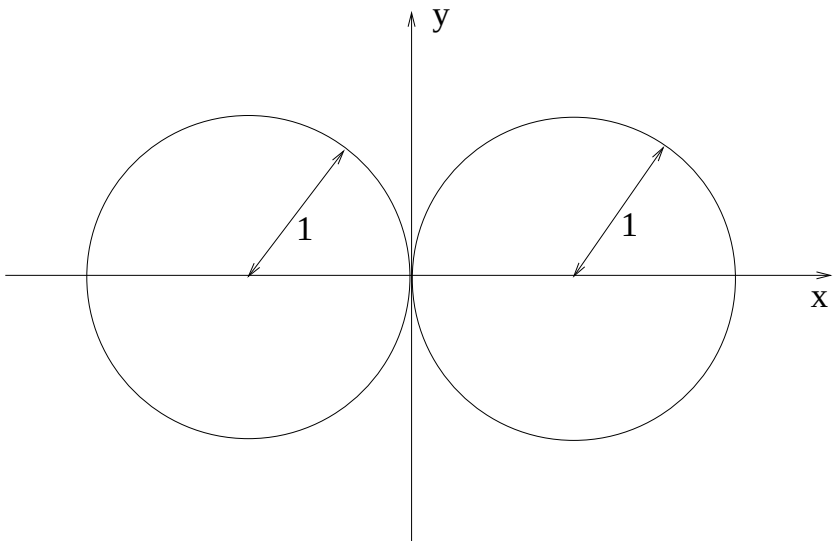


Figure 4.1: The coalescence of two infinite cylinders: initial geometry.

coalescence of two infinite cylinders of unit radius is of particular importance since it provides a useful model for industrial processes involving the viscous sintering of bundles of ceramic fibres [113]. Accurate knowledge of the free-surface dynamics in such problems is useful, in that it allows the final density of materials produced by such processes to be predicted. See [69] for further background material relating to the practical applications of viscous sintering.

In his later papers Hopper extends the approach to include other two-dimensional shapes, and in [47] and [48] he describes solutions for the coalescence of two cylinders of differing radii. Since the point(s) of contact of any two smooth two-dimensional shapes may, locally at least, be approximated by two circular arcs of differing radii, Hopper's solutions in principle allow the initial, extremely stiff, part of such a problem to be dealt with analytically provided, of course, that each such point of contact may be assumed to evolve independently of the others in the earliest stages of such interactions.

The problem considered in this chapter is that of the Stokes-flow coalescence of two parallel infinite cylinders of unit radius under the influence of surface tension, here referred to as the *two-cylinders problem*. Due to its translational symmetry it is reasonable to treat the problem as a two-dimensional one. While it may be argued that this is physically unrealistic, in that instabilities in the third dimension are thus implicitly suppressed, in the viscous sintering of bundles of ceramic fibres [113] such instabilities do not appear to arise in practice and thus the model is a useful one.

The model may also usefully be applied to fibres of finite length, provided that the fibres are sufficiently long that end-effects may be neglected.

The modelling of such flows apparently dates back to Frenkel's 1945 paper [28], in which a simple model of viscous sintering was proposed. Frenkel's model has however been shown by later numerical studies to be incorrect. Ross, Miller and Weatherly's 1981 paper [89] describes the application of the finite element method to the study of an infinite array of cylinders of equal radius. In this they employ a mesh of fixed connectivity that deforms continuously. Martinez-Herrera and Derby [67] apply the finite element method in the modelling of two cylinders of equal radius, as well to other two-dimensional problems involving cylinders of unequal radii, and linear arrays of identical cylinders. They also employ structured meshes of fixed connectivity that deform continuously. Jagota and Dawson [52, 53] employ the finite element method in the modelling of a number of related three-dimensional axisymmetric problems including the coalescence of two spheres of equal radii, the coalescence of two spheres of differing radii, and the coalescence of linear arrays of identical spheres. In their work they employ unstructured meshes that deform continuously, with periodic "repair" of the mesh in the vicinity of the cusp, and periodic remeshing of the entire domain.

In recent years considerable interest has been focused upon the use of boundary integral equation (BIE) methods for the solution of time-dependent free-surface Stokes-flow problems. Also known as boundary-element methods (BEM), these allow the free-surface velocity to be obtained from the stress boundary conditions, without the use of the interior mesh required by the finite element method. Thus at each time step a much smaller, though now dense, system of linear equations must be solved. Kuiken's 1990 paper [60] was one of the first to apply the BEM [51] to the solution of the two-cylinders problem. In this he employs a simple explicit time-integration scheme as a means of updating the free-surface location. With Mattheij and van de Vorst [115] this work was further extended to include shapes such as cylinders of unequal radii, and squares with rounded corners. Mattheij and van de Vorst's 1992 paper [114] further generalises the method by allowing periodic redistribution of collocation points to occur, while van de Vorst's 1994 thesis [113] describes the use of an implicit time-integration scheme (LSODE), as well as the modelling of a large number of related problems, such as the axisymmetric coalescence of two spheres.

Where the BEM is applicable, and where only the evolution of the free surface is required, boundary-element methods appear to have considerable advantages over

finite element methods, since the resulting systems of linear equations are considerably smaller, and the problems of maintaining an adequate interior mesh do not arise. They do not, however, allow the interior velocity field to be computed without considerable additional expense, and thus secondary variables such as the stresses, vorticity, stream function and local rate of viscous dissipation of kinetic energy are similarly difficult to obtain.

4.2 Analytical solution

The problem formulation employed here is based upon that employed by van de Vorst and Mattheij [114, 113]. If two identical infinitely long, parallel cylinders of fluid, are brought into contact, at the line along which they first make contact a pair of cusps arise. Initially the curvature of the free surface at the cusps is unbounded. Figure 4.1 shows the geometry of a cross-section of the domain at the time two cylinders of unit radius initially make contact, while Fig. 4.2 shows part of the evolution of the free surface thereafter. For convenience initial contact is assumed to occur at the dimensionless time $t = 0.0$. The connecting region of

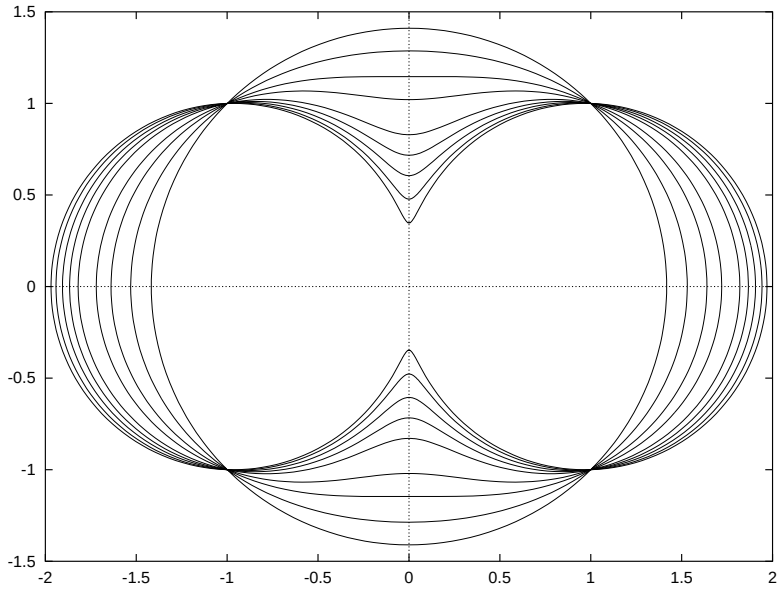


Figure 4.2: The coalescence of two infinite cylinders: free-surface evolution.

fluid that forms between the two cylinders is known as the *neck*. As time passes both the neck width and the neck curvature increase monotonically, the initial neck curvature being negative. Eventually, around the dimensionless time $t = 2.0$, the

neck disappears. Thereafter the evolution of the free surface continues, at an ever decreasing rate, until ultimately a new cylinder arises.

Since the modelling of a highly-curved neck region is difficult it was decided that modelling would begin a short time after the two cylinders initially come into contact. The existence of a known analytical solution for the problem allows this to be done correctly. If Hopper's solution were not known some form of approximation would be required, such as the use of a circular arc to represent the free surface in the neck region.

Mattheij and van de Vorst [114] employ a non-dimensionalisation of the problem similar to that discussed in Section 1.3, which is based upon that of Martinez-Herrera and Derby [67]. Martinez-Herrera and Derby discuss a representative viscous-sintering problem for which the values of the relevant dimensional parameters are $R = 10^{-7} - 10^{-4}m$, $\rho = 10^{3}Kg\,m^{-3}$, $\sigma = 10^{-1}Nm^{-1}$, $\eta = 10^{6} - 10^{9}Kg\,m^{-1}s^{-1}$ and $g = 9.81ms^{-2}$; i.e. length scale, density, surface tension, viscosity and acceleration due to gravity. Combined appropriately, these give a Suratman number of 10^{-23}—10^{-14} and a Bond number of 10^{-9}—10^{-3}. Thus, for this class of problems, the Stokes-flow approximation may be seen to be a particularly good one.

Hopper's analytical method involves the discovery of a time-dependent conformal mapping from the domain onto the unit circle. Further details may be found in [44]. Note that, in this thesis, Mattheij and van de Vorst's formulation for cylinders of unit radius [114, 113] is followed, rather than Hopper's original formulation for cylinders of radius $\frac{\sqrt{2}}{2}$. In [113] van de Vorst gives the following expressions for the location of the free surface,

$$x(\theta,\nu) = \frac{(1-\nu^2)(1-\nu)\sqrt{2}\,\cos\,\theta}{(1-2\nu\,\cos\,2\theta+\nu^2)\sqrt{1+\nu^2}} \tag{4.1}$$

$$y(\theta,\nu) = \frac{(1-\nu^2)(1+\nu)\sqrt{2}\,\sin\,\theta}{(1-2\nu\,\cos\,2\theta+\nu^2)\sqrt{1+\nu^2}} \tag{4.2}$$

where θ is the angle made by a point on the free surface at the origin with respect to the positive x axis, and ν is an increasing function of time, with $0 \leq \nu(t) \leq 1$ for $t > 0$. Using (4.2) the neck radius r may be expressed in the form

$$r(\nu) = y(\frac{\pi}{2},\nu) = \frac{(1-\nu)\sqrt{2}}{\sqrt{(1+\nu^2)}}, \tag{4.3}$$

which may be inverted to give

$$\nu(r) = \frac{2 - r\sqrt{4 - r^2}}{2 - r^2}. \tag{4.4}$$

Van de Vorst gives the following expression for the time t as a function of ν

$$t(\nu) = \frac{\pi}{\sqrt{2}} \int_{\nu}^{1} \frac{dk}{k\sqrt{1 + k^2}K(k)}, \tag{4.5}$$

where $K(k)$ is the complete elliptic integral of the first kind [1, 101], i.e.

$$K(k) = \int_0^{\frac{\pi}{2}} \frac{d\vartheta}{\sqrt{1 - k^2 sin^2\vartheta}}. \tag{4.6}$$

In the experiments described in this chapter (4.5) is integrated numerically using the NAG routine D01BBF [35], employing 64 quadrature points, $K(k)$ being evaluated using the NAG routine S21BBF. Where necessary ν is computed from t by applying the bisection method to (4.5). In practice this need only be done occasionally, for the purposes of data logging etc.

Here an initial configuration at a dimensionless time of $t = 0.2825$ is employed corresponding to a value of $\nu = 0.7$. The resulting simulation, while avoiding extremes of curvature in the neck region, still represents a challenging problem. The following expression for the neck curvature may be obtained using (2.20), (4.1) and (4.2)

$$k_{neck}(\nu) = \frac{(1 - 6\nu + \nu^2)\sqrt{1 + \nu^2}}{(1 - \nu)^3\sqrt{2}}. \tag{4.7}$$

Thus it may be shown that a value of $\nu = 0.7$ corresponds to a neck curvature of approximately -86.633, necessitating a variation in free-surface edge length of nearly two orders of magnitude if curvature is to be equidistributed between edges.

4.3 Method

Five initial meshes were employed in these investigations, three of which are shown in Fig. 4.3. Statistics for the initial meshes are given in Table 4.1. Figure 4.4 shows the mesh in the upper neck region at $t = 0.304$, shortly after the commencement of the simulation, but after the mesh has been regenerated several times. As may be seen, particularly small elements are necessary to resolve the high curvature in the neck region, resulting in a stiff system of equations.

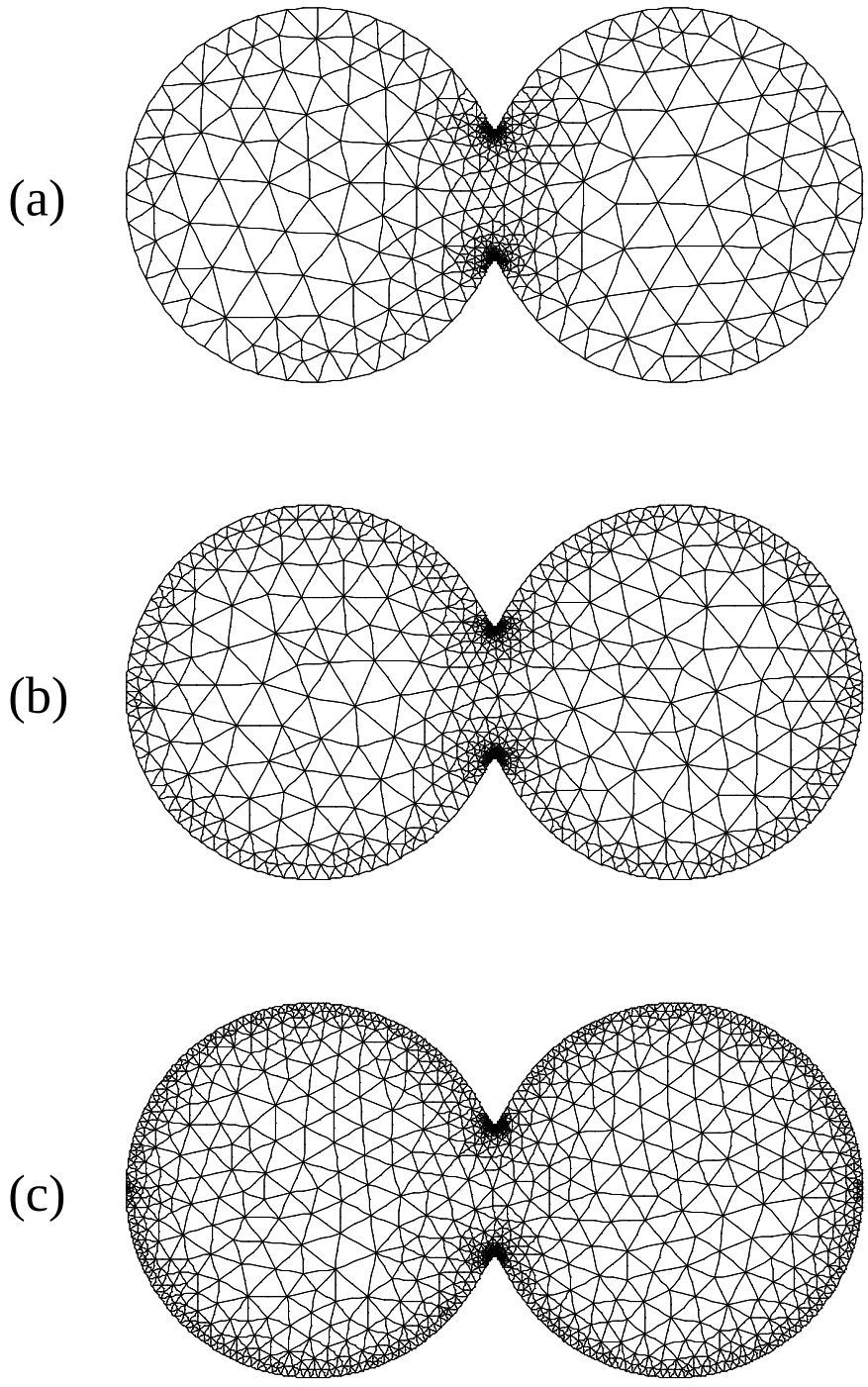


Figure 4.3: The coalescence of two infinite cylinders, selected initial meshes: (a) $k_{tol} = 0.2000$, $h_{max} = 0.3218$, (b) $k_{tol} = 0.1000$, $h_{max} = 0.2554$, (c) $k_{tol} = 0.0500$, $h_{max} = 0.2027$.

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 4.1: The coalescence of two infinite cylinders: initial mesh statistics.

The values of h_{max} were chosen so that each time k_{tol} is halved h_{max} is reduced by a factor of $2^{1/3}$. Thus, as discussed in Section 2.4.4, the error in both the boundary conditions and the velocity on the interior of the mesh should converge linearly. Note, however, that with the unstructured meshes produced by Triangle [96], the actual maximum edge-length need not decrease proportionately to h_{max}. Thus, when halving h_{max}, occasionally a better than average rate of convergence will be observed, while on other occasions the rate will be worse than expected.

Many of the authors who have tackled this problem have elected to make use of the symmetries of the problem, allowing them to model only a quarter of the domain employed here. In the interests of generality here the entire domain is modelled. Thus the current solver may, in principle, be employed to model any simply connected two-dimensional shape. As the experiences recounted in Chapter 6 show, symmetry of the domain may be lost over time where unstructured meshes are used, and thus solving the problem in the present form is a considerably more severe test of the finite element method than would be achieved if only a quarter of the domain were modelled and the symmetries imposed.

Modelling the entire domain, with free-surface stress boundary conditions everywhere, results in the problem being singular; the discrete Stokes operator in this case having a null space of dimension three. The three null vectors correspond to the rigid-body motions of the domain, i.e. translations in the x and y directions and rotation about the domain's centre of mass. Interestingly, if the problem is posed with purely essential boundary conditions, then it is also singular, the null space corresponding to a single, arbitrary, everywhere-constant pressure mode. In this case the problem is sometimes made non-singular by specifying the pressure at a single node. If this approach were to be adopted here, then it would be necessary to impose three constraints on the velocity. While specifying both components of the velocity at a node would provide two of the constraints, imposing the third constraint, on the angular velocity, is more difficult. The obvious approach of using

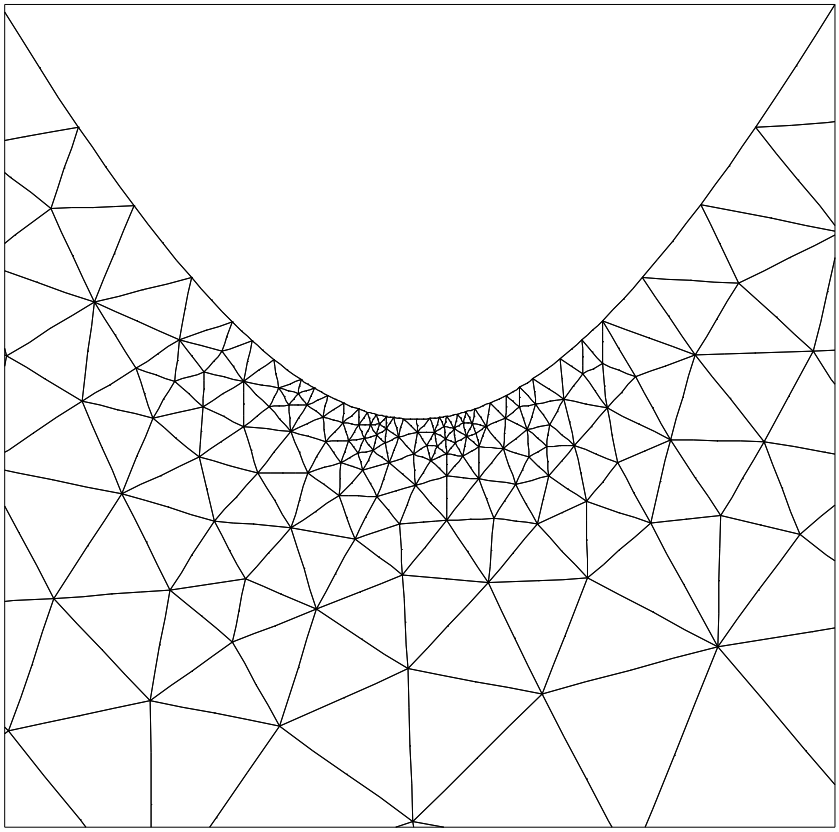


Figure 4.4: The coalescence of two infinite cylinders: mesh detail in the neck region at $t = 0.304$.

the symmetries of the domain in formulating such constraints is problematic, since there is no guarantee that the mesh will remain symmetric as the problem unfolds. An alternative approach, and the one employed here, is to impose global constraints on the velocity and angular velocity, i.e.

$$\int_\Omega u d\Omega = 0, \tag{4.8}$$

$$\int_\Omega v d\Omega = 0, \tag{4.9}$$

$$\int_\Omega \mathbf{u} \times \mathbf{r} d\Omega = 0. \tag{4.10}$$

Thus (4.8) and (4.9) specify that the velocity of the centre of mass of the configuration must be zero, while (4.10) specifies that the configuration must have zero angular velocity about the origin, which is here chosen to be the centre of mass of the fluid for reasons of stability. This approach has the major advantage that it is equally applicable when the domain lacks symmetry.

The three constraints (4.8), (4.9) and (4.10) are assembled as additional rows of the finite element problem, the transposes of the rows also being added so that the resulting matrix is symmetric. Thus one obtains the following modified system of equations

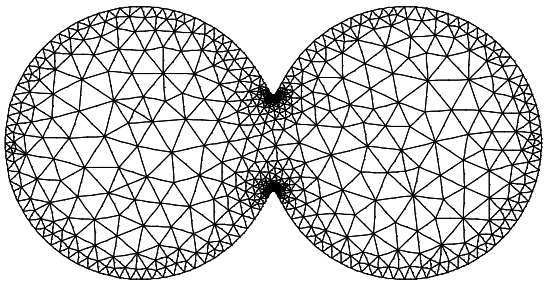
$$\begin{pmatrix} \mathbf{K} & \mathbf{Q}_1{}^T & \mathbf{Q}_2{}^T & \mathbf{Q}_3{}^T \\ \mathbf{Q}_1 & 0 & 0 & 0 \\ \mathbf{Q}_2 & 0 & 0 & 0 \\ \mathbf{Q}_3 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.11}$$

Thus three new variables (Lagrange multipliers) $\hat{\lambda}_1$, $\hat{\lambda}_2$ and $\hat{\lambda}_3$ are introduced. The zero values specified as the right-hand sides of the three new rows are arbitrary, and may be modified to impose any other choices of the linear and angular velocities.

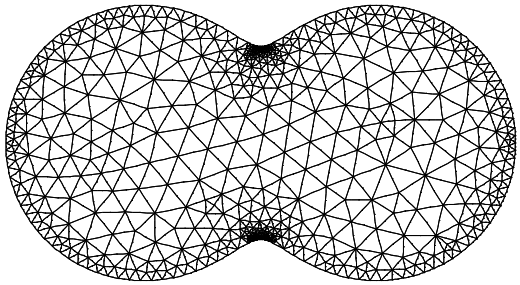
Since the additional constraints must be linearly independent of the rows of the original matrix, and of one another, it is easy to show that if $\mathbf{x}$ satisfies the original (singular) system $\mathbf{K}\mathbf{x} = \mathbf{b}$ and also satisfies the additional constraints $\mathbf{Q}_1\mathbf{x} = 0$ etc. then $(\mathbf{x}, 0, 0, 0)^T$ will be the unique solution of (4.11).

Both diagonal and ILUT preconditioning were investigated as a means of reducing the number of conjugate residual iterations required to solve the linear-algebraic systems that arise in this problem. Diagonal preconditioning, as described in Section 3.12.1, has been found to be useful in this respect, and is the method employed here. Entries in the diagonal preconditioning matrix corresponding to the three additional constraints must however be selected, since the entry corresponding to the diagonal in each constraint row is zero. Setting the corresponding entries in the diagonal preconditioner to equal the area of the domain has been found be effective.

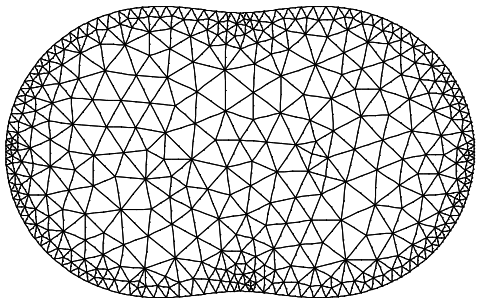
Preconditioners based upon incomplete LU factorisations of the finite element stiffness matrix, while effective in considerably reducing the number of iterations required, have however proved unreliable. Thus, for example, while such preconditioners may prove successful for perhaps several hundred time steps, occasionally the iterative solver will fail completely, stalling once the residual of the preconditioned system has been reduced by only a few orders of magnitude. These failures appear to be due to the dropping of small entries ($a_{ij} < 10^{-6}$) during the incomplete factorisation process, which may, on occasion, prevent convergence to the solution of the correct linear system from occurring, particularly where a small convergence tolerance is required (e.g. 10^{-10}). An in-depth analysis of this problem is not attempted here, since it is noted that many related issues should also be considered, such as that of choosing the values of $lfil$ and *droptol* adaptively, so as to minimise overall run time. In the present work the emphasis has been on achieving reliable



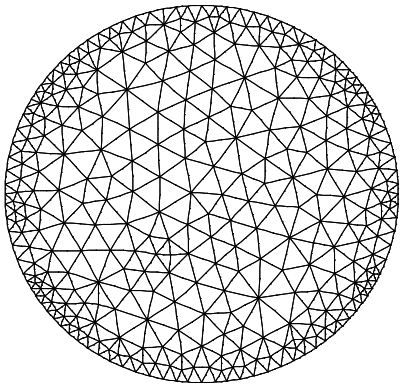
(a) t = 0.2825



(b) t = 0.7675



(c) t = 1.5027



(d) t = 5.0000

Figure 4.5: The coalescence of two infinite cylinders: evolution of mesh 3.

results by the simplest approach possible, the values of $lfil$ and $droptol$ being chosen for reliability rather than efficiency. Unfortunately, this meant that for the larger meshes considered for the two-cylinders problem reliable preconditioners could not be computed within the available storage (approximately 50Mb). Consequently ILUT preconditioners are not employed in this chapter, though it is noted that a hybrid preconditioning strategy, employing an ILUT preconditioner chosen for efficiency whenever possible, and reverting to a diagonal preconditioner when this fails would appear to be an attractive possibility. Note that where ILUT preconditioning is employed, the global constraints (4.8), (4.9) and (4.10) present additional difficulties since, when discretized, they result in rows of the finite element matrix that

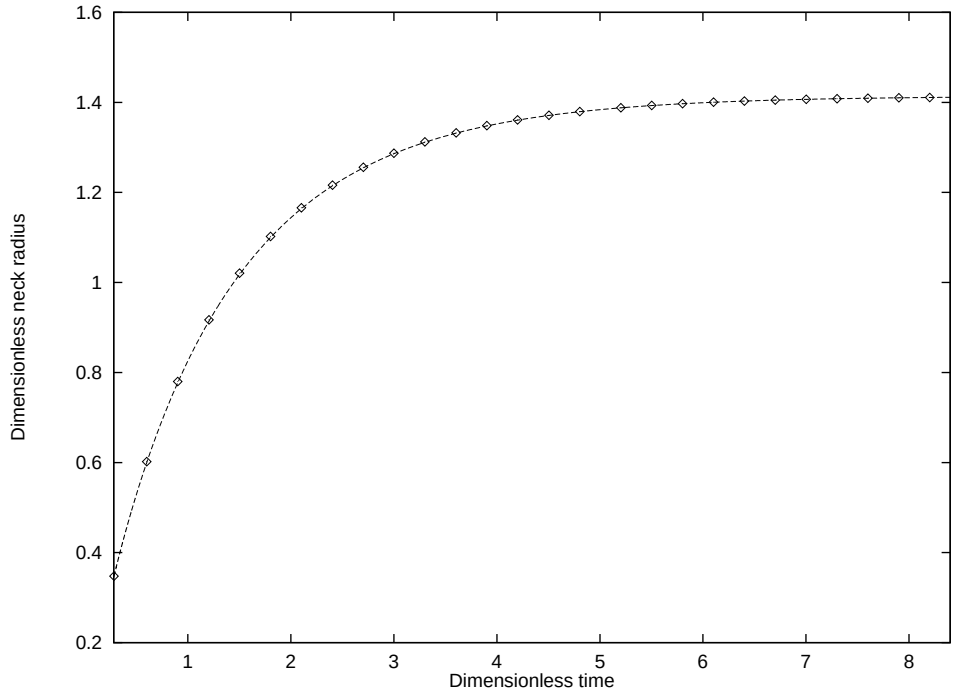


Figure 4.6: Neck radius as a function of time for mesh 3: —– exact; $\diamond$ computed.

have non-zero entries for the u and v unknowns at all the nodes in the mesh. This problem may be circumvented by ordering the rows of the finite element matrix so that the three global constraints are the last rows to be eliminated during the ILU factorisation. Even so, it appears necessary to employ a value of $lfil$ sufficient to store the entire row when eliminating them, though a smaller value of $lfil$ could be employed in factorising the main part of the matrix.

For the problems considered in this chapter the backward-Euler form of the semi-implicit scheme described in Section 3.7 was employed and, unless otherwise stated, the time-step size was chosen adaptively using the stability method described in Section 3.15, taking $lte = 10^{-6}$.

4.4 Results

Figure 4.5 shows the mesh at selected times, (a) corresponding to the initial mesh, while (d) corresponds to the mesh at a time when the free surface is close to circular, the maximum and minimum radii differing by less than 1%. Figure 4.6 shows both the analytical and the computed dimensionless neck radii as functions of time, for mesh 3. As may be seen, the agreement between the computed and analytical solutions is good, the maximum error of approximately 0.01 dimensionless units

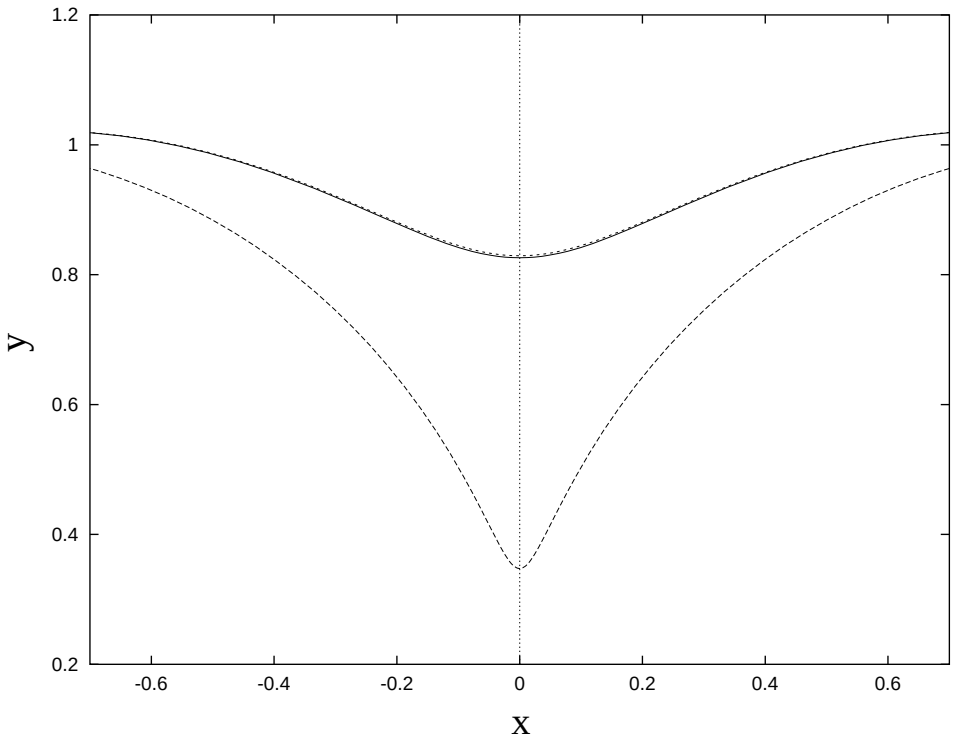


Figure 4.7: Global free-surface error, mesh 3, at $t = 1.000$: - - - - computed free surface; —— analytical free surface; – – – initial free surface.

occurring between $t = 0.5$ and $t = 1.5$. Figure 4.7 shows both the computed and the analytical free surfaces at $t = 1.000$, together with the initial free surface. As may be seen, the maximum discrepancy occurs close to the axis of symmetry. Figure 4.8 shows the dimensionless neck velocity as a function of time. Again the agreement between analytical and computed values is good. Figure 4.9 shows both the velocity field at $t = 0.304$ and the detail in the upper neck region at that time. Note the different scalings of the velocity field employed in the two figures, the greatest velocities occurring in the neck region. As may be seen from 4.9(b), despite the considerable modification of the mesh that has occurred in the neck region by $t = 0.304$, there is no sign of any saw-tooth instability on the free surface, which appears perfectly smooth at this scale.

Figure 4.10 shows two details of the pressure field at $t = 0.304$. The pressure contours shown in all the pressure plots in this thesis were selected automatically, a fixed number of contours being drawn equispaced between the maximum and minimum pressures occurring at that time. Thus neither the contours levels themselves nor their spacing have any absolute significance and cannot be compared between plots.

From Fig. 4.10(b) it may be seen that while the mesh appears adequate, in that

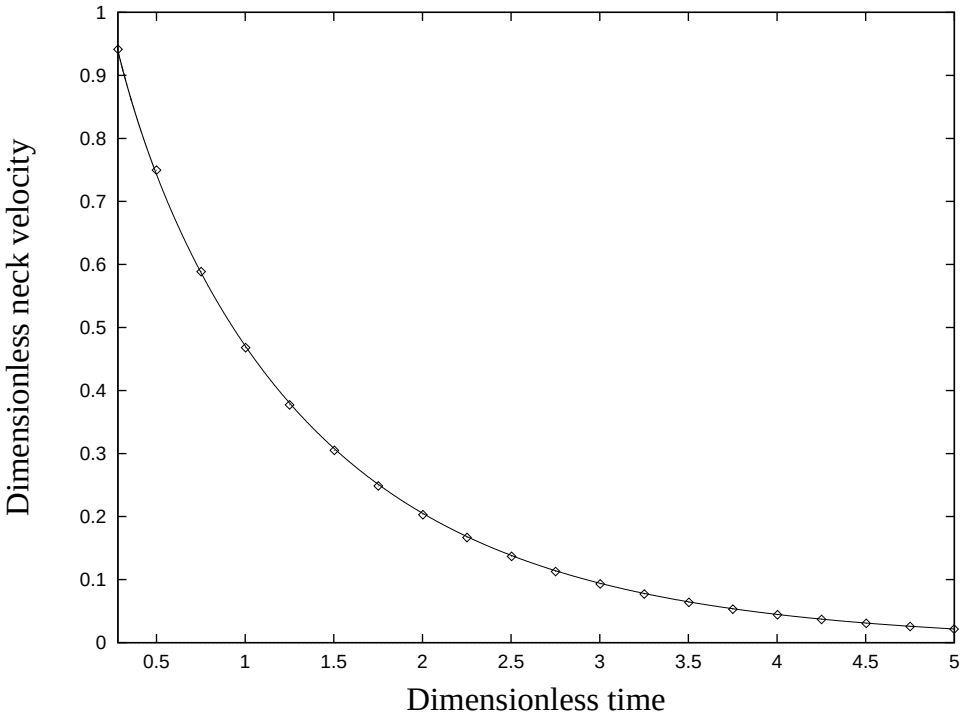
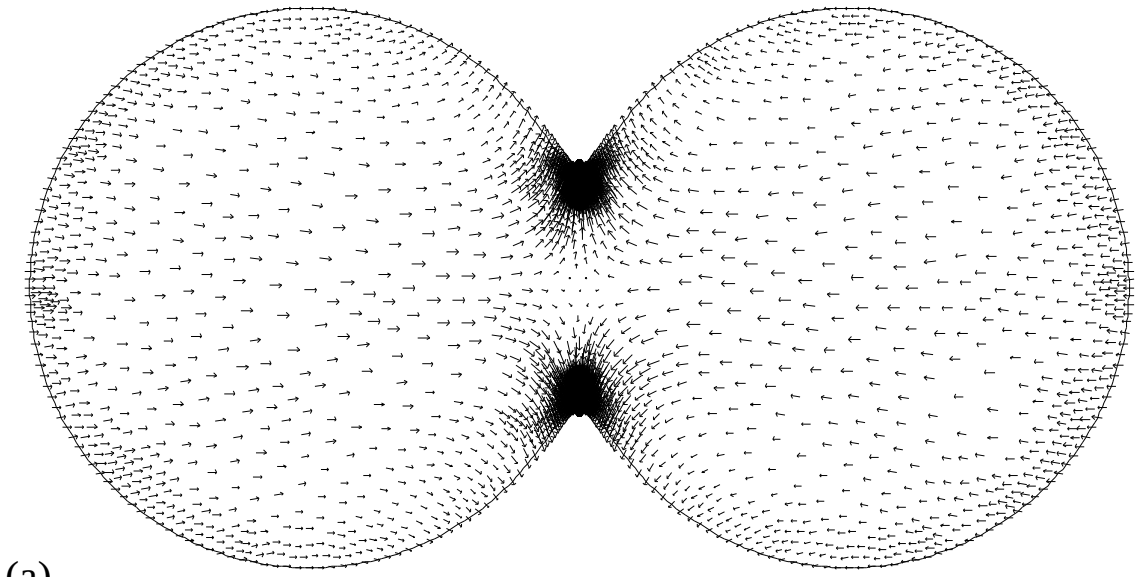


Figure 4.8: Neck velocity as a function of time for mesh 3: —— exact; $\diamond$ computed.

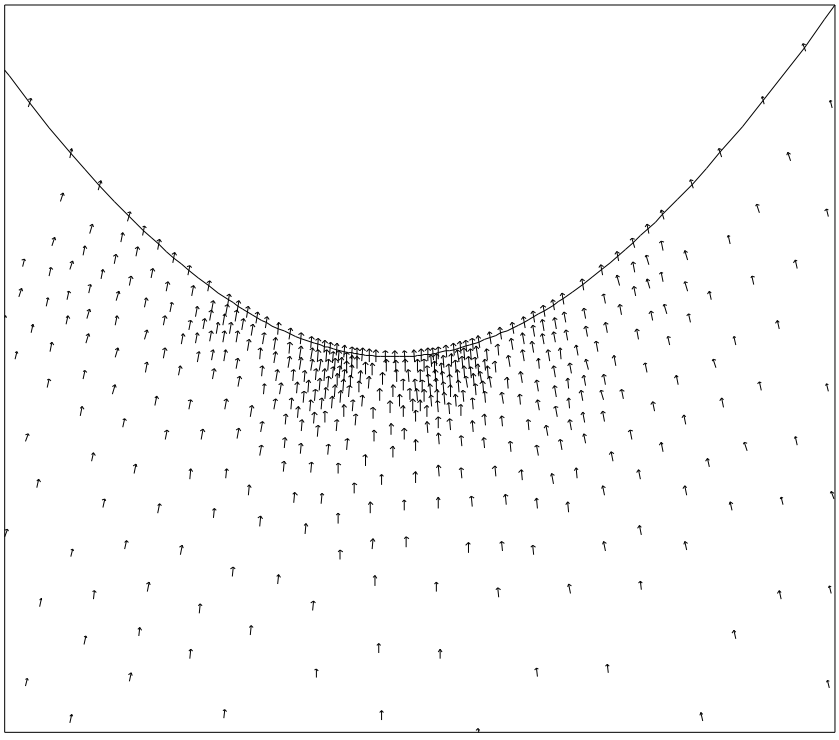
it resolves the main features of the pressure field, kinks are visible in the pressure field approximately one neck radius to the right of the y axis. Reference to Fig. 4.3 shows that these occur in a region of the mesh where the size of elements is changing rapidly in response to reduction in the absolute curvature occurring as one moves away from the neck. Clearly, further refinement of the mesh would be in order here. Note that, as discussed in Section 2.2.3, in this work no attempt is made to refine the free-surface discretisation with respect to the curvature gradient, only equidistribution of curvature being attempted. Thus, if the curvature is small but changing rapidly, a situation in which one would expect large pressure gradients to occur, the current meshing scheme will not take this into account and consequently both the free surface and the interior mesh will be coarser than appropriate.

Figures 4.11 and 4.12 show the velocity and pressure fields, respectively, at $t = 1.000$ and $t = 2.000$. By $t = 2.000$, the concave regions of the free surface have disappeared, and since the pressure gradients driving the flow are now much reduced, the rate of deformation of the domain is now much smaller; the neck velocity in (b) being approximately half that in (a). Note that at $t = 2.000$, the domain still displays a high degree of symmetry.

Table 4.2 shows assorted run statistics for the five meshes considered here. Column two gives the number of PCR iterations required for the first step of the prob-



(a)



(b)

Figure 4.9: The coalescence of two infinite cylinders: (a) velocity field at $t = 0.304$; (b) velocity field in the neck region at $t = 0.304$.

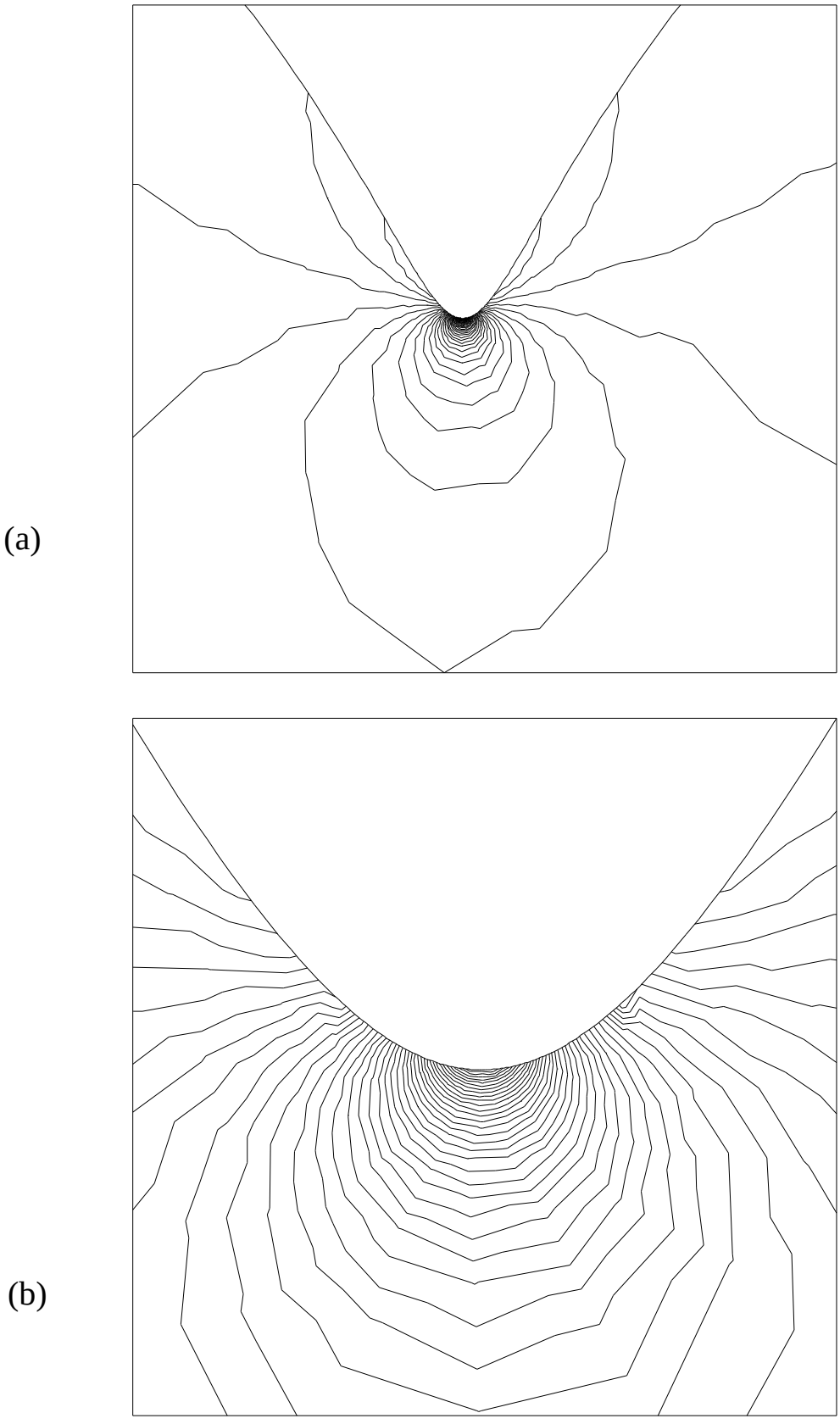


Figure 4.10: The coalescence of two infinite cylinders: (a) pressure field at $t = 0.304$; (b) pressure field in neck region at $t = 0.304$.

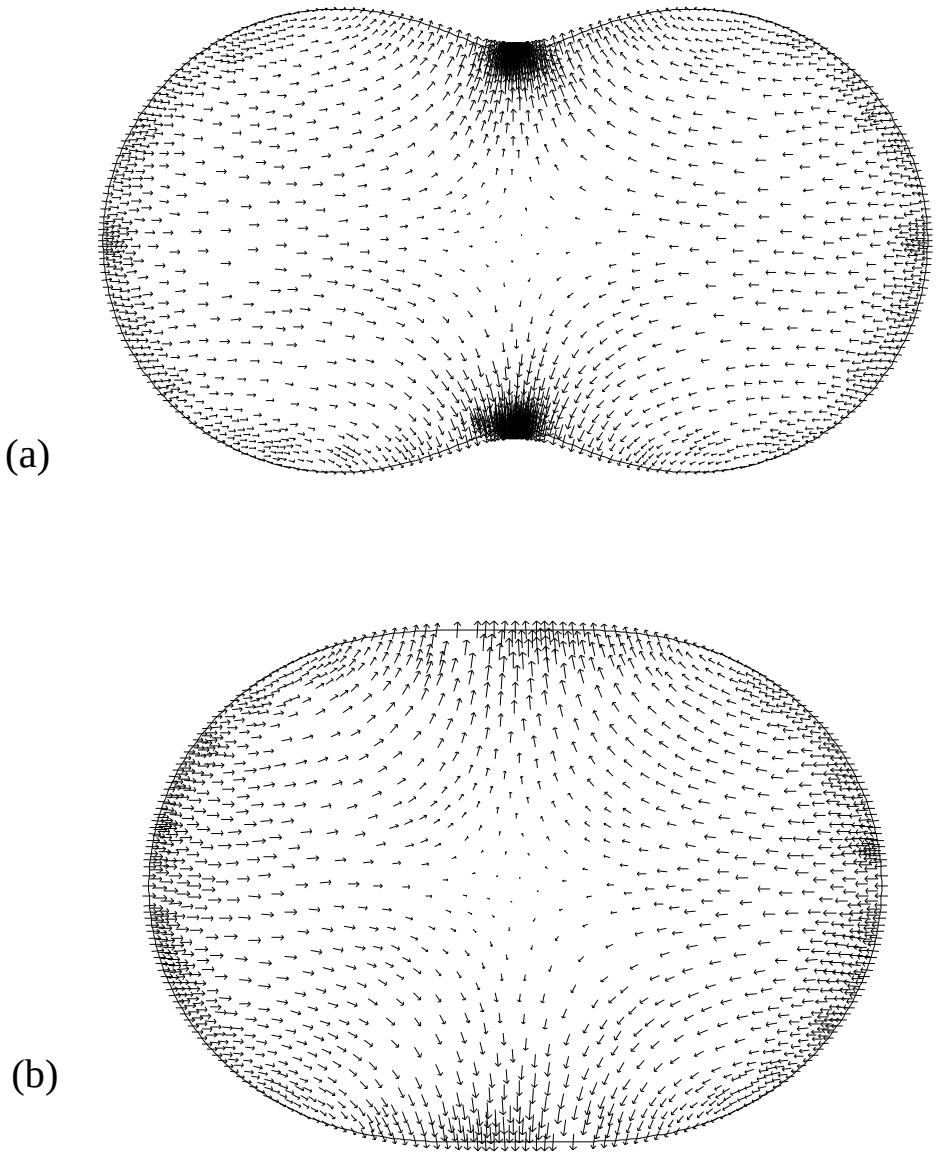
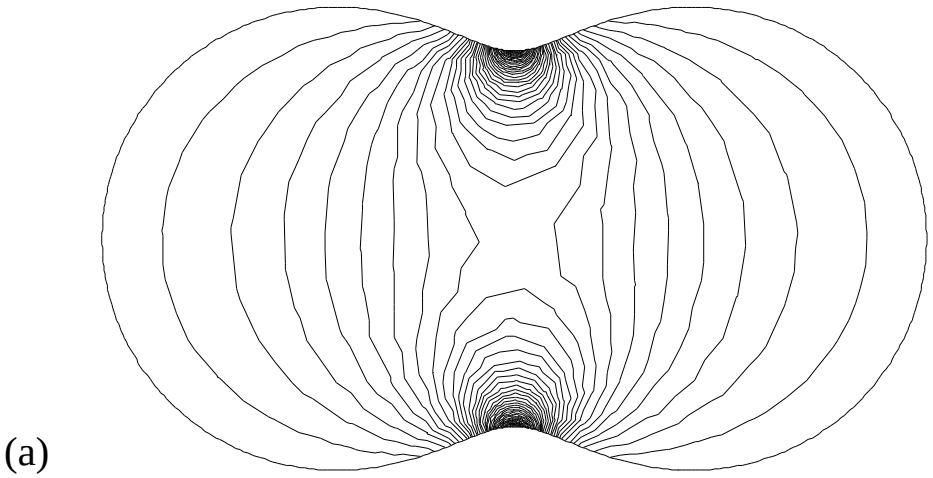
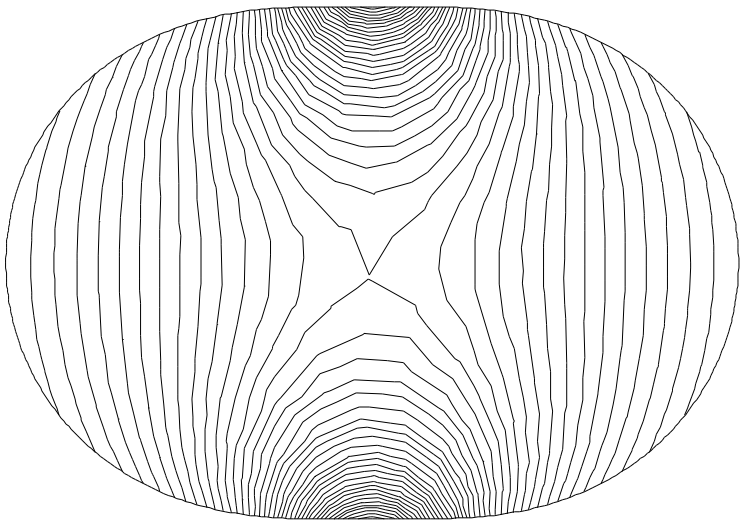


Figure 4.11: Velocity fields at: (a) $t = 1.00$; (b) $t = 2.00$.



(a)



(b)

Figure 4.12: Pressure fields at: (a) $t = 1.00$; (b) $t = 2.00$.

Mesh	Iterations	Steps	Remeshes	Average time per step
1	1619	1548	54	23s
2	1879	1766	149	44s
3	2253	2085	71	69s
4	2686	2648	77	104s
5	3681	3383	130	221s

Table 4.2: The coalescence of two infinite cylinders: run statistics.

lem, column three the number of time steps required to integrate up to $t = 4.0$, column four the number of mesh regenerations required, and the last column the average CPU time per time step as measured on a 180 MHz SGI R5000 workstation. The average CPU time per step, and thus the overall run time, increases by a factor of approximately $2^{1.7}$ each time k_{tol} is halved. The overall run time for mesh 3 was approximately 40 hours, while that for mesh 5 was approximately 208 hours i.e. more than eight days. If uniform refinement were employed, then mesh 5 would have approximately 16 times the number of unknowns as mesh 1, the average time step would take three quarters of an hour and the entire run around four months!

While the number of mesh regenerations required generally increases with the number of time steps as one would expect, the excessively large number required for mesh 2 is hard to explain. It may simply be that for the given domain and choices of h_{max} and k_{tol} the meshes produced by Triangle were poorly suited to the current application, containing triangles that rapidly degenerated as the mesh was deformed. One strategy for dealing with this problem would simply be to perturb h_{max} (or k_{tol}) slightly, in the expectation that this would lead to more satisfactory behaviour. Further investigation of this issue would appear appropriate, particularly with regard to variants of the Jacobi-smoothing operator, which might be designed so as to give greater weight to the need to bound mesh angles away from both 0° and 180°. Since the mesh is regenerated at least every 40 time steps, it may be deduced that in all cases, except for mesh 2, only a minority ($\approx 30\%$) of the mesh regenerations are forced ones, the remainder being periodic regenerations.

4.4.1 Accuracy

Figure 4.13 shows the neck velocity in the early stages of the problem for three of the meshes, as well as the exact velocity. As may be seen, the computed neck

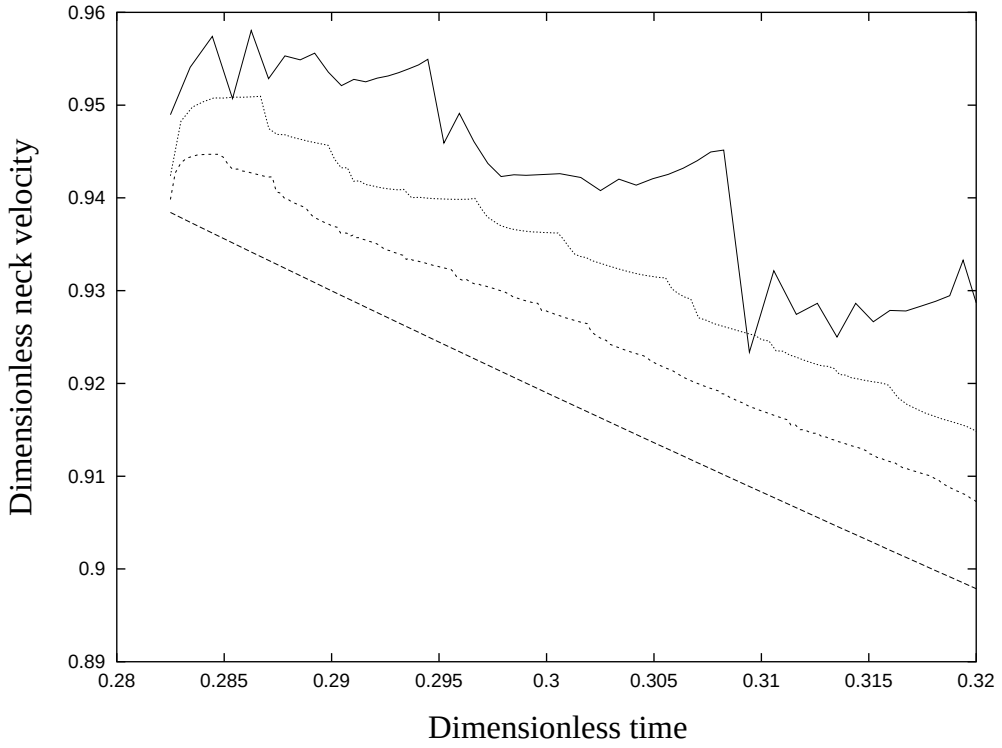


Figure 4.13: Initial stage velocities: —— mesh 1; . . . mesh 3; - - - mesh 5; – – – exact.

velocity contains a large component of 'noise' and is far from monotonic at this scale. This is particularly apparent for the coarsest mesh. As the mesh is refined the amplitude of this noise decreases rapidly, and is in all cases a fraction of the mean error in the velocity. Note the initial upward transient in all three cases, the initial velocities being considerably closer to the exact solution, than those observed once the simulations are underway.

Table 4.3 gives the error in the neck velocity at an arbitrary time, $t = 0.31$, in the early stages of the problem. This time was chosen in order to avoid large errors due to the global error in the free surface's location. The values given were obtained by fitting a straight line to the data shown in Fig. 4.13 for the interval $t \in [0.30, 0.32]$, using a least-squares method [121]. The velocity at $t = 0.31$ was then found by interpolation and the error in the velocity computed using (4.3) and (4.5). The data given in Table 4.3 is also shown in Fig. 4.14, from which it may be seen that the error in the neck velocity decreases linearly, i.e. proportionately to k_{tol}.

4.4.2 Tangential stress

A convenient test of any finite element solution obtained with natural boundary conditions is to check that the boundary conditions recovered from the computed

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 4.3: The coalescence of two infinite cylinders: averaged neck velocities at $t = 0.31$.

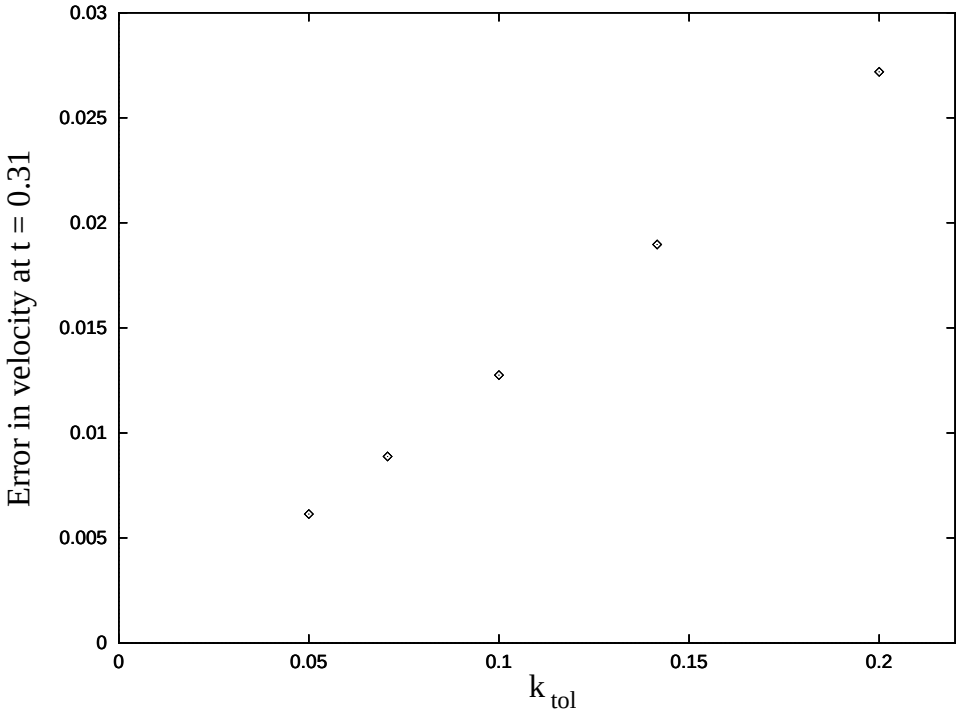


Figure 4.14: Convergence rate: error in the dimensionless averaged neck velocity at $t = 0.31$, as a function of k_{tol}.

solution agree with those imposed. Since the imposed tangential stress is here zero this approach provides a convenient and simple test of the accuracy of the solver, that may be applied to any free-surface problem for which the imposed tangential stress is zero.

At free-surface edge nodes the computed velocity and pressure fields may be employed directly to recover the tangential stress. At free-surface vertices, however, the situation is complicated by the fact that the velocity field is only C^0 continuous at element boundaries. Here this difficulty is dealt with by the simple expedient of taking the average of the tangential stresses computed on the two adjacent free-surface edges. The reader's attention is, however, drawn to the discussion of more

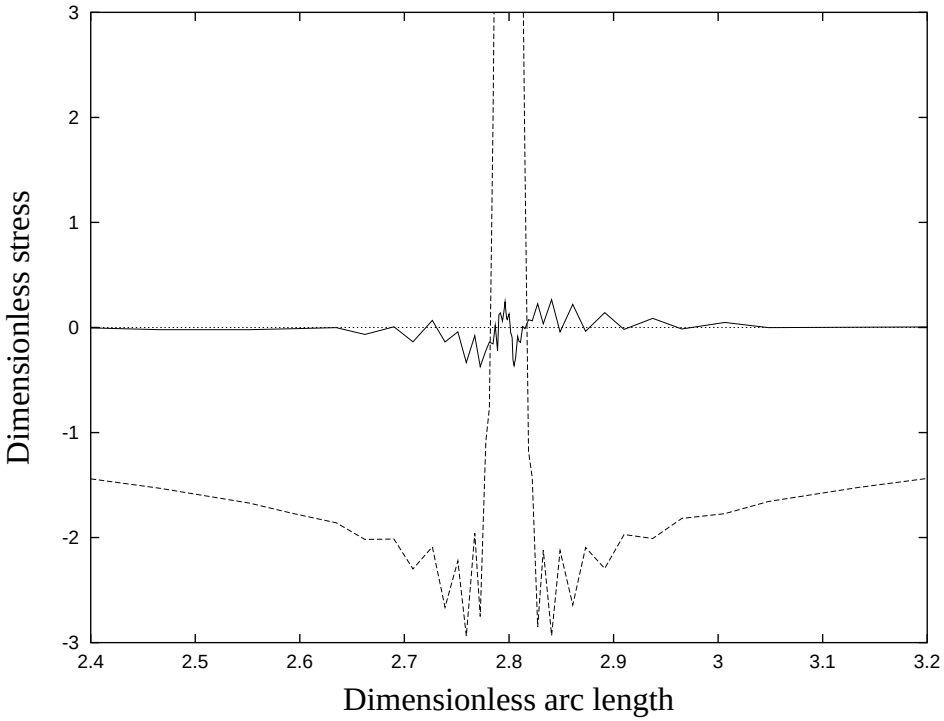


Figure 4.15: Normal and tangential stress in the neck region at $t = 0.29$: – – – normal stress; —— tangential stress.

sophisticated techniques for evaluating gradients at nodes contained in [34]. Figure 4.15 shows both the normal and tangential stresses in the neck region for mesh 5, at $t = 0.29$, when the maximum absolute neck curvature is still large. Note that at this early point in the computation the mesh has already been regenerated three times. As expected, the normal stress has a marked peak in the neck region, the maximum absolute dimensionless stress being approximately 86. The oscillations visible in the normal stress in the neck region, while unwelcome, are not entirely unexpected. The maximum amplitude of the oscillations is around 25% of the imposed normal stress. Oscillations like these are often observed in finite element solutions where the mesh is too coarse to accurately represent the solution [34]. Note that, while the free-surface meshing algorithms employed here do a good job of resolving the regions of maximum free-surface curvature, the adjacent sections of the free surface, being of relatively low curvature, are far less well resolved. Thus it appear that mesh refinement with respect to free-surface curvature alone is insufficient to guarantee a good mesh. Similar oscillations are also visible in the tangential stress with approximately the same amplitude as those observed in the normal stress at a given point on the free surface.

Where the mesh is sufficient to adequately resolve the solution, the tangential-

stress errors observed appear to result from the presence of finite discontinuities in the tangent at free-surface vertices. As discussed in Section 2.2.1, in the current formulation continuity of the tangent at free-surface vertices is by no means guaranteed, and thus, in general, at a free-surface vertex the normal and tangent are not uniquely defined. Regardless of how the imposed normal stress is computed, in practice the specification of a normal stress at a free-surface vertex will inevitably result in the imposition of non-zero tangential stresses on the two edges adjacent to the vertex in question. Thus it is clear that, whenever discontinuities are present at vertices, a certain amount of 'cross-talk' between the imposed normal and tangential boundary conditions will occur.

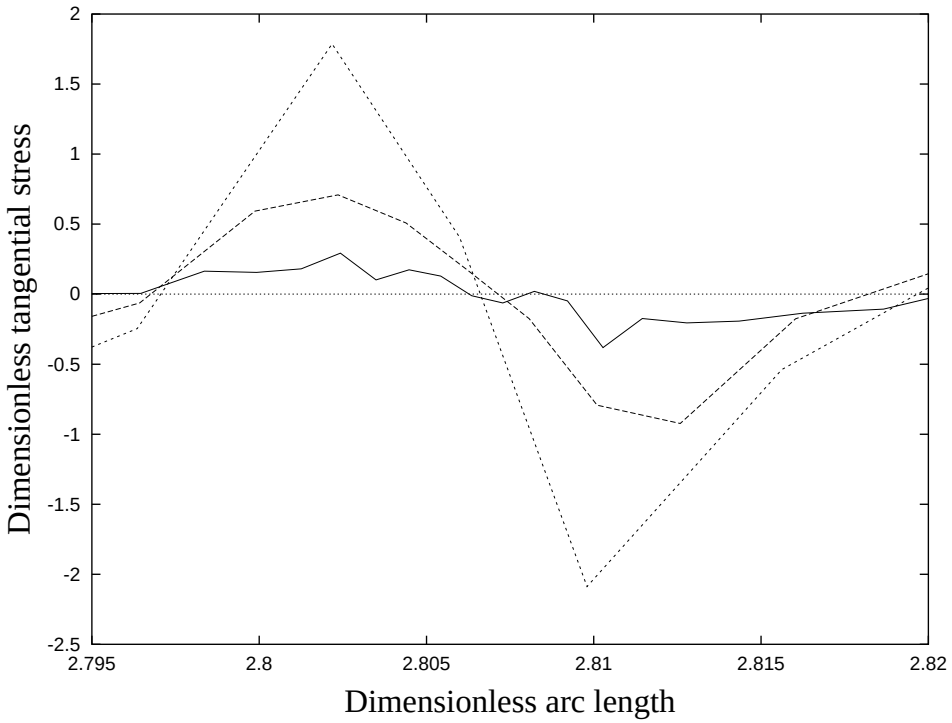


Figure 4.16: The effect of mesh refinement on tangential-stress error in the neck region centred around $x = 0$. Tangential stress at $t = 0.29$: - - - mesh 1; – – – mesh 3; —— mesh 5.

Figure 4.16 shows the tangential stress on a section of the free surface centred on the upper neck for meshes 1, 3 and 5. As may be seen, the tangential-stress error decreases rapidly as the mesh is refined. For mesh 5, the maximum absolute tangential-stress in the region shown is approximately 0.5, i.e. only 0.6% of the magnitude of the imposed normal stress.

4.4.3 Conservation of mass

In a time-dependent free-surface computation the possibility arises that the size of the domain may actually increase or decrease during a computation. Three causes of mass conservation error are immediately apparent in a numerical study of this nature. Firstly, since the set of linear equations is only ever solved approximately, mass may be lost or gained due to failure to impose the incompressibility constraint exactly. In the current finite element formulation, employing Taylor-Hood elements, mass conservation is not guaranteed for each element; incompressibility being imposed only globally through the discretisation of the continuity equation. Thus,

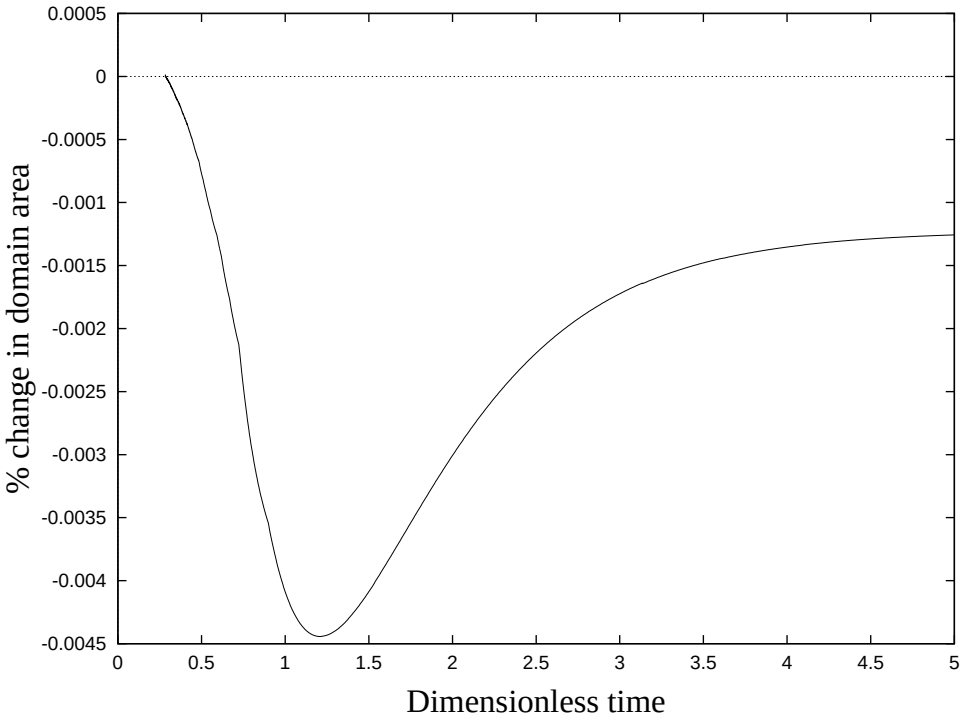


Figure 4.17: Conservation of mass: percentage change in domain area as a function of time for mesh 3.

while locally relatively large mass-conservation errors may occur, over the entire domain one would expect, in the worst case, the rate of mass loss or gain due to this source to be of the same order of magnitude as the tolerance to which the components of the velocity are computed multiplied by the length of the free-surface perimeter. Thus, here, the worst-case rate of mass loss would be expected to be of the order of $4\pi \times 10^{-10}$ in the initial stages of the problem.

In the current scheme, mass may also be gained or lost as a result of the occasional splitting, merging and adjustment of edges necessary to maintain an accurate free-surface discretisation. In practice, these modifications appear not to be a significant

source of global mass conservation error, since the individual errors tend to cancel with one another over a sufficiently long period of time.

Figure 4.17 shows the percentage variation of the domain area as a function of time. Note the initial relatively rapid drop in the area that occurs as the curvature in the neck region decreases rapidly. In the later stages of the problem the area increases until, as the final configuration is approached, the rate of change nears zero. Thus it appears that the rate of change in area is greatest when the free surface is moving most rapidly. Note, however, that the maximum deviation of the area from its initial value is only 0.005% for this problem. From Figure 4.18 it may

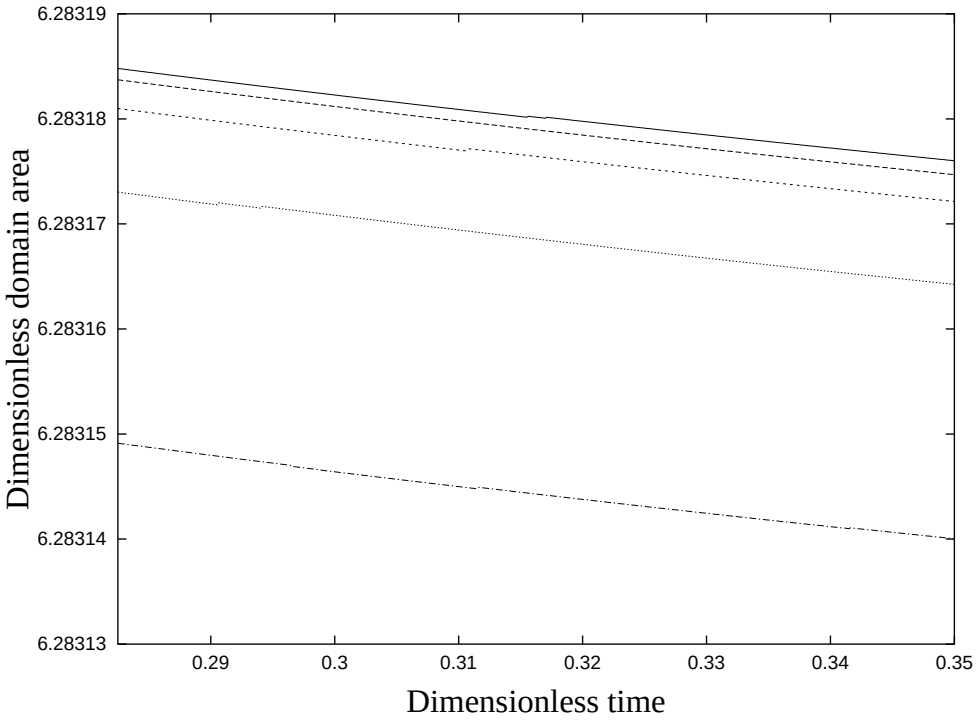


Figure 4.18: The effect of spatial refinement on mass conservation; dimensionless domain area as a function of time, for a fixed time-step size $k = 0.000125$: -·-·-·- mesh 1; · · · · · mesh 2; - - - - - mesh 3; – – – – mesh 4; ——— mesh 5.

be seen that as the boundary of the mesh is refined the rate of area change *does not* decrease significantly. Note the different initial areas for each of the meshes, the correct area of 2π, or 6.2831853 to 8 significant figures, being approached by the finest meshes. In all five cases, a fixed time-step size of $k = 0.000125$ was employed. For mesh 3 an average rate of area change of approximately 2.3×10^{-3} % per dimensionless unit of time was observed over the interval shown.

Figure 4.19 shows the effect that altering the size of the time step has on the initial rate of area change for a mesh of fixed resolution. From Fig. 4.19 it may be

seen that the area lost by time $t = 0.35$ varies approximately linearly with time-step size. This is precisely what one would expect to observe for a first-order explicit advection scheme such as that employed here. In such a scheme the velocity of each

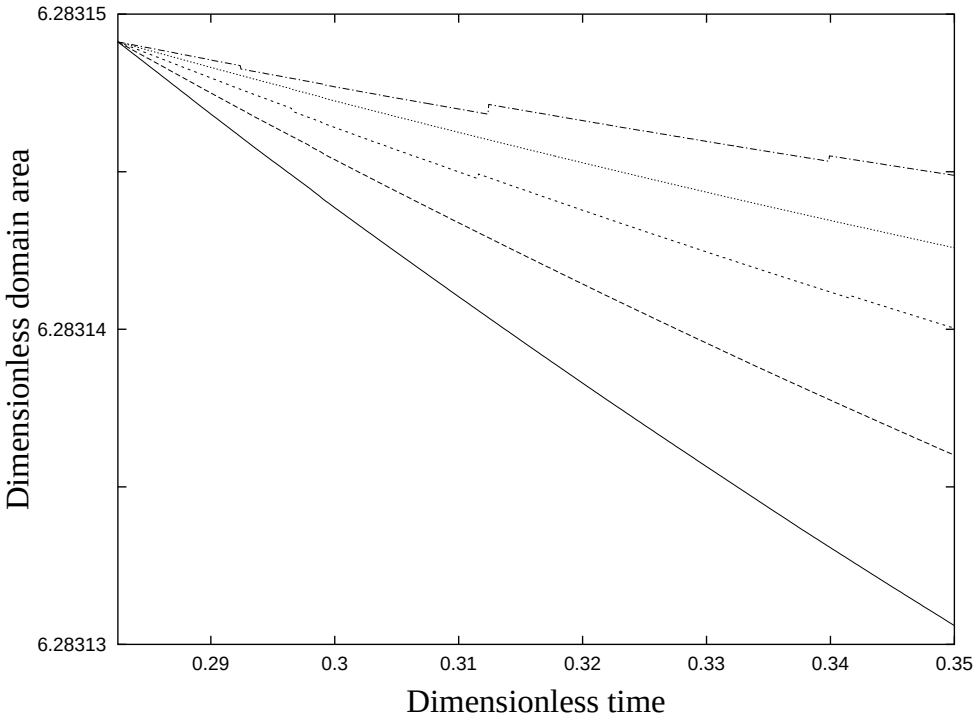


Figure 4.19: The effect of time-step size on mass conservation; dimensionless domain area as a function of time using mesh 3: ——— $k = 0.0002500$; – – – – $k = 0.0001780$; - - - - - $k = 0.0001250$; · · · · · $k = 0.0000884$; -·-·-·-·- $k = 0.0000625$.

free-surface node is held fixed throughout a time step and the position of each node varies linearly with time within the time step. Consequently, the local truncation error for each free-surface node's location at the end of a time step, and thus the area, will be $O(k^2)$. As a result, in the worst case, the global truncation error will be $O(k)$, since the number of steps required is $O(k^{-1})$. Thus it appears that the largest part of the mass-conservation error observed results from the temporal truncation error of the explicit free-surface advection scheme employed here.

A number of discontinuities in the curves are apparent in Figs. 4.18 and 4.19. These are due to errors introduced when the mesh is regenerated and free-surface edges are merged and split. Note that the discontinuities visible in Fig. 4.19 appear worse than those in Fig. 4.18 due to the different scales employed for the y axis in the two figures. The largest discontinuity visible in Fig. 4.18 occurs for mesh 3 around $t = 0.31$ and corresponds to a jump in the area of approximately 10^{-6} dimensionless units. Since mesh 3 requires 77 mesh regenerations for this problem i.e. on average

one every 0.05 time units, if one assumes that the size of the discontinuity at $t = 0.31$ is representative of the others, then, in the worst case, the average rate of change in area via this mechanism will be of the order of $3 \times 10^{-4}\,\%$, and the overall change in area around $2 \times 10^{-3}\,\%$. In practice, it appears that these discontinuities tend to cancel one another out over relatively short periods of time. Furthermore, they appear to be largest in the initial stages of this problem, and, as the maximum absolute neck curvature decreases, similarly decrease in magnitude.

4.4.4 Efficiency

Figure 4.20 illustrates how the number of unknowns varies with time when initial mesh 3 is employed. By the later stages of the problem the number of unknowns, N, has more than halved. Since the linear solver employed here has an $O(N^2)$ run time, this results in a four-fold reduction in the CPU time required per time step. As Fig. 4.21 shows, the reduction in the number of unknowns is matched by a reduction in the number of conjugate residual iterations required per time step. Note the peaks that occur whenever the mesh is regenerated. These are due in part to the fact that immediately after a mesh regeneration no useful predictor is available. On the second step after a mesh regeneration only a first-order accurate predicator is available and thereafter second-order accurate predictors become available. As may be seen, in the later stages of the problem the availability of second-order predictors roughly halves the number of conjugate residual iterations required per time step. Attempts to compute predictors for the time step immediately after a mesh regeneration, by performing interpolation of solutions from previous steps, were however unsuccessful, those computed proving no better than a fixed vector of all ones.

Although the largest part of the reduction in the iteration count occurs over the first three steps after a mesh regeneration, as may be seen from Fig. 4.21, the number of iterations required continues to fall. This is due to the fact that a new mesh generated by Triangle will, in general, undergo a number of Jacobi-smoothing operations before it approaches a stable configuration. Thus, initially, the linear-algebraic problem will be changing rapidly, and, as a consequence, the predictors will be relatively poor. As the mesh nears its optimal configuration however, the algebraic system changes much more slowly, now mainly in response to the evolution of the free surface, and as a result the predictors become increasingly accurate. Note

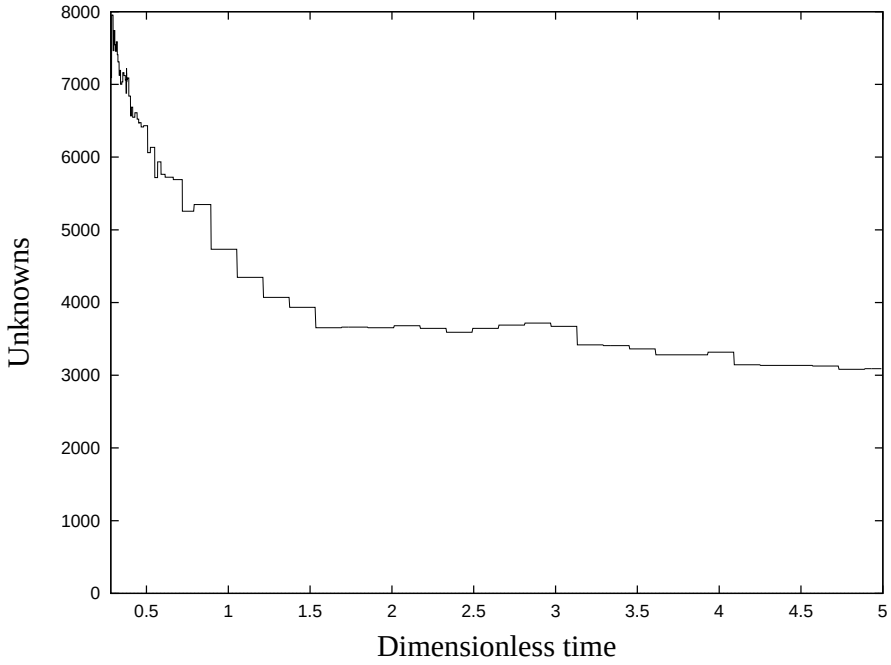


Figure 4.20: Number of unknowns as a function of time for mesh 3.

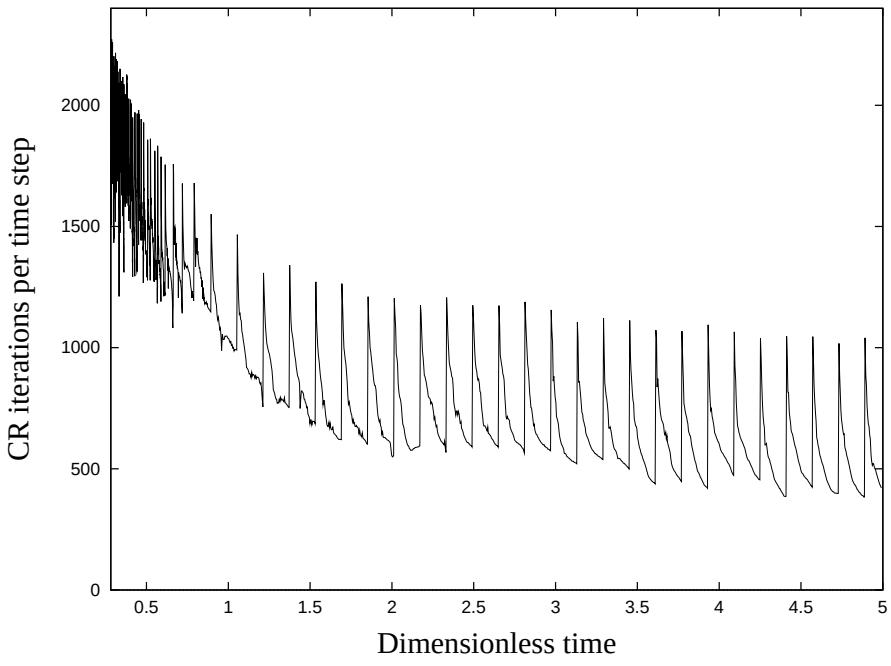


Figure 4.21: Number of PCR iterations per time step as a function of time for mesh 3.

that one would also expect the Laplacian smoother to reduce the condition number of the linear-algebraic problem as it acts to reduce the difference in size of adjacent elements [27].

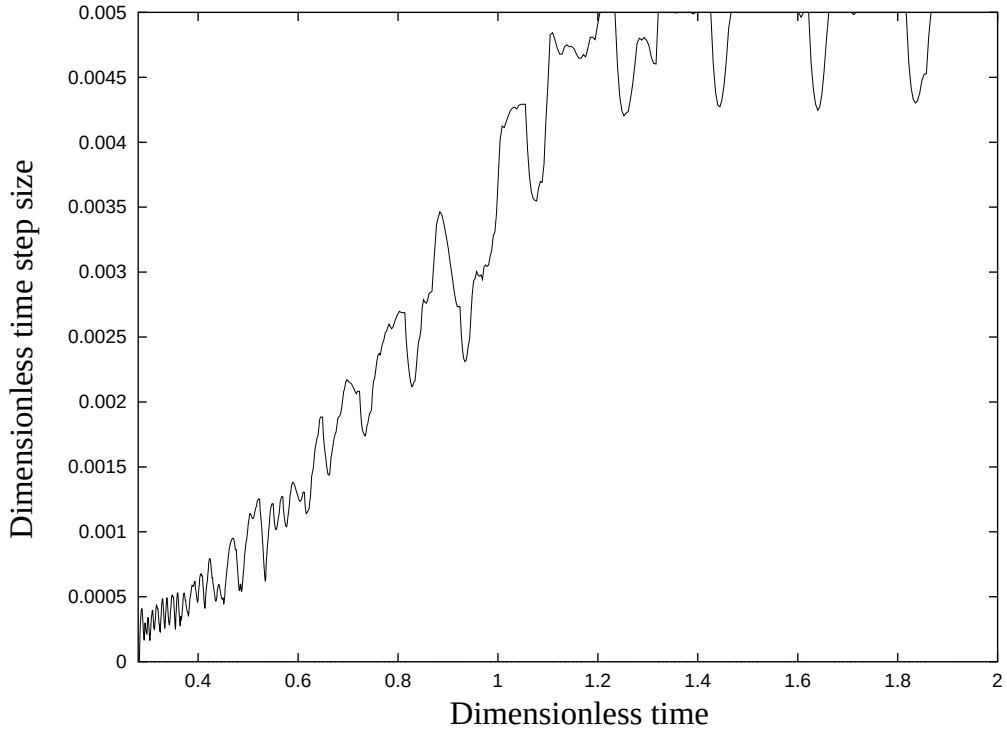


Figure 4.22: Adaptively chosen time-step size as a function of time, $lte = 1 \times 10^6$.

Figure 4.22 shows the dimensionless time-step size, selected using the stability method described in Section 3.15, as a function of dimensionless time. As may be seen, the initial time-step size is of the order of $k = 0.00025$. This gradually increases until, around $t = 1.2$, the upper limit on the time-step size of $k = 0.005$ is attained for the first time. Thus, it may be seen that the time-step size employed increases twenty-fold as the problem progresses, allowing a considerably smaller number of time steps to be taken than would be the case if a time step of fixed size were employed. While the selection of time-step size using the CFL method described in Section 3.15 was also investigated, it was typically found to be less efficient than the stability method, the main difficulty with the CFL approach being that the constant employed ($C_1 = \frac{1}{4}$) often had to be further reduced to guarantee the stable evolution of the free surface throughout the problem.

4.4.5 Shapes with corners

In order to illustrate that the techniques employed above may be applied to problems with more general free-surface geometries, the Stokes-flow evolution of a cross was also briefly investigated. Figure 4.23 shows the evolution of such a shape, the initial width of the domain being 1.2 units. The time-step size was chosen automatically

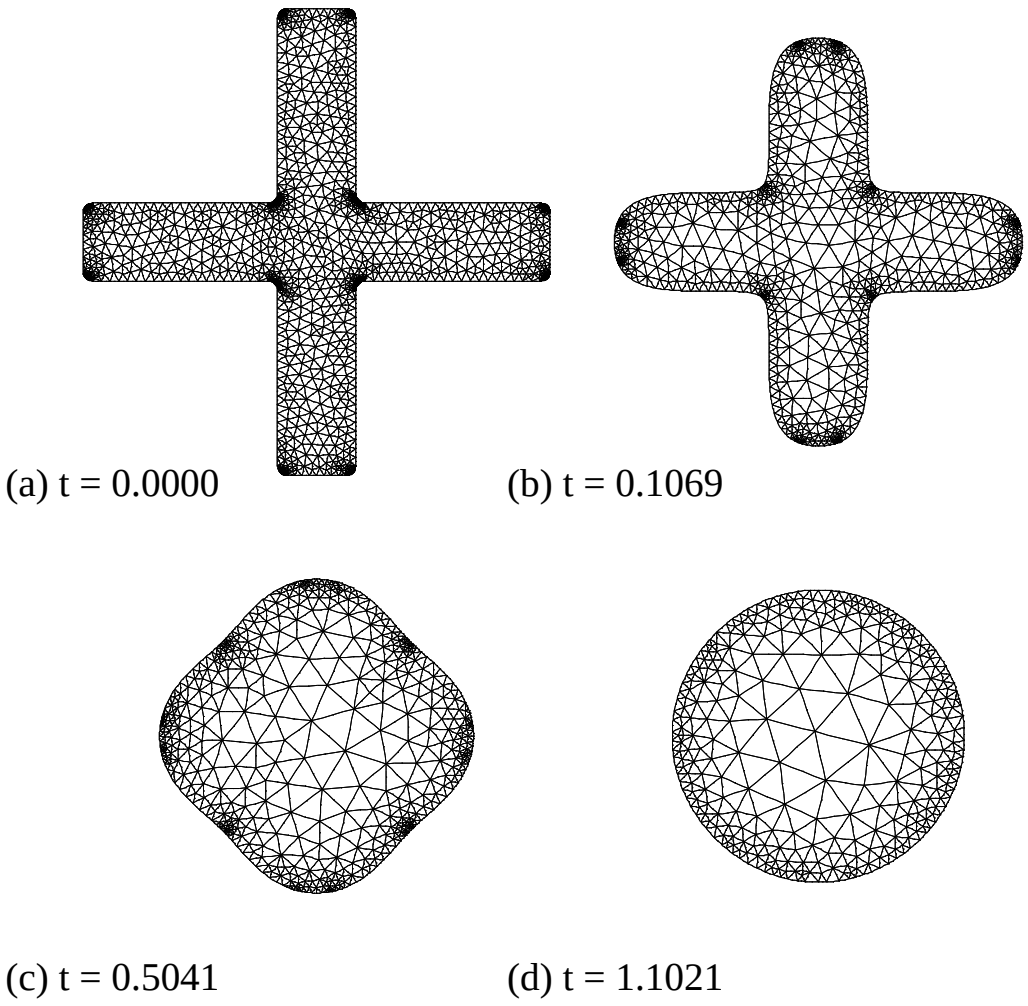


Figure 4.23: Stokes-flow evolution of a cross with rounded corners, mesh at selected times.

using the stability method. For the reasons discussed in Section 2.2.4, the corners of the initial mesh, at which the curvature would otherwise be unbounded, have been replaced with short circular arcs of radius 0.025. In principle these may be made arbitrarily small in radius. In the modelling of such a problem, the rounding of corners appears to be a reasonable modification since it is observed that, in practice, such corners rapidly evolve into smooth arcs, due to the large pressures occurring at the discontinuities. The initial shape was chosen because it contains both concave and convex corners. Initially, the greatest activity occurs near to the corners, where the pressure gradients are largest. After only a short time, i.e. by $t = 0.1069$, the ends of the arms of the cross have been replaced by near-circular arcs. Note that

the interior of the initial mesh, shown in Fig. 4.23(a) is considerably finer than that at, say, $t = 0.1069$. This is due to the use of a smaller value of h_{max} when generating the initial mesh than that employed during the remainder of the computation.

The conservative nature of the mesh derefinement algorithm employed here is apparent in the mesh shown for $t = 0.5041$, in which the sections of the free surface corresponding to the original concave corners still exhibit an apparently higher than necessary degree of refinement. By $t = 1.1021$ this has disappeared, and the mesh is close to being rotationally symmetric with regard to element size.

The success of the mesh regeneration algorithms employed in dealing with this completely different geometry is all the more impressive given that no alterations were necessary to the codes implemented and demonstrates the robustness of the methods employed. While refinement studies were not conducted for this problem, the accuracy of the computations may be gauged by considering the total change in the domain's area during the simulation. This was found to be a gain in area of approximately 0.012%, more than half of which occurred in the first 0.1 dimensionless time units of the simulation. Thus the overall change in the domain area for this problem is approximately equal to that observed for the two-cylinders problem on mesh 3.

4.5 Conclusions

In this chapter a benchmark problem has been solved and the solutions computed shown to be in good agreement with the analytical solution. Furthermore, the initial neck velocity has been demonstrated to converge at the predicted rate towards the exact solution. As far as the author is aware, this is the first time such convergence has been demonstrated conclusively for a time-dependent surface-tension-driven free-surface problem such as this.

The automatic mesh-refinement algorithm described in Chapter 2 has been demonstrated for non-trivial problems, and shown to be robust in practice. That the diagonally-preconditioned conjugate residual method may be used to reliably solve time-dependent problems over many time steps has also been shown, as has the value of simple explicit predictors in considerably reducing the number of iterations required.

The mass conservation properties of the scheme have been investigated. While the rate of mass loss/gain is sufficiently small to allow many useful computations to be carried out, the dependency of the rate of mass loss on the time-step size suggests

that higher-order free-surface advection schemes might well prove useful in reducing the rate of mass loss, without the need to employ excessively small time steps. The nature of the observed tangential-stress errors has also been investigated, and they have been shown to decrease rapidly as the free surface is refined.

Finally, the successful application of the stability method for selecting the time step has been demonstrated, and the approach has been shown to be reliable for a range of mesh resolutions.

Chapter 5

The supported-load problem

The material contained in this chapter is the subject of a paper [77], submitted to Proceedings of the Royal Society of London Series A, in September 1999.

In this chapter the behaviour of a film of viscous fluid adhering to a rotating cylinder in a gravitational field is investigated. The time-dependent form of this problem has, to the author's knowledge, never before been modelled. The maximum supportable cross-sectional area of fluid is shown to be in excellent agreement with that predicted by Hansen and Kelmanson [38], and the existence of steady free-surface profiles is confirmed for certain parameter values. At other parameter-values stable but oscillatory patterns of flow were observed, contrary to expectation. Finally the instabilities that arise when the rate of rotation of the cylinder is too small to support the film are investigated.

5.1 Background

Acheson [2] discusses the problem of finding the maximum cross-sectional area of a viscous fluid that may be supported against gravity by the steady rotation of an infinite cylinder, hereafter referred to as the *supported-load problem*. The problem's geometry is illustrated in Fig. 5.1. Moffatt [70] obtained an expression for the maximum supportable load of fluid at a given rate of rotation by making two assumptions; that flow parallel to the axis of the cylinder is negligible, and that

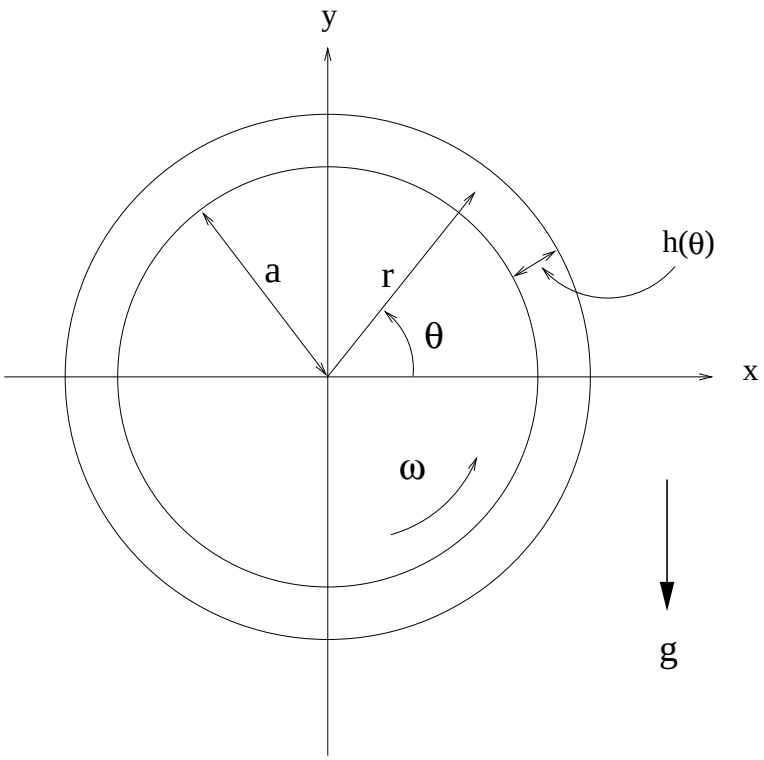


Figure 5.1: Domain for supported-load problems.

the standard approximations of lubrication theory [2] may be employed. To support the first assumption Moffatt presents experimental evidence suggesting that for sufficiently thin films no significant variation in the film thickness occurs in the axial direction; see for example his Fig. 5 [70]. The second assumption appears less satisfactory, since intuitively one would expect any thin-film flow to approximate a rigid-body motion, and for a rigid-body motion the effects of viscosity are non-existent. The solutions computed as part of the investigations described here support the hypothesis that for thin films the pattern of flow is close to that of a rigid-body, significant differences being apparent only for large supported loads.

A rigid-body flow is one in which the velocity field takes the form

$$\mathbf{u} = -\omega \mathbf{r} \times \hat{\mathbf{z}} = (-\omega y, \omega x), \tag{5.1}$$

where $\mathbf{r}$ is the position vector of a fluid particle, ω is the angular velocity of the cylinder and $\hat{\mathbf{z}}$ is a unit normal orthogonal to the x and y axes and projecting out of the page. Note that in the absence of gravity, and provided the pressure is everywhere constant, any rigid-body motion of a fluid, i.e. translations and rotations, satisfies the Stokes equations. Where surface tension is present the free surface will

have minimal surface energy only when it is circular in profile. In the absence of gravity any configuration with a circular cross-section, containing the cylinder, and rotating at the same speed about the same axis, will be an exact solution of the Stokes equations. In these circumstances the everywhere constant pressure will be that required to satisfy the condition of continuity of stress at the free surface. Where gravity is present, for a steady rigid-body motion of the fluid the governing equations reduce to

$$\nabla p = \rho \mathbf{g}, \tag{5.2}$$

the solution of which is simply $p = p_0 + \rho g y$, where ρ is the density and $\mathbf{g}$ is the acceleration due to gravity. If $\rho\mathbf{g}$ is non-zero, then p cannot be constant within the fluid, and consequently the conclusion must be drawn that the pattern of flow must deviate, if only slightly, from a rigid-body one whenever a gravitational field is present.

Interestingly, in the Navier-Stokes case, where gravity is absent a steady rigid-body solution also exists, though now the circular free surface must be coaxial with the cylinder; and since the fluid is assumed to have momentum, the pressure field must be such as to provide the necessary centripetal force. It is easy to verify that

$$p = p_0 + \rho\omega^2 \left(\frac{x^2 + y^2}{2} \right) \tag{5.3}$$

together with (5.1) gives such a solution, where the constant p_0 is again selected to give continuity of stress at the free surface. Indeed, where the effects of surface tension may be neglected, Pukhnachev [81] points out that, by an extension of a result due to Solonnikov [99], such a solution must be unique. Moffatt [70] suggests that the complex three-dimensional free-surface instabilities observed in the laboratory are primarily due to the interaction of the centrifugal and surface tension forces, and Preziosi and Joseph [79] further address this matter.

Figure 5.2 shows the maximum supportable non-dimensional cross-sectional area predicted by three different models as a function of γ, the dimensionless acceleration due to gravity defined by (5.6). Hansen and Kelmanson's results obtained with a Stokes-flow model [38] are shown using diamonds. In addition to the curves given by the thin-film analyses of Moffatt [70] and Kelmanson [57], a third curve, obtained by fitting a Laurent series expansion of degree three to Hansen and Kelmanson's Stokes-flow data, is also shown. As may be seen, for $\gamma \geq 6$, the 'thin-film' region, Hansen and Kelmanson's Stokes-flow results are in good agreement with Kelmanson's thin-film analysis, the agreement with Moffatt being good only at large

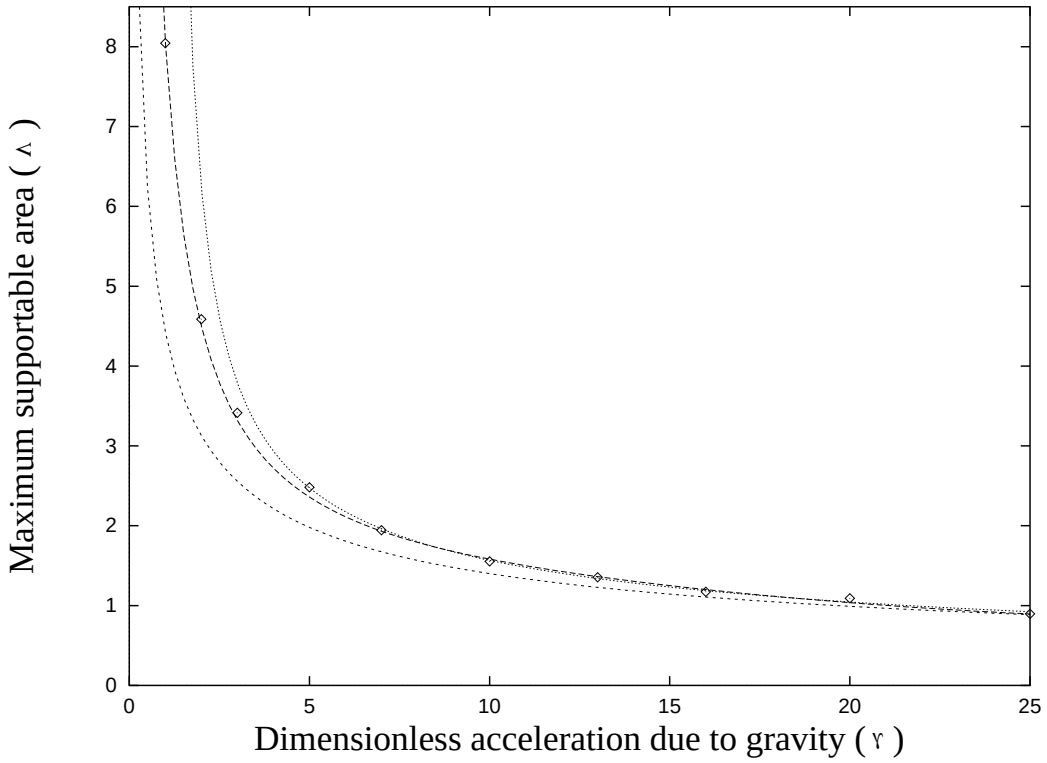


Figure 5.2: Maximum supportable cross-sectional area (Λ) as a function of γ: $\diamond$ Stokes-flow, Hansen and Kelmanson 1994 (data); – – – – Stokes-flow, Hansen and Kelmanson 1994 (LS fitted curve); · · · · · · Thin-film, Kelmanson 1994; - - - - - - Thin-film, Moffatt 1977.

($\gamma \geq 15$) values of γ. For $\gamma < 5$ Kelmanson's thin-film analysis overestimates the maximum supportable load, while Moffatt's underestimates it.

Where the cross-sectional area of the fluid is large in comparison with that of the cylinder, as in Hansen and Kelmanson's Figure 6(a) [38], it is clear that even a near-rigid-body flow requires the radial component of the velocity to be of considerable magnitude. For the configuration they show, the maximum film thickness is approximately twice the minimum film thickness and, since the kinematic boundary condition states that a material point on the free surface must always remain on the free surface, the only way that the radial velocity on the free surface can be negligible is if the tangential component of the free-surface velocity is also everywhere small. Observations conducted as part of investigations presented here appear to rule out such flow patterns, and indeed have shown that the radial velocity need not be negligible, particularly when the supported load is large and gravity forces the free-surface profile to deviate considerably from a coaxial one. This may in part explain the divergence between the predictions of Moffatt's thin-film analysis and Hansen and Kelmanson's Stokes-flow results for large supported loads, since when

the radial component of the velocity is large lubrication theory does not provide an adequate model.

Hansen and Kelmanson [38] attempted to find the maximum steady supportable-load at a given value of γ by employing a search procedure, increasing the load of fluid until their solver failed to converge to a steady-state solution within a prescribed time limit. The results they obtained suggest that the maximum supportable load is a monotonic decreasing function of γ. Caution should however be exercised when interpreting Hansen and Kelmanson results since one cannot rule out the possibility that larger supportable loads might be possible when unsteady configurations are permitted. One intuitively plausible interpretation of Hansen and Kelmanson's results is that for loads lying above the maximum-supportable-load curve no stable steady-state configuration exists, and conversely, that below the curve a stable steady-state configuration always exists. It should be noted, however, that their steady-state iterative boundary integral method gives no means of distinguishing between asymptotically stable and asymptotically unstable steady-state solutions.

Applying the non-dimensionalisation procedure employed by Hansen and Kelmanson (see Section 5.2) to the governing equations one obtains

$$\frac{\rho a^2 \omega}{\mu}\left[\frac{\partial \mathbf{u}^*}{\partial t^*} + (\mathbf{u}^* \cdot \nabla)\mathbf{u}^*\right] = \nabla^2 \mathbf{u}^* - \nabla p^* - \gamma \mathbf{j}, \tag{5.4}$$

where the superscripts $(^*)$ denote dimensionless variables, $\mathbf{j}$ is a unit vector in the positive y direction,

$$Re = \frac{\rho a^2 \omega}{\mu} \tag{5.5}$$

is the Reynolds number and

$$\gamma = \frac{\rho g a}{\omega \mu} \tag{5.6}$$

is the *Stokes number* — the dimensionless acceleration due to gravity. Hansen and Kelmanson [38] calculate a Reynolds number of order 10^{-1} for an apparatus similar to that described by Moffatt [70] and, one infers, a value for γ of order 1. Since for most practical purposes g is fixed, the only way $\frac{\gamma}{Re}$ may be made large, and thus the approximation a good one, is if a and ω are made small. Nevertheless the Stokes-flow model is still of considerable fundamental interest and an important first step in the modelling of the full three-dimensional Navier-Stokes problem.

5.2 The Stokes flow model

For a Stokes-flow problem the only initial condition required is the specification of the initial free-surface profile. Intuitively it was expected that any initial free-surface profile that was sufficiently close to a steady-state profile would rapidly converge towards it, resulting, asymptotically, in a steady solution. As will be described shortly, this does not always happen.

The first difficulty that has to be overcome in the study of the time-dependent form of the supported-load problem is that of selecting suitable initial conditions. The simplest initial condition, and the one adopted here, takes the form of a circular free-surface coaxial with the rotating cylinder. Thus the initial condition is specified by a single parameter, the mean film thickness $\bar{h}$. Alternatively, the cross-sectional area or load Λ, given by

$$\Lambda = \pi((a+\bar{h})^2 - a^2) \tag{5.7}$$

may be specified, where a is the radius of the cylinder. One reason for preferring an initially circular free-surface profile lies in the fact that such an experiment could in principle be carried out in the laboratory by establishing a steady flow on a cylinder at a large angular velocity, giving an approximately circular free-surface profile, and then suddenly reducing the angular velocity of the cylinder.

In their analysis Hansen and Kelmanson [38] start by assuming the existence of a steady solution. They also assume that the Reynolds number is sufficiently small that momentum may be neglected. Thus, non-dimensionalising using the velocity, time and stress scales $U_0 = \omega a$, $T_0 = \omega^{-1}$ and $S_0 = \mu\omega$ they obtain

$$\nabla^2 \mathbf{u}^* - \nabla p^* - \gamma \mathbf{j} = 0. \tag{5.8}$$

where γ is the dimensionless acceleration due to gravity, defined by (5.6).

Note that in the dimensionless model, the cylinder has unit radius and unit angular velocity, taking 2π time units to complete a rotation. The flow is assumed to satisfy a no-slip boundary condition on the cylinder, i.e.

$$\mathbf{u}^* = (-\sin\theta, \cos\theta). \tag{5.9}$$

The free-surface stress boundary condition, $\boldsymbol{\sigma}$, is given by surface tension and has the form

$$\boldsymbol{\sigma} = -\sigma k \mathbf{n} \tag{5.10}$$

where k is the curvature of the cylinder, $\mathbf{n}$ is the outward free-surface normal, and σ is the coefficient of surface tension. On non-dimensionalisation (5.10) becomes

$$\boldsymbol{\sigma}^* = \frac{\sigma}{a\mu\omega}\frac{1}{R_c{}^*} = \alpha\frac{1}{R_c{}^*}, \tag{5.11}$$

where R_c is the radius of curvature and α is the dimensionless surface tension, defined by

$$\alpha = \frac{\sigma}{\omega\mu}. \tag{5.12}$$

Hansen and Kelmanson [38] report that the free-surface profiles they obtain are, to a large extent, independent of the surface tension, i.e. that the difference between the free-surface profiles obtained with $\alpha = 0$ and $\alpha = 100$ is small. Furthermore, they report that near symmetry of the free surface about the x axis is apparent, even when the surface tension is unrealistically large, e.g. with $\alpha = 100$. The inclusion of some surface tension has however proved helpful in the current time-dependent scheme, since it appears necessary to prevent instabilities of the free surface from arising. Bearing these observations in mind it was decided to employ a fixed value of $\alpha = 1$ throughout the current investigations.

Once α has been fixed the problem as formulated here is completely specified by the two-dimensional parameter space $\{\Lambda \times \gamma\}$. Hansen and Kelmanson employ an alternative parametrisation, specifying the dimensionless tangential flux ${\psi_0}^*$ that their steady-state solution must satisfy. The dimensionless flux, at an angle θ, is defined as

$$\psi^*(\theta) = -\int_{r=1}^{r=1+h^*(\theta)} \mathbf{u}^* \cdot (-\sin\theta, \cos\theta)\, dr, \tag{5.13}$$

where the integral is evaluated along a radius of the cylinder. Note that for any steady flow the flux will be independent of θ. In the time-dependent case however, the flux will vary with both time and θ. Furthermore, even if a steady-state solution is located, the flux will in general be different from that found for the initial free-surface configuration. This makes it difficult to obtain solutions that closely match those of Hansen and Kelmanson without a costly iterative search. In practice however, where a steady-state solution is found, the asymptotic flux appears not to differ too greatly from the initial flux, allowing at least qualitative comparisons to be made. Given an initially coaxial free-surface profile, and in the absence of gravity, the dimensionless initial flux may be computed using

$${\psi_0}^* = (\bar{h}^{*2} + 2\bar{h}^*), \tag{5.14}$$

and, conversely, the dimensionless initial mean film-thickness $\bar{h}^*$ may be computed using

$$\bar{h}^* = \sqrt{1 + \psi^*{}_0} \quad - 1. \tag{5.15}$$

Note that, in the remainder of this chapter, the superscripts will be dropped, and the symbols Λ, h, $\bar{h}$, ψ_0 and t will refer to the corresponding dimensionless quantities.

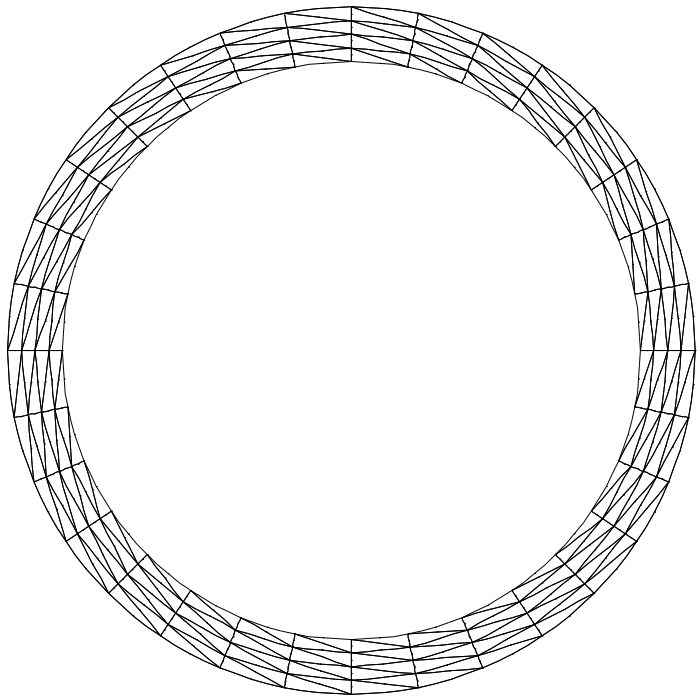


Figure 5.3: Initial mesh (I) for preliminary investigations, 32×5 vertices, $\Lambda = 1.3$.

5.3 Method

For the preliminary investigations a mesh of fixed connectivity was employed, as shown in Fig. 5.3. The initial cross-sectional area of the film, $\Lambda = 1.3$, corresponds to an initial film thickness of approximately $\bar{h} = 0.189$. Statistics for the meshes employed here are given in Table 5.1. Elements that are considerably longer in the circumferential than in the radial direction are employed so as to reduce the number of unknowns. Note that Hansen and Kelmanson employ 32 collocation points on the free surface in their boundary-element scheme, and thus one might reasonably expect that the present scheme will be of similar accuracy. For this, and the other preliminary experiments, a fixed time step of length $k = 0.005$ was

employed. That such a relatively large time step could be employed appears to reflect the fact that here the normal component of the free-surface velocity is typically small. The interior of the mesh was updated at each time step using Lynch's elastic-mesh method. Jacobi-smoothing was not employed for this problem, since it was found that its use resulted in motion of the interior vertices towards the cylinder, causing the elements nearest to the cylinder to become compressed. Thus, eventually, the isoparametric discretisation breaks down near the cylinder. The backward-Euler form of the semi-implicit scheme described in Section 3.7 was employed. An ILUT preconditioner (Section 3.12.2) was recomputed every ten time steps, taking $lfil = 200$ and $droptol = 10^{-6}$. All linear systems were solved to an absolute tolerance of 10^{-10}.

Mesh	N_θ	N_r	Elements	Unknowns
I	32	5	256	1184
II	64	5	512	2368
III	128	5	1024	4736
IV	64	3	256	1216
V	64	9	1024	4672
VI	64	7	768	3520

Table 5.1: Initial mesh statistics.

5.4 Results

Figure 5.4 shows the evolution of the film thicknesses at $\theta = 0°$, $90°$, $180°$ and $270°$ for a computation carried out using mesh I, with a cross-sectional area of fluid $\Lambda = 1.3$, a non-dimensional acceleration due to gravity $\gamma = 12.5$ and a time step of $k = 0.005$. While at this value of γ Kelmanson's analytic predication for the maximum supportable load is in good agreement with Hansen and Kelmanson's Stokes-flow computations, Moffatt's prediction is considerably lower. As may be seen, the oscillations in the four film thicknesses decay rapidly and suggest that the flow is converging towards an asymptotically steady configuration. Figure 5.5, which shows the L_∞ norm of the residual at the start of each time step, appears to confirm that such a configuration will indeed be a steady one. The spikes visible in Fig. 5.5 occur occasionally when the preconditioner is recomputed, and are apparently due to inaccuracies in the incomplete LU factorisation. As the steady solution is

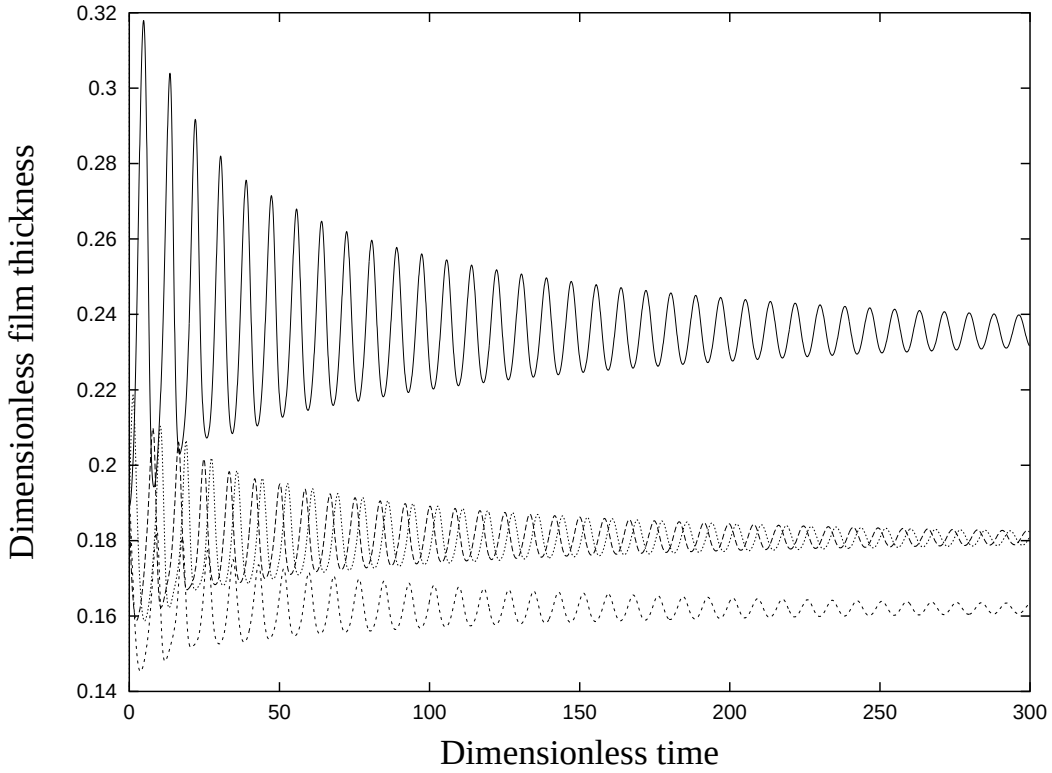


Figure 5.4: Free-surface evolution with $\Lambda = 1.3$, $\gamma = 12.5$, film thicknesses: —— 0°; – – – 90°; - - - - 180°; · · · · · · 270°.

approached the film thicknesses at $\theta = 90°$ and $\theta = 270°$ converge towards nearby limits, the difference in the limits at $t = 300$ being approximately 0.75%. This suggests that the free-surface profile is approximately symmetric in $y = 0$. That the free-surface profile is indeed nearly symmetric may be seen from Fig. 5.6, in which both the free surface and its reflection in the x axis are shown. Exact symmetry is not however expected, in part due to asymmetry of the mesh, but also since here surface tension is not negligible [38].

Plotting the velocity field for the steady-state solution described above is unrevealing, in that, for a film of this thickness, no detectable deviation from a rigid-body motion is apparent. The pressure field is however more interesting. Figure 5.7 shows the pressure field arising at $t = 333$. This confirms at least qualitatively that, as predicted by Hansen and Kelmanson [38], the pressure field is approximately anti-symmetric in $y = 0$. Note that in Fig. 5.7 the maximum pressure of 2.348 occurs on the upper part of cylinder, while the minimum pressure of -0.671 occurs on the lower part of the cylinder. Table 5.2 gives the asymptotic film thicknesses, as estimated from the data shown in Fig. 5.4 (FE), together with the film thicknesses predicted by thin-film theory (TT) [38] and the percentage difference between the

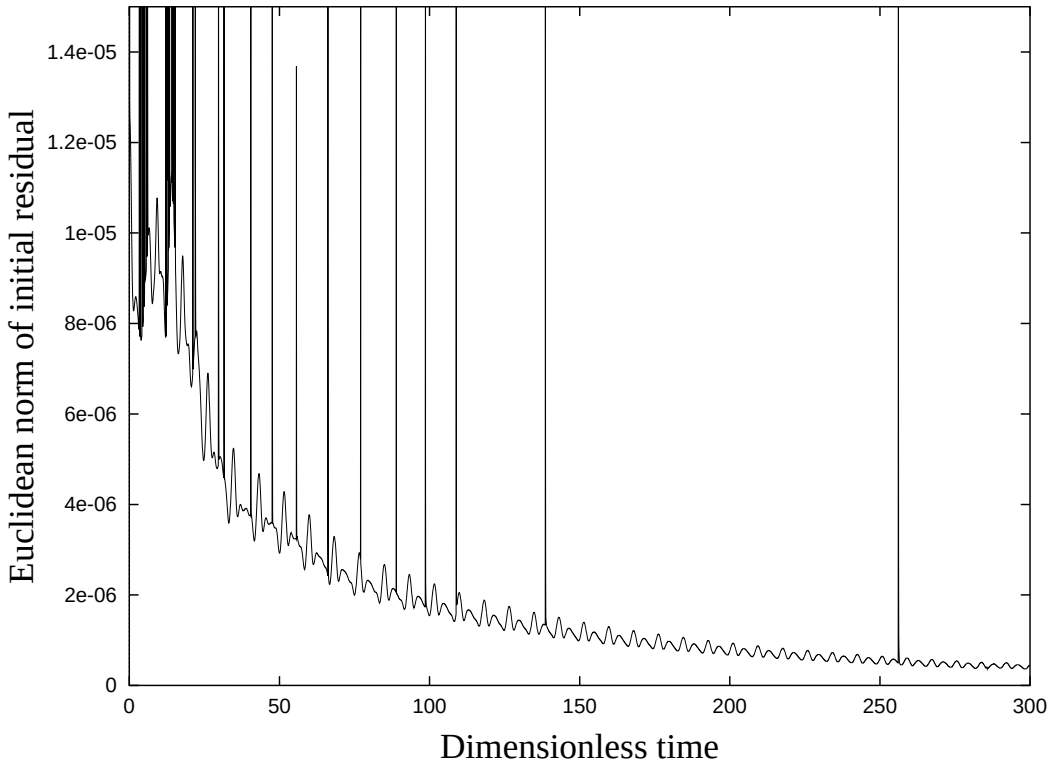


Figure 5.5: L_∞ norm of the residual for an asymptotically steady problem, $\Lambda = 1.3$, $\gamma = 12.5$.

θ	h (FE)	h (TT)	% difference
0	0.236	0.227	4.0
90	0.180	0.190	-5.3
180	0.161	0.153	5.2
270	0.180	0.190	-5.3

Table 5.2: Asymptotic film thicknesses $\Lambda = 1.3$, $\gamma = 7.5$: finite element method (FE); thin-film theory (TT).

two. As these show, the agreement with thin-film theory is reasonably good, the discrepancies being of the order of 5% in all four cases.

When the above experiment was repeated using different values of Λ and γ, a range of behaviours were observed. Figures 5.8 and 5.9 show the results of two simulations in which Λ was reduced, γ being held fixed. As may be seen, with $\Lambda = 1.2$, essentially the same behaviour was observed as in the previous experiment, though now the rate of damping is much reduced. At $\Lambda = 0.75$, however, convergence did not occur and oscillations of increasing amplitude were observed. *This behaviour was totally unexpected.* Numerous further investigations, both at different values of Λ and γ, and employing meshes of different resolutions and different time-step

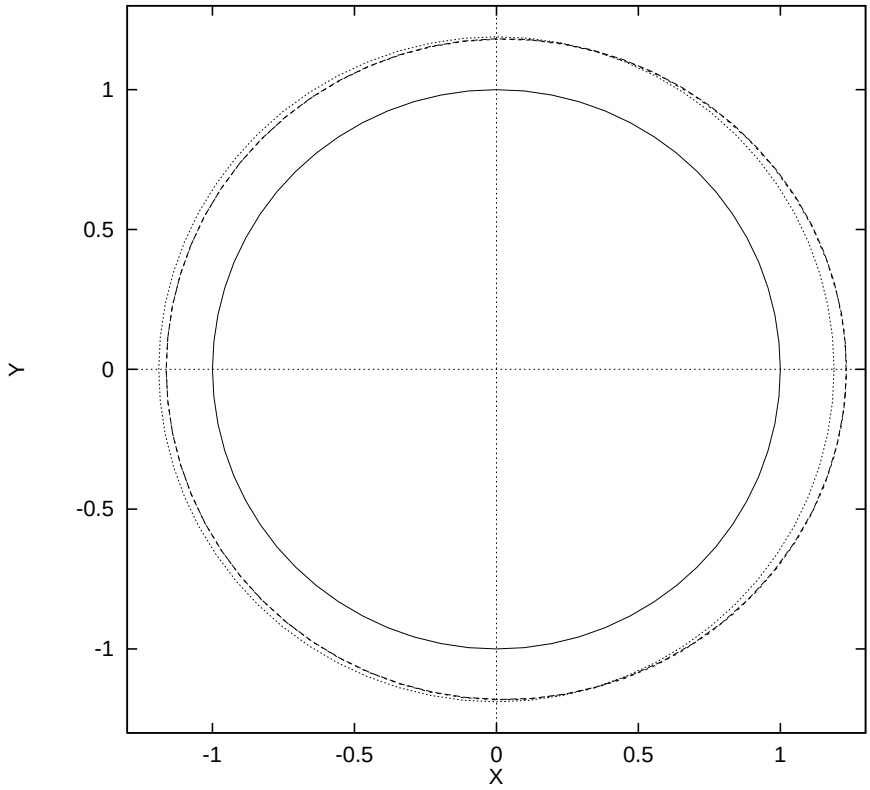


Figure 5.6: Free-surface symmetry $\Lambda = 1.3$, $\gamma = 12.5$: —— cylinder; – – – free surface; - - - - - free surface reflected in $y = 0$; $\cdots\cdots$ initial free surface.

sizes all however appear to confirm that such oscillatory behaviour is a genuine phenomenon. Furthermore, such oscillatory behaviour was found regardless of the exact shape of the initial free-surface profile. Thus, for example, if the free-surface profile given by Kelmanson's steady-state thin-film approximation [57] was employed, similar oscillations were observed, though of smaller amplitude. Modifying the problem, so that γ was increased gradually from zero to its prescribed value over one or more periods of rotation of the cylinder, was similarly found to have no effect on the qualitative nature of the observed behaviour. Note, however, that in many cases where oscillatory behaviour was observed, the rate of increase in the amplitude of oscillation was not so marked as that apparent in Fig. 5.9. In some cases the amplitude would initially grow but then stabilise, in other cases more complex behaviours were observed. The conclusion was eventually reached that there appears to be a large region of the parameter space $\{\Lambda, \gamma\}$ in which stable oscillatory solutions can arise, and that while the existence of steady-state solutions in this region cannot be ruled-out, convergence towards them, even from nearby points in the solution space, may be arbitrarily slow.

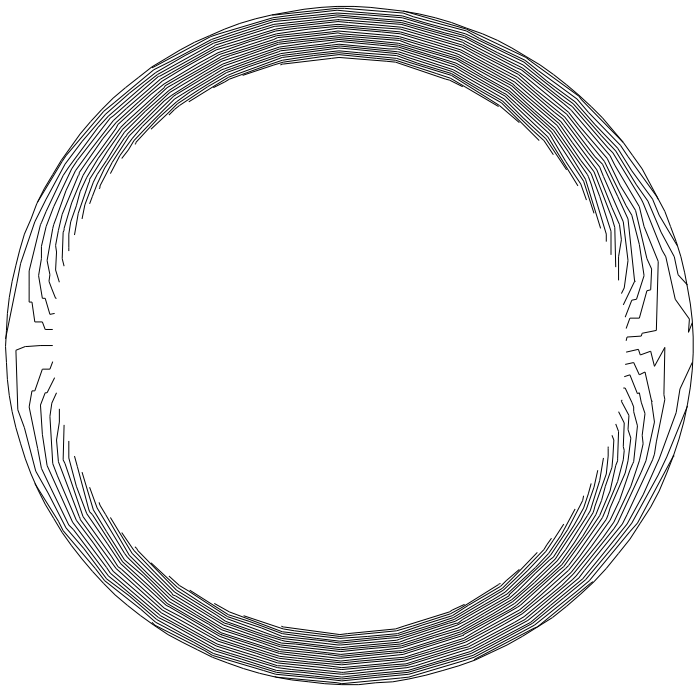


Figure 5.7: Pressure field near steady-state on Mesh I at $t = 333.0$, $\Lambda = 1.3$, $\gamma = 12.5$.

Figure 5.10 shows the results of experiments carried out at a number of points in the parameter space. Points at which convergence to a stable steady-state free-surface configuration was observed are shown using crosses. Squares correspond to points for which no stable configuration was found, the simulations being halted only when it was clear that fluid would be lost from the cylinder. Finally, diamonds correspond to points at which no clear, unequivocal evidence of either instability or convergence was seen. At these points oscillatory behaviours were observed with damping being either absent or negative. Where negative damping was observed the rate of growth of the amplitude of the oscillations was typically small, being of the order of a few percent per rotation of the cylinder. Furthermore, in such cases the growth was often observed to be transient, the amplitude stabilising at a slightly higher value than the original. Such solutions are referred to here as stable oscillatory solutions.

From Fig. 5.10 three regions of the parameter space are apparent. These are labelled A, B and C as shown in Fig. 5.11, the upper curve representing Hansen and Kelmanson's maximum-supportable-load data. For problems in region B, i.e. close to Hansen and Kelmanson's maximum-supportable-load curve, convergence towards

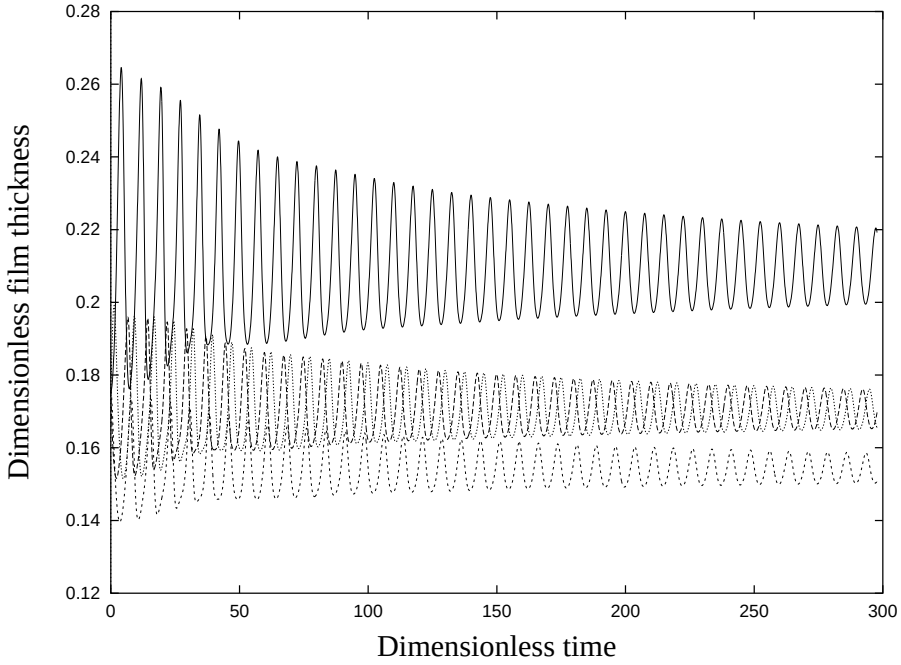


Figure 5.8: Free-surface evolution with $\Lambda = 1.2$, $\gamma = 12.5$: film thicknesses: ——— 0°, – – – 90°, - - - - 180°, · · · · · · 270°.

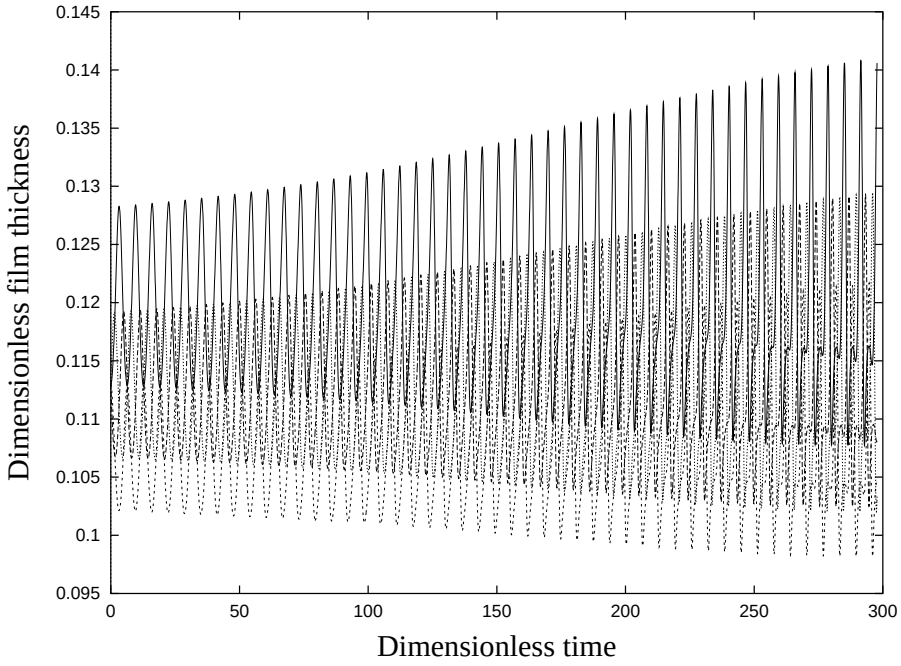


Figure 5.9: Free-surface evolution with $\Lambda = 0.75$, $\gamma = 12.5$: film thicknesses: ——— 0°, – – – 90°, - - - - 180°, · · · · · · 270°.

an asymptotically steady free-surface profile was always observed. In region C only oscillatory solutions were obtained, and in region A fluid was invariably shed by the cylinder. Mapping the boundary between regions A and B is relatively easy,

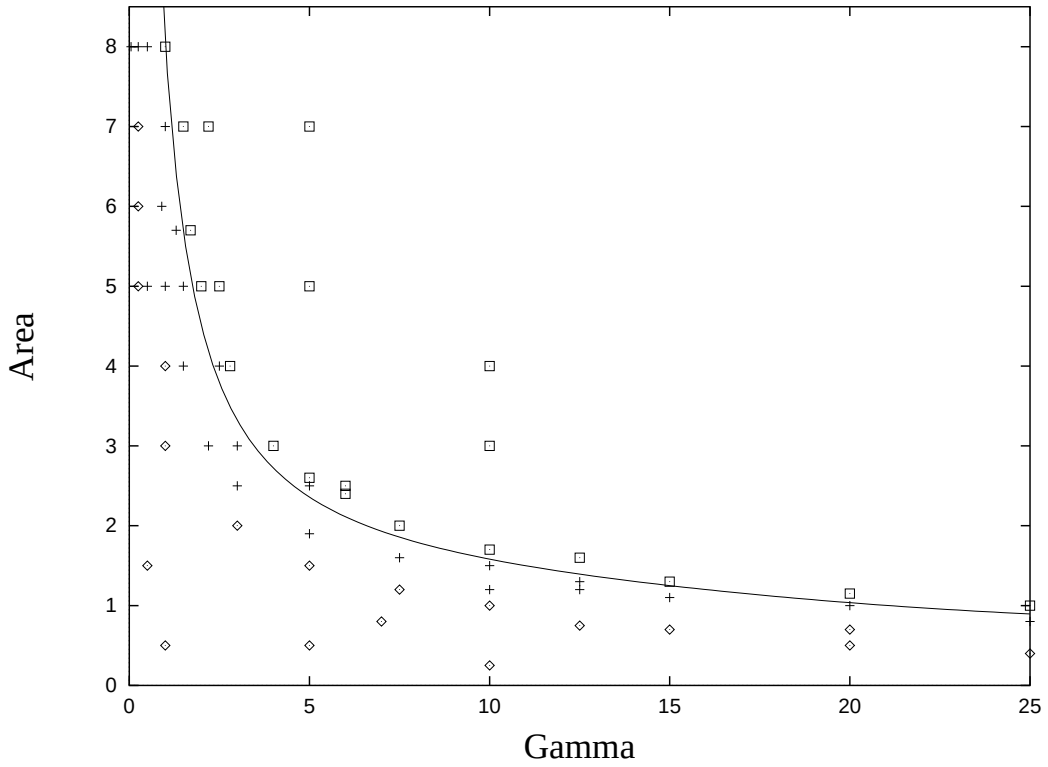


Figure 5.10: Problem parameter space, solution type: □ unstable; + stable, asymptotically steady; ◇ stable, oscillatory; —— least-squares fit to Hansen and Kelmanson's maximum-supportable-load data.

particularly if Hansen and Kelmanson's results are employed as a guide. For points near to this boundary simulations typically show clear signs of either convergence or instability within one or two rotations of the cylinder. Exact reproduction of Hansen and Kelmanson's maximum-supportable-load curve is however impossible using the approach employed here, since the precise curve obtained presumably depends on the initial conditions employed. Mapping the boundary between regions B and C is considerably more difficult, since a simulation may have to be continued for many rotations of the cylinder before it becomes apparent that convergence is not going to occur. Note that the decision to include a point in region C is based upon a negative result — the non-occurrence of any sign of convergence within ten rotations of the cylinder — and thus one cannot rule out entirely the possibility that convergence might occur after some much longer period of time. Simulations of such configurations over more than 100 rotations of the cylinder have led to the conclusion that where oscillatory solutions occur they are stable. The boundary between regions B and C appears to be a simple continuous one, similar in form to Hansen and Kelmanson's maximum-supportable-load curve, but displaced towards

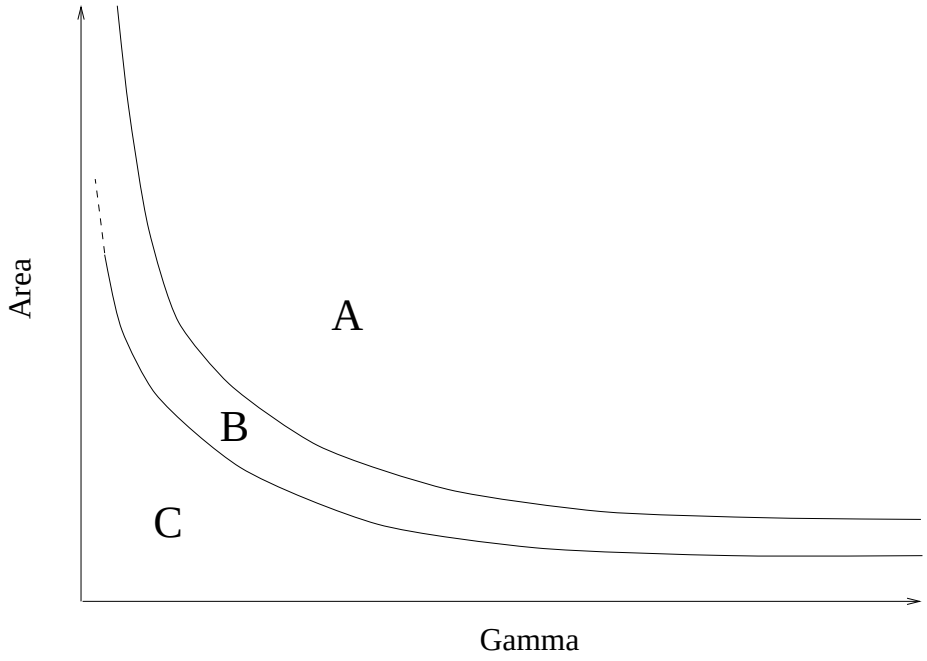


Figure 5.11: Regions of the parameter space: A — unstable; B — stable, asymptotically steady; C — stable, oscillatory.

the origin. A more sophisticated approach to locating the boundary between regions B and C might involve plotting the observed, asymptotic, damping constant as a function of Λ and γ, the contour corresponding to a damping constant of zero presumably giving the boundary. One hypothesis that might be considered is that the boundary is illusory; the rate of damping in region C being merely very small, numerical error masking convergence. The numerical experiments described in the following section would however appear to refute this interpretation. Furthermore, note that while attempts were made to obtain steady-state solutions by employing numerical continuation in γ and Λ from previous steady-state solutions, this was only successful for problems lying in region B.

From Figs. 5.4 and 5.8 it appears that where convergence occurs the amplitude of the oscillation might be exponentially damped, i.e. that the rate of decay is proportional to the amplitude of the oscillation. Figure 5.12 was obtained by plotting the logarithm of the maximum film-thickness at 0° against time for the two asymptotically steady problems described above. As may be seen, asymptotically at least, the rate of damping does appear to be exponential, though initially it appears to be somewhat faster. The straight lines shown in Fig. 5.12 were fitted by eye, for illustrative purposes only. Discovering the precise form of the damping curve would require more sophisticated methods than those employed here, since plotting Fig.

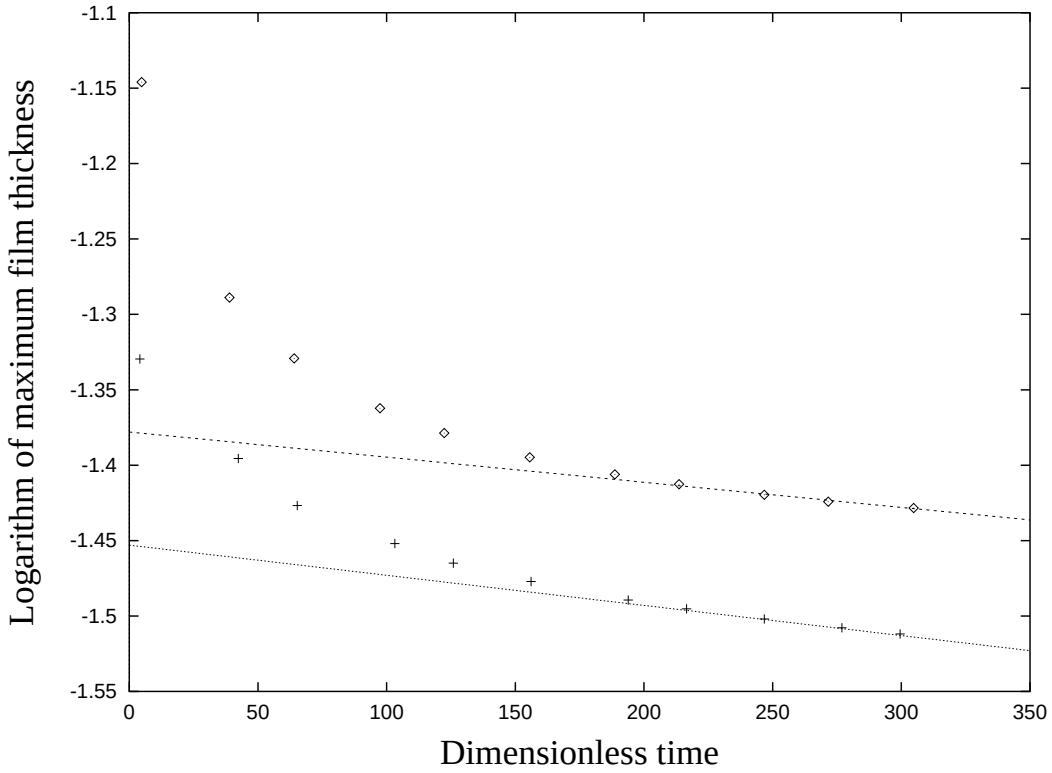


Figure 5.12: Rate of decay of oscillations in film thickness, $\gamma = 12.5$. Maximum film thickness at $\theta = 0$: $\diamond$ $\Lambda = 1.3$; + $\Lambda = 0.75$.

5.12 with, for example, $log(h^2)$ rather than $log(h)$ as the dependent variable results in a curve that is very similar in appearance.

Further investigation shows that, in general, the period of the oscillation in film thickness at $\theta = 0°$ is somewhat greater than that of the rotation of the cylinder — a somewhat surprising observation. For the problem shown in Fig. 5.4 the dimensionless duration of the first period is approximately 8.8, and for the second period 8.45. Thereafter the period decreases gradually until, at $t \approx 320$, the period has fallen to steady value of approximately 8.3, i.e. 32% greater than that of the cylinder. For the problem shown in Fig. 5.8 the initial period is approximately 7.85, falling to approximately 7.5 by time $t = 300$, i.e. approximately 19% greater than that of the cylinder. Finally, the oscillatory solution shown in Fig. 5.9 was found to have an approximately constant period of 6.45 over the first 45 rotations of the cylinder, a value only 2.7% greater than that of the cylinder. In the later stages of this latter problem measurement of the period was complicated by the presence of a higher-frequency component to the oscillations with a period approximately half that of the main component. This may be observed in Fig. 5.9.

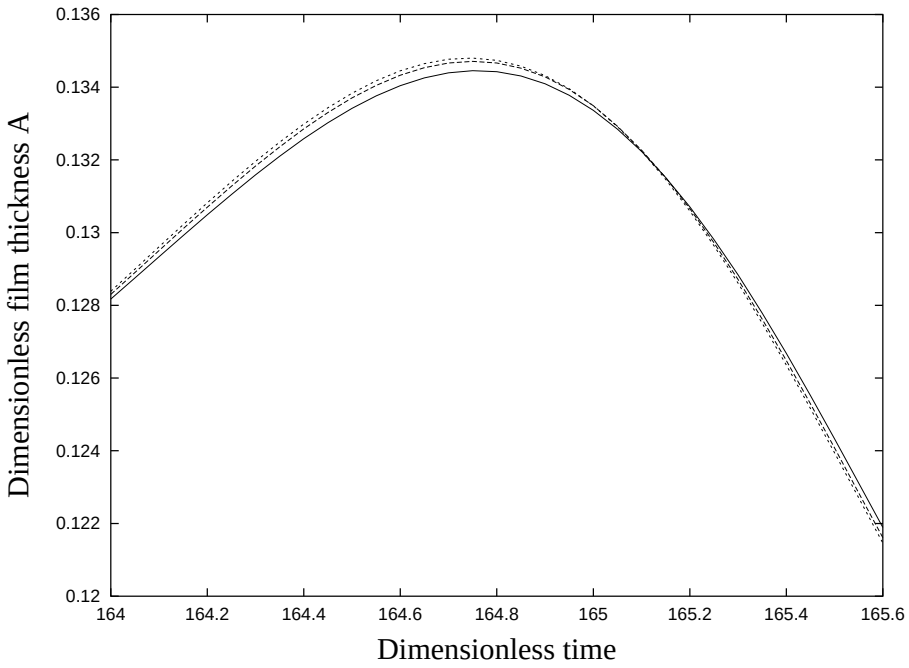


Figure 5.13: The effect of refining the mesh in the circumferential direction on the film thickness at 0°, $\Lambda = 0.75$, $\gamma = 12.5$: —— mesh I; – – – mesh II; - - - - mesh III.

5.4.1 Accuracy

A number of experiments were carried out to ascertain the effects of mesh resolution and time-step size on accuracy. These were carried out using the parameter values $\Lambda = 0.75$ and $\gamma = 12.5$, corresponding to the oscillatory solution shown in Fig. 5.9. First the effect of refining the mesh in the circumferential direction was investigated. The experiment was repeated using meshes II and III with respectively 64 and 128 vertices in the circumferential direction. A time step of $k = 0.005$ was employed. Figure 5.13 shows the evolution of the film thickness at $\theta = 0°$ for meshes I, II and III, after more than 25 rotations of the cylinder. As may be seen, doubling or even quadrupling the number of elements in the circumferential direction resulted in only a small increase in the maximum film thickness computed in the time interval shown. Doubling the number of elements increased the computed maximum film thickness by only 0.23%. Further doubling the number of elements in the circumferential direction increased the maximum film thickness by less than half this amount.

The effect of refining the mesh in the radial direction was next investigated. This time the above experiment was repeated using meshes IV, II and V with, respectively, 3, 5 and 9 vertices in the radial direction. Again a time step of $k = 0.005$ was employed. Figure 5.14 shows the evolution of the film thickness at $\theta = 0°$ for meshes II, IV and V, after six rotations of the cylinder. Note that the piecewise-linear

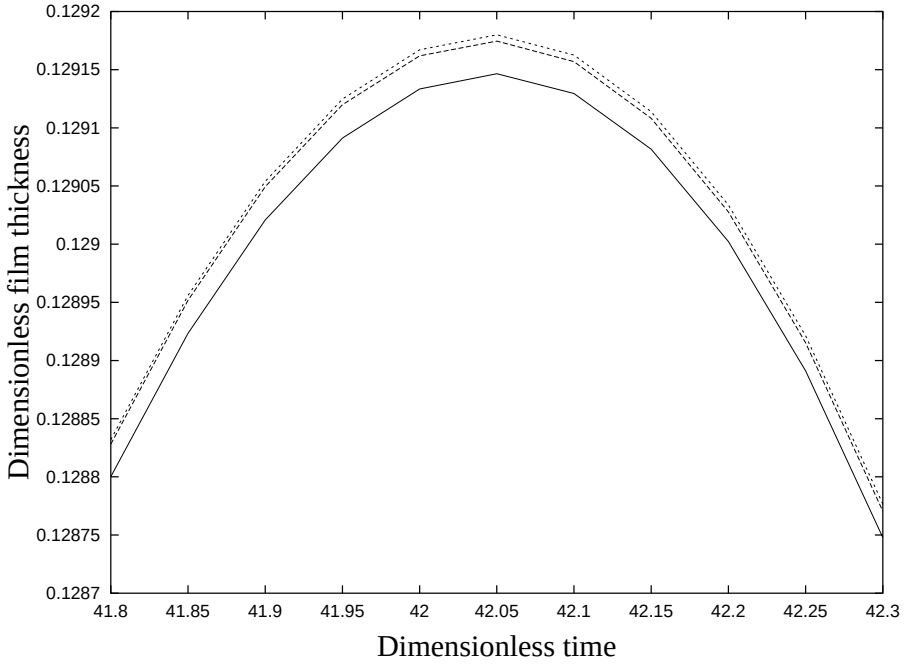


Figure 5.14: The effect of refining the mesh in the radial direction: on the film thickness at 0°, $\Lambda = 0.75$, $\gamma = 12.5$: —— mesh IV; – – – mesh II; - - - - mesh V.

appearance of the curves shown here is a result of the fact that the film thicknesses were recorded only every ten time steps. The increase in the film thickness seen when doubling the number of elements in the radial direction, i.e. moving from mesh II to mesh V, is of the order of 0.016%.

Finally the effect of employing shorter time steps was investigated. The experiment was repeated, using mesh I, with time steps $k = 0.0025$, $k = 0.00125$ and $k = 0.000625$. Figure 5.15 shows the evolution of the film thickness at $\theta = 0°$ over the first eleven rotations of the cylinder, Fig. 5.16 a detail. Clearly, reducing the time step has a considerable impact on the solution obtained. Indeed from Fig. 5.15 it appears that even at $k = 0.00125$ growth in the amplitude of the oscillation is almost negligible. For the solution obtained with a time step of $k = 0.000625$ the maximum film thickness occurring during the first rotation of the cylinder is 0.128181, while that occurring after ten rotations of the cylinder is slightly greater at 0.128215; an increase of approximately 0.027%. It thus appears that for problems in region C of the parameter space, the solutions obtained are particularly sensitive to the accuracy of the time integration scheme employed. A higher-order scheme, such as the explicit Adams-Bashforth two-step method [16], would thus appear attractive in that it might well allow considerable improvements in accuracy to be obtained without the use of excessively small time steps.

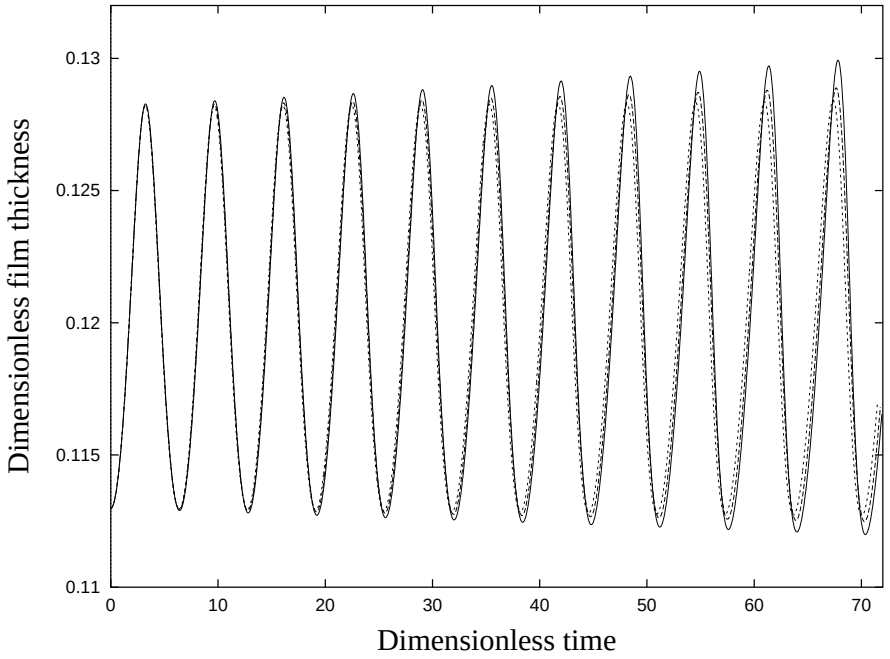


Figure 5.15: The effect of reducing time-step size, mesh I: —— $k = 0.005$; – – – $k = 0.0025$; - - - - - $k = 0.00125$.

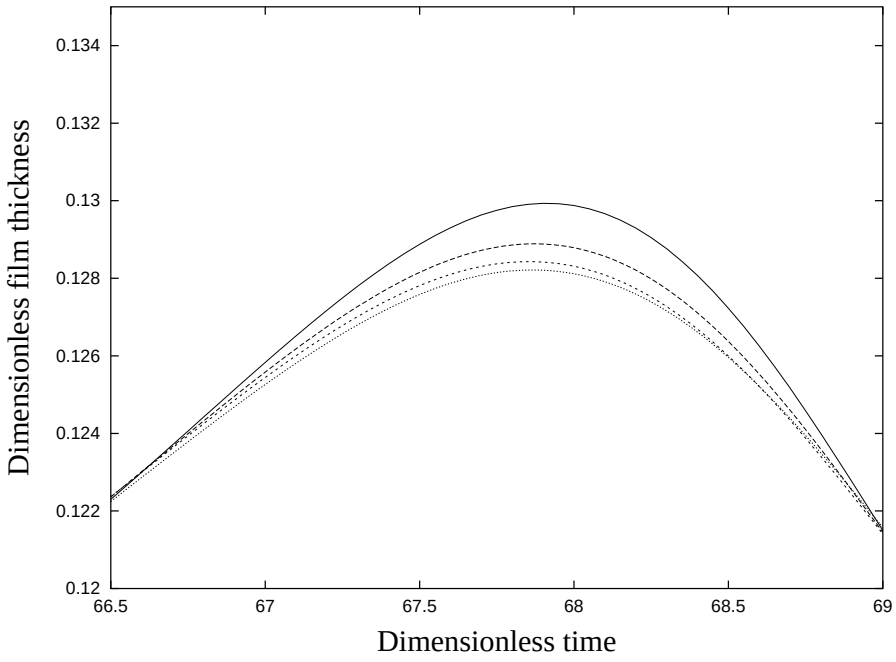


Figure 5.16: The effect of reducing the time-step size, mesh I (detail): —— $k = 0.005$; – – – $k = 0.0025$; - - - - - $k = 0.00125$; · · · · · · $k = 0.000625$.

5.4.2 Further investigation of oscillatory solutions

In order to further investigate the oscillatory solutions observed in region C of the parameter space, the kinetic energy (KE), gravitational potential energy (PE) and

surface energy (SE) were next considered. The dimensionless kinetic, potential and surface energies per unit length of the cylinder were computed using

$$KE = \int_\Omega \frac{1}{2} v^2 d\Omega, \tag{5.16}$$

$$PE = -\int_\Omega \gamma y \, d\Omega, \tag{5.17}$$

$$SE = -\int_{\partial\Omega} \alpha \, ds, \tag{5.18}$$

where the initial gravitational potential energy is defined, arbitrarily, to be zero. Note that in this system the total energy, $TE = KE + PE + SE$, is not in general constant. While kinetic energy may be dissipated due to viscosity it may also be transferred to the fluid from the cylinder through the mechanism of viscosity. Only when a steady-state solution arises will the total energy remain constant. Linear and angular momentum are also, in general, not conserved.

Figures 5.17 and 5.19 show the evolution of the components of the energy for two of the above problems. In both cases the greatest variations occur in the potential energy. In Fig. 5.17, which corresponds to a point in region B of the parameter space, periodic damped oscillations in both the kinetic and potential energy may be seen. In Fig. 5.19, which corresponds to a point in region C of the parameter space, no oscillation is apparent in the kinetic energy, while that in the potential energy appears to be close to sinusoidal, and of fixed amplitude. In both cases the surface energy remains very nearly constant.

Figures 5.18 and 5.20 show the variation of mass with time for the two problems. For the asymptotically steady problem, the change in the mass over the period shown is approximately $1.8 \times 10^{-3}\%$, the rate of mass gain being greatest when the amplitude of the oscillations is largest. For the oscillatory problem the rate of mass gain is approximately constant, the overall change over the first eleven rotations of the cylinder being approximately $8 \times 10^{-4}\%$.

5.4.3 Large supported loads

Large supported loads were next considered, the intention being to observe the velocity field directly and thus gain insight into the mechanisms involved in supporting the fluid. A new mesh (VI) with 32 vertices in the circumferential direction and 7 vertices in the radial direction was employed for this problem. Parameter values of $\Lambda = 5.7$ and $\gamma = 1.3$ were chosen, corresponding to a point in region B. Figure 5.21

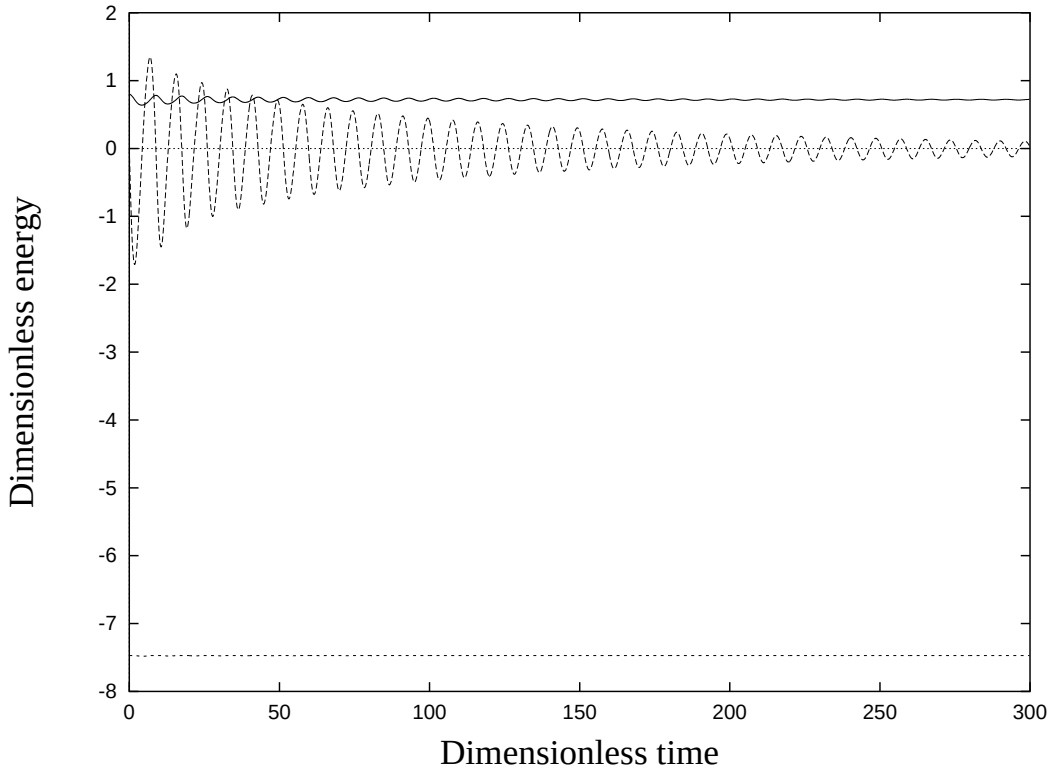


Figure 5.17: Potential, surface and kinetic energy region B, $\Lambda = 1.3$, $\gamma = 12.5$: ——— kinetic energy; – – – potential energy; - - - - - surface energy.

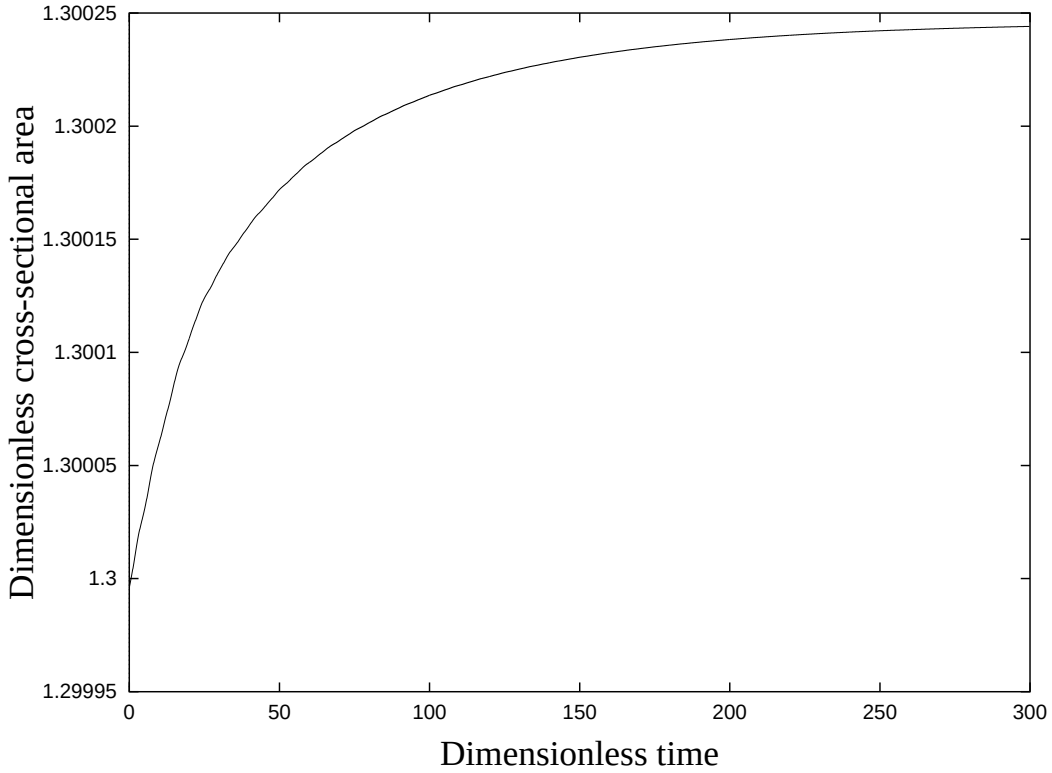


Figure 5.18: Conservation of mass region B, $\Lambda = 1.3$, $\gamma = 12.5$.

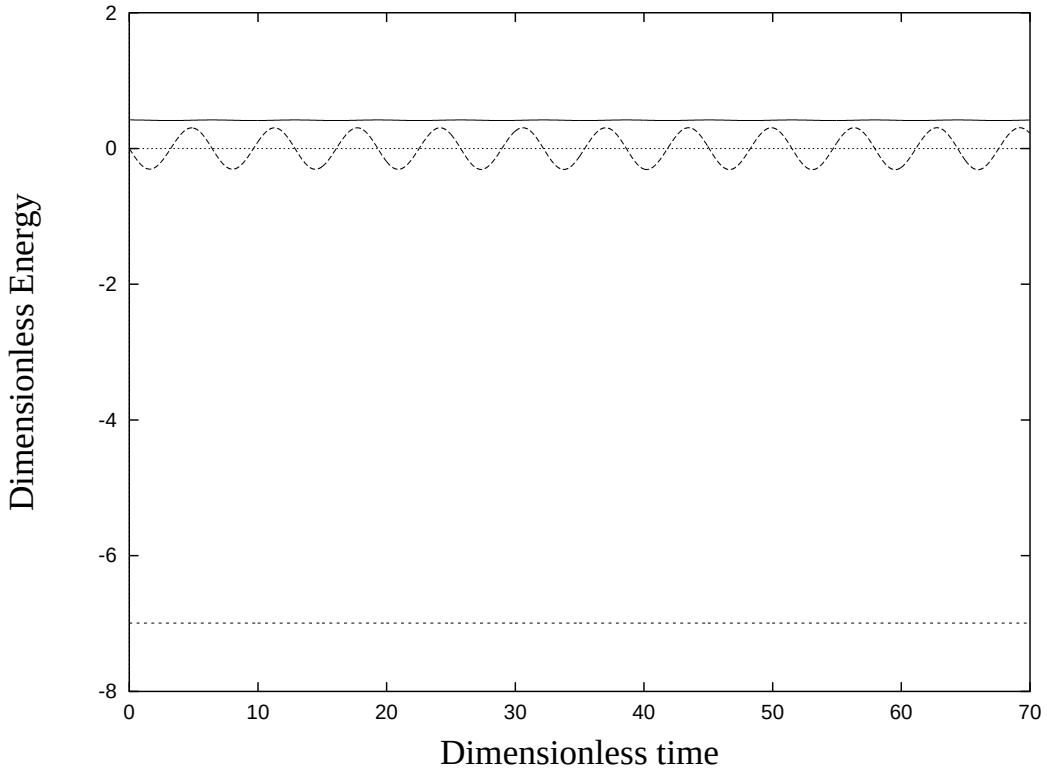


Figure 5.19: Potential, surface and kinetic energy region C, $\Lambda = 0.75$, $\gamma = 12.5$: ——— kinetic energy; – – – potential energy; - - - - - surface energy.

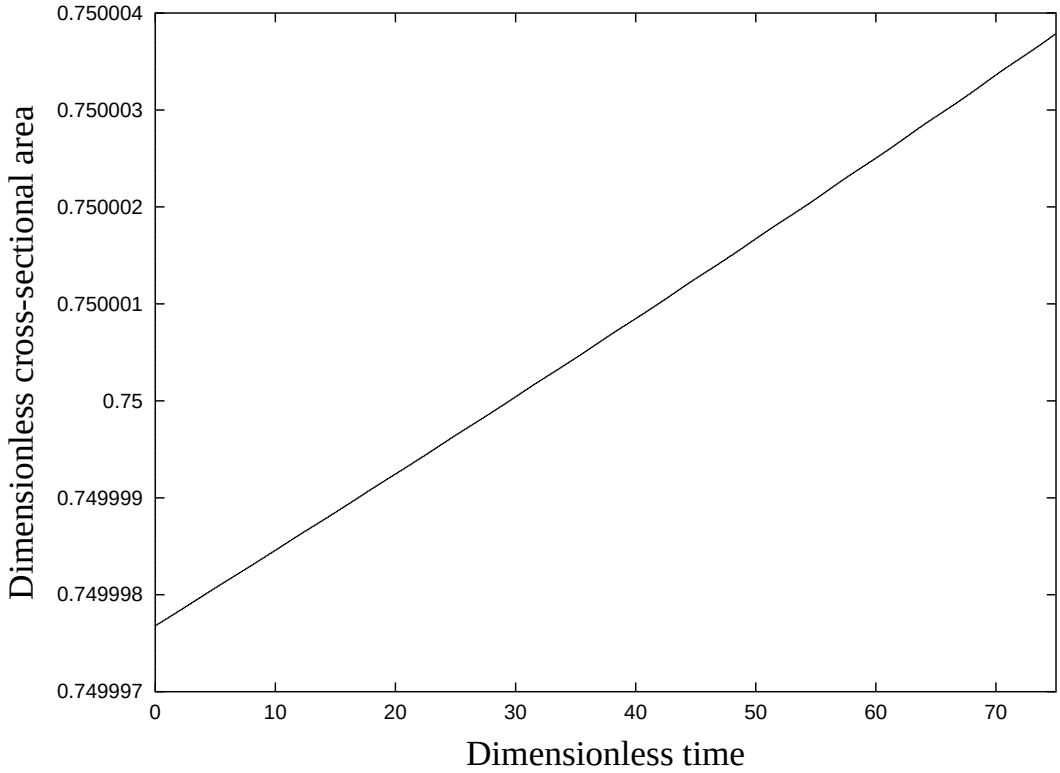


Figure 5.20: Conservation of mass region C, $\Lambda = 0.75$, $\gamma = 12.5$.

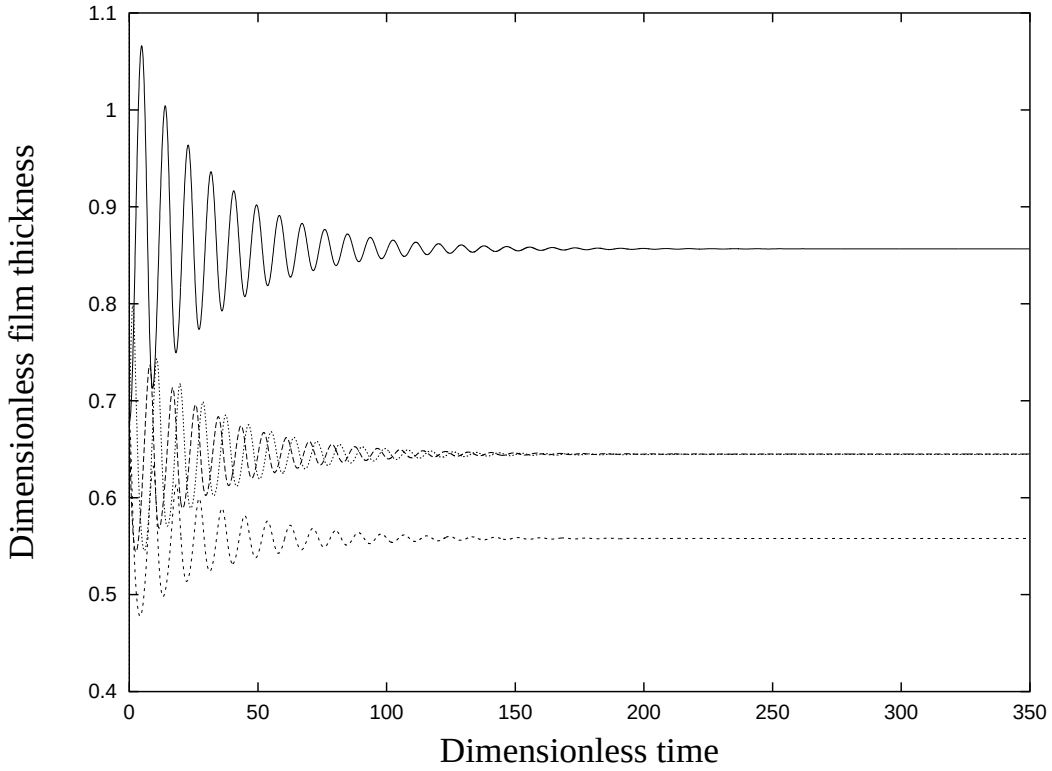


Figure 5.21: Large load free-surface evolution, $\Lambda = 5.7$, $\gamma = 1.3$: film thicknesses: ——— 0°, – – – 90°, - - - - 180°, · · · · · · 270°.

shows the evolution of the film thicknesses, while Figs. 5.22 and 5.23 show respectively the velocity and pressure fields at $t = 300$. As may be seen from Fig. 5.21 the film thicknesses at $\theta = 90°$ and $\theta = 270°$ again converge, suggesting a near symmetric free-surface profile, the difference in the thicknesses being approximately 0.056% at $t = 300$. In Fig. 5.22 the tangential velocity at $\theta = 180°$ is approximately 1.85, i.e. nearly twice that of the cylinder's surface; while at $\theta = 0°$ it is approximately 1.12, only slightly greater than that of cylinder's surface. Thus, qualitatively at least, the velocity distribution is in agreement with Hansen and Kelmanson's numerical results [38]; see for example their Fig. 7(a).

5.4.4 Load shedding

Finally, the evolution of loads that could not be supported was considered. For problems corresponding to points lying in region A of the parameter space, meshes of fixed connectivity like those employed above are unsuitable. Typically, in such problems, a bulge develops on the free surface during the first rotation of the cylinder. This then grows and, as it does so, the free-surface curvature increases considerably until, at some point, the isoparametric discretisation fails. Note that if the method

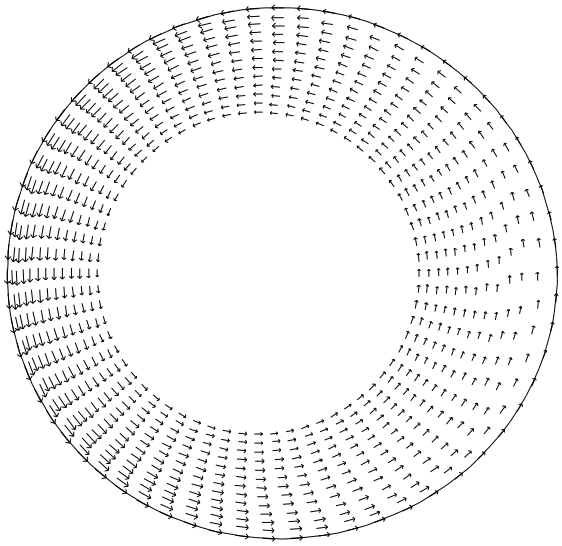


Figure 5.22: Large load problem, $\Lambda = 5.7$, $\gamma = 1.3$: velocity at $t = 300$.

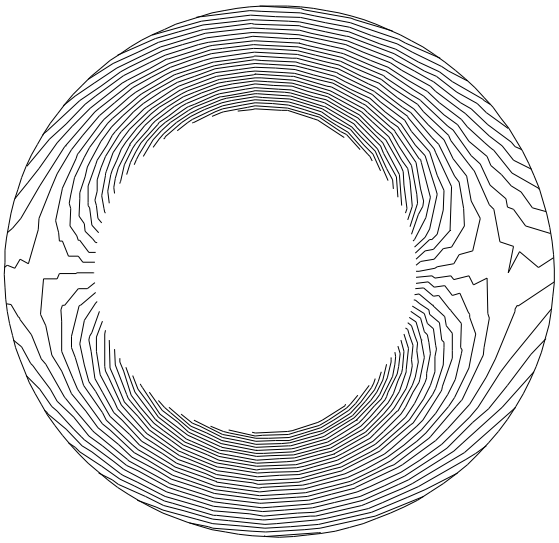


Figure 5.23: Large load problem, $\Lambda = 5.7$, $\gamma = 1.3$: pressure at $t = 300$ — range -0.065 to 1.192 dimensionless units.

of spines were to be employed, then it will fail if the free surface ever becomes tangent to one of the spines. As Figure 5.24 shows, this will inevitably happen in this problem unless multiple origins are employed for the system of spines.

An initial unstructured mesh with 64 equally-spaced vertices on both the free surface and the cylinder was employed, involving 256 elements and 1216 unknowns.

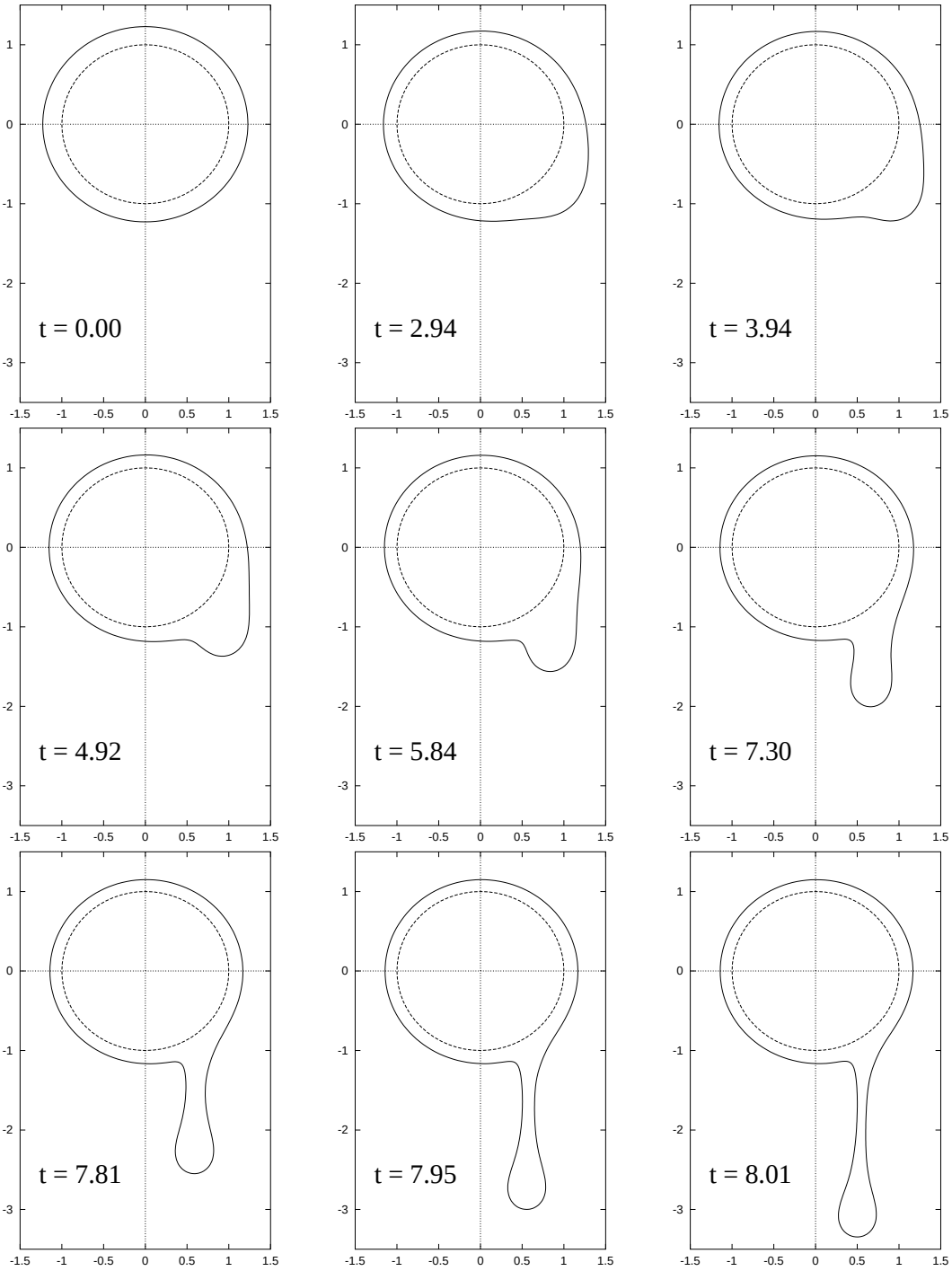


Figure 5.24: Load-shedding problem 1, free-surface evolution: $\Lambda = 1.6$, $\gamma = 12.5$.

Parameter values $\Lambda = 1.6$ and $\gamma = 12.5$ were selected, corresponding to a load only slightly greater than the maximum supportable-load predicted by thin-film theory [57]. Automatic refinement of both the free surface and the interior of the mesh was performed, as described in Section 2.4, values of $k_{tol} = 0.4$ and $h_{max} = 0.3$ being em-

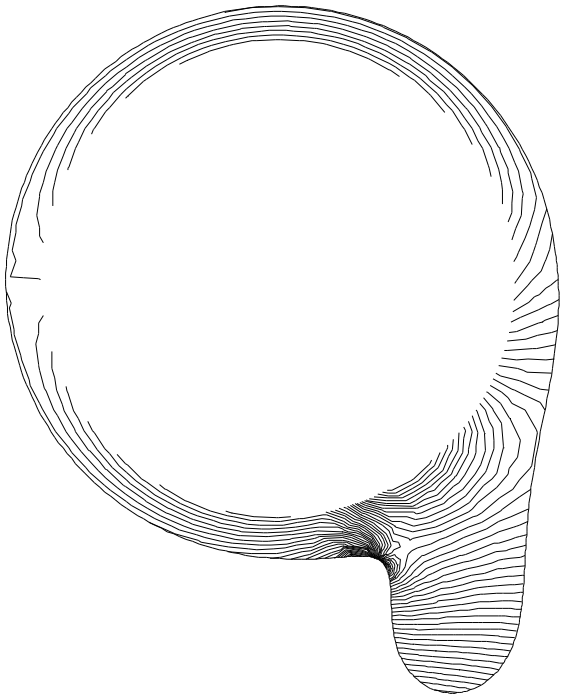


Figure 5.25: Load-shedding problem 1, $\Lambda = 1.6$, $\gamma = 12.5$: pressure at $t = 7.00$.

ployed. The interior of the mesh was updated at each step using Jacobi-smoothing. The free-surface node lying initially on the positive x axis was used as the initial node for the Cuthill-McKee ordering algorithm throughout the computation. Time integration was performed using time steps chosen by the stability method described in Section 3.15, taking $lte = 10^{-6}$. An ILUT preconditioner was computed every ten time steps and whenever remeshing required it, employing values of $lfil = 300$ and $droptol = 10^{-6}$. Between five and twenty conjugate residual iterations were typically required at each time step, though very occasionally a much larger number were necessary. The number of iterations required was observed to increase gradually as the mesh became larger, from an average of five per time step initially to approximately ten per time step in the later stages of the problem. On a shorter time-scale the number of iterations per time step was observed to grow approximately linearly as the preconditioner aged, the number required approximately doubling before the preconditioner was recomputed.

Figure 5.24 shows the evolution of the free surface. A bulge is clearly apparent at $t = 2.94$, and by $t = 5.84$ has assumed a characteristic lobe shape. Such lobes are reminiscent of those observed by Moffatt in his experimental work; see for example his Figure 7 [70]. Note, however, that the lobes observed by Moffatt are three-

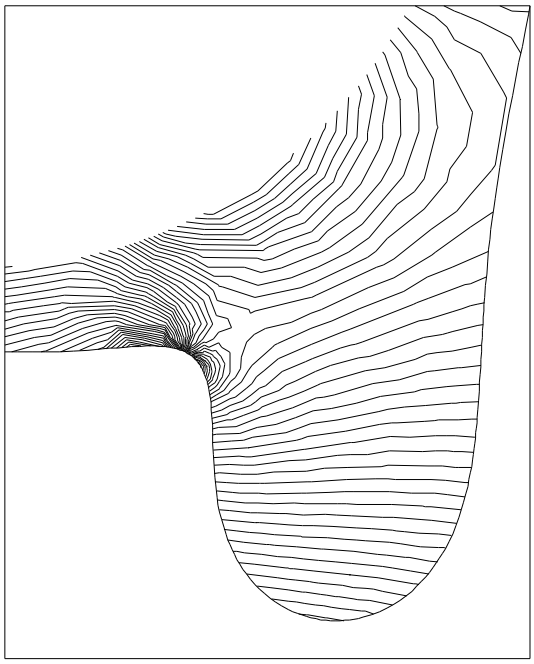


Figure 5.26: Load-shedding problem 1, $\Lambda = 1.6$, $\gamma = 12.5$: pressure at $t = 7.00$.

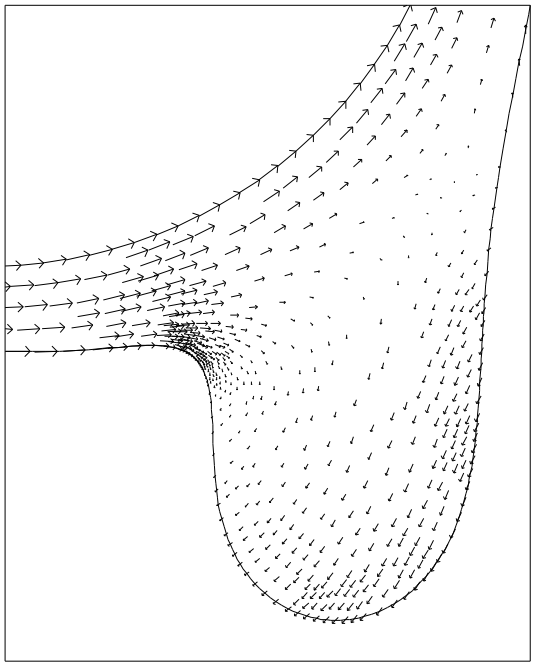


Figure 5.27: Load-shedding problem 1, $\Lambda = 1.6$, $\gamma = 12.5$: velocity at $t = 7.00$.

dimensional in nature and rotate about the cylinder with an angular velocity slightly lower than that of the cylinder. By $t = 7.30$ a droplet (or more correctly a curtain of fluid) has started to form. As the simulation continues, the droplet now starts to accelerate rapidly downwards. Shortly after $t = 8.01$ the solver failed. This appears to have been due to the now-large bandwidth of the finite element stiffness matrix, which could not be accurately factorised with the values of $lfil$ and $droptol$ employed. The final free-surface profile, at $t = 8.01$, corresponds to a mesh with 671 elements, nearly three times the number in the initial mesh. The cross-sectional area of the fluid was found to have increased by approximately +0.14% by the end of the computation.

Figure 5.25 shows the pressure field at $t = 7.00$, while Fig. 5.26 shows the detail in the lobe and Fig. 5.27 shows the corresponding velocity field. As may be seen, large variations in free-surface curvature occur near to where the upstream side of the droplet is attached to the remaining rotating film. The large velocity gradients and discontinuities in the pressure gradient apparent in this region suggest that additional refinement of the mesh would here be appropriate. Over much of the droplet the pressure contours are approximately horizontal, suggesting that there the flow is dominated by gravity.

5.4.5 A second load-shedding problem

Further investigations have shown that provided the initial load does not greatly exceed the maximum supportable load, then the mechanism by which fluid is shed is essentially independent of the initial configuration. As a final test the above experiment was repeated with $\Lambda = 5.7$ and $\gamma = 1.7$ i.e. with a considerably greater load, though again one that only slightly exceeds the maximum supportable load. The initial mesh is shown in Fig. 5.28 and again has 64 equispaced vertices on both the free surface and the cylinder. This time values of $k_{tol} = 0.3$ and $h_{max} = 0.2$ were employed, resulting in a somewhat larger initial mesh, with 520 elements and 2404 unknowns. Values of $lfil = 400$ and $droptol = 10^{-6}$ were employed for the preconditioner.

The evolution of the free-surface evolution is shown in Fig. 5.29. In this problem the mass of fluid shed is much larger, and it is clear that it considerably exceeds the minimum required to result in a supportable load. Indeed, it would appear that as much as half of the initial load will be shed. The change in the cross-sectional area is approximately +0.59% by time $t = 14.85$, most of which occurs in the final stages

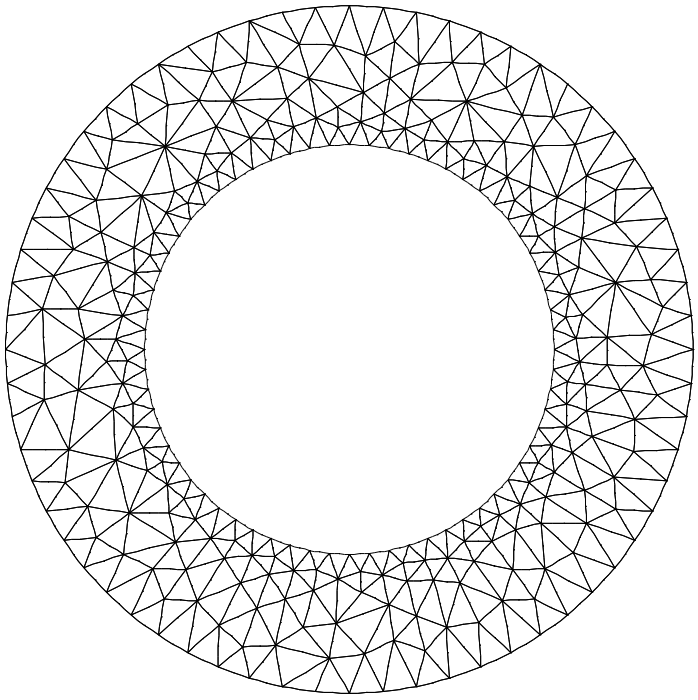


Figure 5.28: Load-shedding problem 2: initial mesh.

of the problem as the droplet beings to fall rapidly, the gain in mass by $t = 14.46$ being only 0.11%. Two phases of the development of the flow appear of particular interest. The first is the initial development of the lobe. Figure 5.30 shows the velocity and pressure fields at $t = 3.00$. As Fig. 5.31 shows, by $t = 6.95$ the flow regime has completely changed; the droplet is rapidly growing and a stagnation point is now present. The pressure field is also markedly different, a new saddle point being present where the lobe is attached to the up stream side of the rotating film.

The next potentially interesting phase of the problem occurs around the time that the down-stream side of the lobe changes from being convex to being concave — the point at which the lobe becomes a genuine 'droplet'. As may be seen from Figs. 5.32 and 5.33, no great change in the flow regime occurs around this time; that apparent being due to the different scalings employed in the two figures.

From $t = 12.80$ onwards the droplet evolves primarily under the influence of gravity and, as it begins to fall downwards, an elongated neck develops. As this happens, fluid continues to be drawn off from the rotating cylinder. Figure 5.34 shows the velocity field at time $t = 14.85$, shortly before the solver failed, while Fig. 5.35 shows the mesh at this time, which contains 839 elements, nearly twice the

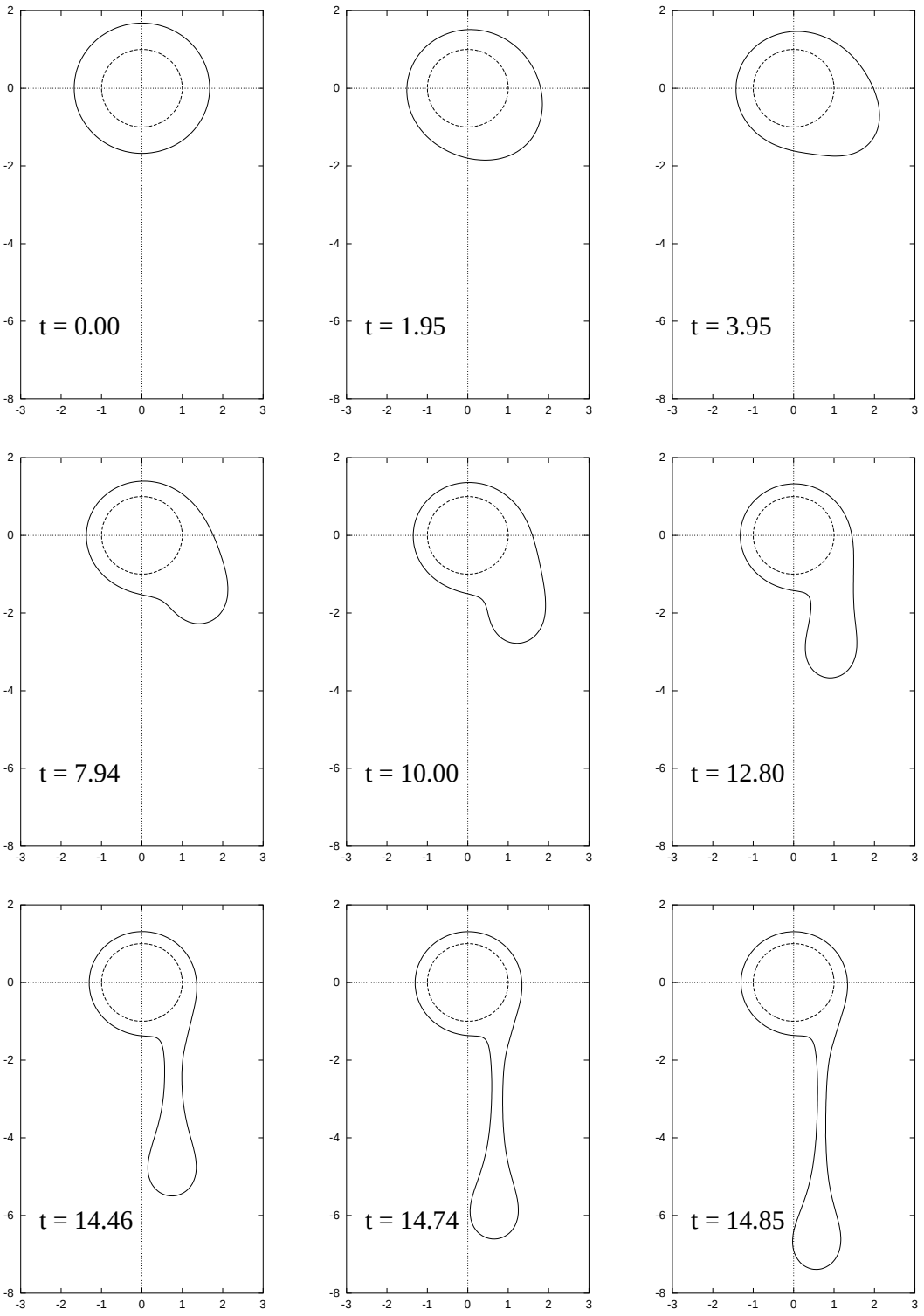


Figure 5.29: Load-shedding problem 2, $\Lambda = 5.7$, $\gamma = 1.7$: free-surface evolution.

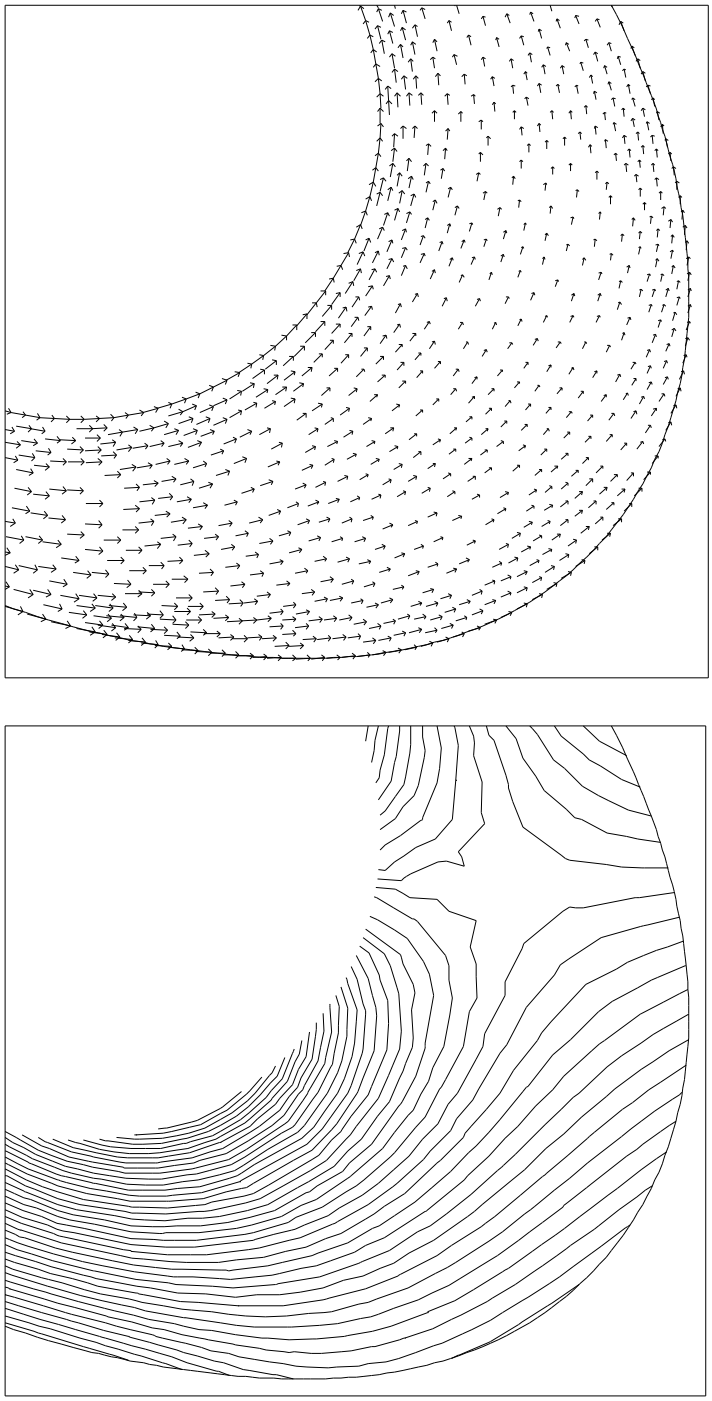


Figure 5.30: Load-shedding problem 2, break down of rigid-body flow (a) $\Lambda = 5.7$, $\gamma = 1.7$: velocity and pressure at $t = 3.00$.

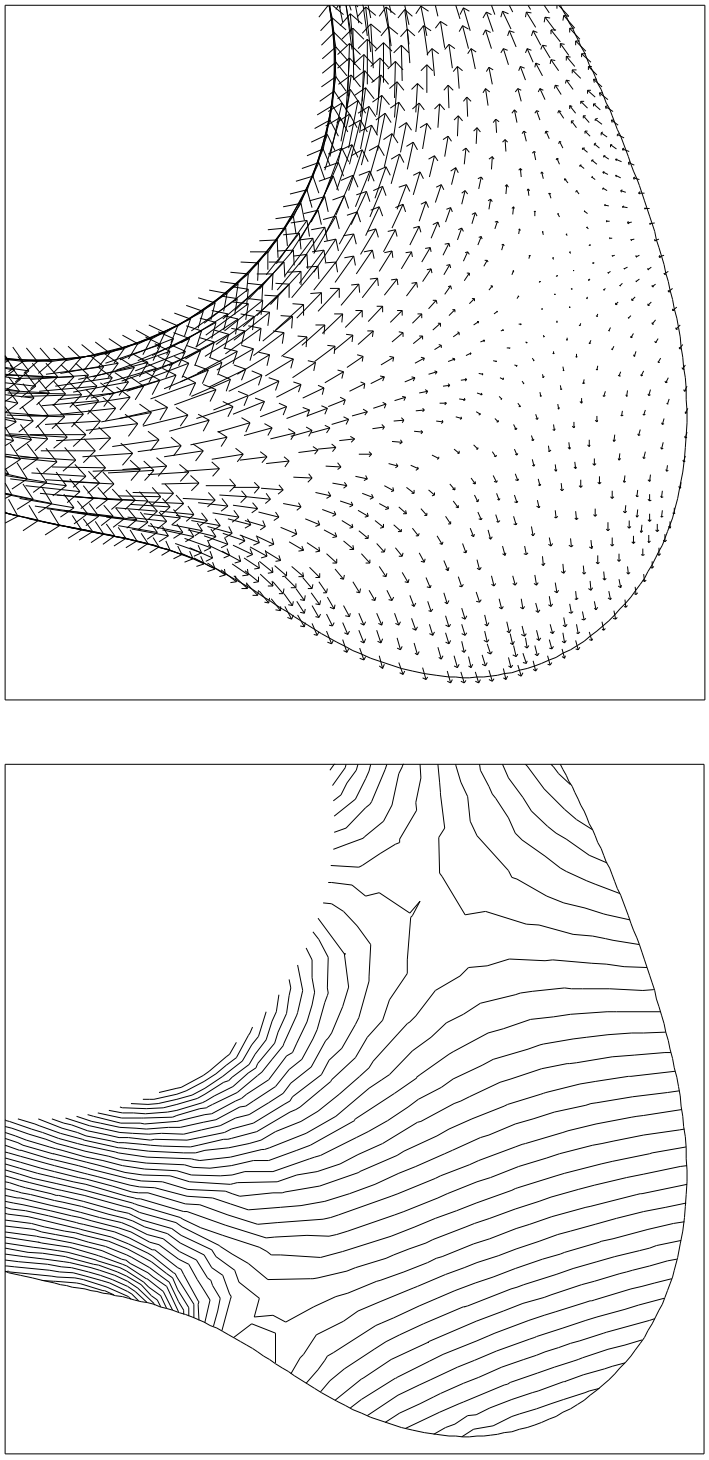


Figure 5.31: Load-shedding problem 2, break down of rigid-body flow (b) $\Lambda = 5.7$, $\gamma = 1.7$: velocity and pressure at $t = 6.95$.

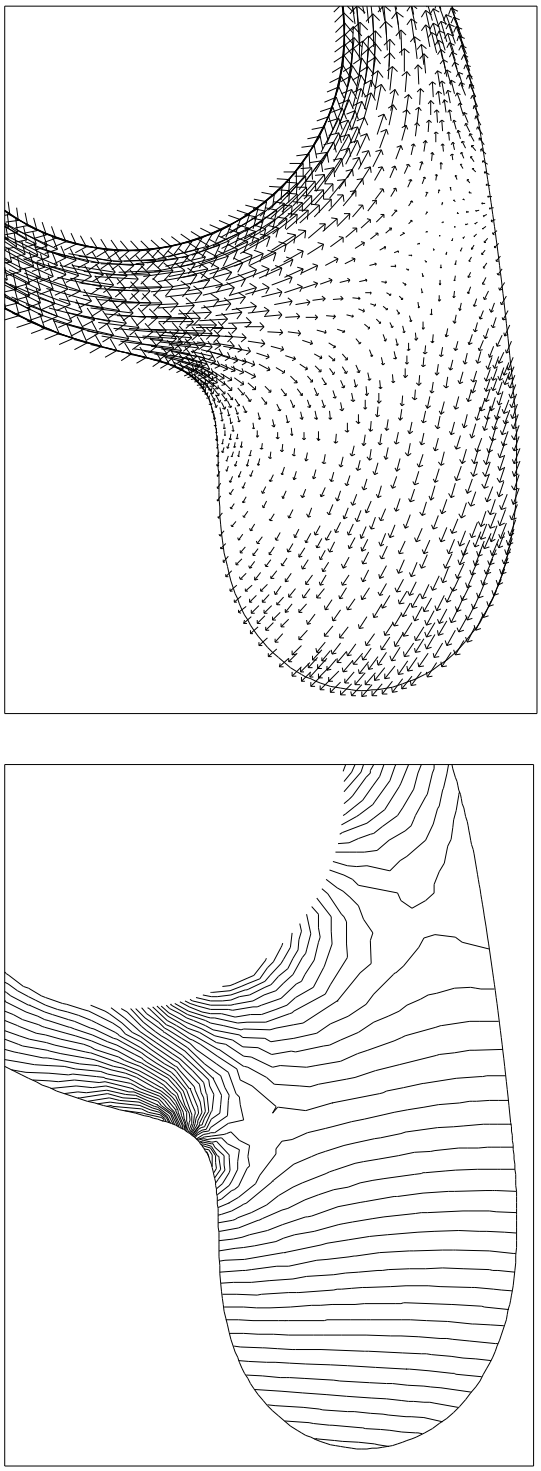


Figure 5.32: Load-shedding problem 2, droplet formation (a) $\Lambda = 5.7$, $\gamma = 1.7$: velocity and pressure at $t = 11.00$.

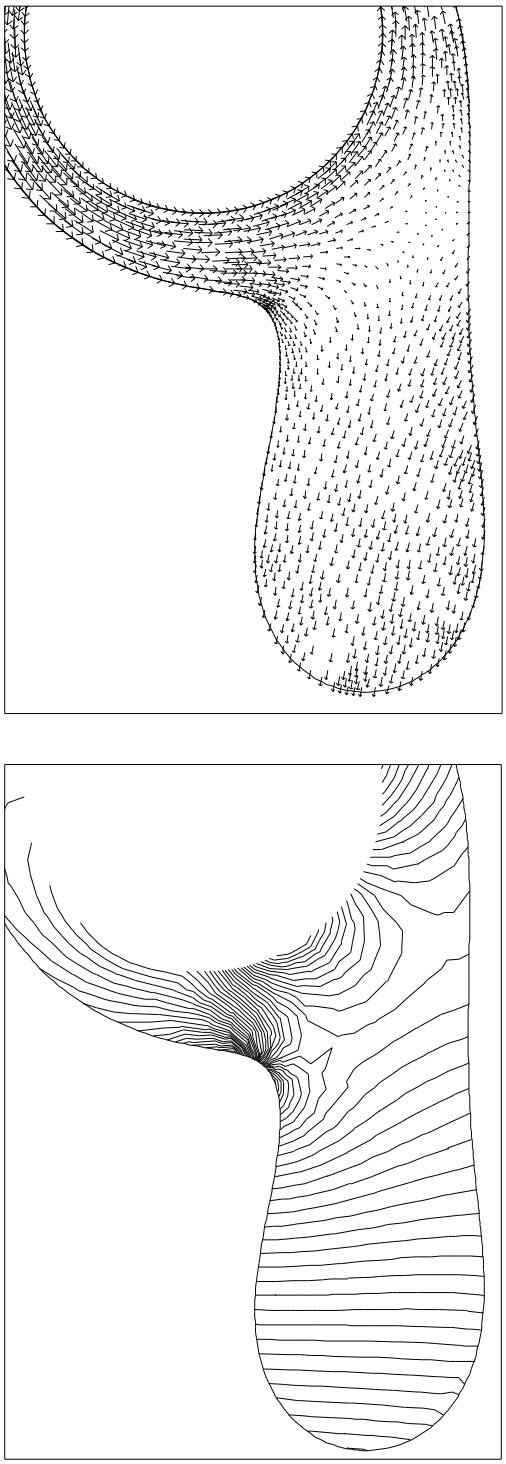


Figure 5.33: Load-shedding problem 2, droplet formation (b) $\Lambda = 5.7$, $\gamma = 1.7$: velocity and pressure at $t = 12.80$.

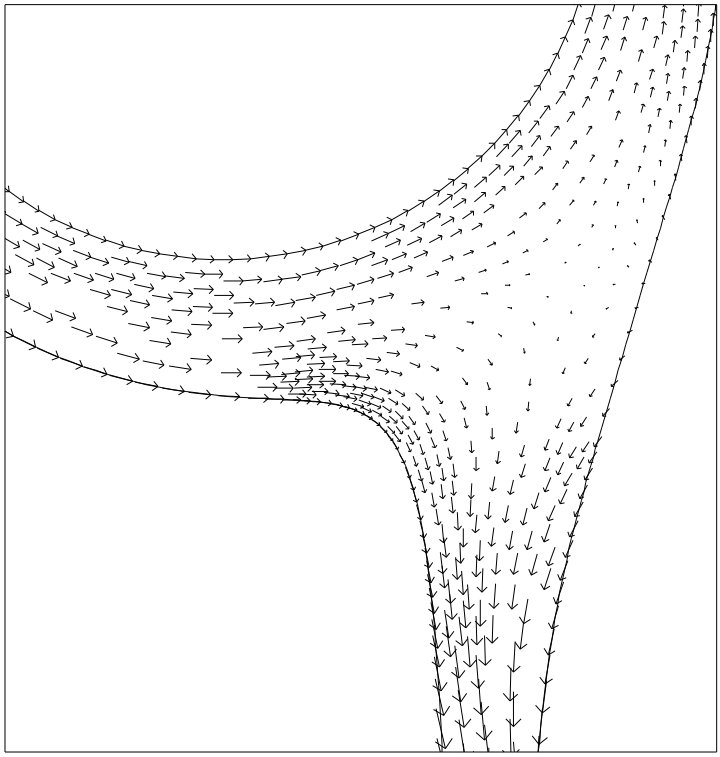


Figure 5.34: Load-shedding problem 2, $\Lambda = 5.7$, $\gamma = 1.7$: velocity at $t = 14.85$.

number in the initial mesh.

5.5 Conclusions

The time-dependent behaviour of films of viscous fluid supported on a rotating cylinder has been investigated using a Stokes-flow model. While broadly confirming the predictions of the maximum supportable load made by Moffat [70], and by Hansen and Kelmanson [38], the computations reported here suggest that for much of the parameter space, convergence from an arbitrary initial configuration towards a stable steady-state solution does not occur.

Where asymptotically steady solutions were found, the free-surface profiles were close to symmetric about the horizontal plane drawn through the axis of the cylinder, confirming the observations of Hansen and Kelmanson [38]. The film thicknesses computed have been shown to be in reasonable agreement with those predicted by thin-film theory.

Stable oscillatory solutions have also been demonstrated, and have been shown to be independent of mesh resolution. The apparent tendency of such oscillatory solutions to grow in amplitude has been shown to be linked to the accuracy of the

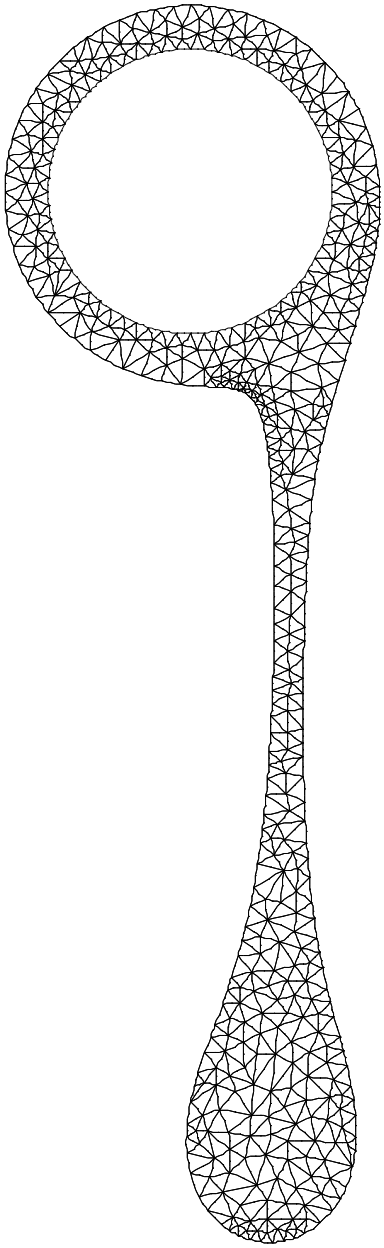


Figure 5.35: Load-shedding problem 2, $\Lambda = 5.7$, $\gamma = 1.7$: mesh at $t = 14.85$.

time-integration scheme employed.

The phenomenon of load shedding has been investigated, and illustrations of this process are presented here for the first time. Such simulations have been continued far beyond the point at which conventional spine-based methods would have failed.

Finally it has been shown that the adaptive mesh regeneration techniques orig-

inally developed for the problem described in Chapter 4 may be applied without modification to a very different problem, involving large changes in domain geometry, and in which both considerable increases and decreases in free-surface curvature occur.

Chapter 6

Navier-Stokes problems

In this chapter the application of the automatic mesh generation algorithm to free-surface Navier-Stokes problems is considered. First, the axisymmetric form of the Navier-Stokes equations is described. Next, small-amplitude axisymmetric oscillations of droplets are modelled, as a means of validating the implementation of the Navier-Stokes solver. Finally, the unstructured moving-mesh method is briefly explored as a means of solving free-surface Navier-Stokes problems of moderate to large amplitude, by considering first oscillations of ellipsoidal droplets and then oscillations of droplets perturbed by a second-spherical-harmonic component of large amplitude.

6.1 Axisymmetric oscillations of droplets

It appears that, apart from in artificial or trivial cases, analytic solutions of free-surface Navier-Stokes flow problems are unknown. One route to the validation of a free-surface scheme lies through the simulation of small-amplitude oscillations of three-dimensional droplets driven by surface tension [7, 68]. When the amplitude of such an oscillation is small and the Reynolds number is large the period may be estimated using Prosperetti's analytical model for inviscid droplets [80].

Foote pioneered the computational modelling of viscous droplets as early as 1973 [26], employing a finite-difference method which incorporated a marker-and-cell (MAC) scheme for tracking the free surface. More recently problems involving

droplets have been studied by Basaran [7], using the finite element method, and by Mashayek and Ashgriz [68], using a volume-of-fluid (VOF) finite element method. The work of Basaran [7] is of particular interest here, in that it contains numerous computational results that may be used for comparison when modelling oscillations of moderate to large amplitude. In addition, many papers have been published [84, 111, 9, 8, 102, 75, 118] containing experimental and analytical results for moderate- to large-amplitude nonlinear oscillations of droplets in both the viscid and inviscid cases, allowing insight to be gained into the physics of such problems. Thus, in addition to being of fundamental scientific importance, the study of the oscillation of axisymmetric droplets is a convenient source of test problems for numerical methods.

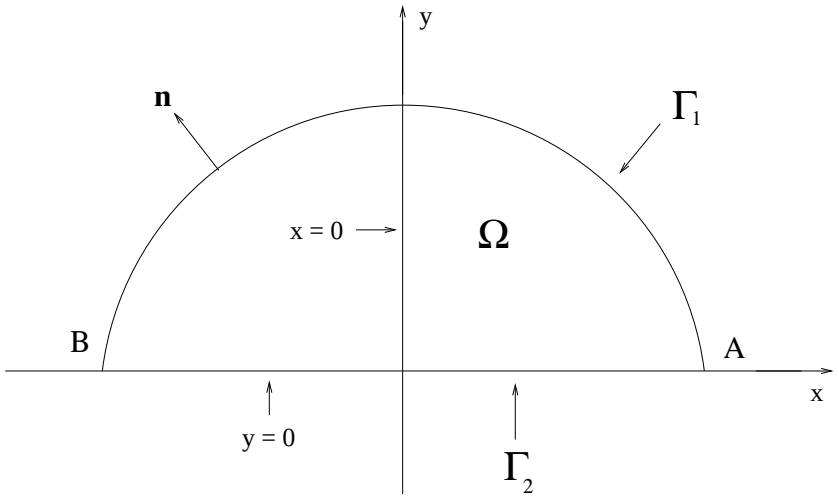


Figure 6.1: Domain for axisymmetric problems.

In the study of the oscillation of viscous droplets two types of initial free-surface configuration are commonly encountered. The first is a volume of revolution obtained by rotating an ellipse around one of its axes of symmetry. The second is a volume of revolution formed by rotating a circular domain perturbed by a spherical-harmonic, about an axis of symmetry. It was Rayleigh, in an appendix to his 1879 study of the capillary phenomena of jets of inviscid fluids [84] who first identified the modes of inviscid droplets with the spherical-harmonics. Basaran [7] gives the following expression for the cross-section of a sphere perturbed by a single spherical-harmonic component

$$f(\theta) = \gamma_n \left[1 + f_n P_n(\cos\theta)\right], \tag{6.1}$$

where $f(\theta)$ is the radial distance from the origin to the free surface at an angle $0 \leq \theta \leq \pi$ to the positive x-axis, f_n is the amplitude of the initial perturbation,

γ_n is a constant chosen to normalise the volume of the droplet, and P_n is the nth Legendre polynomial written in terms of θ [101]. Thus, for example

$$P_0 = 1, \tag{6.2}$$

$$P_1 = \cos\theta, \tag{6.3}$$

$$P_2 = \frac{1}{4}\left(1 + 3\cos 2\theta\right), \tag{6.4}$$

$$P_3 = \frac{1}{8}(3\cos\theta + 5\cos 3\theta), \tag{6.5}$$

$$P_4 = \frac{1}{64}(9 + 20\cos 2\theta + 35\cos 4\theta), \tag{6.6}$$

etc. The curve defined by (6.1) is rotated about the x axis to give a three-dimensional shape. The first spherical-harmonic, P_0, is not considered here since it corresponds to a change in droplet volume. The odd-numbered spherical-harmonics result in free-surface profiles that are not symmetric in the plane $x = 0$.

The constant γ_n is chosen so that the initial volume of the droplet is $\frac{4}{3}\pi R^3$, where R is the radius of the unperturbed droplet which is normally chosen to equal one in the dimensionless model. Thus, in the $n = 2$ case, Mashayek and Ashgriz [68] give the expression

$$\gamma_2 = \left(\frac{35}{35 + 21{f_2}^2 + 2{f_2}^3}\right)^{\frac{1}{3}}. \tag{6.7}$$

Figure 6.1 illustrates in schematic form the geometry of the domain for axisymmetric problems. The axis of rotational symmetry lies along the x axis. The boundary may be divided into two parts. The first, Γ_1, corresponds to the surface of the volume of revolution, on which surface tension provides the boundary condition. The second, Γ_2, corresponds to the domain's axis of rotational symmetry, on which an artificial symmetry boundary condition must be imposed. The points A and B lie at the junctions of Γ_1 and Γ_2. Note that in many problems of fundamental interest a further bilateral symmetry is present in the plane $x = 0$.

6.2 Axisymmetric problem formulation

The axisymmetric formulation of the Navier-Stokes equations may be derived by writing the three-dimensional Navier-Stokes equations in cylindrical coordinate form and then simplifying, using rotational symmetry, and by assuming that the droplet does not rotate about the x axis, to give a two-dimensional system of equations [34]. Further details of this derivation are given in Appendix B. Thus, if the effects of

gravity are ignored, (3.1) and (3.2) become

$$\begin{aligned}
\frac{\partial u}{\partial t}+\left(u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right) &= -\frac{\partial p}{\partial x}+\frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{1}{y}\frac{\partial}{\partial y}\left(y\frac{\partial u}{\partial y}\right)\right) \\
&\quad +\frac{1}{Re}\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{1}{y}\frac{\partial}{\partial y}\left(yv\right)\right), \qquad (6.8)\\
\frac{\partial v}{\partial t}+\left(u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right) &= -\frac{\partial p}{\partial y}+\frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{1}{y}\frac{\partial}{\partial y}\left(y\frac{\partial v}{\partial y}\right)-\frac{v}{y^2}\right) \\
&\quad +\frac{1}{Re}\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{1}{y}\frac{\partial}{\partial y}\left(yv\right)\right), \qquad (6.9)\\
\frac{\partial u}{\partial x}+\frac{1}{y}\frac{\partial}{\partial y}\left(yv\right) &= 0. \qquad (6.10)
\end{aligned}$$

Equations (6.8), (6.9) and (6.10) may now be written in Galerkin weighted-residual form, by multiplying by the appropriate test functions and integrating over the domain using the axisymmetric volume element $2\pi\, y\, dx\, dy$. The factor 2π, being present in each term, may safely be ignored when forming the finite element stiffness matrix. If (6.8) and (6.9) are rewritten (see Appendix A) so as to give the appropriate natural boundary conditions then the majority of the integrals that result are identical to their Cartesian counterparts, except for the addition of a factor y in the integrand. There are however a number of terms that are not present in the original Cartesian formulation. The term

$$-\frac{1}{Re}\frac{v}{y^2} \qquad (6.11)$$

in (6.9), which arises when the Laplacian of the velocity field is written in cylindrical coordinates, appears to be a particular source of difficulty since at $y = 0$ it is undefined. In Galerkin weighted-residual form this term gives rise to integrals of the form[1]

$$-\int_\Omega q_i \frac{v}{y} d\,\Omega. \qquad (6.12)$$

The corresponding entries in the finite element stiffness matrix are thus of the form

$$-\int_\Omega \frac{q_i q_j}{y} d\,\Omega, \qquad (6.13)$$

[1]Note that in this chapter the notation $d\Omega$ is employed to denote $dx\, dy$ rather than, as is more conventional in axisymmetric formulations, $y\, dx\, dy$. Similarly the notation $d\Gamma$ is used to denote a Cartesian rather than an axisymmetric line element i.e. ds rather than $y\, ds$.

and clearly make a symmetric contribution. Analytically, if one assumes that $v(x,y)$ is C^2 continuous on $\Omega \cup \Gamma_1 \cup \Gamma_2$ then, since by symmetry v must be zero on the x axis and v must be an odd function of y, one may write any admissible velocity field $v(x,y)$ in the form

$$v(x,y) = v_1(x)y + O(y^3). \tag{6.14}$$

Thus, by L'Hôpital's rule

$$\lim_{y\to 0} \frac{v(x,y)}{y} = v_1(x) \tag{6.15}$$

for any admissible $v(x,y)$, and consequently integrals of the form (6.12) are well defined.

From a practical point of view, if open sets of Gauss-Legendre quadrature points are employed, i.e. sets containing no points on the master element's boundary, then there is no need to evaluate the integrand in (6.13) on $y = 0$, thus avoiding the singularity, or indeed particularly near to it, avoiding numerical rounding problems.

The second new term in the axisymmetric formulation results from the integration by parts of the pressure-gradient operator in the momentum equation for v. This takes the form

$$\begin{aligned} \int_\Omega q_i y \frac{\partial p}{\partial y}\, d\Omega &= \int_{\partial\Omega} q_i y p n_y \, d\Gamma - \int_\Omega p \frac{\partial}{\partial y}\left(q_i y\right)\, d\Omega & (6.16)\\ &= \int_{\partial\Omega} q_i y p n_y \, d\Gamma - \int_\Omega p y \frac{\partial q_i}{\partial y}\, d\Omega - \int_\Omega p q_i \, d\Omega. & (6.17) \end{aligned}$$

When discretized the last term of (6.17), which has no counterpart in its Cartesian form (2.27), becomes

$$-\int_\Omega l_j q_i \, d\Omega. \tag{6.18}$$

This term, on its own, would lead to the finite element stiffness matrix being non-symmetric if it were not for a matching contribution deriving from the axisymmetric form of the continuity equation (6.10), i.e.

$$\begin{aligned} -\int_\Omega l_i y \left(\frac{\partial u}{\partial x} + \frac{1}{y}\frac{\partial}{\partial y}\left(yv\right)\right)\, d\Omega &= \\ -\int_\Omega l_i y \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\, d\Omega + \int_\Omega l_i v \, d\Omega &= 0, \end{aligned} \tag{6.19}$$

where again the continuity equation has been multiplied by -1 in order to give a symmetric stiffness matrix. The first two terms in (6.19) are identical to their Cartesian counterparts, except for the additional factor y. The last term is present

only in the axisymmetric formulation, and takes the discrete form

$$-\int_\Omega l_i q_j \, d\Omega, \tag{6.20}$$

i.e. the transpose of (6.18). Thus these two new terms together make a symmetric contribution to the finite element stiffness matrix.

The final new term arises only when the stress-divergence form of the viscous term is employed, and is absent in the more conventional formulation. It arises when the $\nabla(\nabla \cdot \mathbf{u})$ term in the v momentum equation is integrated by parts to obtain the natural stress boundary condition, i.e.

$$\begin{aligned} \int_\Omega q_i y \frac{\partial}{\partial y}\left(\frac{1}{y}\frac{\partial}{\partial y}(yv)\right) d\Omega &= \int_\Omega q_i y \frac{\partial}{\partial y}\left(\frac{v}{y}+\frac{\partial v}{\partial y}\right) d\Omega \qquad (6.21) \\ &= \int_{\partial\Omega} q_i y n_y \frac{\partial v}{\partial y}\, d\Gamma - \int_\Omega y \frac{\partial q_i}{\partial y}\frac{\partial v}{\partial y}\, d\Omega \\ &\quad - \int_\Omega q_i \frac{v}{y}\, d\Omega. \qquad (6.22) \end{aligned}$$

The first two terms on the right-hand side of (6.22) are again identical to their Cartesian counterparts, except for the additional factor y. The last term is new and is identical to (6.12). Thus it may be seen that the switch to the axisymmetric formulation introduces no new asymmetry into the finite element stiffness matrix. In full, the weak form of the axisymmetric Navier-Stokes equations employed here is thus

$$\int_\Omega q_i y \frac{\partial u}{\partial t} d\Omega + \int_\Omega q_i y \left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) d\Omega +$$

$$\frac{2}{Re}\int_\Omega y \frac{\partial q_i}{\partial x}\frac{\partial u}{\partial x} d\Omega + \frac{1}{Re}\int_\Omega y \frac{\partial q_i}{\partial y}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) d\Omega - \int_\Omega y p \frac{\partial q_i}{\partial x} d\Omega$$

$$= \int_{\partial\Omega} q_i y \left(-p n_x + \frac{2}{Re}\frac{\partial u}{\partial x} n_x + \frac{1}{Re}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) n_y\right) d\Gamma \tag{6.23}$$

$$\int_\Omega q_i y \frac{\partial v}{\partial t} d\Omega + \int_\Omega q_i y \left(u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} \right) d\Omega +$$

$$\frac{2}{Re} \int_\Omega y \frac{\partial q_i}{\partial y} \frac{\partial v}{\partial y} d\Omega + \frac{1}{Re} \int_\Omega y \frac{\partial q_i}{\partial x} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) d\Omega - \int_\Omega y p \frac{\partial q_i}{\partial y} d\Omega$$

$$+ \frac{2}{Re} \int_\Omega q_i \frac{v}{y} d\Omega - \int_\Omega q_i p d\Omega$$

$$= \int_{\partial\Omega} q_i y \left(-p n_y + \frac{2}{Re} \frac{\partial v}{\partial y} n_y + \frac{1}{Re} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_x \right) d\Gamma \tag{6.24}$$

$$- \int_\Omega l_i y \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) d\Omega - \int_\Omega l_i v d\Omega = 0, \tag{6.25}$$

where $\mathbf{n} = (n_x, n_y)$ is the outward free-surface normal on $\partial\Omega = \Gamma_1 \cup \Gamma_2$.

6.3 Boundary conditions

The bracketed expressions in the boundary integrals in (6.23) and (6.24), i.e.

$$-p n_x + \frac{2}{Re} \frac{\partial u}{\partial x} n_x + \frac{1}{Re} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_y \tag{6.26}$$

and

$$-p n_y + \frac{2}{Re} \frac{\partial v}{\partial y} n_y + \frac{1}{Re} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) n_x, \tag{6.27}$$

are simply the x and y components of the stress at a point in a Newtonian fluid, the additional factor y in the boundary integrals in (6.23) and (6.24) being due to the form of the axisymmetric volume element. Thus, arbitrary stress boundary conditions $\sigma = (\sigma_x, \sigma_y)$ may be imposed by assembling

$$\int_{\partial\Omega} q_i y \sigma_x d\Gamma, \tag{6.28}$$

and

$$\int_{\partial\Omega} q_i y \sigma_y d\Gamma, \tag{6.29}$$

in place of the right-hand sides of (6.23) and (6.24). Here, on Γ_1, the relevant boundary integral takes the form

$$\frac{1}{We}\int_{\partial\Omega} q_i y(k_1(s)+k_2(s))\mathbf{n}\, d\Gamma, \tag{6.30}$$

where We is the Weber number introduced in Section 1.2, and k_1 and k_2 are the principle curvatures of the volume of revolution, which may be computed [113] using

$$k_1(s) = \frac{x_s y_{ss} - y_s x_{ss}}{({x_s}^2 + {y_s}^2)^{\frac{3}{2}}}, \quad k_2(s) = \frac{x_s}{y({x_s}^2+{y_s}^2)^{\frac{1}{2}}}, \tag{6.31}$$

where s is the arc length. Considering first the integral involving k_1 i.e. the component of the curvature in the x–y plane, and ignoring the constant $\frac{1}{We}$ integration by parts gives

$$\begin{aligned}\int_A^B q_i y k_1 \mathbf{n}\, d\Gamma &= \int_A^B q_i y \frac{\partial \mathbf{t}}{\partial s}\, d\Gamma & (6.32)\\ &= [q_i y \mathbf{t}]_A^B - \int_A^B y\mathbf{t}\frac{\partial q_i}{\partial s}\, d\Gamma - \int_A^B q_i \mathbf{t}\frac{\partial y}{\partial s}\, d\Gamma, & (6.33)\end{aligned}$$

where A and B are the appropriate limits of integration for a given edge. Note the presence of an additional boundary integral on the right-hand side of (6.33) that has no corresponding equivalent in the Cartesian formulation. Evaluation of both integrals is considerably simplified if carried out using local element coordinates (ξ,η).

The expression for k_2 may be rewritten, using local element coordinates, in the equivalent form

$$\frac{x_\xi}{y({x_\xi}^2+{y_\xi}^2)^{\frac{1}{2}}} \tag{6.34}$$

and thus, by a change in the variable of integration, the corresponding contributions to the boundary conditions may be evaluated using

$$\int_{\partial\Omega} q_i y k_2 \mathbf{n} d\Gamma = \int_{\partial\Omega} q_i \frac{\partial x}{\partial \xi}\mathbf{n}\, d\xi. \tag{6.35}$$

On Γ_2, two new boundary conditions are required, one for each of the momentum equations. By symmetry $v = 0$ on Γ_2, which provides a convenient essential bound-

ary condition. Furthermore, by symmetry, u and p must be even functions of y, while v must be an odd function of y. Thus on Γ_2

$$\frac{\partial u}{\partial y} = 0, \tag{6.36}$$

and

$$\frac{\partial p}{\partial y} = 0, \tag{6.37}$$

but in general,

$$\frac{\partial v}{\partial y} \neq 0. \tag{6.38}$$

Furthermore, since $v = 0$ on $x = 0$,

$$\frac{\partial v}{\partial x} = 0 \tag{6.39}$$

on Γ_2. The tangential stress $\sigma.\mathbf{t}$ on Γ_2 may be shown, using (6.26) and (6.27), to be

$$-\frac{1}{Re}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right), \tag{6.40}$$

which by (6.36) and (6.39) is equal to zero. Thus the second new boundary condition required on Γ_2 is simply the imposition of zero tangential stress, a convenient homogeneous natural boundary condition.

6.4 Mesh update procedures

The locations of free-surface nodes are updated using the kinematic boundary condition (1.4), the discrete axisymmetric form of which,

$$\int_{\partial\Omega} y q_i (\dot{\mathbf{s}} - \mathbf{u}) \cdot \mathbf{n} \, d\Gamma = 0, \tag{6.41}$$

is satisfied if one takes $\dot{\mathbf{s}}_i \cdot \mathbf{n}_i = \mathbf{u}_i \cdot \mathbf{n}_i$ for each free-surface node i. Thus no modification is required to the existing explicit implementation of the kinematic boundary condition.

Anticipating that the update of the two free-surface nodes on the axis of symmetry might prove problematic, the direct imposition of the free-surface symmetry

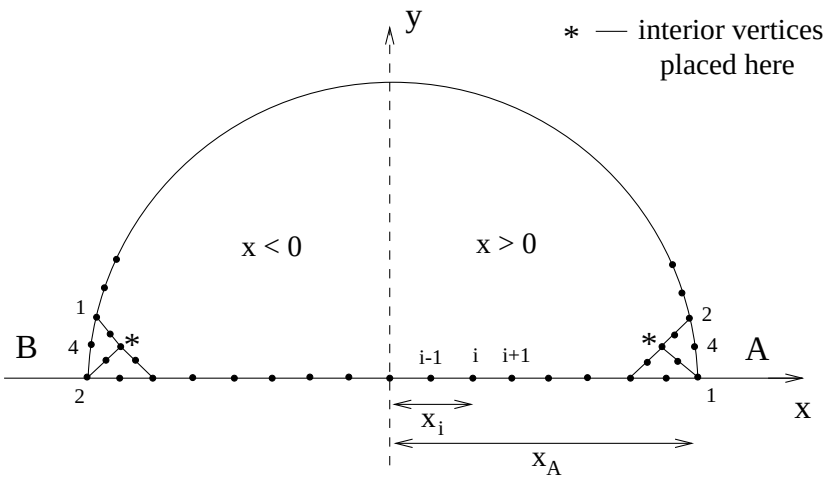


Figure 6.2: Updating of the axial free-nodes: local numbering of nodes for axial node update.

boundary condition was investigated. In this approach, instead of updating the two axial free-surface nodes using the kinematic boundary condition, their new locations are selected at the end of each time step so that

$$\frac{dx}{ds} = 0 \tag{6.42}$$

at A and B, i.e. that the free surface at A and B is vertical. In discrete form these constraints become

$$\frac{dx}{ds}(A) \quad = -3x_1 - x_2 + 4x_4 \quad = 0, \tag{6.43}$$

$$\frac{dx}{ds}(B) \quad = x_1 + 3x_2 - 4x_4 \quad = 0, \tag{6.44}$$

where the x_i are the x coordinates of the nodes comprising the free-surface edges adjacent to A and B, numbered locally as shown in Fig. 6.2. Thus at A one updates the node's position using

$$x_A^{(n+1)} = x_1^{(n+1)} = \frac{4x_4^{(n+1)} - x_2^{(n+1)}}{3}, \tag{6.45}$$

while at B one uses

$$x_B^{(n+1)} = x_2^{(n+1)} = \frac{4x_4^{(n+1)} - x_1^{(n+1)}}{3}. \tag{6.46}$$

Once this has been done the locations of the interior nodes lying on Γ_2 are updated

using

$$x_i^{(n+1)} = x_i^{(n)} + u_A^{(n+1)} \frac{x_i{}^{(n)}}{x_A{}^{(n)}} k, \tag{6.47}$$

for $x_i^{(n)} \geq 0$, and

$$x_i^{(n+1)} = x_i^{(n)} + u_B^{(n+1)} \frac{x_i{}^{(n)}}{x_B{}^{(n)}} k, \tag{6.48}$$

for $x_i^{(n)} \leq 0$, where k is the length of the current time step, and where

$$u_A^{(n+1)} = \frac{u_A^{(n+1)} - u_A^{(n)}}{k}, \tag{6.49}$$

$$u_B^{(n+1)} = \frac{u_B^{(n+1)} - u_B^{(n)}}{k}. \tag{6.50}$$

Thus nodes on the axis move proportionately to their distance from the origin and so maintain their original positions relative to one another. Once this has been done, the interior mesh is updated using the techniques described in Section 2.5. Note that when generating meshes a pair of interior nodes are specified, one near A and one near B, their positions being selected according to the spacing of the adjacent free-surface nodes, so as to prevent the automatic mesh generator creating elements with two boundary edges. Thus for example if the free-surface vertex at A has coordinates $(x_1, 0)$ and the next free-surface vertex has coordinates (x_2, y_2), then the corresponding interior node is placed at $(x_1 - \frac{y_2}{2}, \frac{y_2}{2})$.

In the semi-implicit framework employed here, the symmetry boundary condition was generally found to give superior results for large-amplitude problems, while the kinematic boundary condition was found to be superior for small-amplitude problems.

6.5 Results

For all the problems described in this chapter time integration was performed using the backward-Euler form of the semi-implicit scheme discussed in Section 3.5. The convective term was treated explicitly, i.e. it was evaluated at the start of each time step, using the solution computed at the end of the previous time step. Thus only a single linear algebraic problem needed to be solved at each time step. This modification was found to place no additional stability constraint on time step size, while giving results that were very similar to those obtained when the convective term was treated implicitly. The moving-mesh corrections described in Section 3.2

Mesh	k_{tol}	h_{max}	Nodes	Bndry. nodes	Elements	Unknowns	Time step
1	0.20	0.4	59	28	88	469	0.00100
2	0.10	0.3	133	52	212	1087	0.00050
3	0.05	0.2	301	96	504	2511	0.00025

Table 6.1: Small-amplitude axisymmetric droplet oscillations: initial mesh data.

were not included for the runs shown, since they were found to have no great effect on accuracy of the solutions computed [2].

ILUT preconditioning as described in Section 3.12.2, was employed for all problems. Unless otherwise stated the preconditioner was recomputed every ten time steps, employing values of $droptol = 5 \times 10^{-7}$ and $lfil = 200$. This typically resulted in a preconditioner with around twice the number of entries as the original finite element stiffness matrix. Profiling of the code showed that the run time was dominated by the costs of re-assembling the stiffness matrix at each time step. Each time step typically required fewer than five conjugate residual iterations to achieve convergence to an absolute tolerance of 10^{-10} in all components of the solution.

6.5.1 Small-amplitude oscillations

In order to verify the accuracy of the Navier-Stokes solver, spherical droplets of unit radius, perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.01$, were first considered. The initial meshes employed, shown in Fig. 6.3, were generated using the parameter values given in Table 6.1. The values of h_{max} were chosen so that the maximum edge length in the mesh decreased by a factor of at least $2^{-\frac{1}{3}}$ each time k_{tol} was halved. The initial locations of the nodes lying on the axis of symmetry were selected automatically using a grading algorithm similar to that employed for the mesh itself. In contrast to the meshes employed by Basaran [7], and Mashayek and Ashgriz [68], here the initial meshes are unstructured and while the initial boundary

[2] It should be noted, however, that for free-surface nodes the tangential mesh velocity $\dot{s}_t$ is zero and the normal mesh velocity $\dot{s}_n$ is equal to u_n. Thus the moving-mesh term $-(\dot{\mathbf{s}} \cdot \nabla)\mathbf{u}$ simplifies to

$$\left(-u_n \frac{\partial u_t}{\partial n} , -u_n \frac{\partial u_n}{\partial n} \right)^T \tag{6.51}$$

and consequently, provided $|\frac{\partial \mathbf{u}}{\partial n}|$ is small near to the free surface, the errors introduced by neglecting these terms will also be small. As the distance from the free surface increases the mesh velocity $\dot{\mathbf{s}}$ typically reduces rapidly and thus the errors introduced at nodes in the interior of the mesh are also small.

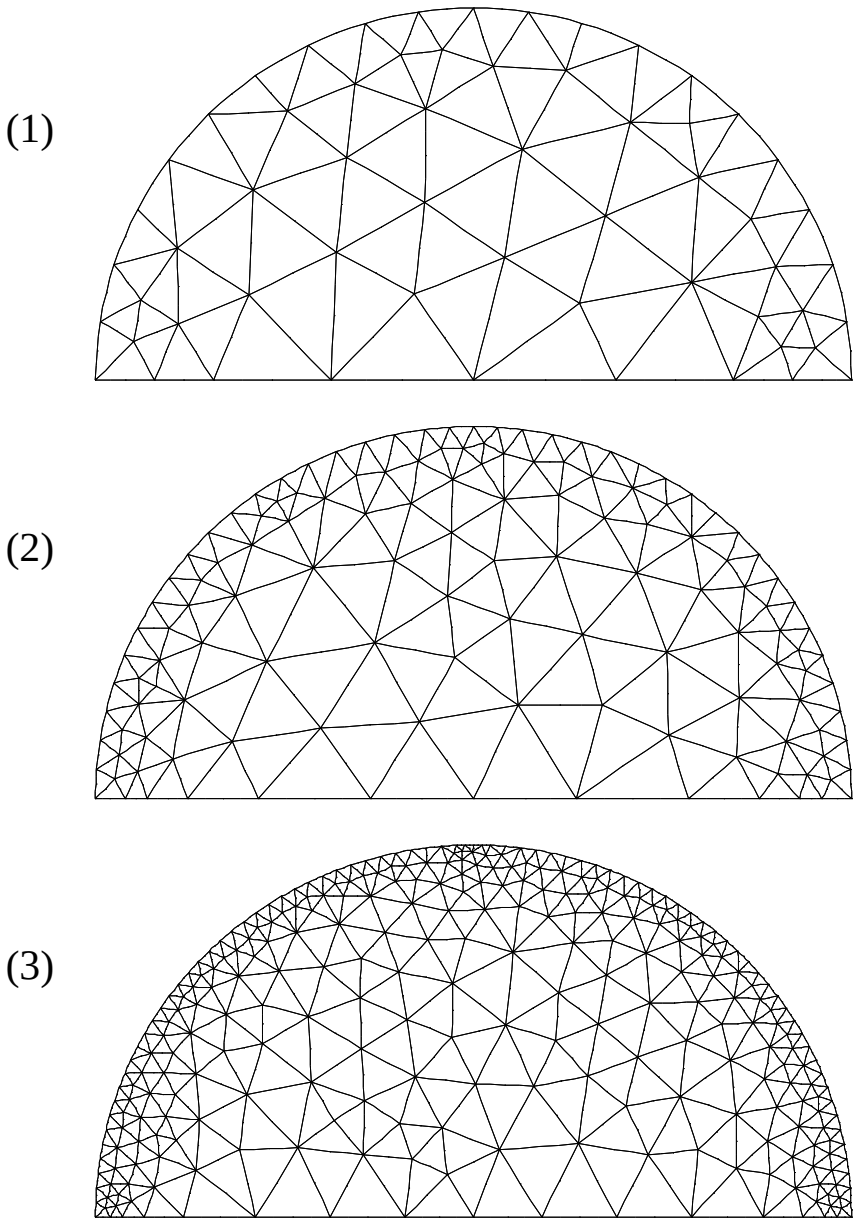


Figure 6.3: Initial meshes for a second-spherical-harmonic problem with amplitude $f_2 = 0.01$, as given in Table 6.1: (1) $k_{tol} = 0.2$, $h_{max} = 0.4$; (2) $k_{tol} = 0.1$, $h_{max} = 0.3$; (3) $k_{tol} = 0.05$, $h_{max} = 0.2$.

discretisations are symmetric in the plane $x = 0$, the initial meshes are not. Note that an amplitude of $f_2 = 0.01$ corresponds to a perturbation of the order of 1.5% of the unperturbed sphere's radius.

Time integration was performed using time steps of fixed length, the values employed being shown in Table 6.1. These were selected so as to prevent free-surface instabilities of the form described in Section 3.15 from occurring.

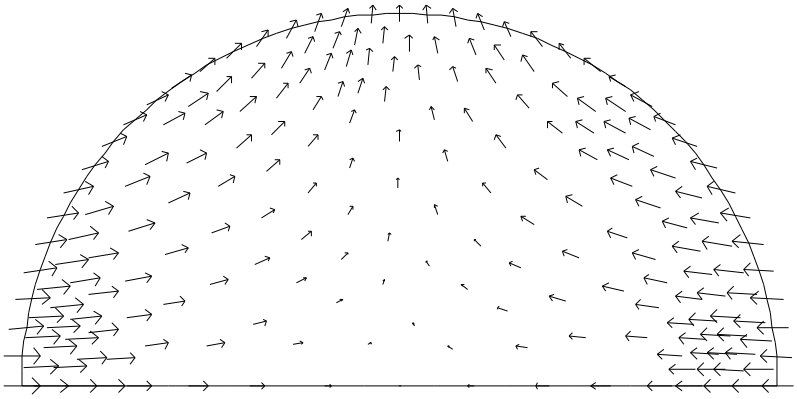


Figure 6.4: Velocity field for mesh 1 at $t = 0.1$, $Re = 10$.

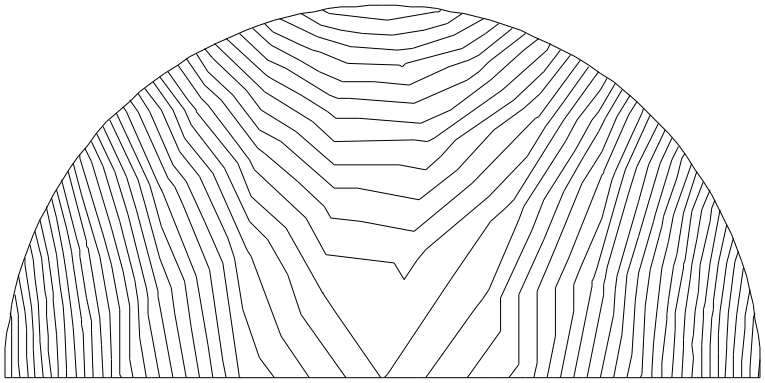


Figure 6.5: Pressure field for mesh 1 at $t = 0.1$, $Re = 10$.

Note that for all the problems in this chapter, unless otherwise stated, boundary mesh refinement and interior mesh regeneration were not performed; the only mesh update operations performed being Laplacian mesh-smoothing, as discussed in Section 2.5, and edge adjustment, as discussed in Section 2.4.3. At the two free-surface nodes lying on the axis of symmetry it was found preferable here to employ the kinematic boundary condition rather than free-surface symmetry boundary condition, since the latter was observed to give rise to small disturbances of the pressure field in elements close to the axial free-surface nodes. While these disturbances were small in magnitude, here the range of pressures involved is also small, and thus the disturbances when plotted are conspicuous. For larger-amplitude problems, in which typically the range of pressures is much larger, the distortions are much less

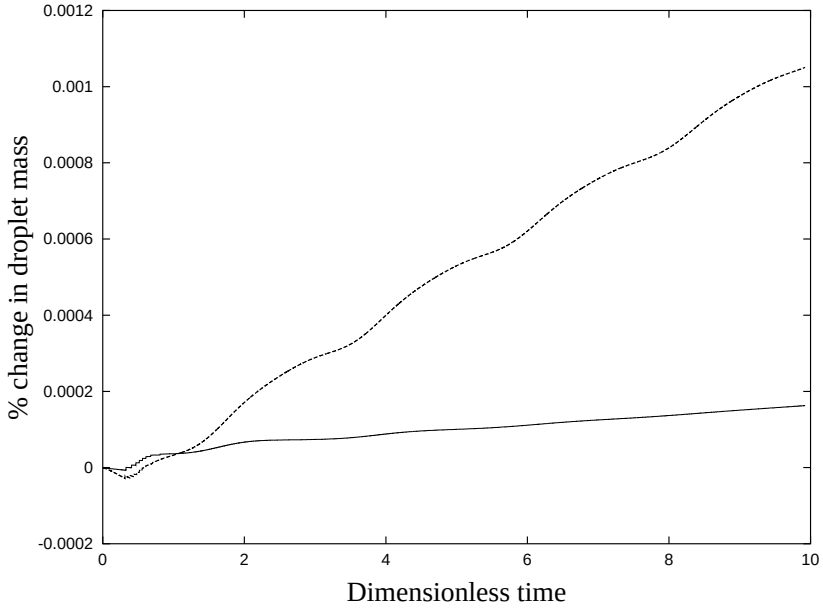


Figure 6.6: Percentage change in droplet mass as a function of dimensionless time using mesh 1: —— $Re = 10$; - - - - $Re = 100$.

apparent, and consequently the symmetry boundary condition is preferred for stability reasons. Regardless of the boundary condition employed, the solver produced almost identical results for the periods of oscillation computed.

Two problems were considered at Reynolds numbers of 10 and 100, a Weber number of 1 being employed in both cases. The droplet was released from a state of rest at the initial dimensionless time $t = 0$. Figures 6.4 and 6.5 show the velocity and pressure fields computed on mesh 1 at a dimensionless time $t = 0.1$, at a Reynolds number of 10. The pressure contours shown in Fig. 6.5, as with all the pressure plots in this work, were selected by requiring that a fixed number of contours be equispaced throughout the pressure range. Note that the pressures displayed in Fig 6.5 all lie in the narrow range 1.98 to 2.04. The velocity field shown in Fig. 6.4 appears smooth over the entire domain, while the pressure field appears smooth except in the region were the mesh is coarsest. This failure to accurately model the pressure appears however to have little impact on the accuracy of the solution, occurring as it does where the pressure gradients are smallest. On finer meshes the computed pressure fields are invariably far smoother. Thus it appears that for this problem, even a mesh as coarse as mesh 1 is adequate, in that it resolves the main features of the velocity and pressure fields. Consequently, convergence of the solution should be observed if the mesh is further refined. The corresponding velocity and pressure fields at a Reynolds number of 100 are very similar at this

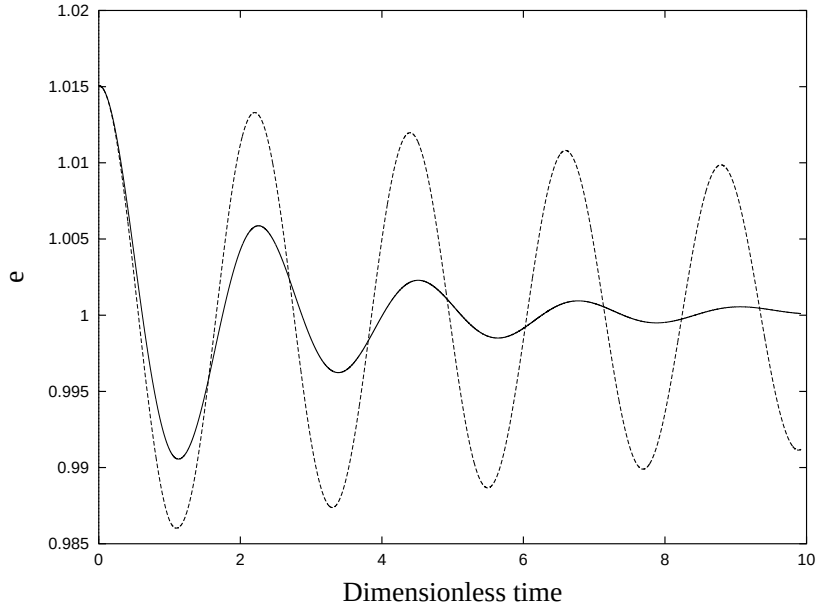


Figure 6.7: Variation of e with dimensionless time for small-amplitude oscillations of a droplet perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.01$, on mesh 1: —— $Re = 10$; - - - - $Re = 100$.

early stage of the flow, and so are not shown. Figure 6.6 shows the variation in the mass of the droplet with time, at both $Re = 10$ and $Re = 100$. In both cases the droplet's mass increases by less than 0.001% over the first four periods of oscillation, the greater rate of mass gain being observed at the higher Reynolds number as one would expect. The quantity $e = \frac{a}{b}$, where a and b are the distances from the centre of the droplet to the free surface along the x and y axes respectively, was employed in determining the period of oscillation of a droplet. Figure 6.7 shows how e varies with time for mesh 1 at the two Reynolds numbers. The period of oscillation of the droplet was estimated by observing the time interval between maxima in e. The length of the first period, t_1, together with the amplitude e_1 at the end of the first period, are listed in Table 6.2, for each of the three meshes. Column five gives the average processor time per time step, as measured over a considerable number of time steps (> 1000), on a Silicon Graphics R5000 workstation running at 180 MHz. The run time, shown in the last column, is a notional one found by multiplying the average processor time per time step by the number of time steps required to integrate up to a time $t = 2.3$, i.e. the approximate length of a period of oscillation. From Table 6.2 it may be seen that convergence is evident in both t_1 and e_1 as the mesh is refined, at both Reynolds numbers. Further decreasing the length of the time step was not found to result in an appreciable increase in accuracy. At the higher Reynolds number, t_1 appears to converge to a value close to 2.2245, which

Re	Mesh	t_1	e_1	time/step	Run time
10	1	2.2750	1.00588	1.79s	1.1h
10	2	2.2785	1.00597	4.47s	5.7h
10	3	2.2788	1.00602	9.50s	24.3h
100	1	2.2200	1.01330	1.85s	1.2h
100	2	2.2240	1.01345	3.90s	5.0h
100	3	2.2245	1.01353	9.63s	24.6h

Table 6.2: Small-amplitude axisymmetric droplet oscillations: results.

is in good agreement with the value of 2.2287 computed by Basaran [7], and differs from the analytical value of 2.2218 he quotes by only 0.12%. The amplitude of the oscillation at the end of the first period, e_1, varies little between the meshes, though it again shows signs of convergence. The value computed on mesh 3 of 1.01353 is in excellent agreement with that found by Basaran of 1.0136 and is, up to rounding error, identical with the analytical value of 1.0135 given by Basaran.

At the lower Reynolds number agreement is again good, with the period computed on mesh 3 of 2.2788 being within 0.5% of that found by Basaran. The value of e_1 computed is also in good agreement with Basaran's value of 1.0061.

It was observed that the scheme did not conserve momentum. Using mesh 1 an average acceleration of 5×10^{-4} non-dimensional units was observed over the first period, while for mesh 2 the average acceleration was approximately half that. This behaviour was observed irrespective of Reynolds number. Such accelerations appear to be due to asymmetries in the imposed free-surface boundary conditions.

6.5.2 Large-amplitude oscillations of ellipsoidal droplets

Two large-amplitude problems were next considered in order to investigate the wider applicability of the method. The first problem relates to the behaviour of droplets, the initial shape of which is obtained by rotating an ellipse centred at the origin around its major axis, which is assumed collinear with the x axis. The intersection of the initial droplet's surface with the x-y plane takes the form

$$x = \hat{a} \cos \theta, \tag{6.52}$$

$$y = \hat{b} \sin \theta, \tag{6.53}$$

where $\hat{a}$ and $\hat{b}$ are the minimum and the maximum radii of the ellipse. If only droplets with unit volume are considered then each problem is characterised by the

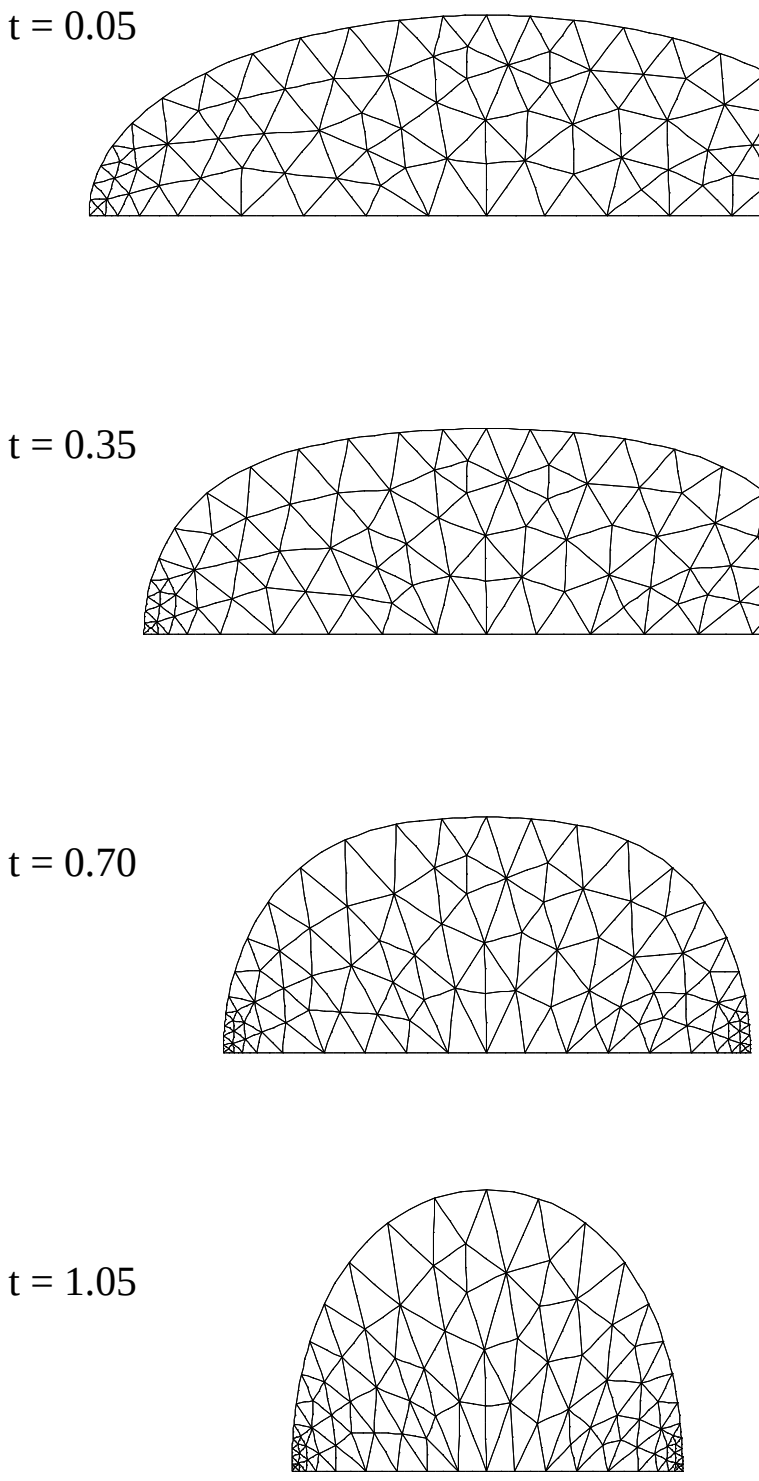


Figure 6.8: Large-amplitude ellipsoidal oscillations, $Re = 10$: mesh at selected dimensionless times.

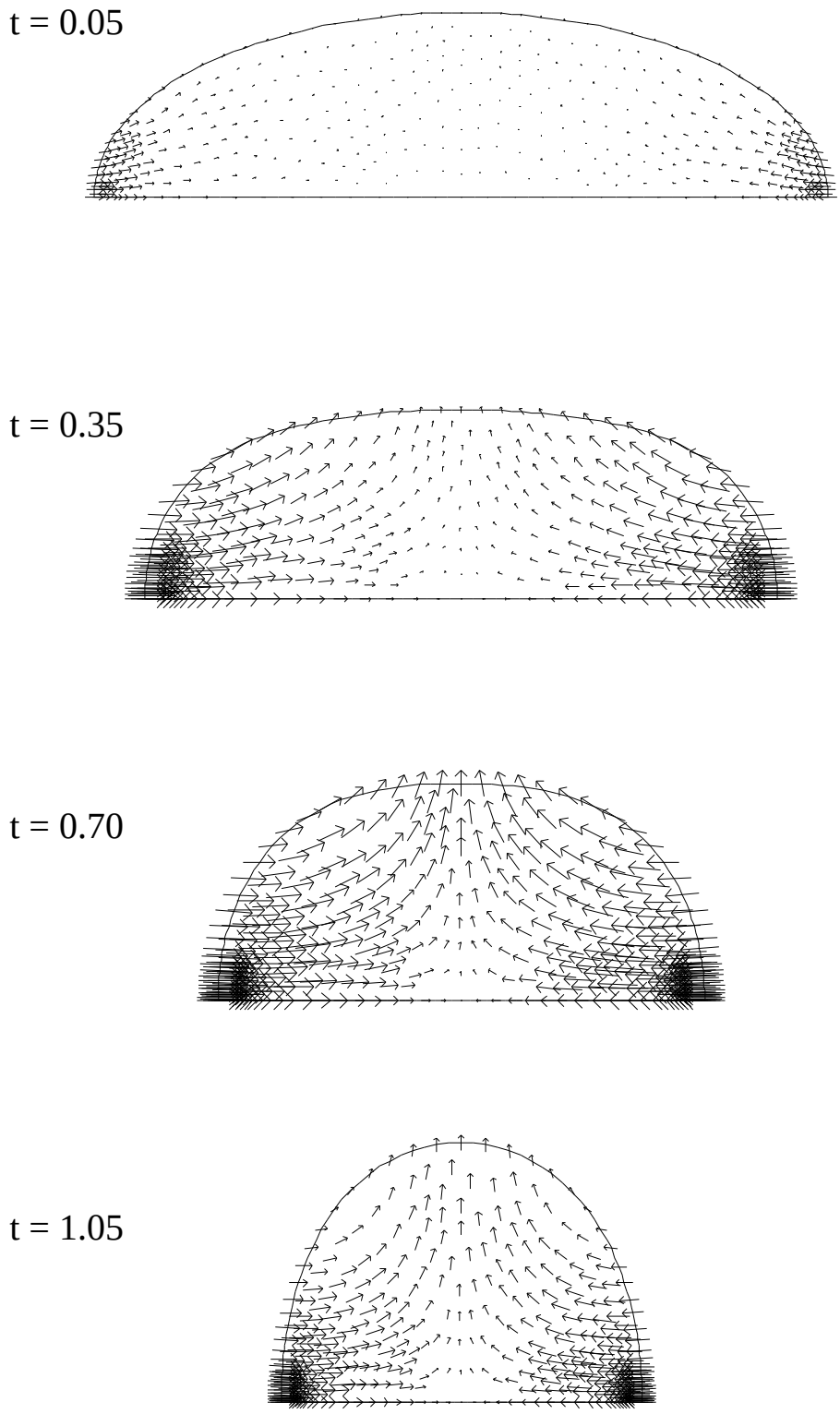
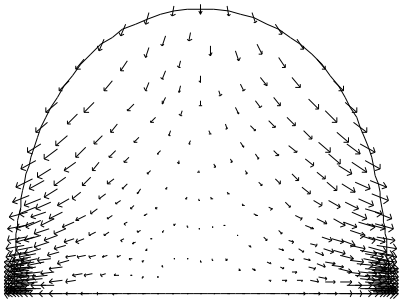
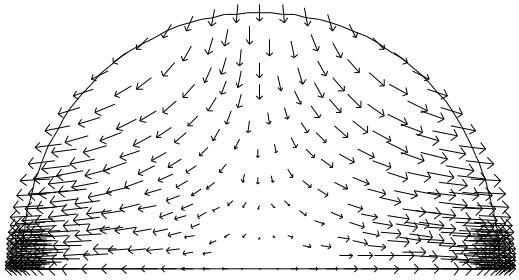


Figure 6.9: Large-amplitude ellipsoidal oscillations, $Re = 10$: velocity field at selected dimensionless times (a).

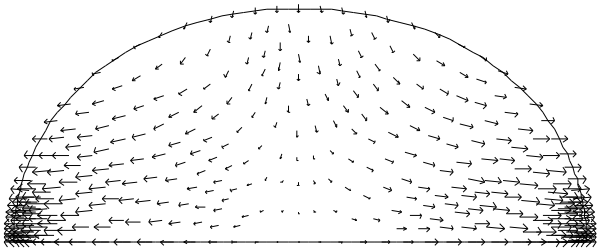
t = 1.40



t = 1.75



t = 2.10



t = 2.40

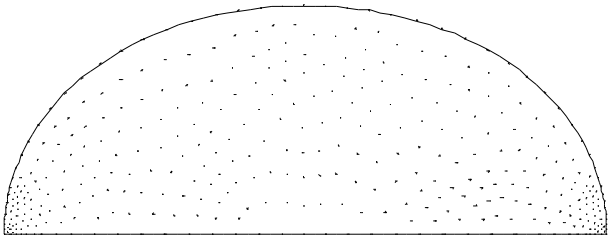


Figure 6.10: Large-amplitude ellipsoidal oscillations $Re = 10$: velocity field at selected dimensionless times (b).

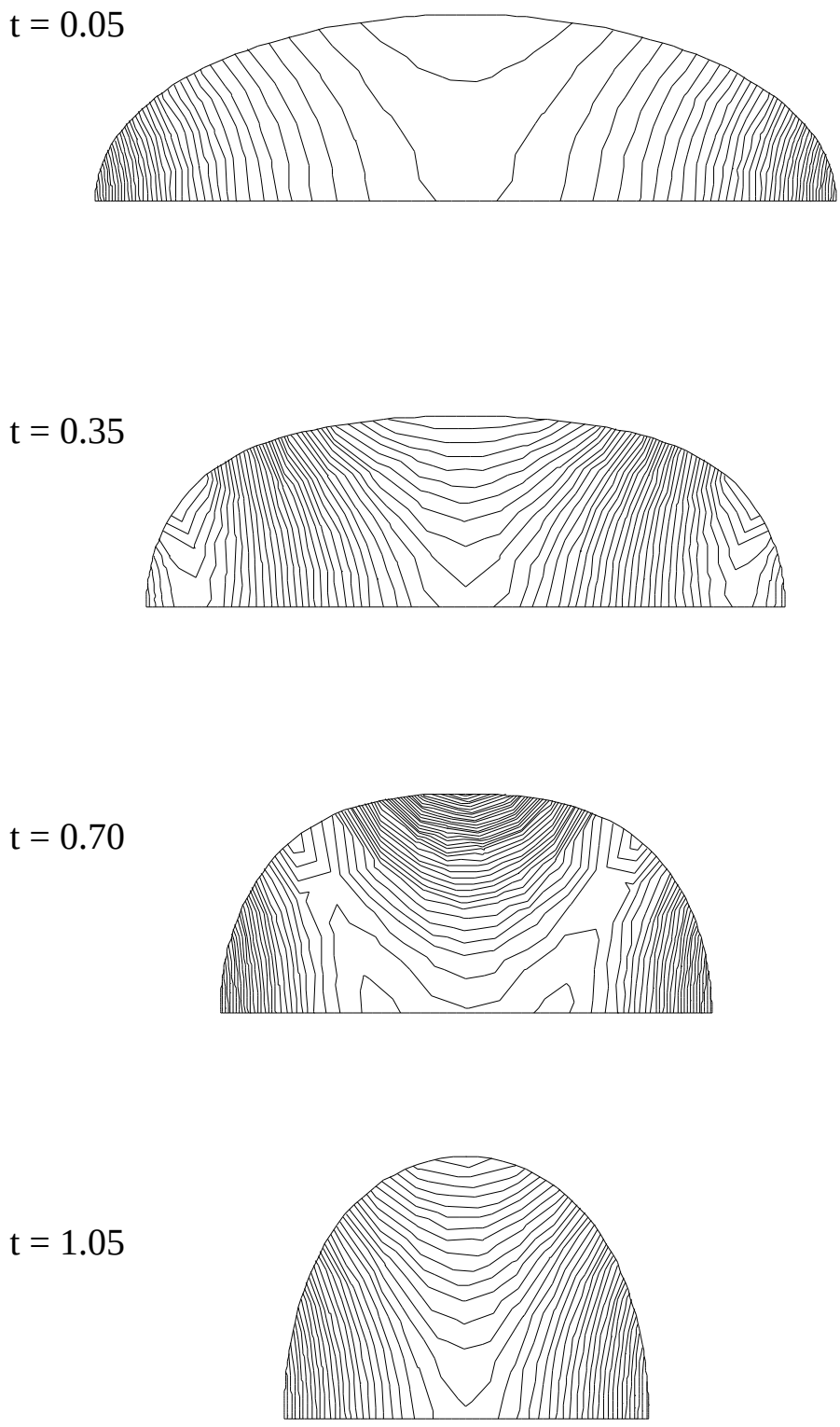


Figure 6.11: Large-amplitude ellipsoidal oscillations, $Re = 10$: pressure field at selected dimensionless times (a).

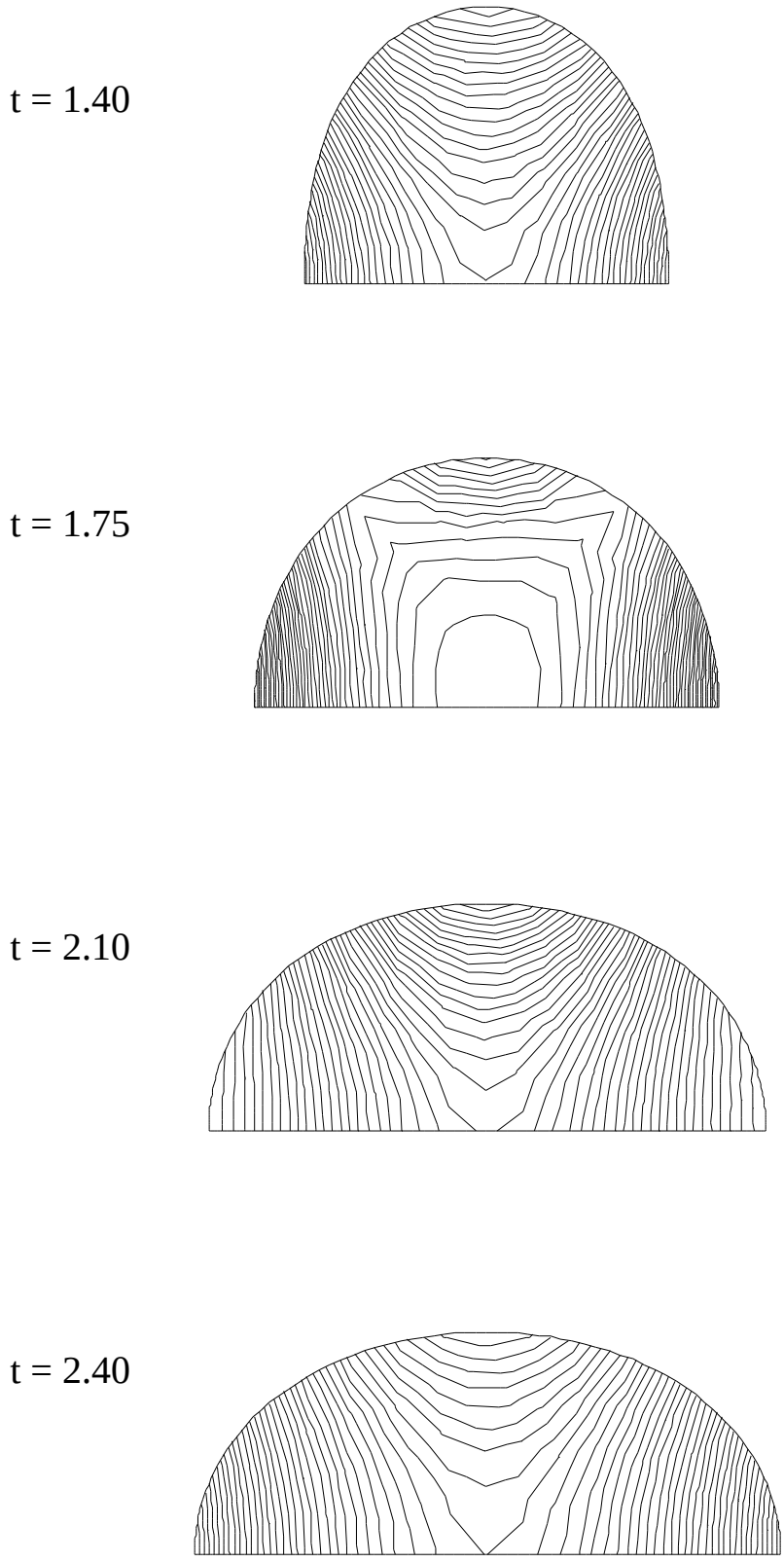


Figure 6.12: Large-amplitude ellipsoidal oscillations, $Re = 10$: pressure field at selected dimensionless times (b).

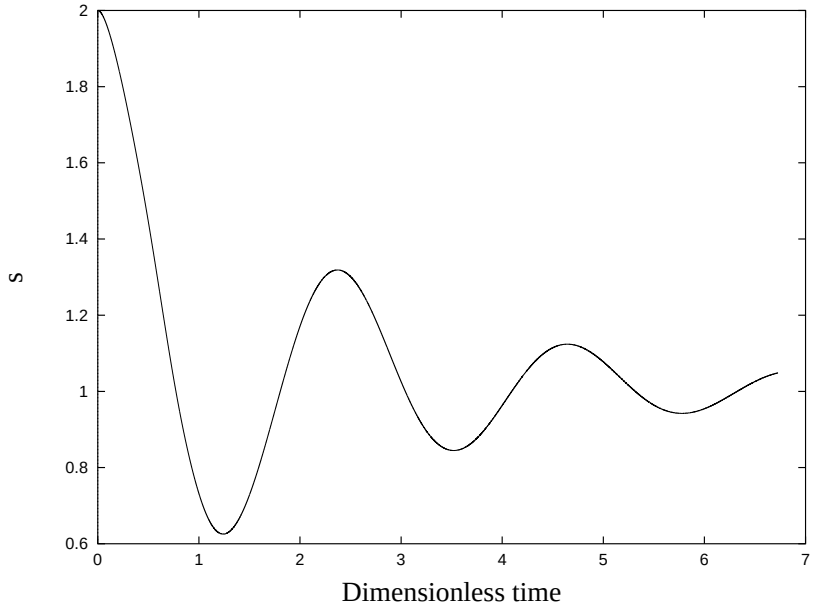


Figure 6.13: Large-amplitude ellipsoidal oscillations, $Re = 10$, $\hat{s} = 2$: s as a function of dimensionless time.

three parameters Re, We and $\hat{s}$, where

$$\hat{s} = \frac{\hat{a}}{\hat{b}}\,. \tag{6.54}$$

Conversely, given $\hat{s}$, $\hat{a}$ and $\hat{b}$ may be computed using

$$\hat{a} = \hat{s}^{\frac{2}{3}}, \tag{6.55}$$

$$\hat{b} = \hat{s}^{-\frac{1}{3}}. \tag{6.56}$$

Here the values $\hat{s} = 2$, $Re = 10$ and $We = 1$ were employed. To facilitate measurements the quantity

$$s = \frac{a}{b}, \tag{6.57}$$

was defined, where a and b are the distances, along the x and y axes respectively, from the centre of the droplet to the free surface. An initial mesh was generated by taking $k_{tol} = 0.2$ and $h_{max} = 0.25$, resulting in a mesh with 170 elements and 841 unknowns. A fixed time step of length 0.0005 was employed for this problem. Fig. 6.8 shows the mesh at selected times during the first half-period of the oscillation. Over the second half-period the motion of the nodes is essentially reversed and by the end of the first period the mesh has returned approximately to its initial configuration. Figures 6.9, 6.10, 6.11, and 6.12 show the velocity and pressure fields

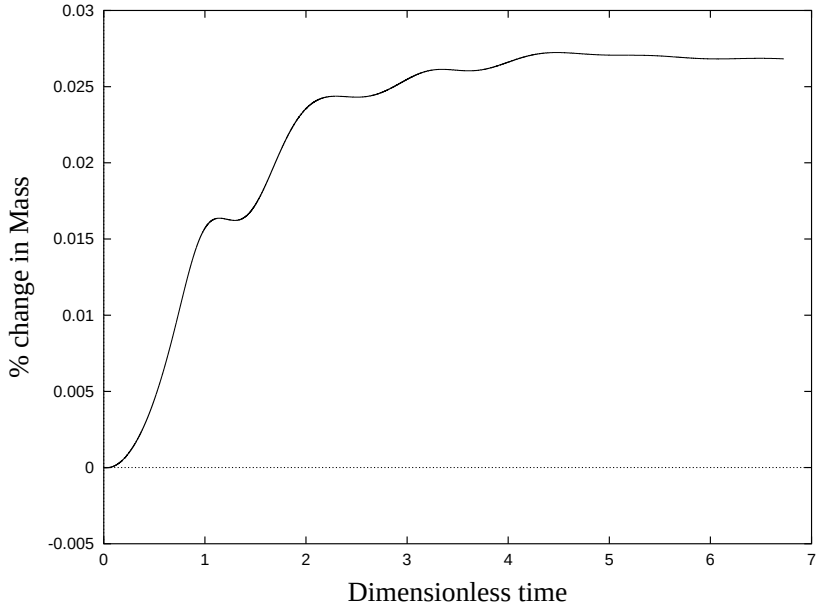


Figure 6.14: Large-amplitude ellipsoidal droplet oscillations, $\hat{s} = 2$: percentage change in mass as a function of dimensionless time.

at selected times after the droplet has been released from an initial state of rest.

Figure 6.13 shows the variation of s with time. The computed length of the first period, 2.39, is in reasonable agreement with that given by Mashayek and Ashgriz [68] of 2.41, a difference of approximately 0.8%. Agreement between the value of s computed at the end of the first period and that given by Mashayek and Ashgriz is also reasonable, i.e. 1.32 against their value of 1.28, a difference of approximately 3%. This perhaps reflects the neglect here of the moving-mesh correction terms which effectively adds momentum.

Figure 6.14 shows the variation of the droplet's mass with time as a percentage of its initial mass, the maximum change in the mass being approximately 0.03%. The rate of change of mass is clearly greatest when the free surface is moving most rapidly and, as the asymptotic configuration is approached, the rate of mass loss becomes negligible.

Figure 6.15 shows the velocity field at a time $t = 2.40$, shortly after the end of the first period of oscillation, and is obtained by scaling-up the velocity field shown in Fig. 6.10 by a factor of sixteen, a line through the centre of mass of the domain having also been added. A pair of counter-rotating vortices can clearly be seen and a certain degree of asymmetry of the flow is also apparent. This is presumably due to the accumulation of effects due to the asymmetries of the meshes employed up to this point.

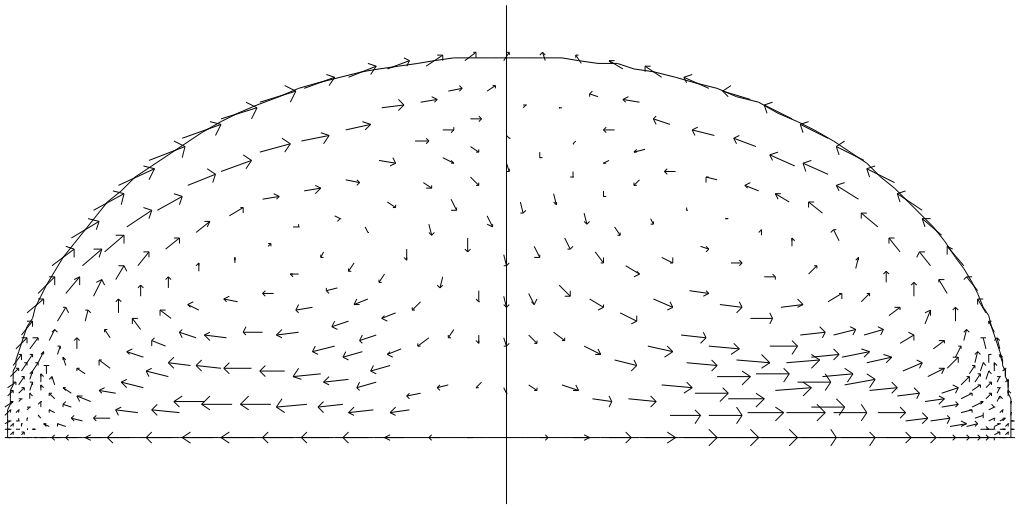


Figure 6.15: Large-amplitude ellipsoidal oscillations, $Re = 10$: velocity field at $t = 2.4$.

6.5.3 A large-amplitude second-spherical-harmonic problem

The next problem considered involves the evolution, at $Re = 100$, of a spherical droplet that has been perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.9$. Two initial meshes, shown in Fig. 6.16, were considered. Mesh A was generated by taking $k_{tol} = 0.2$ and $h_{max} = 0.4$, and has 132 vertices and 692 unknowns. Mesh B was generated by taking $k_{tol} = 0.1$ and $h_{max} = 0.3$, and has 185 vertices and 1511 unknowns. A fixed time step of $k = 0.00025$ was employed for mesh A, while one of $k = 0.000125$ was employed for mesh B. Note the poor quality of mesh B in the neck region, due to the failure of the free-surface meshing algorithm to detect the changes in the sign of the curvature, as discussed in Section 4.4.2.

Figure 6.17 shows the evolution of e with time for the two meshes. As may be seen, the two curves are surprisingly similar given the coarseness of the two meshes. For mesh A a period of 2.750 was measured, which differs from Basaran's [7] value of 2.906 by approximately 5.4%. A better agreement was observed for the amplitude, e_1; at the end of the first period the value computed was 2.329 as against Basaran's value of 2.331, a difference of only 0.09%. For mesh B values of $t_1 = 2.772$ and $e_1 = 2.327$ were computed. Thus, while the period computed was marginally closer to Basaran's value, the discrepancy in the amplitudes was approximately twice as large. Figure 6.18 shows the computed free surface at selected times. A visual comparison with Mashayek and Ashgriz's Figure 9 [68], however, shows that considerable differences in free-surface shape occur after $t = 0.4$. Since Mashayek and Ashgriz's free-surface profiles appear to be in good agreement with those computed

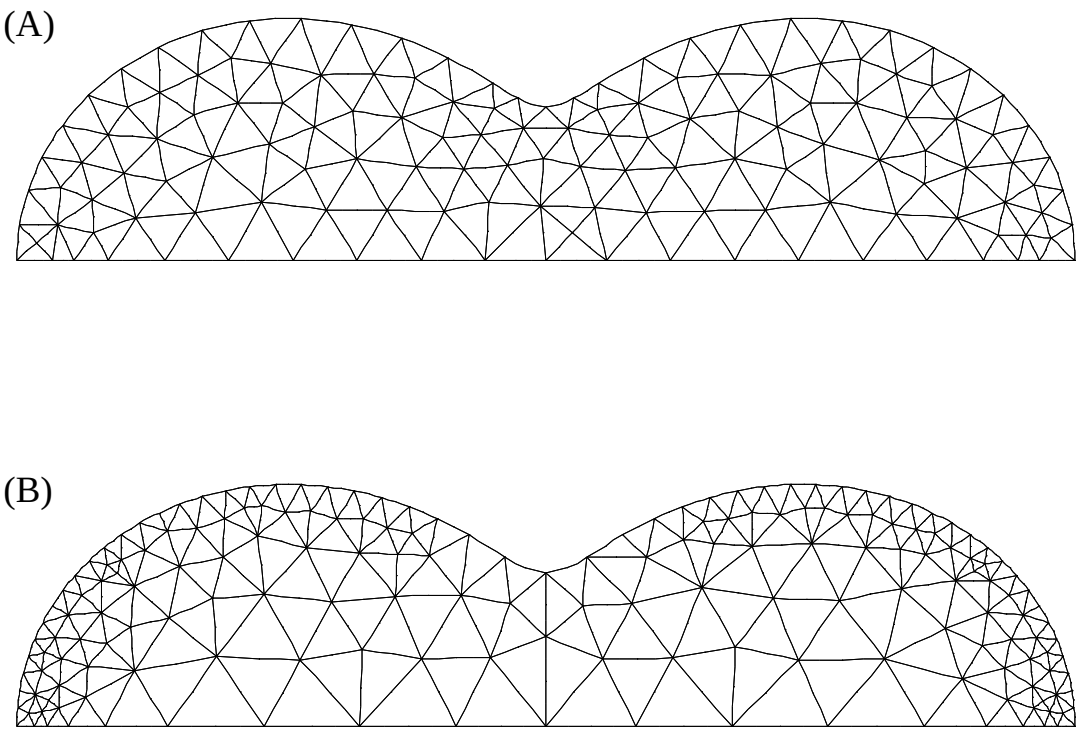


Figure 6.16: Initial meshes A and B for a second-spherical-harmonic problem with $f_2 = 0.9$.

by Basaran, one is forced to conclude that it is the current computation that is in error. A closer study of the free surface's evolution shows that oscillations of considerable amplitude occur in the locations of the axial free-surface nodes around time $t = 1.2$. The behaviour observed suggests that modes of oscillation are being excited in the current model that are not observed in those employed by the other authors. Thus it is apparent that agreement between two different models as to the overall period of oscillation, does not allow one to conclude that the details of the two solutions are in similar agreement. A lack of mesh resolution in the initial neck region appears likely to be the main cause of the discrepancies observed, though another may lie in the large deformations of elements that occur. Figure 6.19 shows mesh B at time $t = 1.3$, and illustrates the severe deformations of elements that arise.

The use of interior remeshing was next investigated, with the aim of avoiding the

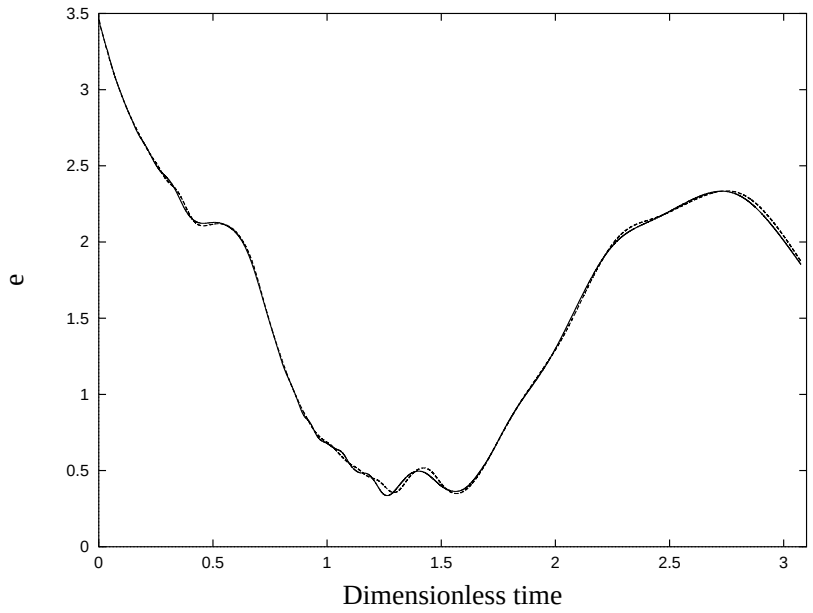


Figure 6.17: Oscillations of a spherical droplet perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.9$ at $Re = 100$, e as a function of dimensionless time: —— mesh A; - - - - mesh B.

above problems. These attempts however proved unsuccessful. Failure was invariably due to large local curvatures arising on the free surface, resulting in singularity of the finite element discretisation. Figure 6.20 shows the evolution of the free surface when interior remeshing was performed, employing mesh B as the initial mesh. Asymmetry in the plane $x = 0$ is clearly visible at time $t = 1.0$. The cause of these asymmetries appears to be the asymmetry of the interior mesh. If the solution of problems that are symmetric in $x = 0$ is all that is required, then the symmetry may be imposed directly by modelling only half of the above domain, employing the appropriate symmetry boundary condition on $x = 0$. This has the additional benefit of approximately halving the number of unknowns that must be solved for.

Free-surface remeshing was also briefly investigated, employing adaptive time-step size selection to reduce the run time. Mesh B was again used as the initial mesh. Free-surface remeshing invariably failed due to stalling of the adaptive time integration scheme when excessively small time steps became necessary. This in turn was due to the attempts of the automatic refinement algorithm to track free-surface features of high curvature that arose during the simulations.

The source of these high-curvature free-surface features appears to lie in the fact that errors introduced at an early stage of the computation are only weakly damped in a Navier-Stokes flow. While for surface-tension-driven Stokes-flow problems the velocity field is specified entirely by the instantaneous shape of the free surface, for

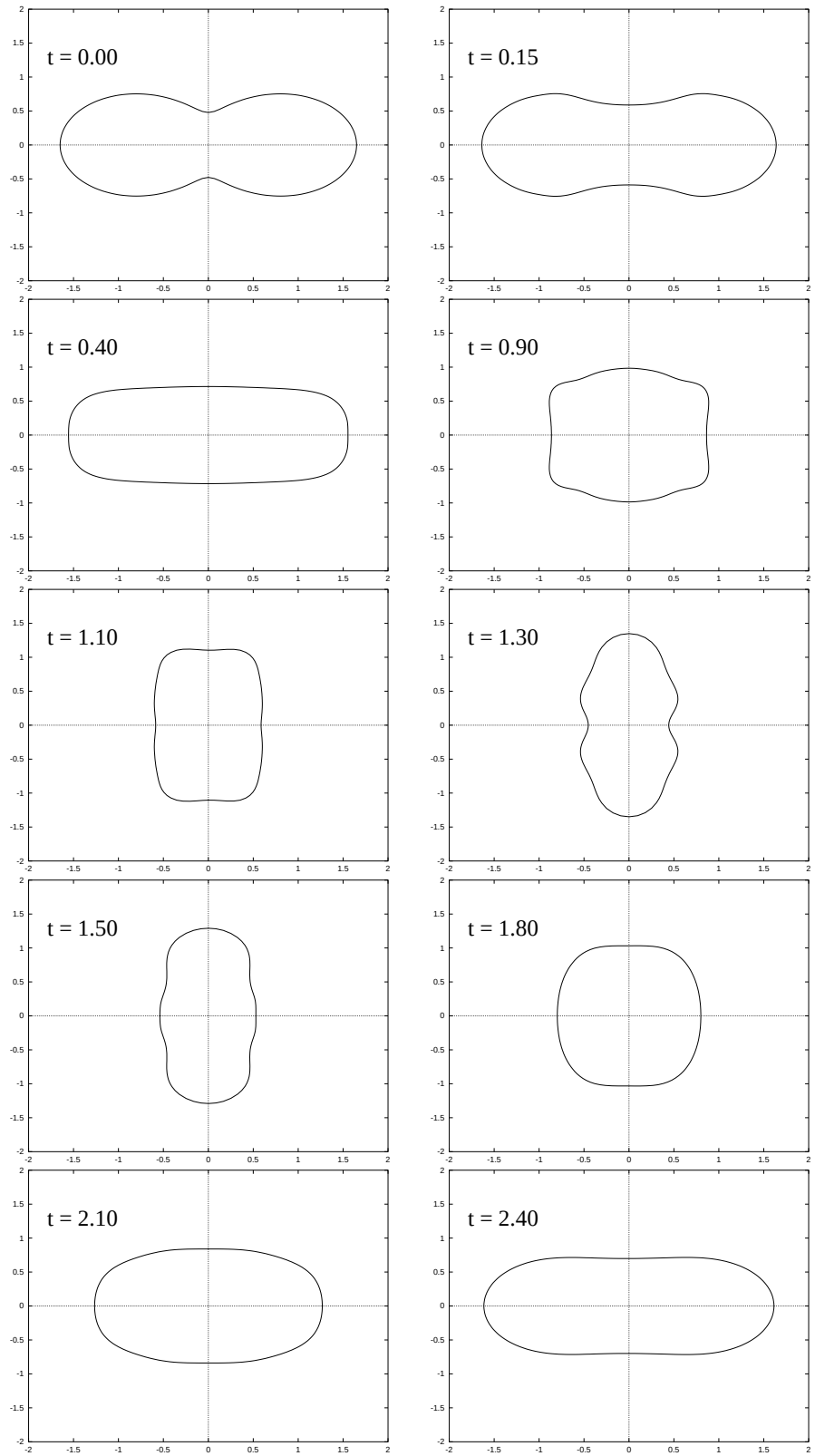


Figure 6.18: Oscillations of a spherical droplet perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.9$, at $Re = 100$: free-surface evolution for initial mesh B, with no mesh regeneration, at selected dimensionless times.

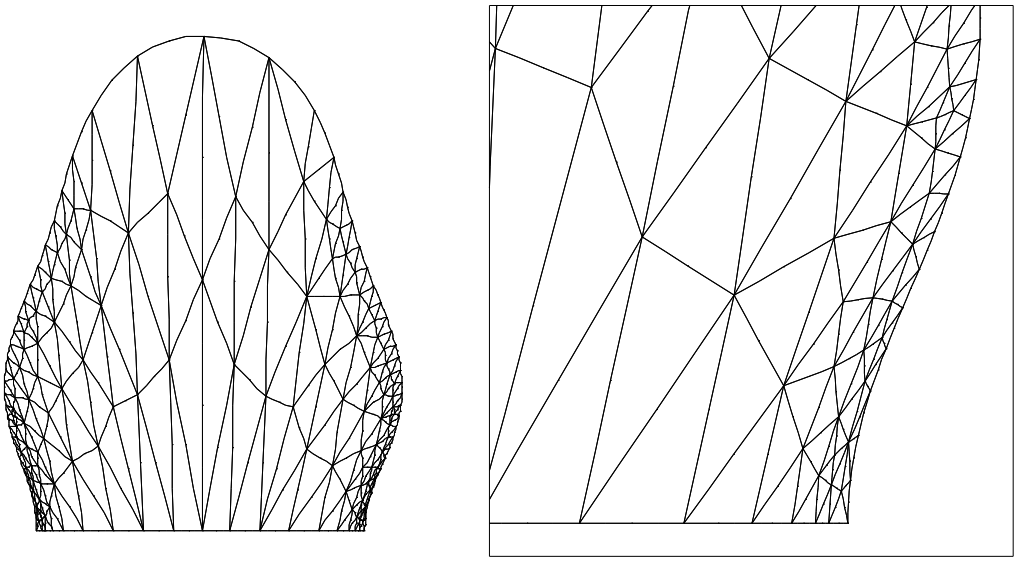


Figure 6.19: Oscillations of a spherical droplet perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.9$, at $Re = 100$: (a) free surface at $t = 1.3$; (b) detail.

Navier-Stokes problems considerable internal structure may develop in the velocity field, as illustrated by Fig. 6.15. It appears reasonable to assume that features such as the vortices shown in Fig. 6.15 can have a considerable influence on the evolution of the free surface, and thus that errors in the flow computed for the interior of the droplet can affect the free surface far from the source of the initial error. In these situations refinement of the free surface, merely allows the solver to more accurately model features that have arisen due to errors earlier in the computation; *the refinement is both too late and in the wrong place.* One solution to this problem lies in the implementation of adaptive interior mesh generation, employing error indicators based on local solution gradients. A simpler, though perhaps less efficient solution, would be to employ a finer mesh for the entire domain.

6.5.4 Interior-mesh regeneration

As a final experiment the above problem was repeated at $Re = 10$, employing interior-mesh regeneration but no automatic refinement of the boundary. A new initial mesh C, shown as Fig. 6.21(a), was generated by taking $k_{tol} = 0.2$ and $h_{max} = 0.2$. In addition the algorithm used to generate the initial free-surface node distribution was modified so that no edge of length greater than 0.05 was gener-

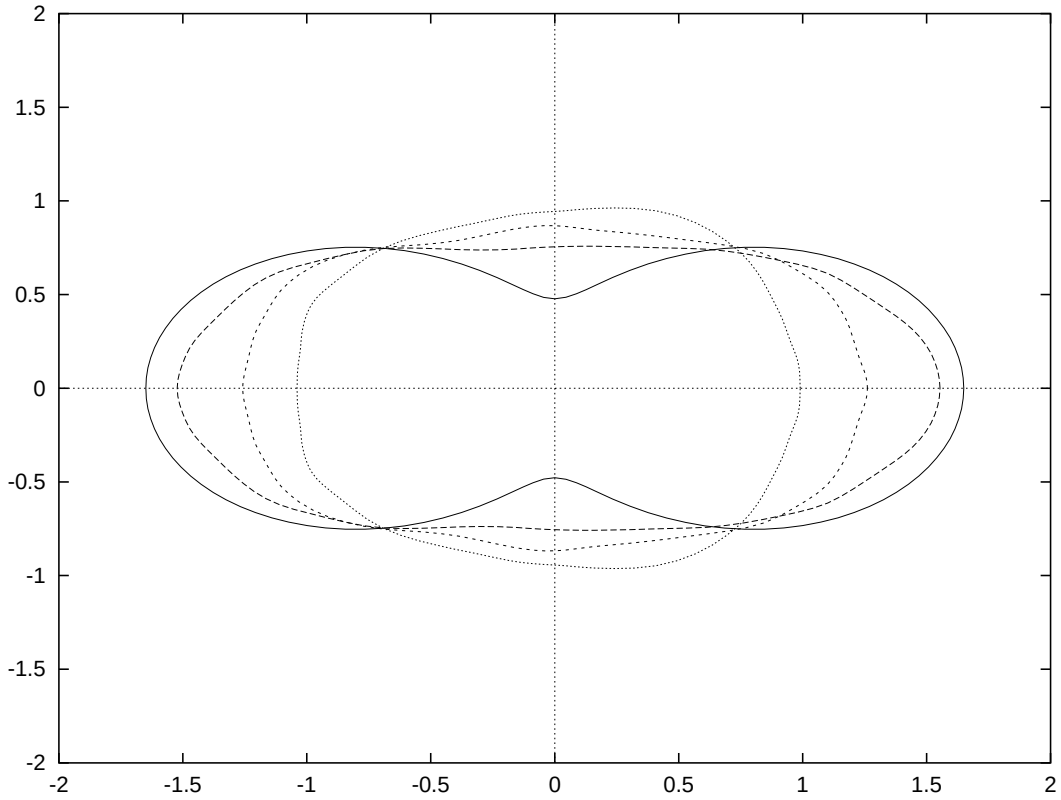


Figure 6.20: Free-surface profiles arising in a large-amplitude, $f_2 = 0.9$, second-spherical-harmonic problem at $Re = 100$, employing interior mesh regeneration with same parameters as mesh B: —— $t = 0.0$; – – – $t = 0.6$; - - - - $t = 0.8$; $\cdots$ $t = 1.0$.

ated, thus ensuring that initially the free-surface nodes were approximately equidistributed. As may be seen, the density of free-surface nodes is now far greater in the neck region than it was with mesh B. This has the side-effect of forcing Triangle to generate a finer interior mesh in the neck region. Boundary node spacing along the axis of symmetry has also been reduced. Mesh C contains 536 elements and involves 2662 unknowns. It is thus roughly comparable in resolution to Basaran's mesh IVB, which contains 192 quadrilateral elements and involves 1904 unknowns. While mesh C involves more unknowns, quadrilateral elements are generally acknowledged to be more accurate that triangular ones, and thus solutions of a similar accuracy might be expected.

The interior of the mesh was regenerated every forty time steps and whenever the minimum angle fell below 10° or the maximum angle rose above 150°, the velocity field being interpolated onto the new mesh using the quadratic scheme described in Section 3.14.1 so as to give initial conditions for the next time step. A preconditioner was recomputed every ten time steps and whenever the mesh was regenerated. An

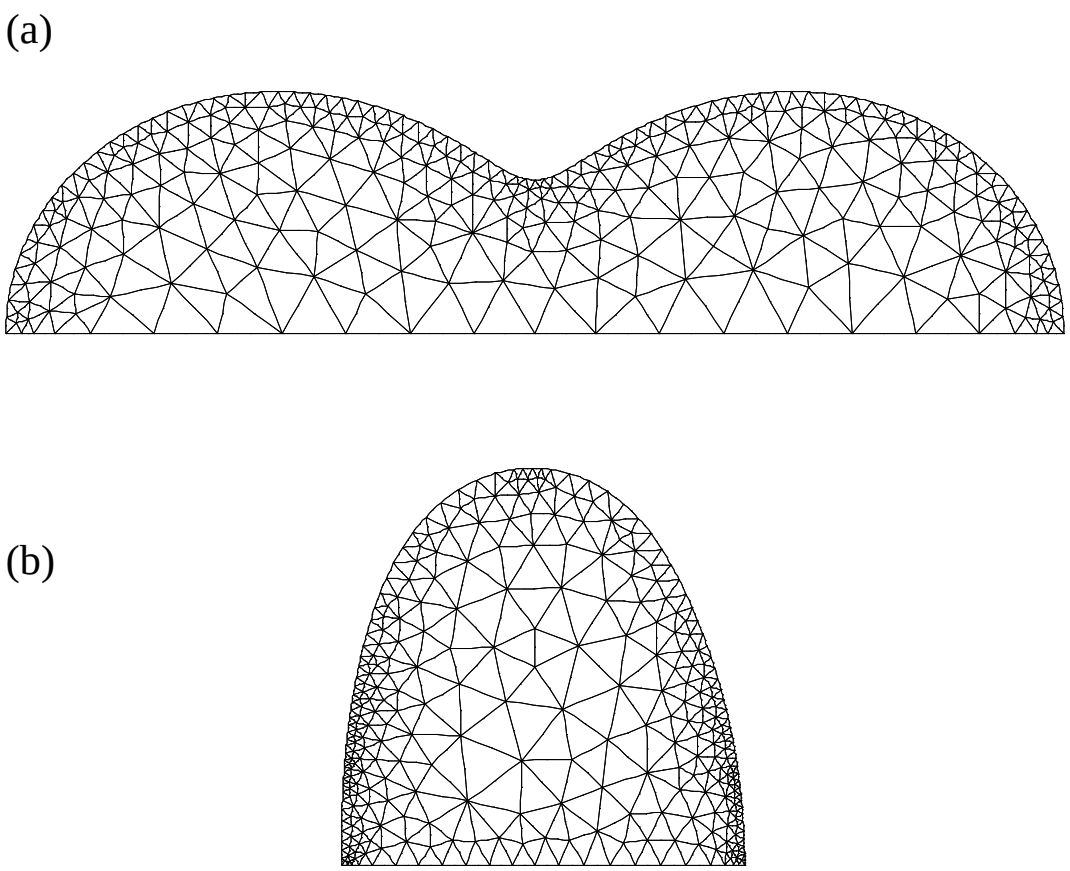


Figure 6.21: Large-amplitude, $f_2 = 0.9$, second-spherical-harmonic problem at $Re = 10$, mesh C with initially equispaced free-surface nodes: (a) initial mesh; (b) mesh at $t = 1.43$.

increased amount of fill-in was allowed in the incomplete factorisation of the preconditioner by taking $lfil = 300$. Adaptive time-stepping as described in Section 3.15 was employed, with a prescribed tolerance of 5×10^{-5}. This reduced the estimated 65 hour run time, that a fixed time step size would have entailed, down to a more reasonable 10 hours for a single period.

Figure 6.21(b) shows the mesh at time $t = 1.43$, from which it may be seen that the maximum element aspect-ratio is approximately equal to two. The uneven distribution of free-surface nodes, that does not truly reflect the curvature of the boundary, is however unwelcome. Free-surface remeshing would thus be appropriate here, provided that the difficulties mentioned above, with regard to the accuracy of the velocity in the interior of the mesh, can be resolved. Figure 6.22 shows the

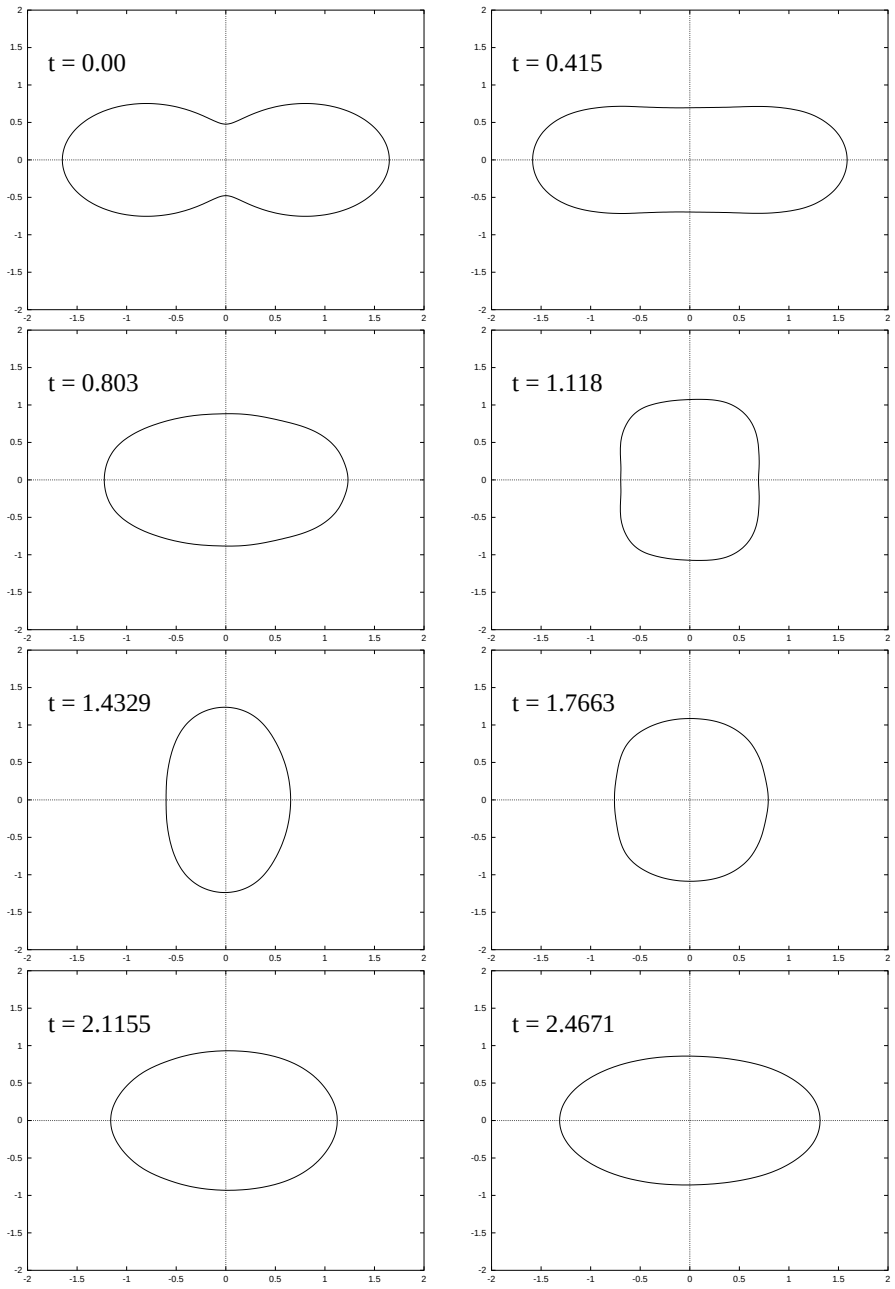


Figure 6.22: Oscillations of a spherical droplet perturbed by a second-spherical-harmonic of amplitude $f_2 = 0.9$ at $Re = 10$: free-surface evolution, for initial mesh C with interior mesh regeneration, at selected dimensionless times.

evolution of the free surface over most of the first period. While a certain degree of asymmetry is still apparent in these free-surface profiles, it is less pronounced than that visible in Fig 6.20 at the higher Reynolds number and on a coarser mesh.

A period of 2.57 was computed, which is in reasonable agreement with Basaran's value of 2.66 [7], a difference of 3.4%. For the amplitude at the end of the first

period a value of 1.54 was computed, as against Basaran's value of 1.43, a difference of 7.7%. These results suggest that additional refinement of the interior of the mesh would further reduce the tendency for asymmetry to arise, and thus allow results comparable in accuracy to Basaran's to be obtained.

6.6 Conclusions

This chapter has demonstrated that unstructured meshes of triangular elements may be used successfully to solve axisymmetric Navier-Stokes problems involving small-amplitude perturbations of spherical droplets. Good agreement has been demonstrated with both Prosperetti's theoretical model and Basaran's numerical results for small-amplitude oscillations.

The accurate modelling of larger-amplitude problems appears however to be considerably more difficult; this being particularly apparent when the droplet is expected to remain symmetric. For Navier-Stokes problems, in contrast to Stokes-flow problems, it appears necessary to give additional consideration to the need to ensure the accuracy of the solution on the interior of the mesh; employing a good boundary representation is no longer sufficient to guarantee a stable and accurate solution. While the moving-mesh approach has been employed with some success, both with and without automatic regeneration of the interior mesh, automatic regeneration of the free-surface discretisation has proved more problematic. Further investigation of the additional difficulties faced when regeneration of the free surface is attempted is thus appropriate. The preliminary results obtained however suggest that when the problems mentioned above are resolved the approach will be a useful one.

The use of ILU preconditioning has been demonstrated for this class of problem, and the technique has proved to be robust and easy to use in practice. Finally, the considerable reductions in run time made possible by adaptive time-step size selection have been demonstrated.

Chapter 7

Conclusions

As conclusions have already been presented at the ends of the preceding chapters, here only the main points will be reiterated. First, the prospects for the methods described in this thesis are reviewed. Comparison is then made with the main alternative approaches currently available for free-surface incompressible-flow problems. Finally, a number of suggestions are made for the further development of the unstructured mesh approach.

7.1 The unstructured mesh approach

With regard to its main objective the current work appears to have been broadly successful in that a methodology has been developed that allows a wide range of time-dependent surface-tension-driven free-surface problems, involving domains with arbitrary initial free-surface profiles, to be simulated accurately over extended periods of time. The use of automatically generated unstructured finite element meshes in a time-dependent free-surface incompressible-flow solver has been demonstrated for the first time. The approach has been shown to be practical, to be applicable to a range of problems, and to allow solutions of demonstrable accuracy to be obtained.

The primary advantage of an unstructured finite element method is that no *a priori* decisions need be made about the structure of the interior mesh and thus, once the initial free-surface profile and the boundary and initial conditions have been specified, the codes developed may be employed to automatically compute time-

dependent flows in domains that change considerably in shape. While here only surface-tension-driven incompressible-flow problems are considered, the use of the finite element methodology allows the methods outlined to be directly generalised to other types of problem such as non-Newtonian flows, elasticity problems with free surfaces, and phase change problems.

Experience has shown that surface-tension-driven free-surface flow problems are considerably more difficult to solve than incompressible-flow problems for which essential boundary conditions are known *a priori*. The additional difficulties are primarily a result of the fact that the shape of the free surface, and thus the boundary conditions, depends on the solution at previous time steps. In addition mass-conservation considerations mean that the systems of equations must be solved to high accuracy.

The investigations undertaken have led to the conclusion that for many transient problems, relatively simple semi-implicit schemes may well be competitive with fully implicit ones; in part due to the possibility of the reuse of preconditioners when the time step is small in size, as is typically the case with semi-implicit schemes, but also due to the considerably greater costs associated with solving a fully implicit system of equations on a moving mesh. Experience has shown that, for the semi-implicit schemes described here, the additional costs associated with managing a moving mesh are typically small in comparison with the solution costs associated with the flow problem itself.

The work has highlighted the issue of the accuracy of boundary conditions computed using piecewise-continuous boundary representations. In particular the potential for the rate of convergence of the boundary conditions to compromise the overall rate of convergence of the solution has been highlighted, a problem that has apparently hitherto been overlooked.

An important feature of the approach described here is that no artificial smoothing of the free surface is performed. Furthermore, the stability-based adaptive time-step size selection procedures described in Section 3.15 allow maximal time steps to be taken while ensuring stability of the free surface.

The use of iterative methods for the solution of the systems of linear equations has been explored and the results obtained suggest that they are highly advantageous, particularly when the problem allows preconditioners to be reused many times. The need to restrict time-step size when semi-implicit schemes are employed may be turned to an advantage in that it allows powerful ILUT preconditioners to be reused over many time steps, offsetting the considerable cost of computing them.

Indeed, where a steady-state solution is being approached the preconditioner may not need to be recomputed at all.

7.2 Comparison with alternative approaches

In the approach proposed by Mashayek and Ashgriz [68] penalty methods are employed for the solution of the incompressible-flow problem, and a volume-of-fluid (VOF) scheme is employed to update the free surface. Although the VOF approach appears attractive in that it conserves mass exactly, the fact that it does not impose the kinematic boundary condition directly makes it less attractive from a theoretical point of view. While the use of unstructured meshes would complicate the implementation of a VOF scheme, there appears to be no reason why, in principle, such an approach could not be integrated with the unstructured mesh approach described here. Similarly there is no reason why penalty methods could not also employed as the basis of the incompressible-flow solver.

For problems in which the mesh need not change in connectivity during a simulation the implicit methods proposed by Basaran [7] would appear to have considerable advantages, particularly where accuracy considerations allow large time steps to be taken. Basaran's scheme for updating the positions of interior nodes is representative of those adopted by many other authors. In it an interior vertex moves along the fixed spine on which it lies, at a velocity that is a continuous function of the velocity of the free surface where it intercepts the spine. Interior edge nodes are constrained to move so as to keep interior edges straight. One important advantage arising from this approach is that the motion of an interior node, and thus each basis function associated with the node, depends on the motion of only three free-surface nodes. Consequently the Jacobian of the system of nonlinear algebraic equations that results when Newton's method is applied is sparse. Furthermore the Jacobian may be computed analytically and thus accurately and efficiently. Where an unstructured mesh is employed, and Laplacian or elastic-mesh smoothing used to update the positions of interior nodes, the computation of the Jacobian will be considerably more complicated since now each basis function depends on the locations of all free-surface nodes. As a result the Jacobian will contain a dense block and will be much more expensive to compute. Since the main justification for employing implicit methods is that large time steps may be taken, and since in such circumstances one would expect that the mesh would change considerably over a time step and thus that the Jacobian and any preconditioner would need to be recomputed

frequently, the question must be addressed as to whether this is preferable to taking a relatively large number of much cheaper semi-implicit steps.

The large additional costs associated with the solution of a time-dependent nonlinear moving-mesh problem lead the author to suspect that more efficient approaches might exist in which the mesh remains fixed throughout a computation. Thus it may be the case that, when combined with efficient solvers, the phase-field methods mentioned in Chapter 1 will prove to be more efficient than moving-mesh methods. Such methods are, however, at an early stage of development and it remains to be seen if they live up to their promise.

7.3 Suggestions for further work

The potential further gains in efficiency made possible by the use of optimal ILUT preconditioners have already been remarked upon in Section 4.3. If the approach outlined were adopted, then a two- to four-fold decrease in run times might well be obtainable. The use of an explicit second-order accurate free-surface advection scheme should also be considered as a priority.

Further theoretical work is also appropriate. The stability method for choosing the time-step length would appear to invite further investigation. In particular it would be valuable to confirm experimentally that the effective time-step-size constraint is indeed first order. Analysis linking the global error to the local spatial and temporal truncation errors would also now appear to be appropriate. An improved understanding of the error analysis is necessary if the various tolerances for the solution of the systems of linear equations are to be selected optimally; i.e. so as to minimise run time while, at the same time, delivering a solution of guaranteed accuracy.

There is considerable scope for the further development of the Laplacian mesh-smoothing algorithm currently employed. In particular, the algorithm might be modified so as to prevent large and small angles from developing. Alternatively, the fully Lagrangian approach might be adopted for Navier-Stokes problems. Since experience has shown that mesh generation costs are small when compared to the costs of solving the flow equations the only additional difficulty that would arise is that interpolation of the velocity field would be required more often due to more rapid degeneration of mesh quality. Since the Lagrangian approach may conveniently be implemented within the framework employed here, with minimal additional work, there appears little reason why it should not be further investigated. The devel-

opment of accurate methods for transferring solutions between meshes should now also be considered a priority.

While the current implementation allows a large family of interesting problems to be tackled, the range would be further extended by the inclusion of models for inflow and outflow boundaries at which free surfaces are present. The inclusion of simple models for static and dynamic contact lines would also considerably extend the range of problems that might be investigated, and would allow solutions to be obtained for many industrially important problems, such as those described in Chapter 1, for which to date only steady-state solutions have been obtained. For coating-flow problems such as those mentioned in Chapter 1, the domain geometries allow narrow bandwidths to be obtained by appropriate node orderings, and thus allow good preconditioners to be computed efficiently. Furthermore, for many coating-flow problems one would not expect to observe rapid motions of the free surface, and thus any deformation of the interior mesh would also be expected to occur slowly, allowing preconditioners to be reused over many time steps. For such problems the rate of mass loss due to inaccuracy in the explicit update of the free surface would be very small, depending as it does primarily on the normal free-surface velocity. Thus, while such problems are simple enough that fully implicit methods might be expected to be successful, the very features that make implicit schemes simple to code make the semi-implicit schemes described here particularly efficient, and thus potentially still competitive.

In the present work adaptive refinement has been employed only when necessary for the resolution of the free surface's shape, any grading of the mesh being performed for purely geometrical reasons. As the problems mentioned in Chapter 4 with regard to tangential stress errors highlight, the use of more sophisticated error indicators to control free-surface resolution may often be appropriate. Moreover, there is considerable scope for the inclusion of error indicators based on solution gradients, which might be used to control the resolution of both the free surface and the interior of the mesh. Within the framework described here such modifications would be straightforward given Triangle's capabilities.

Finally, the extension of the current methods to three-dimensional problems should be considered, opening up the possibility of the study of genuine physical problems without the need for mathematical idealisation. The experience gained in the course of the research described here suggests that only the simplest of truly three-dimensional problems are at present feasible using even the computational resources of, say, ten workstations, and that such computations would require days

or even weeks of run time. While the development of efficient solvers and preconditioning strategies are necessary prerequisites for such work, the finite element methodology employed in this work should allow the techniques described to be generalised in a straightforward manner to the three-dimensional case.

Appendix A

Some useful identities

The identities (A.4) and (A.5) are convenient when manipulating the integrals in the finite element formulation in order to obtain weak forms. They may be derived using the divergence theorem

$$\int_\Omega \nabla \cdot \mathbf{A} d\Omega = \int_{\partial\Omega} \mathbf{A} \cdot \mathbf{n}\, d\Gamma, \tag{A.1}$$

by substituting respectively

$$\mathbf{A} = (\phi\xi_1, 0) \tag{A.2}$$

and

$$\mathbf{A} = (0, \phi\xi_2) \tag{A.3}$$

into A.1, where ϕ, ξ_1 and ξ_2 are C^1 scalar functions defined on a simply-connected region Ω bounded by the curve $\partial\Omega$ and with outward normal $\mathbf{n}$. Thus one obtains

$$\int_\Omega \phi \frac{\partial \xi_1}{\partial x} d\Omega = -\int_\Omega \xi_1 \frac{\partial \phi}{\partial x} d\Omega + \int_{\partial\Omega} \phi n_x \xi_1\, d\Gamma, \tag{A.4}$$

$$\int_\Omega \phi \frac{\partial \xi_2}{\partial y} d\Omega = -\int_\Omega \xi_2 \frac{\partial \phi}{\partial y} d\Omega + \int_{\partial\Omega} \phi n_y \xi_2\, d\Gamma. \tag{A.5}$$

Appendix B

Operators in axisymmetric form

In this appendix the axisymmetric forms of the terms in the Navier-Stokes equations are derived using standard methods that may be found in many textbooks [101]. In cylindrical coordinates the position of a point $\mathbf{r}$ may be specified in the form

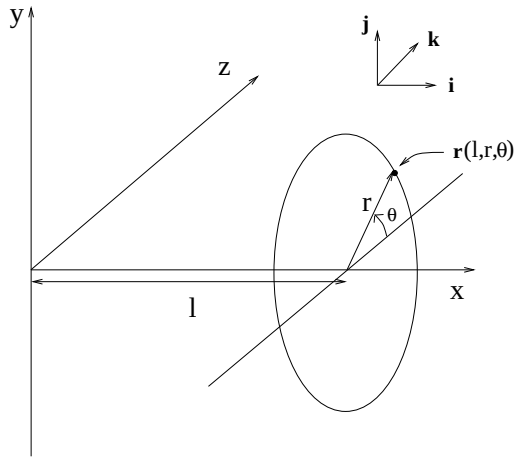


Figure B.1: Geometry of the cylindrical coordinate system

(l, r, θ), as illustrated in Fig. B.1. Alternatively, $\mathbf{r}$ may be written in the form $\mathbf{r} = x\,\mathbf{i} + y\,\mathbf{j} + z\,\mathbf{k}$, where $\mathbf{i}$, $\mathbf{j}$ and $\mathbf{k}$ are unit vectors parallel to, and with the same sense as, the Cartesian coordinate axes. Thus

$$x = l, \tag{B.1}$$

$$y = r sin\,\theta, \tag{B.2}$$

$$z = r cos\,\theta. \tag{B.3}$$

Since

$$\mathbf{r} = l\,\mathbf{i} + r sin\theta\,\mathbf{j} + r cos\theta\,\mathbf{k}, \tag{B.4}$$

the scale factors h_1, h_2 and h_3 are given by

$$h_1 = \left|\frac{\partial \mathbf{r}}{\partial l}\right| = 1, \qquad h_2 = \left|\frac{\partial \mathbf{r}}{\partial r}\right| = 1, \qquad h_3 = \left|\frac{\partial \mathbf{r}}{\partial \theta}\right| = r. \tag{B.5}$$

Furthermore, one may define three mutually perpendicular unit vectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ in the following manner

$$\frac{\partial \mathbf{r}}{\partial l} = h_1\mathbf{e}_1 = \mathbf{i}, \tag{B.6}$$

$$\frac{\partial \mathbf{r}}{\partial r} = h_2\mathbf{e}_2 = sin\theta\,\mathbf{j} + cos\theta\,\mathbf{k}, \tag{B.7}$$

$$\frac{\partial \mathbf{r}}{\partial \theta} = h_3\mathbf{e}_3 = rcos\theta\,\mathbf{j} - rsin\theta\,\mathbf{k}. \tag{B.8}$$

Thus, the velocity at any point in space may be written in the form

$$\mathbf{u} = u_l\mathbf{e}_1 + u_r\mathbf{e}_2 + u_\theta\mathbf{e}_3, \tag{B.9}$$

where u_l, u_r and u_θ are the components parallel to $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ respectively.

Since for an axisymmetric model one need consider only a half-plane with edge lying along the x-axis, here we choose the upper half of the x–y plane. Thus, setting $\theta = 90°$ gives

$$\mathbf{e}_1 = \mathbf{i}, \qquad \mathbf{e}_2 = \mathbf{j}, \qquad \mathbf{e}_3 = -\mathbf{k}, \tag{B.10}$$

and

$$u_l = u, \qquad u_r = v, \qquad u_\theta = -w, \tag{B.11}$$

u, v and w being the Cartesian velocity components. By definition, in any axisymmetric problem, the derivatives of all physical quantities with respect to θ must be zero i.e. $\frac{\partial}{\partial \theta} = 0$. While u_θ need not be zero in an axisymmetric flow, here for simplicity it will be assumed that $u_\theta = 0$. If u_θ is not zero then only the form of the convective terms given here will require modification.

Using standard results given in [101], in cylindrical coordinates the gradient of a scalar function ϕ may be written as

$$\nabla\phi = \frac{\partial \phi}{\partial l}\mathbf{e}_1 + \frac{\partial \phi}{\partial r}\mathbf{e}_2 + \frac{1}{r}\frac{\partial \phi}{\partial \theta}\mathbf{e}_3, \tag{B.12}$$

the divergence of a vector function $\mathbf{A} = A_l\mathbf{e}_1 + A_r\mathbf{e}_2 + A_\theta\mathbf{e}_3$ as

$$\nabla \cdot \mathbf{A} = \frac{1}{r}\left(\frac{\partial (rA_l)}{\partial l} + \frac{\partial (rA_r)}{\partial r} + \frac{\partial A_\theta}{\partial \theta}\right) \tag{B.13}$$

and the curl of $\mathbf{A}$ as

$$\nabla \times \mathbf{A} = \frac{1}{r}\left[\frac{\partial (rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial \theta}\right]\mathbf{e}_1 + \frac{1}{r}\left[\frac{\partial A_l}{\partial \theta} - \frac{\partial (rA_\theta)}{\partial l}\right]\mathbf{e}_2 + \left[\frac{\partial A_r}{\partial l} - \frac{\partial A_l}{\partial r}\right]\mathbf{e}_3. \tag{B.14}$$

B.1 The continuity equation

Applying (B.13) to the velocity field $\mathbf{u} = u_l\mathbf{e}_1 + u_r\mathbf{e}_2 + u_\theta\mathbf{e}_3$ one obtains the following expression for divergence of the velocity field

$$\nabla \cdot \mathbf{u} = \frac{1}{r}\left(\frac{\partial (ru_l)}{\partial l} + \frac{\partial (ru_r)}{\partial r} + \frac{\partial u_\theta}{\partial \theta}\right). \tag{B.15}$$

Since $\frac{\partial}{\partial \theta} = 0$, this simplifies to

$$\nabla \cdot \mathbf{u} = \frac{\partial u_l}{\partial l} + \frac{1}{r}\frac{\partial (ru_r)}{\partial r} = \frac{\partial u_l}{\partial l} + \frac{\partial u_r}{\partial r} + \frac{u_r}{r}, \tag{B.16}$$

or, in the notation employed in Chapter 6,

$$\nabla \cdot \mathbf{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{v}{y}. \tag{B.17}$$

B.2 The pressure gradient

Expression (B.12) may be used to write the pressure gradient in the cylindrical coordinate form

$$\nabla p = \frac{\partial p}{\partial l}\mathbf{e}_1 + \frac{\partial p}{\partial r}\mathbf{e}_2 + \frac{1}{r}\frac{\partial p}{\partial \theta}\mathbf{e}_3. \tag{B.18}$$

Since $\frac{\partial}{\partial \theta} = 0$, this simplifies to

$$\nabla p = \frac{\partial p}{\partial l}\mathbf{e}_1 + \frac{\partial p}{\partial r}\mathbf{e}_2, \tag{B.19}$$

or alternatively

$$\nabla p = \frac{\partial p}{\partial x}\mathbf{i} + \frac{\partial p}{\partial y}\mathbf{j}, \tag{B.20}$$

where the vectors **i** and **j** are employed to indicate the contributions to the x and y momentum equations respectively.

B.3 The convective terms

The convective term in the Navier-Stokes equations is normally written using the notation $(\mathbf{u}.\nabla)\mathbf{u}$. When written out in full, using (B.12), it takes the form

$$\left(u_l\frac{\partial}{\partial l}+u_r\frac{\partial}{\partial r}+\frac{u_\theta}{r}\frac{\partial}{\partial\theta}\right)(u_l\mathbf{e}_1+u_r\mathbf{e}_2+u_\theta\mathbf{e}_3)\,. \tag{B.21}$$

Expanding this, and simplifying using the assumption that $u_\theta = 0$, one obtains

$$\left(u_l\frac{\partial u_l}{\partial l}+u_r\frac{\partial u_l}{\partial r}\right)\mathbf{e}_1+\left(u_l\frac{\partial u_r}{\partial l}+u_r\frac{\partial u_r}{\partial r}\right)\mathbf{e}_2+0\mathbf{e}_3, \tag{B.22}$$

and thus

$$\left(u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}\right)\mathbf{i}+\left(u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}\right)\mathbf{j}. \tag{B.23}$$

B.4 The viscous terms

In a Cartesian coordinate system the viscous term is frequently written using the notation

$$\nabla^2\mathbf{u}, \tag{B.24}$$

the understanding being that this expression may be evaluated by applying the Laplacian operator independently to each rectangular component of the velocity field. In the present work the alternative stress-divergence form of the viscous term is employed, i.e.

$$\nabla(\nabla\cdot\mathbf{u})+\nabla^2\mathbf{u}, \tag{B.25}$$

so as to give physically meaningful free-surface stress boundary conditions. For a continuous divergence-free velocity field (B.24) and (B.25) have identical values. The independent application of the Laplacian to each spatial component of the velocity is, however, legitimate only in a Cartesian coordinate system. When a coordinate system other than a Cartesian one is employed the viscous term (B.24) must be written in the alternative *div-curl form* [34]

$$\nabla(\nabla\cdot\mathbf{u})-\nabla\times(\nabla\times\mathbf{u}), \tag{B.26}$$

which is valid in a general coordinate system. Note that, despite its appearance, (B.26) *does not* include the $\nabla(\nabla \cdot \mathbf{u})$ term required to give stress natural boundary conditions shown in (B.25). Using (B.14), the fact that $\frac{\partial}{\partial \theta} = 0$, and the assumption that $u_\theta = 0$, one may obtain the following expression

$$\nabla \times \mathbf{u} = \left[\frac{\partial u_r}{\partial l} - \frac{\partial u_l}{\partial r} \right] \mathbf{e}_3. \tag{B.27}$$

Applying (B.14) a second time, and simplifying, gives

$$\nabla \times (\nabla \times \mathbf{u}) = \left[\frac{1}{r}\frac{\partial u_r}{\partial l} - \frac{1}{r}\frac{\partial u_l}{\partial r} + \frac{\partial^2 u_r}{\partial r \partial l} - \frac{\partial^2 u_l}{\partial r^2} \right] \mathbf{e}_1 + \left[-\frac{\partial^2 u_r}{\partial l^2} + \frac{\partial^2 u_l}{\partial l \partial r} \right] \mathbf{e}_2 + 0\mathbf{e}_3. \tag{B.28}$$

Considering next the $\nabla(\nabla \cdot \mathbf{u})$ term in (B.26) and employing (B.12) and (B.16) one obtains

$$\nabla(\nabla \cdot \mathbf{u}) = \left[\frac{\partial^2 u_r}{\partial l \partial r} + \frac{\partial^2 u_l}{\partial l^2} + \frac{1}{r}\frac{\partial u_r}{\partial l} \right] \mathbf{e}_1 + \left[\frac{\partial^2 u_r}{\partial r^2} + \frac{1}{r}\frac{\partial u_r}{\partial r} + \frac{\partial^2 u_l}{\partial r \partial l} - \frac{u_r}{r^2} \right] \mathbf{e}_2 + 0\mathbf{e}_3. \tag{B.29}$$

Finally, subtracting (B.28) from (B.29) one obtains

$$\left(\frac{\partial^2 u_l}{\partial r^2} + \frac{\partial^2 u_l}{\partial l^2} + \frac{1}{r}\frac{\partial u_l}{\partial r} \right) \mathbf{e}_1 + \left(\frac{\partial^2 u_r}{\partial l^2} + \frac{\partial^2 u_r}{\partial r^2} + \frac{1}{r}\frac{\partial u_r}{\partial r} - \frac{u_r}{r^2} \right) \mathbf{e}_2 + 0\mathbf{e}_3, \tag{B.30}$$

or, in the notation employed in Chapter 6,

$$\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{1}{y}\frac{\partial u}{\partial y} \right) \mathbf{i} + \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{1}{y}\frac{\partial v}{\partial y} - \frac{v}{y^2} \right) \mathbf{j}. \tag{B.31}$$

The additional $\nabla(\nabla \cdot \mathbf{u})$ term required to give stress natural boundary conditions may be found using (B.29), which in the notation employed in Chapter 6 takes the form

$$\nabla(\nabla \cdot \mathbf{u}) = \left[\frac{\partial^2 u}{\partial x^2} + \frac{1}{y}\frac{\partial v}{\partial x} + \frac{\partial^2 v}{\partial x \partial y} \right] \mathbf{i} + \left[\frac{\partial^2 v}{\partial y^2} + \frac{1}{y}\frac{\partial v}{\partial y} + \frac{\partial^2 u}{\partial y \partial x} - \frac{v}{y^2} \right] \mathbf{j}. \tag{B.32}$$

Bibliography

[1] M. Abramowitz and I.A. Stegun. *Handbook of mathematical functions*. Applied mathematics series. Dover Publications, New York, 1964.

[2] D.J. Acheson. *Elementary Fluid Dynamics*. Clarendon Press, Oxford, 1990.

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

[4] P. Bach and O. Hassager. An algorithm for the use of the Lagrangian specification in newtonian fluid mechanics and applications to free-surface flow. *Journal of Fluid Mechanics*, 152:173–190, 1985.

[5] R.E. Bank and R.K. Smith. The incomplete factorization multigraph algorithm. *SIAM Journal on Scientific Computing*, 20(4):1349–1364, 1999.

[6] R.E. Bank and J. Xu. An algorithm for coarsening unstructured meshes. *Numerische Mathematik*, 73:1–36, 1996.

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

[8] E. Becker, W.J. Hiller, and T.A. Kowalewski. Experimental and theoretical investigation of large-amplitude oscillations of liquid droplets. *Journal of Fluid Mechanics*, 231:189–210, 1991.

[9] T. Brooke Benjamin and A.T. Ellis. Self-propulsion of asymmetrically vibrating bubbles. *Journal of Fluid Mechanics*, 212:65–80, 1990.

[10] M. Benzi, C.D. Meyer, and M. Tuma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM Journal on Scientific Computing*, 17(5):1135–1149, 1996.

[11] M. Berzins, P.J. Capon, and P.K. Jimack. On spatial adaptivity and interpolation when using the method of lines. *Applied Numerical Mathematics*, 26:1–17, 1997.

[12] M. Berzins and R.M. Furzeland. An adaptive theta method for the solution of stiff and nonstiff differential equations. *Applied Numerical Mathematics*, 9:1–19, 1992.

[13] D.L. Book. *Finite difference techniques for vectorised fluid dynamics calculations*. Springer-Verlag, 1981.

[14] A. Brandt. Multigrid Techniques: 1984 Guide, with applications to fluid dynamics. Technical report, Weizmann Institute of Science, Israel, 1984.

[15] W.L. Briggs. *A Multigrid Tutorial*. SIAM, 1987.

[16] R.L. Burden, J.D. Faires, and A.C. Reynolds. *Numerical Analysis*. Prindle, Weber and Schmidt, Boston, U.S.A., second edition, 1981.

[17] P.J. Capon and P.K. Jimack. An inexact Newton method for systems arising from the finite element method. *Applied Mathematics Letters*, 10(3):9–12, 1997.

[18] G.F. Carey and J.T. Oden. *Finite Elements*, volume 6 - Fluid Mechanics. Prentice-Hall, 1986.

[19] J.C. Cavendish. Automatic triangulation of arbitrary planar domains. *International Journal for Numerical Methods in Engineering*, 8:679–696, 1974.

[20] E. Chow and Y. Saad. Experimental study of ILU preconditioners for indefinite matrices. *Journal of Computational and Applied Mathematics*, 86(2):387–414, 1997.

[21] S.D. Conte and C. de Boor. *Elementary numerical analysis: an algorithmic approach*. McGraw-Hill Book Company, Inc., second edition, 1980.

[22] G.R. Cowper. Gaussian quadrature formulas for triangles. *International Journal for Numerical Methods in Engineering*, 7:405–408, 1973.

[23] J. Donea. Arbitrary Lagrangian-Eulerian finite element methods. In T. Belytschko and T.J.R. Hughes, editors, *Computational Methods for Transient Analysis*, chapter 10, pages 474–516. Elsevier Science Publishers, 1983.

[24] M.S. Engelman, R.L. Sani, 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.

[25] D.A. Field. Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods*, 4:709–712, 1988.

[26] G.B. Foote. A numerical method for studying liquid drop behaviour: simple oscillation. *Journal of Computational Physics*, 11:507–530, 1973.

[27] L.A. Freitag and C. Ollivier-Gooch. A cost/benefit analysis of simplical mesh improvement techniques as measured by solution efficiency. Preprint ANL/MCS-P722-0598, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1988. also to appear in International Journal of Computational Geometry.

[28] J. Frenkel. Viscous flow of crystalline bodies under the action of surface tension. *Journal of Physics USSR*, 9:385–391, 1945.

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

[30] A. George and J.W-H. Liu. *Computer solution of large sparse positive definite systems*. Prentice-Hall, 1981.

[31] P.L. George. *Automatic mesh generation: application to finite element methods*. John Wiley, London, 1991.

[32] G.H. Golub and C.F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, second edition, 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] P.M. Gresho and R.L. Sani. *Incompressible Flow and the Finite Element Method: Advection-Diffusion and Isothermal Laminar Flow*, volume 1. John Wiley and Sons Ltd., 1998.

[35] The Numerical Algorithms Group. *The NAG Fortran Library Manual, Mark 15*. The Numerical Algorithms Group Ltd., Oxford, U.K., 1991.

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

[37] E.B. Hansen. Stokes flow down a wall into an infinite pool. *Journal of Fluid Mechanics*, 178:243–256, 1987.

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

[39] F.H. Harlow and J.E. Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *The Physics of Fluids*, 8(12):2182–2189, 1965.

[40] F.B. Hildebrand. *Introduction to Numerical Analysis*. McGraw-Hill Book Company, Inc., 1956.

[41] C.W. Hirt, A.A. Amsden, and J.L. Cook. An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds. *Journal of Computational Physics*, 14(3):227–253, 1974.

[42] C.W. Hirt and B.D. Nichols. Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries. *Journal of Computational Physics*, 39:201–225, 1981.

[43] P. Hood. Frontal solution program for unsymmetric matrices. *International Journal for Numerical Methods in Engineering*, 10:379–399, 1976.

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

[45] R.W. Hopper. Plane Stokes flow driven by capillarity on a free surface. Part 2. Further developments. *Journal of Fluid Mechanics*, 230:355–364, 1991.

[46] R.W. Hopper. Stokes flow of a cylinder and half-space driven by capillarity. *Journal of Fluid Mechanics*, 243:171–181, 1992.

[47] R.W. Hopper. Coalescence of Two Viscous Cylinders by Capillarity: Part 1, Theory. *Journal of the American Ceramic Society*, 76(12):2947–2952, 1993.

[48] R.W. Hopper. Coalescence of Two Viscous Cylinders by Capillarity: Part 2, Shape Evolution. *Journal of the American Ceramic Society*, 76(12):2953–2960, 1993.

[49] A. Huerta and W.K. Liu. Viscous flow with large free surface motion. *Computer Methods in Applied Mechanics and Engineering*, 69:277–324, 1988.

[50] W.F. Hughes and J.A. Brighton. *Theory and Problems of Fluid Dynamics.* Schaum's outline series. McGraw-Hill, Inc., Second edition, 1991.

[51] D.B. Ingham and M.A. Kelmanson. *Lecture Notes in Engineering Vol. 7 Boundary Integral Equation Analyses of Singular Potential, and Biharmonic Problems.* Springer-Verlag, 1984.

[52] A. Jagota and P.R. Dawson. Micromechanical modeling of powder compacts – I. unit problems for sintering and traction induced deformation. *Acta Metallica*, 36(9):2551–2561, 1988.

[53] A. Jagota and P.R. Dawson. Simulation of the Viscous Sintering of Two Particles. *Journal of the American Ceramic Society*, 73(1):173–177, 1990.

[54] 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):990–1103, 1991.

[55] B. Joe. Geompack — A Software Package for the Generation of Meshes using Geometric Algorithms. *Advanced Engineering Software*, 13(5/6):325–331, 1991.

[56] M.A. Kelmanson. Boundary integral equation solution of viscous flows with free surfaces. *Journal of Engineering Mathematics*, 17:329–343, 1983.

[57] M.A. Kelmanson. Theoretical and experimental analyses of the maximum-supportable fluid load on a rotating cylinder. *Journal of Engineering Mathematics*, 29:271–285, 1995.

[58] S.O. Kim and H.C. No. Second-order model for free surface convection and interface reconstruction. *International Journal for Numerical Methods in Fluids*, 26(1):79–100, 1998.

[59] 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.

[60] 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.

[61] R.J. Leveque and Zhilin Li. Immersed interface methods for Stokes flow with elastic boundaries or surface tension. *SIAM Journal on Scientific Computing*, 18(3):709–735, 1997.

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

[63] J. Liou and T.E. Tezduyar. A clustered element-by-element iteration method for finite element computations. In Glowinski et al., editor, *Domain decomposition methods for partial differential equations*, pages 140–150. SIAM, Philadelphia, PA., 1991.

[64] S.H. Lo. A new mesh generation scheme for arbitrary planar domains. *International Journal for Numerical Methods in Engineering*, 21:1403–1426, 1985.

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

[66] D.R. Lynch. Unified Approach to Simulation on Deforming Elements with Application to Phase Change Problems. *Journal of Computational Physics*, 47:387–411, 1982.

[67] 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.

[68] F. Mashayek and N. Ashgriz. A Spine-Flux Method for Simulating Free Surface Flows. *Journal of Computational Physics*, 122:367–379, 1995.

[69] 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, Department of Mathematics and Computer Science, 1995.

[70] H.K. Moffatt. Behaviour of a viscous film on the outer surface of a rotating cylinder. *Journal de Mecanique*, 16(5):651–673, 1977.

[71] B.D. Nichols and C.W. Hirt. Improved Free Surface Boundary Conditions for Numerical Incompressible-Flow Calculations. *Journal of Computational Physics*, 8(3):434–448, 1971.

[72] C. Ollivier-Gooch. GRUMMP Version 0.1.3 User's Manual. Technical report, The University of British Columbia and the University of Chicago, Argonne National Laboratory, 1998.

[73] C.F. Ollivier-Gooch. An Unstructured Mesh Improvement Toolkit with Application to Mesh Improvement, Generation, and (De-)Refinement. Technical Report AIAA 98-0218, 1998. Presented at the AIAA 36th Aerospace Sciences Meeting, Reno, Nevada.

[74] B. O'Neill. *Elementary Differential Geometry.* Academic Press, Inc., 1966.

[75] T.W. Patzek, O.A. Basaran, R.E. Benner, and L.E. Scriven. Nonlinear Oscillations of Two-Dimensional Rotating Inviscid Drops. *Journal of Computational Physics*, 116:3–25, 1995.

[76] R.C. Peterson, P.K. Jimack, and M.A. Kelmanson. Automatic Generation of Finite Element Meshes for Evolving Gas/Liquid Interfaces with Arbitrary Geometry. In *Proceedings of the 6th ICFD Conference on Numerical Methods in Fluids*, Oxford, U.K., May 1998. ICFD, Oxford.

[77] R.C. Peterson, P.K. Jimack, and M.A. Kelmanson. On the stability of viscous, free-surface flow supported by a rotating cylinder. Submitted Proc. Roy. Soc. Series A, September 1999.

[78] R.C. Peterson, P.K. Jimack, and M.A. Kelmanson. The Solution of Two-Dimensional Free-surface Problems Using Automatic Mesh Generation. *International Journal for Numerical Methods in Fluids*, 1999. To appear.

[79] L. Preziosi and D.D. Joseph. The run-off condition for coating and rimming flows. *Journal of Fluid Mechanics*, 187:99–113, 1988.

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

[81] V.V. Pukhnachev. Motion of a liquid film on the surface of a rotating cylinder in a gravitational field. *Journal of Applied Mechanics and Technical Physics*, 18:344–351, 1977.

[82] 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.

[83] J.D. Ramshaw. Conservative Rezoning Algorithm for Generalized Two-Dimensional Meshes. *Journal of Computational Physics*, 59:193–199, 1985.

[84] J.W.S. Rayleigh. On the Capillary Phenomena of Jets. *Proceeding of the Royal Society of London*, 29:71–97, 1879.

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

[86] M.P. Reddy and J.N. Reddy. Multigrid methods to accelerate convergence of element-by-element solution algorithms for viscous incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 132:179–193, 1996.

[87] O. Reglat, R. Labrie, and P.A. Tanguy. A New Free Surface Model for the Dip Coating Process. *Journal of Computational Physics*, 109:238–246, 1993.

[88] N.C. Reis, R.F. Griffiths, and E.P.L. Roberts. Finite Volume Method to Solve Free-Surface Fluid Flow Problems. In *Proceedings of the 6th ICFD Conference on Numerical Methods in Fluids*, Oxford, U.K., May 1998. ICFD, Oxford.

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

[90] J. Ruppert. A Delaunay Refinement Algorithm for Quality 2-Dimensional Mesh Generation. *Journal of Algorithms*, 18(3):548–585, 1995.

[91] 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.

[92] Y. Saad. SPARSKIT: a basic toolkit for sparse matrix computations, version 2. Technical report, Computer Science Department, University of Minnesota, 1994.

[93] Y. Saad. ILUM: A multi-elimination ILU preconditioner for general sparse matrices. *SIAM Journal on Scientific Computation*, 17(4):830–847, 1996.

[94] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS publishing company, Boston, 1996.

[95] V.V. Shaidurov. *Multigrid Methods for Finite Elements*. Kluwer Academic Publishers, Dordrecht, 1995.

[96] J.R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *First workshop on Applied Computational Geometry*, pages 123–133, Philadelphia, Pennsylvania, May 1996. ACM.

[97] D. Silvester and A. Wathen. Fast iterative solution of stabilised Stokes systems. Part 2: Using general block preconditioners. *SIAM Journal on Numerical Analysis*, 31(5):1352–1367, 1994.

[98] G.D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Clarendon Press, Oxford University, U.K., third edition, 1992.

[99] V.A. Solonnikov. Solvability of the problem of the motion of a viscous incompressible liquid bounded by a free surface. In *Dynamics of a continuous medium, No. 23*, pages 182–197. Akad. Nauk. SSSR, Novosibirsk, 1973.

[100] A. Soulaimani and Y. Saad. An arbitrary Lagrangian-Eulerian finite element method for solving three-dimensional free surface flows. *Computer Methods in Applied Mechanics and Engineering*, 162:79–106, 1998.

[101] M.R. Spiegel. *Mathematical Handbook of Formulas and Tables*. Schaum's outline series. McGraw-Hill Company, 1968.

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

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

[104] M. Sussman, P. Smereka, and S. Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114:146–159, 1994.

[105] D. Tabor. *Gases, liquids and solids.* Cambridge University Press, Cambridge, U.K., second edition, 1979.

[106] T.E. Tezduyar, M. Behr, and S. Mittal. A new strategy for finite element computations involving moving boundaries and interfaces — the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. *Computer Methods in Applied Mechanics and Engineering*, 94:353–371, 1992.

[107] T.E. Tezduyar and J. Liou. Grouped element-by-element iteration schemes for incompressible-flow computations. *Computer Physics Communications*, 53(1–3):441–453, 1989.

[108] R.W. Thatcher. Locally mass-conserving Taylor-Hood elements for two- and three-dimensional flow. *International Journal for Numerical Methods in Fluids*, 11:341–353, 1990.

[109] J.F. Thompson, Z.U.A. Warsi, and C.W. Mastin. *Numerical Grid Generation: Foundations and Applications.* North-Holland, 1985.

[110] W. Tsai and D.K.P. Yue. Computation of nonlinear free-surface flows. *Annual Review of Fluid Mechanics*, 28:249–278, 1996.

[111] J.A. Tsampoulos and R.A. Brown. Nonlinear oscillations of inviscid drops and bubbles. *Journal of Fluid Mechanics*, 127:519–537, 1983.

[112] C. Tu and C.S. Peskin. Stability and instability in the computation of flows with moving immersed boundaries: A comparison of 3 methods. *SIAM Journal on Scientific and Statistical Computing*, 13(6):1361–1376, 1992.

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

[114] 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.

[115] G.A.L. van de Vorst, R.M.M. Mattheij, and H.K. Kuiken. A Boundary Element Solution for Two-Dimensional Viscous Sintering. *Journal of Computational Physics*, 100:50–63, 1992.

[116] M. Wang and A.W. Troesch. Numerical stability analysis for free surface flows. *International Journal for Numerical Methods in Fluids*, 24:893–912, 1997.

[117] Q.X. Wang. The Evolution of a Gas Bubble Near an Inclined Wall. *Theoretical and Computational Fluid Dynamics*, 12:29–51, 1998.

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

[119] 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.

[120] P. Wesseling. *An introduction to multigrid methods.* John Wiley and Sons, 1992.

[121] S. Wolfram and G. Beck. *Mathematica: The student book.* Addison-Wesley Publishing Company, Inc., 1994.

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

[123] O.C. Zienkiewicz and K. Morgan. *Finite Elements and Approximation.* John Wiley and Sons, 1983.