Direct Numerical Simulations of Gas–Liquid Multiphase Flows

Grétar Tryggvason,
Ruben Scardovelli and Stéphane Zaleski

CAMBRIDGE

CAMBRIDGE more information - www.cambridge.org/9780521782401

DIRECT NUMERICAL SIMULATIONS OF GAS–LIQUID MULTIPHASE FLOWS

Accurately predicting the behavior of multiphase flows is a problem of immense industrial and scientific interest. Using modern computers, researchers can now study the dynamics in great detail, and computer simulations are yielding unprecedented insight. This book provides a comprehensive introduction to direct numerical simulations of multiphase flows for researchers and graduate students.

After a brief overview of the context and history, the authors review the governing equations. A particular emphasis is placed on the "one-fluid" formulation, where a single set of equations is used to describe the entire flow field and interface terms are included as singularity distributions. Several applications are discussed, such as atomization, droplet impact, breakup and collision, and bubbly flows, showing how direct numerical simulations have helped researchers advance both our understanding and our ability to make predictions. The final chapter gives an overview of recent studies of flows with relatively complex physics, such as mass transfer and chemical reactions, solidification, and boiling, and includes extensive references to current work.

GRÉTAR TRYGGVASON is the Viola D. Hank Professor of Aerospace and Mechanical Engineering at the University of Notre Dame, Indiana.

RUBEN SCARDOVELLI is an Associate Professor in the Dipartimento di Ingegneria Energetica, Nucleare e del Controllo Ambientale (DIENCA) of the Università degli Studi di Bologna.

STÉPHANE ZALESKI is a Professor of Mechanics at the Université Pierre et Marie Curie (UPMC) in Paris and Head of the Jean Le Rond d'Alembert Institute (CNRS UMR 7190).

DIRECT NUMERICAL SIMULATIONS OF GAS–LIQUID MULTIPHASE FLOWS

GRÉTAR TRYGGVASON
University of Notre Dame, Indiana

RUBEN SCARDOVELLI
Università degli Studi di Bologna

STÉPHANE ZALESKI
Université Pierre et Marie Curie, Paris 6

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town,
Singapore, São Paulo, Delhi, Tokyo, Mexico City

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521782401

© Cambridge University Press 2011

This publication is in copyright. Subject to statutory exception
and to the provisions of relevant collective licensing agreements,
no reproduction of any part may take place without the written
permission of Cambridge University Press.

First published 2011

Printed in the United Kingdom at the University Press, Cambridge

A catalogue record for this publication is available from the British Library

ISBN 978-0-521-78240-1 Hardback

Cambridge University Press has no responsibility for the persistence or
accuracy of URLs for external or third-party internet websites referred to
in this publication, and does not guarantee that any content on such
websites is, or will remain, accurate or appropriate.

Contents

Preface *page* ix

1 Introduction 1
- 1.1 Examples of multiphase flows 3
- 1.2 Computational modeling 7
- 1.3 Looking ahead 18

2 Fluid mechanics with interfaces 21
- 2.1 General principles 21
- 2.2 Basic equations 22
- 2.3 Interfaces: description and definitions 30
- 2.4 Fluid mechanics with interfaces 36
- 2.5 Fluid mechanics with interfaces: the one-fluid formulation 41
- 2.6 Nondimensional numbers 42
- 2.7 Thin films, intermolecular forces, and contact lines 44
- 2.8 Notes 47

3 Numerical solutions of the Navier–Stokes equations 50
- 3.1 Time integration 51
- 3.2 Spatial discretization 55
- 3.3 Discretization of the advection terms 59
- 3.4 The viscous terms 61
- 3.5 The pressure equation 64
- 3.6 Velocity boundary conditions 69
- 3.7 Outflow boundary conditions 70
- 3.8 Adaptive mesh refinement 71
- 3.9 Summary 72
- 3.10 Postscript: conservative versus non-conservative form 73

4 Advecting a fluid interface 75
- 4.1 Notations 76

4.2 Advecting the color function 77
4.3 The volume-of-fluid (VOF) method 81
4.4 Front tracking 84
4.5 The level-set method 87
4.6 Phase-field methods 90
4.7 The CIP method 91
4.8 Summary 93

5 The volume-of-fluid method 95
5.1 Basic properties 95
5.2 Interface reconstruction 98
5.3 Tests of reconstruction methods 106
5.4 Interface advection 108
5.5 Tests of reconstruction and advection methods 122
5.6 Hybrid methods 128

6 Advecting marker points: front tracking 133
6.1 The structure of the front 134
6.2 Restructuring the fronts 143
6.3 The front-grid communications 145
6.4 Advection of the front 150
6.5 Constructing the marker function 152
6.6 Changes in the front topology 158
6.7 Notes 160

7 Surface tension 161
7.1 Computing surface tension from marker functions 161
7.2 Computing the surface tension of a tracked front 168
7.3 Testing the surface tension methods 177
7.4 More sophisticated surface tension methods 181
7.5 Conclusion on numerical methods 186

8 Disperse bubbly flows 187
8.1 Introduction 187
8.2 Homogeneous bubbly flows 189
8.3 Bubbly flows in vertical channels 194
8.4 Discussion 201

9 Atomization and breakup 204
9.1 Introduction 204
9.2 Thread, sheet, and rim breakup 205
9.3 High-speed jets 214
9.4 Atomization simulations 219

10 Droplet collision, impact, and splashing 228
10.1 Introduction 228
10.2 Early simulations 229
10.3 Low-velocity impacts and collisions 229
10.4 More complex slow impacts 232
10.5 Corolla, crowns, and splashing impacts 235
11 Extensions 243
11.1 Additional fields and surface physics 243
11.2 Imbedded boundaries 256
11.3 Multiscale issues 266
11.4 Summary 269
Appendix A Interfaces: description and definitions 270
A.1 Two-dimensional geometry 270
A.2 Three-dimensional geometry 272
A.3 Axisymmetric geometry 274
A.4 Differentiation and integration on surfaces 275
Appendix B Distributions concentrated on the interface 279
B.1 A simple example 281
Appendix C Cube-chopping algorithm 284
C.1 Two-dimensional problem 285
C.2 Three-dimensional problem 286
Appendix D The dynamics of liquid sheets: linearized theory 288
D.1 Flow configuration 288
D.2 Inviscid results 288
D.3 Viscous theory for the Kelvin–Helmholtz instability 293
References 295
Index 322

Preface

Progress is usually a sequence of events where advances in one field open up new opportunities in another, which in turn makes it possible to push yet another field forward, and so on. Thus, the development of fast and powerful computers has led to the development of new numerical methods for direct numerical simulations (DNS) of multiphase flows that have produced detailed studies and improved knowledge of multiphase flows. While the origin of DNS of multiphase flows goes back to the beginning of computational fluid dynamics in the early sixties, it is only in the last decade and a half that the field has taken off. We, the authors of this book, have had the privilege of being among the pioneers in the development of these methods and among the first researchers to apply DNS to study relatively complex multiphase flows. We have also had the opportunity to follow the progress of others closely, as participants in numerous meetings, as visitors to many laboratories, and as editors of scientific journals such as the *Journal of Computational Physics* and the *International Journal of Multiphase Flows*. To us, the state of the art can be summarized by two observations:

- Even though there are superficial differences between the various approaches being pursued for DNS of multiphase flows, the similarities and commonalities of the approaches are considerably greater than the differences.
- As methods become more sophisticated and the problems of interest become more complex, the barrier that must be overcome by a new investigator wishing to do DNS of multiphase flows keeps increasing.

This book is an attempt to address both issues.

The development of numerical methods for flows containing a sharp interface, as fluids consisting of two or more immiscible components inherently do, is currently a "hot" topic and significant progress has been made by a number of groups. Indeed, for a while there was hardly an issue of the *Journal of Computational Physics* that did not contain one or more papers describing such methods. In the present book we have elected to focus mostly on two specific classes of methods: volume of fluid (VOF) and front-tracking methods. This choice reflects our own background, as well as the fact that both types of method have been very successful and are responsible for some of the most significant new insights into multiphase flow dynamics that DNS has revealed. Furthermore, as emphasized by the first bullet point, the similarities in the different approaches are sufficiently great that

a reader of the present book would most likely find it relatively easy to switch to other methods capable of capturing the interface, such as level set and phase field.

The goal of DNS of multiphase flows is the understanding of the behavior and properties of such flows. We believe that while the development of numerical methods is important, it is in their applications where the most significant rewards are to be found. Thus, we include in the book several chapters where we describe the use of DNS to understand specific systems and what has been learned up to now. This is inherently a somewhat biased sample (since we elected to talk about studies that we know well – our own!), but we feel that the importance of these chapters goes beyond the specific topics treated. We furthermore firmly believe that the methods that we describe here have now reached sufficient maturity so that they can be used to probe the mysteries of a large number of complex flows. Therefore, the application of existing methods to problems that they are suited for and the development of new numerical methods for more complex flows, such as those described in the final chapter, are among the most exciting immediate directions for DNS of multiphase flows.

Our work has benefitted from the efforts of many colleagues and friends. First and foremost we thank our students, postdoctoral researchers and visitors for the many and significant contributions they have made to the work presented here. GT would like to thank his students, Drs. D. Yu, M. Song, S.O. Unverdi, E. Ervin, M.R. Nobari, C.H.H. Chang, Y.-J. Jan, S. Nas, M. Saeed, A. Esmaeeli, F. Tounsi, D. Juric, N.C. Suresh, J. Han, J. Che, B. Bunner, N. Al-Rawahi, W. Tauber, M. Stock, S. Biswas, and S. Thomas, as well as the following visitors and postdoctoral researchers: S. Homma, J. Wells, A. Fernandez, and J. Lu. RS would like to thank his students, Drs. E. Aulisa, L. Campioni, A. Cervone, and V. Marra, as well as his collaborators S. Manservisi, P. Yecko, and G. Zanetti. SZ would like to thank his students, Drs. B. Lafaurie, F.-X. Keller, J. Li, D. Gueyffier, S. Popinet, A. Leboissetier, L. Duchemin, O. Devauchelle, A. Bagué, and G. Agbaglah, as well as his collaborators, visitors, and postdoctoral researchers, G. Zanetti, A. Nadim, J.-M. Fullana, C. Josserand, P. Yecko, M. and Y. Renardy, E. Lopez-Pages, T. Boeck, P. Ray, D. Fuster, G. Tomar, and J. Hoepffner. SZ would also like to thank his trusted friends and mentors, Y. Pomeau, D.H. Rothman, and E.A. Spiegel, for their invaluable advice. We also thank Ms. Victoria Tsengué Ingoba for reading the complete book twice and pointing out numerous typos and mistakes. Any errors, omissions, and ambiguities are, of course, the fault of the authors alone.

And last, but certainly not least, we would like to thank our families for unerring support, acceptance of long working hours, and tolerance of (what sometimes must have seemed) obscure priorities.

Grétar Tryggvason
Ruben Scardovelli
Stéphane Zaleski

1

Introduction

Gas–liquid multiphase flows play an essential role in the workings of Nature and the enterprises of mankind. Our everyday encounter with liquids is nearly always at a free surface, such as when drinking, washing, rinsing, and cooking. Similarly, such flows are in abundance in industrial applications: heat transfer by boiling is the preferred mode in both conventional and nuclear power plants, and bubble-driven circulation systems are used in metal processing operations such as steel making, ladle metallurgy, and the secondary refining of aluminum and copper. A significant fraction of the energy needs of mankind is met by burning liquid fuel, and a liquid must evaporate before it burns. In almost all cases the liquid is therefore atomized to generate a large number of small droplets and, hence, a large surface area. Indeed, except for drag (including pressure drops in pipes) and mixing of gaseous fuels, we would not be far off to assert that nearly all industrial applications of fluids involve a multiphase flow of one sort or another. Sometimes, one of the phases is a solid, such as in slurries and fluidized beds, but in a large number of applications one phase is a liquid and the other is a gas. Of natural gas–liquid multiphase flows, rain is perhaps the experience that first comes to mind, but bubbles and droplets play a major role in the exchange of heat and mass between the oceans and the atmosphere and in volcanic explosions. Living organisms are essentially large and complex multiphase systems.

Understanding the dynamics of gas–liquid multiphase flows is of critical engineering and scientific importance and the literature is extensive. From a mathematical point of view, multiphase flow problems are notoriously difficult and much of what we know has been obtained by experimentation and scaling analysis. Not only are the equations, governing the fluid flow in both phases, highly nonlinear, but the position of the phase boundary must generally be found as a part of the solution. Exact analytical solutions, therefore, exist only for the simplest problems, such as the steady-state motion of bubbles and droplets in Stokes flow, linear inviscid waves, and small oscillations of bubbles and droplets. Experimental studies of

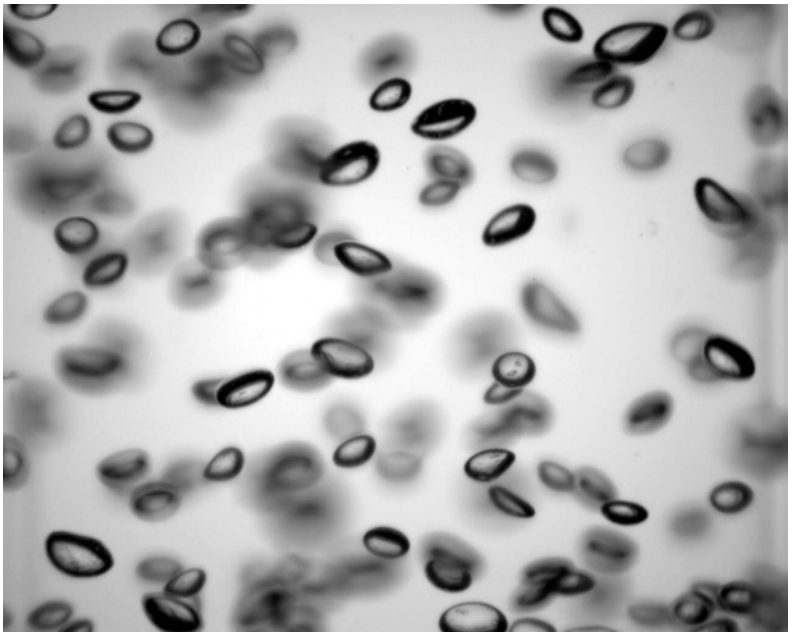


Fig. 1.1. A picture of many buoyant bubbles rising in an otherwise quiescent liquid pool. The average bubble diameter is about 2.2 mm and the void fraction is approximately 0.75%. From Bröder and Sommerfeld (2007). Reproduced with permission.

multiphase flows are not easy either. For many flows of practical interest the length scales are small, the time scales are short, and optical access to much of the flow is limited. The need for numerical solutions of the governing equations has, therefore, been felt by the multiphase research community since the origin of computational fluid dynamics, in the late fifties and early sixties. Although much has been accomplished, simulations of multiphase flows have remained far behind homogeneous flows where direct numerical simulations (DNS) have become a standard tool in turbulence research.

In this book we use DNS to mean simulations of unsteady flow containing a non-trivial range of scales, where the governing equations are solved using sufficiently fine grids so that all continuum time- and length-scales are fully resolved. We believe that this conforms reasonably well with commonly accepted usage, although we recognize that there are exceptions. Some authors feel that DNS refers exclusively to fully resolved simulations of turbulent flows, while others seem to use DNS for any computation of fluid flow that does not include a turbulence model. Our definition falls somewhere in the middle. We also note that some authors, especially in the field of atomization – which is of some importance in this book – refer to unresolved simulations without a turbulence model as LES. We also

Fig. 1.2. A photograph of an atomization experiment performed with coaxial water and air jets reproduced from Villermaux *et al*. (2004). Reproduced with permission. Copyright American Physical Society.

prefer to call such computations DNS, especially as a continuous effort is made in such simulations to check the results as the grid is refined. While it is not surprising that DNS of multiphase flows lags behind homogeneous flows, considering the added difficulty, the situation is certainly not due to lack of effort. However, in the last decade and a half or so, these efforts have started to pay off and rather significant progress has been accomplished on many fronts. It is now possible to do DNS for a large number of fairly complex systems and DNS are starting to yield information that are likely to be unobtainable in any other way. This book is an effort to assess the state of the art, to review how we came to where we are, and to provide the foundation for further progress, involving even more complex multiphase flows.

1.1 Examples of multiphase flows

Since this is a book about numerical simulations, it seems appropriate to start by showing a few "real" systems. The following examples are picked somewhat randomly, but give some insight into the kind of systems that can be examined by direct numerical simulations.

Bubbles are found in a large number of industrial applications. For example, they carry vapor away from hot surfaces in boiling heat transfer, disperse gases and provide stirring in various chemical processing systems, and also affect the propagation of sound in the ocean. To design systems that involve bubbly flows it is necessary to understand how the collective rise velocity of many bubbles depends

Fig. 1.3. The splash generated when a droplet hits a free surface. From A. Davidhazy. Rochester Institute of Technology. Reproduced with permission.

on the void fraction and the bubble size distribution, how bubbles disperse and how they stir up the fluid. Figure 1.1 is a picture of air bubbles rising through water in a small bubble column. The average bubble diameter is about 2.2 mm and the void fraction is approximately 0.75%. At these parameters the bubbles rise with an average velocity of roughly 0.27 m/s, but since the bubbles are not all of the same size they will generally rise with different velocities.

To generate sprays for combustion, coating and painting, irrigation, humidification, and a large number of other applications, a liquid jet must be atomized. Predicting the rate of atomization and the resulting droplet size distribution, as well as droplet velocity, is critical to the successful design of such processes. In Fig. 1.2, a liquid jet is ejected from a nozzle of diameter 8 mm with a velocity of 0.6 m/s. To accelerate its breakup, the jet is injected into a co-flowing air stream, with a velocity of 35 m/s. Initially, the shear between the air and the liquid leads to large axisymmetric waves, but as the waves move downstream the air pulls long filaments from the crest of the wave. The filaments then break into droplets by a capillary instability. See Marmottant and Villermaux (2001) and Villermaux *et al.* (2004) for details.

Droplets impacting solid or liquid surfaces generally splash, often disrupting the surface significantly. Rain droplets falling on the ground often result in soil erosion,

Fig. 1.4. Massive cavitation near the maximum thickness of an airfoil. The flow is from the left to right. In addition to volume change due to phase change, the compressibility of the bubbles is often important (see Section 11.1.5). From Kermeen (1956). Reproduced with permission.

for example. But droplet impact can also help to increase the heat transfer, such as in quenching and spray cooling, and rain often greatly enhances the mixing at the ocean surface. Figure 1.3 shows the splash created when a droplet of a diameter of about 3 mm, released from nearly 0.5 m above the surface, impacts a liquid layer a little over a droplet-diameter deep. The impact of the droplet creates a liquid crater and a rim that often breaks into droplets. As the crater collapses, air bubbles are sometimes trapped in the liquid.

While bubbles are often generated by air injection into a pool of liquid or are formed by entrainment at a free surface, such as when waves break, they also frequently form when a liquid changes phase into vapor. Such a phase change is often nucleated at a solid surface and can take place either by heating the liquid above the saturation temperature, as in boiling, or by lowering the pressure below the vapor pressure, as in cavitation. Figure 1.4 shows massive cavitation near the maximum thickness of an airfoil submerged in water. The chord of the airfoil is 7.6 cm, the flow speed is 13.7 m/s from left to right, and the increase in the liquid velocity as it passes over the leading edge of the airfoil leads to a drop in pressure that is sufficiently large so that the liquid "boils." As the vapor bubbles move into regions of higher pressure at the back of the airfoil, they collapse. However, residual gases, dissolved in the liquid, diffuse into the bubbles during their existence, leaving traces that are visible after the vapor has condensed.

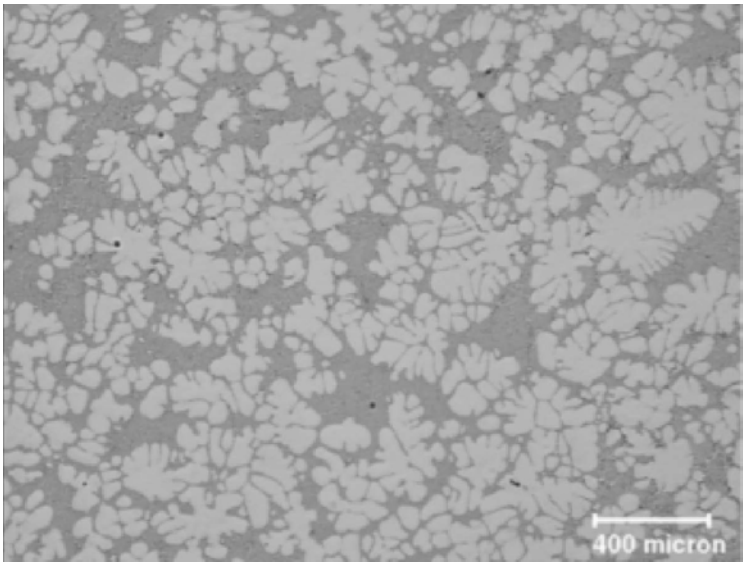


Fig. 1.5. Microstructure of an aluminum–silicon alloy. From D. Apelian, Worcester Polytechnic Institute. Reproduced with permission.

In many multiphase systems one phase is a solid. Suspensions of solid particles in liquids or gases are common and the definition of multiphase flows is sometimes extended to cover flows through or over complex stationary solids, such as packed beds, porous media, forests, and cities. The main difference between gas–liquid multiphase flows and solid–gas or solid–liquid multiphase flows is usually that the interface maintains its shape in the latter cases, even though the location of the solid may change. In some instances, however, that is not the case. Flexible solids can change their shape in response to fluid flow, and during solidification or erosion the boundary can evolve, sometimes into shapes that are just as convoluted as encountered for gas–liquid systems. When a metal alloy solidifies, the solute is initially rejected by the solid phase. This leads to constitutional undercooling and an instability of the solidification front. The solute-rich phase eventually solidifies, but with a very different composition than the material that first became solid. The size, shape, and composition of the resulting microstructures determine the properties of the material, and those are usually sensitively dependent on the various process parameters. A representative micrograph of an Al–Si alloy prepared by metallographic techniques and etching to reveal phase boundaries and interfaces is shown in Fig. 1.5. The light gray phase is almost pure aluminum and solidifies first, but constitutional undercooling leads to dendritic structures of a size on the order of a few tens of micrometers.

Living systems provide an abundance of multiphase flow examples. Suspended blood cells and aerosol in pulmonary flow are obvious examples at the “body”

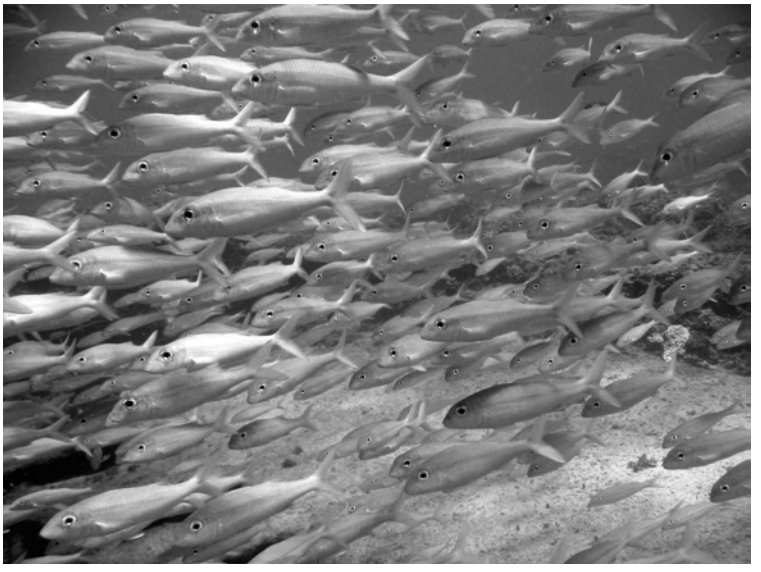


Fig. 1.6. A school of yellow-tailed goatfish (*Mulloidichthys flavolineatus*) near the Northwest Hawaiian Islands. Since self-propelled bodies develop a thrust-producing wake, their collective dynamics is likely to differ significantly from rising or falling bodies. From the NOAA Photo Library.

scale, as are the motion of organs and even complete individuals. But even more complex systems, such as the motion of a flock of birds through air and a school of fish through water, are also multiphase flows. Figure 1.6 show a large number of yellow-tailed goatfish swimming together and coordinating their movement. An understanding of the motion of both a single fish and the collective motion of a large school may have implication for population control and harvesting, as well as the construction of mechanical swimming and flying devices.

1.2 Computational modeling

Computations of multifluid (two different fluids) and multiphase (same fluid, different phases) flows are nearly as old as computations of constant-density flows. As for such flows, a number of different approaches have been tried and a number of simplifications used. In this section we will attempt to give a brief but comprehensive overview of the major efforts to simulate multi-fluid flows. We make no attempt to cite every paper, but hope to mention all major developments.

1.2.1 Simple flows ($Re = 0$ and $Re = \infty$)

In the limit of either very large or very small viscosity (as measured by the Reynolds number, see Section 2.2.6), it is sometimes possible to simplify considerably the

flow description by either ignoring inertia completely (Stokes flow) or by ignoring viscous effects completely (inviscid flow). For inviscid flows it is usually further necessary to assume that the flow is irrotational, except at fluid interfaces. Most success has been achieved for disperse flows of undeformable spheres where, in both these limits, it is possible to reduce the governing equations to a system of coupled ordinary differential equations (ODEs) for the particle positions. For Stokes flow the main developer was Brady and his collaborators (see Brady and Bossis (1988) for a review of early work) who have investigated extensively the properties of suspensions of particles in shear flows, among other problems. For inviscid flows, Sangani and Didwania (1993) and Smereka (1993) simulated the motion of spherical bubbles in a periodic box and observed that the bubbles tended to form horizontal clusters, particularly when the variance of the bubble velocity was small.

For both Stokes flows and inviscid potential flows, problems with deformable boundaries can be simulated with boundary integral techniques. One of the earliest attempts was due to Birkhoff (1954), where the evolution of the interface between a heavy fluid initially on top of a lighter one (the Rayleigh–Taylor instability) was followed by a method tracking the interface between two inviscid and irrotational fluids. Both the method and the problem later became a staple of multiphase flow simulations. A boundary integral method for water waves was presented by Longuet-Higgins and Cokelet (1976) and used to examine breaking waves. This paper had enormous influence and was followed by a large number of very successful extensions and applications, particularly for water waves (e.g. Vinje and Brevig, 1981; Baker *et al.*, 1982; Schultz *et al.*, 1994). Other applications include the evolution of the Rayleigh–Taylor instability (Baker *et al.*, 1980), the growth and collapse of cavitation bubbles (Blake and Gibson, 1981; Robinson *et al.*, 2001), the generation of bubbles and droplets due to the coalescence of bubbles with a free surface (Oguz and Prosperetti, 1990; Boulton-Stone and Blake, 1993), the formation of bubbles and droplets from an orifice (Oguz and Prosperetti, 1993), and the interactions of vortical flows with a free surface (Yu and Tryggvason, 1990), just to name a few. All boundary integral (or boundary element, when the integration is element based) methods for inviscid flows are based on following the evolution of the strength of surface singularities in time by integrating a Bernoulli-type equation. The surface singularities give one velocity component and Green's second theorem yields the other, thus allowing the position of the surface to be advanced in time. Different surface singularities allow for a large number of different methods (some that can only deal with a free surface and others that are suited for two-fluid problems), and different implementations multiply the possibilities even further. For an extensive discussion and recent progress, see Hou *et al.* (2001). Although continuous improvements are being made and new applications continue

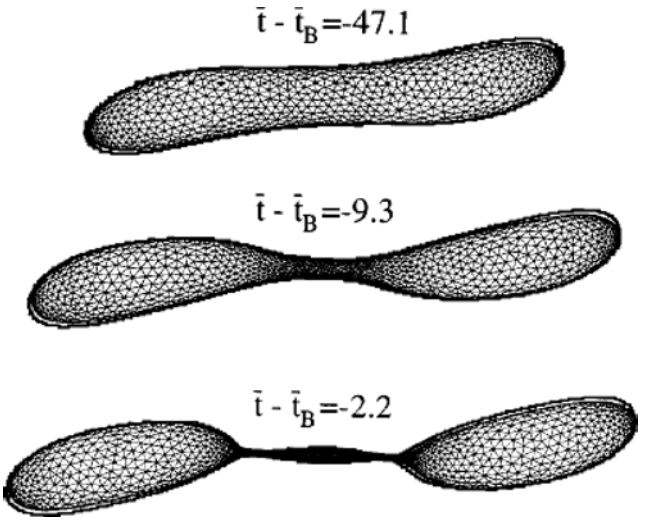


Fig. 1.7. A Stokes flow simulation of the breakup of a droplet in a linear shear flow. The barely visible line behind the numerical results is the outline of a drop traced from an experimental photograph. Reprinted with permission from Cristini *et al.* (1998). Copyright 2005, American Institute of Physics.

to appear, two-dimensional boundary integral techniques for inviscid flows are by now – more than 30 years after the publication of the paper by Longuet-Higgins and Cokelet – a fairly mature technology. Fully three-dimensional computations are, however, still rare. Chahine and Duraiswami (1992) computed the interactions of a few inviscid cavitation bubbles and Xue *et al.* (2001) have simulated a three-dimensional breaking wave. While the potential flow assumption has led to many spectacular successes, particularly for short-time transient flows, its inherent limitations are many. The lack of a small-scale dissipative mechanism makes those models susceptible to singularity formation and the absence of dissipation usually makes them unsuitable for the predictions of the long-time evolution of any system.

The key to the reformulation of inviscid interface problems with irrotational flow in terms of a boundary integral is the linearity of the potential equation. In the opposite limit, where inertia effects can be ignored and the flow is dominated by viscous dissipation, the Navier–Stokes equations become linear (the so-called Stokes flow limit) and it is also possible to recast the governing equations as an integral equation on a moving surface. Boundary integral simulations of unsteady two-fluid Stokes problems originated with Youngren and Acrivos (1976) and Rallison and Acrivos (1978), who simulated the deformation of a bubble and a droplet, respectively, in an extensional flow. Subsequently, several authors have

investigated a number of problems. Pozrikidis and collaborators have examined several aspects of suspensions of droplets, starting with a study by Zhou and Pozrikidis (1993) of the suspension of a few two-dimensional droplets in a channel. Simulations of fully three-dimensional suspensions have been done by Loewenberg and Hinch (1996) and Zinchenko and Davis (2000). The method has been described in detail in the book by Pozrikidis (1992), and Pozrikidis (2001) gives a very complete summary of the various applications. An example of a computation of the breakup of a very viscous droplet in a linear shear flow, using a method that adaptively refines the surface grid as the droplet deforms, is shown in Fig. 1.7.

In addition to inviscid flows and Stokes flows, boundary integral methods have been used by a number of authors to examine two-dimensional, two-fluid flows in Hele–Shaw cells. Although the flow is completely viscous, away from the interface it is a potential flow. The interface can be represented by the singularities used for inviscid flows (de Josselin de Jong, 1960), but the evolution equation for the singularity strength is different. This was used by Tryggvason and Aref (1983, 1985) to examine the Saffman–Taylor instability, where an interface separating two fluids of different viscosity deforms if the less viscous fluid is displacing the more viscous one. They used a fixed grid to solve for the normal velocity component (instead of Green's theorem), but Green's theorem was subsequently used by several authors to develop boundary integral methods for interfaces in Hele–Shaw cells. See, for example, DeGregoria and Schwartz (1985), Meiburg and Homsy (1988), and the review by Hou *et al.* (2001).

Under the heading of simple flows we should also mention simulations of the motion of solid particles, in the limit where the fluid motion can be neglected and the dynamics is governed only by the inertia of the particles. Several authors have followed the motion of a large number of particles that interact only when they collide with each other. Here, it is also sufficient to solve a system of ODEs for the particle motion. Simulations of this kind are usually called "granular dynamics." For an early discussion, see Louge (1994); a more recent one can be found in Pöschel and Schwage (2005), for example. While these methods have been enormously successful in simulating certain types of solid–gas multiphase flows, they are limited to a very small class of problems. One could, however, argue that simulations of the motion of particles interacting through a potential, such as simulations of the gravitational interactions of planets or galaxies and molecular dynamics, also fall into this class. Discussing such methods and their applications would enlarge the scope of the present work enormously, and so we will confine our coverage by simply suggesting that the interested reader consults the appropriate references, such as Schlick (2002) for molecular simulations and Hockney and Eastwood (1981) for astrophysical and other systems.

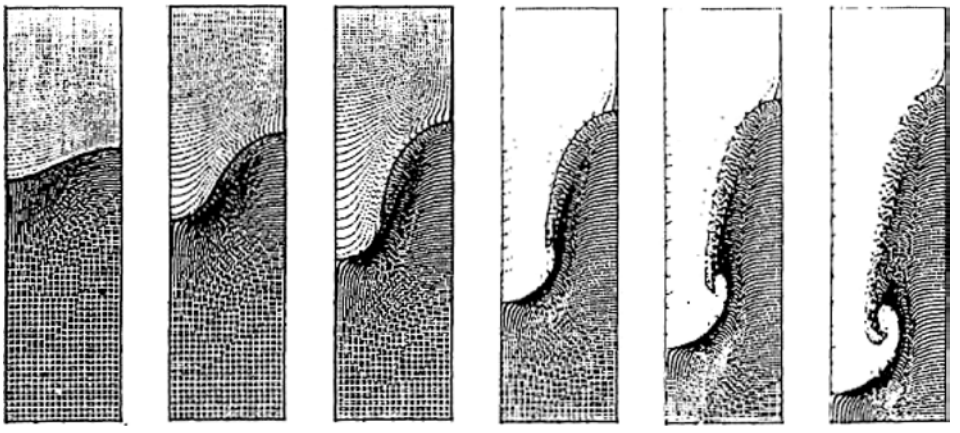


Fig. 1.8. The beginning of computational studies of multiphase flows. The evolution of the nonlinear Rayleigh–Taylor instability, computed using the two-fluid MAC method. Reprinted with permission from Daly (1969b). Copyright 2005, American Institute of Physics.

1.2.2 Finite Reynolds number flows

For intermediate Reynolds numbers it is necessary to solve the full Navier–Stokes equations. Nearly 10 years after Birkhoff's effort to simulate the inviscid Rayleigh–Taylor problem by a boundary integral technique, the marker-and-cell (MAC) method was developed at Los Alamos by Harlow and collaborators. In the MAC method the fluid is identified by marker particles distributed throughout the fluid region and the governing equations solved on a regular grid that covers both the fluid-filled and the empty part of the domain. The method was introduced in Harlow and Welch (1965) and two sample computations of the so-called dam breaking problem were shown in that first paper. Several papers quickly followed: Harlow and Welch (1966) examined the Rayleigh–Taylor problem (Fig. 1.8) and Harlow and Shannon (1967) studied the splash when a droplet hits a liquid surface. As originally implemented, the MAC method assumed a free surface, so there was only one fluid involved. This required boundary conditions to be applied at the surface and the fluid in the rest of the domain to be completely passive. The Los Alamos group realized, however, that the same methodology could be applied to two-fluid problems. Daly (1969b) computed the evolution of the Rayleigh–Taylor instability for finite density ratios and Daly and Pracht (1968) examined the initial motion of density currents. Surface tension was then added by Daly (1969a) and the method again used to examine the Rayleigh–Taylor instability. The MAC method quickly attracted a small group of followers that used it to study several problems: Chan and Street (1970) applied it to free-surface waves, Foote (1973)

and Foote (1975) simulated the oscillations of an axisymmetric droplet and the collision of a droplet with a rigid wall, respectively, and Chapman and Plesset (1972) and Mitchell and Hammitt (1973) followed the collapse of a cavitation bubble. While the Los Alamos group did a number of computations of various problems in the sixties and early seventies and Harlow described the basic idea in a *Scientific American* article (Harlow and Fromm, 1965), the enormous potential of this newfound tool did not, for the most part, capture the fancy of the fluid mechanics research community. Although the MAC method was designed specifically for multifluid problems (hence the M for markers!), it was also the first method to successfully solve the Navier–Stokes equation using the primitive variables (velocity and pressure). The staggered grid used was a novelty, and today it is a common practice to refer to any method using a projection-based time integration on a staggered grid as a MAC method (see Chapter 3).

The next generation of methods for multifluid flows evolved gradually from the MAC method. It was already clear in the Harlow and Welch (1965) paper that the marker particles could cause inaccuracies, and of the many algorithmic ideas explored by the Los Alamos group, the replacement of the particles by a marker function soon became the most popular alternative. Thus, the volume-of-fluid (VOF) method was born. VOF was first discussed in a refereed journal article by Hirt and Nichols (1981), but the method originated earlier (DeBar, 1974; Noh and Woodward, 1976). The basic problem with advecting a marker function is the numerical diffusion resulting from working with a cell-averaged marker function (see Chapter 4). To prevent the marker function from continuing to diffuse, the interface is "reconstructed" in the VOF method in such a way that the marker does not start to flow into a new cell until the current cell is full. The one-dimensional implementation of this idea is essentially trivial, and in the early implementation of VOF the interface in each cell was simply assumed to be a vertical plane for advection in the horizontal direction and a horizontal plane for advection in the vertical direction. This relatively crude reconstruction often led to large amount of "floatsam and jetsam" (small unphysical droplets that break away from the interface) that degraded the accuracy of the computation. To improve the representation, Youngs (1982), Ashgriz and Poo (1991), and others introduced more complex reconstructions of the interface, representing it with a line (two dimensions) or a plane (three dimensions) that could be oriented arbitrarily in such a way as to best fit the interface. This increased the complexity of the method considerably, but resulted in greatly improved advection of the marker function. Even with higher order representation of the fluid interface in each cell, the accurate computation of surface tension remained a major problem. In his simulations of surface tension effects on the Rayleigh–Taylor instability, using the MAC method, Daly (1969b) introduced explicit surface markers for this purpose. However, the premise behind the

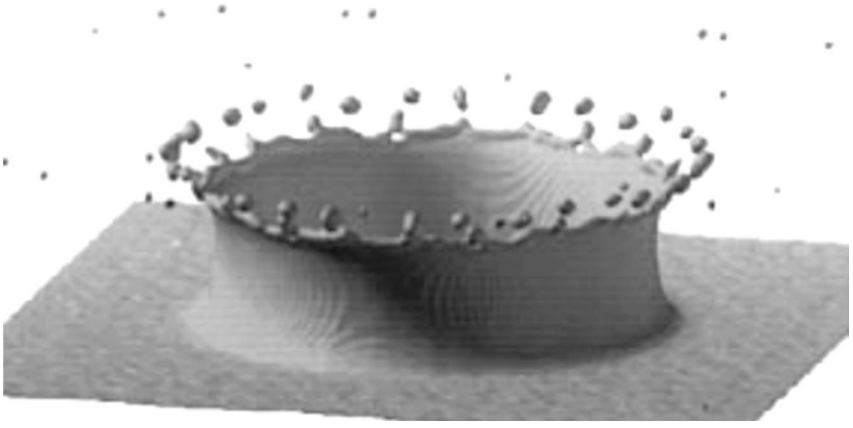


Fig. 1.9. Computation of a splashing drop using an advanced VOF method. Reprinted from Rieber and Frohn (1999) with permission from Elsevier.

development of the VOF method was to get away from using any kind of surface marker so that the surface tension had to be obtained from the marker function instead. This was achieved by Brackbill *et al.* (1992), who showed that the curvature (and hence surface tension) could be computed by taking the discrete divergence of the marker function. A "conservative" version of this "continuum surface force" method was developed by Lafaurie *et al.* (1994). The VOF method has been extended in various ways by a number of authors. In addition to better ways to reconstruct the interface (Rider and Kothe, 1998; Scardovelli and Zaleski, 2000; Aulisa *et al.*, 2007) and compute the surface tension (Renardy and Renardy, 2002; Popinet, 2009), more advanced advection schemes for the momentum equation and better solvers for the pressure equation have been introduced (see Rudman (1997), for example). Other refinements include the use of sub-cells to keep the interface as sharp as possible (Chen *et al.*, 1997a). VOF methods are in widespread use today, and many commercial codes include VOF to track interfaces and free surfaces. Figure 1.9 shows one example of a computation of the splash made when a liquid droplet hits a free surface, done by a modern VOF method. We will discuss the use of VOF extensively in later chapters.

The basic ideas behind the MAC and the VOF methods gave rise to several new approaches in the early nineties. Unverdi and Tryggvason (1992) introduced a front-tracking method for multifluid flows where the interface was marked by connected marker points. The markers are used to advect the material properties (such as density and viscosity) and to compute surface tension, but the rest of the computations are done on a fixed grid as in the VOF method. Although using

connected markers to update the material function was new, marker particles had already been used by Daly (1969a), who used them to evaluate surface tension in simulations with the MAC method, in the immersed-boundary method of Peskin (1977) for one-dimensional elastic fibers in homogeneous viscous fluids and in the vortex-in-cell method of Tryggvason and Aref (1983) for two-fluid interfaces in a Hele–Shaw cell, for example. The front-tracking method of Unverdi and Tryggvason (1992) has been very successful for simulations of finite Reynolds number flows of immiscible fluids and Tryggvason and collaborators have used it to explore a large number of problems.

The early nineties also saw the introduction of the level-set, the CIP, and the phase-field methods to track fluid interfaces on stationary grids. The level-set method was introduced by Osher and Sethian (1988), but its first use to track fluid interfaces appears to be in the work of Sussman *et al.* (1994) and Chang *et al.* (1996), who used it to simulate the rise of bubbles and the fall of droplets in two dimensions. An axisymmetric version was used subsequently by Sussman and Smereka (1997) to examine the behavior of bubbles and droplets. Unlike the VOF method, where a discontinuous marker function is advected with the flow, in the level-set method a continuous level-set function is used. The interface is then identified with the zero contour of the level-set function. To reconstruct the material properties of the flow (density and viscosity, for example) a marker function is constructed from the level-set function. The marker function is given a smooth transition zone from one fluid to the next, thus increasing the regularity of the interface over the VOF method where the interface is confined to only one grid space. However, this mapping from the level-set function to the marker function requires the level-set function to maintain the same shape near the interface and to deal with this problem, Sussman *et al.* (1994) introduced a reinitialization procedure where the level-set function is adjusted in such a way that its value is equal to the shortest distance to the interface at all times. This step was critical in making level-sets work for fluid-dynamics simulations. Surface tension is found in the same way as in the continuous surface force technique introduced for VOF methods by Brackbill *et al.* (1992). The early implementation of the level-set method did not conserve mass very well, and a number of improvements and extensions followed its original introduction. Sussman *et al.* (1998) and Sussman and Fatemi (1999) introduced ways to improved mass conservation, Sussman *et al.* (1999) coupled level-set tracking with adaptive grid refinement and a hybrid VOF/level-set method was developed by Sussman and Puckett (2000), for example.

The constrained interpolated propagation (CIP) method introduced by Takewaki *et al.* (1985) has been particularly popular with Japanese authors, who have applied it to a wide variety of multiphase problems. In the CIP method, the transition from one fluid to another is described by a cubic polynomial. Both the marker function

and its derivative are then updated to advect the interface. In addition to simulating two-fluid problems, the method has been used for a number of more complex applications, such as those involving floating solids; see Yabe *et al.* (2001).

In the phase-field method the governing equations are modified in such a way that the dynamics of the smoothed region between the different fluids is described in a thermodynamically consistent way. In actual implementations the thickness of the transition is, however, much larger than it is in real systems and the net effect of the modification is to provide an "antidiffusive" term that keeps the interface reasonably sharp. While superficially there are considerable similarities between phase-field and level-set methods, the fundamental ideas behind the methods are very different. In the level-set method the smoothness of the phase boundary is completely artificial and introduced for numerical reasons only. In phase-field methods, on the other hand, the transition zone is real, although it is made much thicker than it should be for numerical reasons. It is not clear, at the time of this writing, whether keeping the correct thermodynamic conditions in an artificially thick interface has any advantages over methods that start with a completely sharp interface. The key drawback seems to be that since the propagation and properties of the interface depend sensitively on the dynamics in the transition zone, it must be well resolved. For the motion of two immiscible fluids, that are well described by assuming a sharp interface, this adds a resolution requirement that is more stringent than for other "one-fluid" methods. The phase-field approach was originally introduced to model solidification (see Kobayashi (1992, 1993)) and has found widespread use in such simulations. With the exception of the modeling of solidification in the presence of flows (Beckermann *et al.*, 1999; Tonhardt and Amberg, 1998), its use for fluid dynamic simulations is relatively limited (Jacqmin, 1999; Jamet *et al.*, 2001). The main appeal of the phase-field methods appears to be for problems where small-scale physics must be accounted for and it is difficult to do so in the sharp interface limit.

In the "one-fluid" methods described above, where a single set of governing equations is used to describe the fluid motion in both fluids, the fluid motion is mostly computed on regular structured grids and the main difference between the various methods is how a marker function is advected (and how surface tension is found). The thickness of the interface varies from one cell in VOF methods to a few cells in level-set and front-tracking methods, but once the marker function has been found, the specific scheme for the interface advection is essentially irrelevant for the rest of the computations. While these methods have been enormously successful, their accuracy is generally somewhat limited. There have, therefore, recently been several attempts to generate methods that retain most of the advantages of these methods but treat the interface as "fully sharp." The origin of these attempts can be traced to the work of Glimm and collaborators (Glimm *et al.*, 1981; Glimm

and McBryan, 1985; Chern *et al.*, 1986), who used grids that were modified locally near an interface in such a way that the interface coincided with a cell boundary, and more recent "cut-cell" methods for the inclusion of complex bodies in simulations of inviscid flows (Quirk, 1994; Powell, 1998). In their modern incarnation, sharp interface methods include the ghost fluid method, the immersed-interface method and the method of Udaykumar *et al.* (2001). In the "ghost fluid" method introduced by Fedkiw *et al.* (1999) the interface is marked by advecting a level-set function, but to find numerical approximations to derivatives near the interface, where the finite difference stencil involves values from the other side of the interface, fictitious values are assigned to those grid points. The values are obtained by extrapolation, and a few different possibilities for doing so are discussed by Glimm *et al.* (2001), for example. The "immersed-interface" method of Lee and LeVeque (2003), on the other hand, is based on modifying the numerical approximations near the interface by explicitly incorporating the jump across the interface into the finite difference equations. While this is easily done for relatively simple jump conditions, it becomes more involved for complex situations. Lee and LeVeque (2003) thus found it necessary to limit their development to fluids with the same viscosity. In the method of Udaykumar *et al.* (2001), complex solid boundaries are represented on a regular grid by merging cells near the interface and using polynomial fitting to find field values at the interface. This method, which is related to the "cut-cell" methods used for inviscid compressible flows (Powell, 1998), has so far only been implemented for solids and fluids, including solidification (Yang and Udaykumar, 2005), but there seems to be no reason why the method cannot be used for multifluid problems. For an extension to three dimensions, see Marella *et al.* (2005).

While the original "one-fluid" methods require essentially no modification of the flow solver near the interface (except allowing for variable density and viscosity), the sharp interface methods all require localized modifications of the basic scheme. This results in considerably more complex numerical schemes, but is also likely to improve the accuracy. That may be important for extreme values of the governing parameters, such as large differences between the material properties of the different fluids and low viscosities. The sharp interface approach may also be required for flows with very complex interface physics. However, methods based on a straightforward implementation of the "one-fluid" formulation of the governing equations, coupled with advanced schemes to advect the interface (or marker function), have already demonstrated their usefulness for a large range of problems, and it is likely that their simplicity will ensure that they will continue to be widely used.

In addition to the development of more accurate implementations of the "one-fluid" approach, many investigators have pursued extension of the basic schemes to problems that are more complex than the flow of two immiscible liquids. More

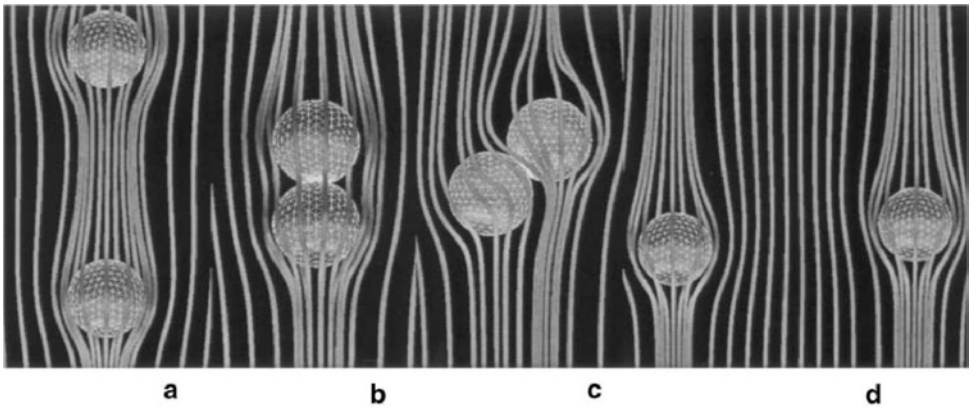


Fig. 1.10. The interaction of two falling spheres. The spheres are shown at four different times, going from left to right. Reprinted from Hu *et al.* (2001), with permission from Elsevier.

complex physics has been incorporated to simulate contaminated interfaces, mass transfer and chemical reactions, electrorheological effects, boiling, solidification, and the interaction of solid bodies with a free surface or a fluid interface. We will briefly review such advanced applications at the very end of the book, in Chapter 11.

While methods based on the "one-fluid" approach were being developed, other techniques were also explored. Hirt *et al.* (1970) describe one of the earliest use of structured, boundary-fitted Lagrangian grids. In this approach a logically rectangular structured grid is used, but the grid points move with the fluid velocity, thus deforming the grid. This approach is particularly well suited when the interface topology is relatively simple and no unexpected interface configurations develop. In a related approach, a grid line is aligned with the fluid interface, but the grid away from the interface is generated using standard grid-generation techniques, such as conformal mapping, or other more advanced elliptic grid-generation schemes. The method was used by Ryskin and Leal (1984) to compute the steady rise of buoyant, deformable, axisymmetric bubbles. They assumed that the fluid inside the bubble could be neglected, but Dandy and Leal (1989) and Kang and Leal (1987) extended the method to two-fluid problems and unsteady flows. Several authors have used this approach to examine relatively simple problems, such as the steady-state motion of single particles or moderate deformation of free surfaces. Fully three-dimensional simulations are relatively rare (see, though, Takagi *et al.* (1997)), and it is probably fair to say that it is unlikely that this approach will be the method of choice for very complex problems such as the three-dimensional unsteady motion of several particles.

A much more general approach to continuously represent a fluid interface by a grid line is to use fully unstructured grids. This allows grid points to be inserted and deleted as needed and distorted grid cells to be reshaped. While the grid was moved with the fluid velocity in some of the early applications of this method, the more modern approach is either to move only the interface points or to move the interior nodes with a velocity different from the fluid velocity, in such a way that the grid distortion is reduced but adequate resolution is still maintained. A large number of methods have been developed that fall into this general category, but we will only reference a few examples. Oran and Boris (1987) simulated the breakup of a two-dimensional drop; Shopov *et al.* (1990) examined the initial deformation of a buoyant bubble; Feng *et al.* (1994, 1995) and Hu (1996) computed the unsteady two-dimensional motion of several particles and Fukai *et al.* (1995) followed the collision of a single axisymmetric droplet with a wall. Although this appears to be a fairly complex approach, Johnson and Tezduyar (1997) and Hu *et al.* (2001) have produced very impressive results for the three-dimensional unsteady motion of many spherical particles. Figure 1.10 shows an example of a simulation done using the arbitrary-Lagrangian–Eulerian method of Hu *et al.* (2001). Here, two solid spheres are initially falling in-line (left frame). Since the trailing sphere is sheltered from the flow by the leading one, it catches up and "kisses" the leading one. The in-line configuration is unstable and the spheres "tumble" (two middle frames). After tumbling, the spheres drift apart (right frame).

The most recent addition to the collection of methods to simulate finite Reynolds number multiphase flows is the lattice–Boltzmann method (LBM). It is now clear that LBM can be used to obtain results of accuracy comparable to more conventional methods. It is still not clear, however, whether the LBM is significantly faster or simpler than other methods (as sometimes claimed), but most likely these methods are here to stay. For a discussion see, for example, Shan and Chen (1993) and Sankaranarayanan *et al.* (2002). A comparison of results obtained by the LBM method and the front-tracking method of Unverdi and Tryggvason (1992) can be found in Sankaranarayanan *et al.* (2003). We will not discuss LBM in this book, but refer the reader to Rothman and Zaleski (1997) and Chapter 6 in Prosperetti and Tryggvason (2007).

1.3 Looking ahead

Direct numerical simulations of multiphase flows have come a long way in the last decade and a half or so. It is now possible to simulate accurately the evolution of disperse flows of several hundred bubbles, droplets, and particles for sufficiently long times so that reliable values can be obtained for various statistical quantities. Similarly, major progress has been achieved in the development of methods

for more complex flows, including those where a liquid solidifies or evaporates. Simulations of large systems undergoing boiling and solidification are therefore within reach.

Much remains to be done, however, and it is probably fair to say that the use of direct numerical simulations of multiphase flows for research and design is still in the embryonic state. The possibility of computing the evolution of complex multiphase flows – such as churn-turbulent bubbly flow undergoing boiling, or the breakup of a jet into evaporating droplets – will transform our understanding of flows of enormous economic significance. Currently, control of most multiphase flow processes is fairly rudimentary and almost exclusively based on intuition and empirical observations. Industries that deal primarily with multiphase flows are, however, multibillion dollar operations, and the savings realized if atomizers for spray generation, bubble injectors in bubble columns, and inserts into pipes to break up droplets, just to name a few examples, could be improved by just a little bit would add up to a substantial amount of money. Reliable predictions would also reduce the design cost significantly for situations such as space vehicles and habitats where experimental investigations are expensive. And, as the possibilities of manipulating flows at the very smallest scales by either stationary or free flowing microelectromechanical devices become more realistic, the need to predict the effect of such manipulations becomes critical.

While speculating about the long-term impact of any new technology is a dangerous thing – and we will simply state that the impact of direct numerical simulations of multiphase flows will without doubt be significant – it is easier to predict the near future. Apart from the obvious prediction that computers will continue to become faster and more available, we expect that the development of numerical methods will focus mainly on flows with complex physics. Although some progress has already been achieved for flows with variable surface tension, flows coupled to temperature and electric fields, and flows with phase change, simulations of such systems are still far from being commonplace. In addition to the need to solve a large number of equations, coupled systems generally possess much larger ranges of length and time scales than simple two-fluid systems. Thus, the incorporation of implicit time-integrators for stiff systems and adaptive gridding will become even more important. It is also likely, as more and more complex problems are dealt with, that the differences between direct numerical simulations – where everything is resolved fully – and simulations where the smallest scales are modeled will become blurred. Simulations of atomization where the evolution of thin films are computed by "subgrid" models and very small droplets are included as point particles are relatively obvious examples of such simulations (for a discussion of the point-particle approximation, see Chapter 9 in Prosperetti and Tryggvason (2007), for example). Other examples include possible couplings of continuum

approaches such as those described in this book with microscopic simulations of moving contact lines, kinetics effects at a solidifying interface, and reactions in thin flames. Simulations of non-Newtonian fluids, where the microstructure has to be modeled in such a way that the molecular structure is accounted for in some way, also fall under this category.

In addition to the development of more powerful numerical methods, it is increasingly critical to deal with the "human" aspect of large-scale numerical simulations. The physical problems that we must deal with and the computational tools that are available are rapidly becoming very complex. The difficulty of developing fully parallelized software to solve the continuum equations (fluid flow, mass and heat transfer, etc.), where three-dimensional interfaces must be handled and the grids must be dynamically adapted, is putting such simulations beyond the reach of a single graduate student. In the future these simulations may even be beyond the capacity of small research groups. It is becoming very difficult for a graduate student to learn everything that they need to know and make significant new progress in 4 to 5 years. Lowering the "knowledge barrier" and ensuring that new investigators can enter the field of direct numerical simulations of multiphase flow may well become as important as improving the efficiency and accuracy of the numerical methods. The present book is an attempt to ease the entry of new researchers into this field.

2

Fluid mechanics with interfaces

The equations governing multiphase flows, where a sharp interface separates immiscible fluids or phases, are presented in this chapter. We first derive the equations for flows without interfaces, in a relatively standard manner. Then we discuss the mathematical representation of a moving interface and the appropriate jump conditions needed to couple the equations across the interfaces. Finally, we introduce the so-called "one-fluid" approach, where the interface is introduced as a singular distribution in equations written for the whole flow field. The "one-fluid" form of the equations plays a fundamental rôle for the numerical methods discussed in the rest of the book.

2.1 General principles

The derivation of the governing equations is based on three general principles: the continuum hypothesis, the hypothesis of sharp interfaces, and the neglect of intermolecular forces. The assumption that fluids can be treated as a continuum is usually an excellent approximation. Real fluids are, of course, made of atoms or molecules. To understand the continuum hypothesis, consider the density or amount of mass per unit volume. If this amount were measured in a box of sufficiently small dimensions ℓ, it would be a wildly fluctuating quantity (see Batchelor (1970), for a detailed discussion). However, as the box side ℓ increases, the density becomes ever smoother, until it is well approximated by a smooth function ρ. For liquids in ambient conditions this happens for ℓ above a few tens of nanometers ($1\text{ nm} = 10^{-9}$ m). In some cases, such as in dilute gases, the discrete nature of matter may be felt over much larger length scales. For dilute gases, the average distance between molecular collisions, or the *mean free path* ℓ_{mfp}, is the important length scale. The gas obeys the Navier–Stokes equations for scales $\ell \gg \ell_{\text{mfp}}$. Molecular simulations, where the motion of many individual molecules is followed for sufficiently long times so that meaningful averages can be computed, show that

the fluid behaves as a continuum for a surprisingly small number of molecules. Koplik *et al.* (1988) found, for example, that under realistic pressure and temperature a few hundred molecules in a channel resulted in a Poiseuille flow that agreed with the predictions of continuum theory.

Beyond the continuum hypothesis, for multiphase flows we shall make the *assumption of sharp interfaces*. Interfaces separate different fluids, such as air and water, oil and vinegar, or any other pair of *immiscible* fluids and different thermodynamic phases, such as solid and liquid or vapor and liquid. The properties of the fluids, including their equation of state, density, viscosity and heat conductivity, generally change across the interface. The transition from one phase to another occurs on very small scales, as described above. For continuum scales we may safely assume that interfaces have vanishing thickness.

We also impose certain restrictions on the type of forces that are taken into account. Long-range forces between fluid particles, such as electromagnetic forces in charged fluids, shall not be considered. Intermolecular forces, such as van der Waals forces that play an important rôle in interface physics, are modelled by retaining their most important effect: capillarity. This effect, also called surface tension, amounts to a stress concentrated at the sharp interfaces.

The three assumptions above also reflect the fact that it would be nearly impossible, with the current state of the art, to describe complex droplet and bubble interactions while keeping the microscopic physics. For instance, simulating physical phenomena from the nanometer to the centimeter scale would require 10^7 grid points in every direction, an extravagant requirement for any type of computation, even with the use of cleverly employed adaptive mesh refinement, at least at present.

Beyond the three assumptions above, we mostly deal with incompressible flows in this book, although in the present chapter we derive the equations initially for general flow situations.

2.2 Basic equations

Expressing the basic principles of conservation of mass, momentum, and energy mathematically leads to the governing equations for fluid flow. In addition to the general *conservation* principles, we also need *constitutive* assumptions about the specific nature of each fluid. Here we will work only with Newtonian fluids.

2.2.1 Mass conservation

The principle of conservation of mass states that mass cannot be created or destroyed. Therefore, if we consider a volume V, fixed in space, then the mass inside this volume can only change if mass flows in or out through its boundary S. The

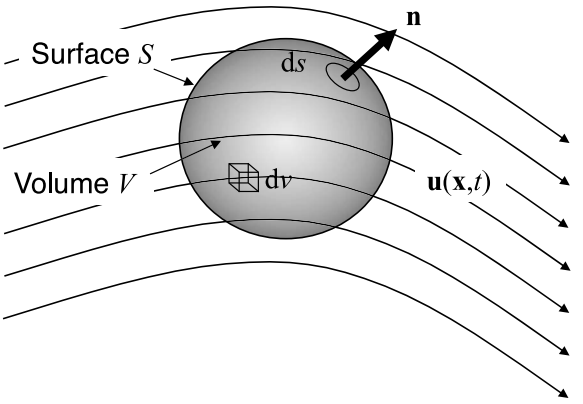


Fig. 2.1. A stationary control volume V. The surface is denoted by S.

flow out of V, through a surface element ds, is $\rho\mathbf{u}\cdot\mathbf{n}$ ds where $\mathbf{n}$ is the outward normal, ρ is the density, and $\mathbf{u}$ is the velocity. The notation is shown in Fig. 2.1. Stated in integral form, the principle of mass conservation is

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho \,\mathrm{d}v = -\oint_S \rho\mathbf{u}\cdot\mathbf{n}\,\mathrm{d}s. \tag{2.1}$$

Here, the left-hand side is the rate of change of mass in the volume V and the right-hand side represents the net flow through its boundary S. Since the volume is fixed in space, we can take the derivative inside the integral and, by applying the divergence theorem to the integral of the fluxes through the boundary, we have

$$\int_V \left[\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u})\right]\mathrm{d}v = 0. \tag{2.2}$$

This relation must hold for any arbitrary volume, no matter how small, and that can only be true if the quantity inside the square brackets is zero. The partial differential equation expressing conservation of mass is therefore

$$\frac{\partial\rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0. \tag{2.3}$$

By using the definition of the substantial derivative

$$\frac{D()}{Dt} = \frac{\partial()}{\partial t} + \mathbf{u}\cdot\nabla(), \tag{2.4}$$

and expanding the divergence, $\nabla\cdot(\rho\mathbf{u}) = \mathbf{u}\cdot\nabla\rho + \rho\nabla\cdot\mathbf{u}$, the continuity equation can be rewritten in *convective form* as

$$\frac{D\rho}{Dt} = -\rho\nabla\cdot\mathbf{u}, \tag{2.5}$$

emphasizing that the density of a material particle can only change if the fluid is compressed or expanded ($\nabla \cdot \mathbf{u} \neq 0$).

2.2.2 Momentum conservation

The equation of motion is derived by using the momentum-conservation principle, stating that the rate of change of fluid momentum in the fixed volume V is the difference in momentum flux across the boundary S plus the net forces acting on the volume. Therefore,

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\mathbf{u}\,\mathrm{d}v = -\oint_S \rho\mathbf{u}(\mathbf{u}\cdot\mathbf{n})\,\mathrm{d}s + \int_V \mathbf{f}\,\mathrm{d}v + \oint_S \mathbf{n}\cdot\mathbf{T}\,\mathrm{d}s. \tag{2.6}$$

The first term on the right-hand side is the momentum flux through the boundary of V and the next term is the total body force on V. Frequently, the force per unit volume $\mathbf{f}$ is only the gravitational force, $\mathbf{f} = \rho\mathbf{g}$. The last term is the total surface force. Here, the tensor $\mathbf{T}$ is a symmetric stress tensor constructed in such a way that $\mathbf{n}\cdot\mathbf{T}\,\mathrm{d}s$ is the force on a surface element $\mathrm{d}s$ with a normal $\mathbf{n}$.

By the same argument as applied to the mass conservation equation, Equation (2.6) must be valid at every point in the fluid, so that

$$\frac{\partial\rho\mathbf{u}}{\partial t} = -\nabla\cdot(\rho\mathbf{u}\mathbf{u}) + \mathbf{f} + \nabla\cdot\mathbf{T}. \tag{2.7}$$

Here we denote with $\mathbf{ab}$, whose ijth component is $a_i b_j$, the dyadic product of the two vectors $\mathbf{a}$ and $\mathbf{b}$; hence, $\mathbf{uu}$ has components $u_i u_j$. The nonlinear advection term can be written as

$$\nabla\cdot(\rho\mathbf{u}\mathbf{u}) = \rho\mathbf{u}\cdot\nabla\mathbf{u} + \mathbf{u}\nabla\cdot(\rho\mathbf{u}), \tag{2.8}$$

and using the definition of the substantial derivative and the continuity equation we can rewrite Equation (2.7) as

$$\rho\frac{D\mathbf{u}}{Dt} = \mathbf{f} + \nabla\cdot\mathbf{T}. \tag{2.9}$$

This is Cauchy's equation of motion and is valid for any continuous medium. For fluids like water, oil, and air (as well as many others that are generally referred to as Newtonian fluids) the stress may be assumed to be a linear function of the rate of strain

$$\mathbf{T} = (-p + \lambda\nabla\cdot\mathbf{u})\mathbf{I} + 2\mu\mathbf{S}. \tag{2.10}$$

Here, $\mathbf{I}$ is the unit tensor, p the pressure, μ the viscosity, and $\mathbf{S} = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}})$ is the rate-of-strain or deformation tensor whose components are

$$S_{ij} = \frac{1}{2}\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\Big). \tag{2.11}$$

λ is the second coefficient of viscosity, and if Stokes' hypothesis is assumed to hold, then $\lambda = -(2/3)\mu$.[1] Substituting the expression for the stress tensor into Cauchy's equation of motion results in

$$\rho \frac{D\mathbf{u}}{Dt} = \mathbf{f} - \nabla p + \nabla(\lambda \nabla \cdot \mathbf{u}) + \nabla \cdot (2\mu \mathbf{S}), \tag{2.12}$$

which is the Navier–Stokes equation for fluid flow.

2.2.3 Energy conservation

In integral form the conservation of energy principle, as applied to a control volume V fixed in space, is

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t} \int_V \rho \left(e + \frac{1}{2} u^2 \right) \, \mathrm{d}v \\ &\quad = - \oint_S \rho \left(e + \frac{1}{2} u^2 \right) \mathbf{u} \cdot \mathbf{n} \, \mathrm{d}s + \int_V \mathbf{u} \cdot \mathbf{f} \, \mathrm{d}v + \oint_S \mathbf{n} \cdot (\mathbf{u} \cdot \mathbf{T}) \, \mathrm{d}s - \oint_S \mathbf{q} \cdot \mathbf{n} \, \mathrm{d}s. \end{aligned} \tag{2.13}$$

Here, $u^2 = \mathbf{u} \cdot \mathbf{u}$ and e is the total internal energy per unit mass. The left-hand side is the rate of change of the internal and kinetic energy, the first term on the right-hand side is the flow of internal and kinetic energy across the boundary, the second term represents the work done by body forces, the third term is the work done by the stresses at the boundary (pressure and viscous shear), and $\mathbf{q}$ in the fourth term is the heat-flux vector. This equation can be simplified by using the momentum equation. Taking the dot product of the velocity with Equation (2.9) gives

$$\rho \frac{\partial u^2/2}{\partial t} = -\rho \mathbf{u} \cdot \nabla u^2/2 + \mathbf{u} \cdot \mathbf{f} + \mathbf{u} \cdot (\nabla \cdot \mathbf{T}) \tag{2.14}$$

for the *mechanical* energy. After using this equation to cancel terms in (2.13) and applying the same arguments as before, we obtain the convective form of the energy equation:

$$\rho \frac{De}{Dt} - \mathbf{T} : \nabla \mathbf{u} + \nabla \cdot \mathbf{q} = 0. \tag{2.15}$$

Here we denote with $\mathbf{A} : \mathbf{B} = \sum_i \sum_j A_{ij} B_{ji}$ the scalar product of the two tensors $\mathbf{A}$ and $\mathbf{B}$.

This equation needs to be supplemented by constitutive equations for the specific fluids we are considering. We will assume that the flux of heat is proportional to the gradient of the temperature T, so that

$$\mathbf{q} = -k \nabla T. \tag{2.16}$$

[1] In several texts the discussion is based on the bulk viscosity $\kappa = 2/3\mu + \lambda$. Stokes' hypothesis is that the bulk viscosity vanishes.

This is called Fourier's law and k is the thermal conductivity. Using Fourier's law, assuming a Newtonian fluid, so that the stress tensor is given by Equation (2.10), and that radiative heat transfer is negligible, then the energy equation takes the form

$$\rho \frac{De}{Dt} + p\nabla \cdot \mathbf{u} = \Phi + \nabla \cdot k\nabla T, \tag{2.17}$$

where $\Phi = \lambda(\nabla \cdot \mathbf{u})^2 + 2\mu \mathbf{S} : \mathbf{S}$ is called the dissipation function. It can be shown that Φ, which represents the rate at which work is converted into heat, is always greater or equal to zero.

In general we also need an equation of state giving, say, pressure as a function of density and internal energy

$$p = p(e, \rho), \tag{2.18}$$

as well as equations for the transport coefficients μ, λ, and k as functions of the state of the fluid.

The governing equations are summarized in Panels 2.1 to 2.3. The integral form obtained by applying the conservation principles directly to a small control volume is shown in Panel 2.1. While usually not very convenient for analytical work, the integral form is the starting point for *finite-volume* numerical methods. In Panel 2.2 we show the differential form obtained directly by assuming that the integral laws hold at a point. This form can be used for *finite difference* numerical methods and usually leads to discretizations that are essentially identical to those obtained by the finite-volume method. In Panel 2.3 we have regrouped the terms to obtain the *convective* or the *non-conservative* form of the equations. This is the form usually shown in textbooks and is often used as a starting point for finite difference methods.

2.2.4 Incompressible flow

For an important class of flows the density of each fluid particle does not change as it moves. This is generally the case when the maximum or characteristic flow velocity U is much smaller than the velocity of sound c_s, or equivalently when the Mach number $\mathrm{Ma} = U/c_s$ is much smaller than unity. The equation for the evolution of the density is then

$$\frac{D\rho}{Dt} = \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho = 0, \tag{2.19}$$

and the mass conservation equation, Equation (2.5), becomes

$$\nabla \cdot \mathbf{u} = 0. \tag{2.20}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho \,\mathrm{d}v + \oint_S \rho \mathbf{u}\cdot\mathbf{n}\,\mathrm{d}s = 0,$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\,\mathbf{u}\,\mathrm{d}v = \int_V \mathbf{f}\,\mathrm{d}v + \oint_S \Big(\mathbf{n}\cdot\mathbf{T} - \rho\mathbf{u}(\mathbf{u}\cdot\mathbf{n})\Big)\mathrm{d}s,$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\left(e+\frac{1}{2}u^2\right)\mathrm{d}v = \int_V \mathbf{u}\cdot\mathbf{f}\,\mathrm{d}v + \oint_S \mathbf{n}\cdot\left(\mathbf{u}\cdot\mathbf{T} - \rho\left(e+\frac{1}{2}u^2\right)\mathbf{u} - \mathbf{q}\right)\mathrm{d}s.$$

Panel 2.1. The equations of fluid motion in integral form. The flux terms have been moved under the same surface integral as the stress terms.

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\mathbf{u}) = 0,$$

$$\frac{\partial \rho\mathbf{u}}{\partial t} = \mathbf{f} + \nabla\cdot(\mathbf{T} - \rho\mathbf{u}\mathbf{u}),$$

$$\frac{\partial}{\partial t}\rho\left(e+\frac{1}{2}u^2\right) = -\nabla\cdot\left(\rho\left(e+\frac{1}{2}u^2\right)\mathbf{u} - \mathbf{T}\cdot\mathbf{u} + \mathbf{q}\right) + \mathbf{u}\cdot\mathbf{f}.$$

Panel 2.2. The equations of fluid motion in conservative form.

$$\frac{D\rho}{Dt} + \rho\nabla\cdot\mathbf{u} = 0,$$

$$\rho\frac{D\mathbf{u}}{Dt} = \mathbf{f} + \nabla\cdot\mathbf{T},$$

$$\rho\frac{De}{Dt} = \mathbf{T}:\nabla\mathbf{u} - \nabla\cdot\mathbf{q}.$$

Panel 2.3. The equations of fluid motion in convective or non-conservative form.

Equation (2.20) states that the volume of any fluid element cannot be changed and these flows are therefore referred to as incompressible flows. If we integrate this equation over a finite volume V with boundary S and use the divergence theorem (or expand the divergence in (2.2) and use Equation (2.3)), we find that the integral

form of the mass conservation equation for incompressible flows is

$$\oint_S \mathbf{u}\cdot\mathbf{n}\,\mathrm{d}s = 0, \tag{2.21}$$

stating that inflow balances outflow.

Notice that there is no requirement that the density is the same everywhere, for incompressible flows. The density of a material particle can vary from one particle to the next one, but the density of each particle must stay constant. When the density is not the same everywhere its value at any given point in space can change with time as material particles of different density are advected with the flow. In this case the density field must be updated using Equation (2.19). If the density is constant everywhere, then this is, of course, not necessary.

The pressure plays a special rôle for incompressible flows. Instead of being a thermodynamic function of (say) density and temperature, it is determined solely by the velocity field and will take on whatever value is necessary to make the flow divergence free. It is sometimes convenient – and we will use this extensively – to think of the pressure as projecting the velocity field into the space of incompressible functions. That is, we imagine the velocity first being predicted by (2.12) without the pressure, then we find the pressure necessary to enforce incompressibility and correct the velocity field. Notice that for incompressible flow it is not necessary to solve the energy equation to find the velocity and the pressure, unless the material properties are functions of the temperature. Thus, the flow field is found by solving the momentum equation, Equation (2.12), along with the incompressibility condition, Equation (2.20) or (2.21). The governing equations for incompressible, Newtonian flows in convective form are summarized in Panel 2.4. The total derivative in the momentum equation in Panel 2.4 can be written in several different ways, all of which are equivalent analytically but that generally lead to slightly different numerical approximations. The most common ones are

$$\frac{D\mathbf{u}}{Dt} = \frac{\partial\mathbf{u}}{\partial t} + \nabla\cdot(\mathbf{u}\mathbf{u}) = \frac{\partial\mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = \frac{\partial\mathbf{u}}{\partial t} + \frac{1}{2}\nabla(\mathbf{u}\cdot\mathbf{u}) - \mathbf{u}\times(\nabla\times\mathbf{u}). \tag{2.22}$$

We will usually work with the second form, but we note that the third one is very commonly used as well. For compressible flows the fully conservative form of the momentum equations is usually used, but as discussed at the end of Chapter 3, doing so for incompressible flow with a density that changes abruptly can lead to certain numerical difficulties.

The special case of incompressible fluids with constant density and viscosity is of considerable importance. By writing the deformation tensor in component form it is easily shown that $\nabla\cdot(\nabla\mathbf{u}+\nabla\mathbf{u}^{\mathrm{T}}) = \nabla^2\mathbf{u}$ and the momentum equation becomes

$$\frac{\partial\mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} = -\frac{\nabla p}{\rho} + \frac{\mathbf{f}}{\rho} + \nu\nabla^2\mathbf{u}, \tag{2.23}$$

$$\nabla \cdot \mathbf{u} = 0,$$

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \mathbf{f} + \nabla \cdot \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^{\mathrm{T}}).$$

Panel 2.4. The equations of fluid motion for incompressible Newtonian flow in convective form.

where $\nu = \mu/\rho$ is the *kinematic viscosity*.

2.2.5 Boundary conditions

One of the major difficulties in numerical simulations of fluid flows is the correct implementation of the boundary conditions. In principle the conditions at boundaries are well defined. For viscous, incompressible fluids we require the fluid to stick to the wall so the fluid velocity there is equal to the wall velocity:

$$\mathbf{u} = \mathbf{U}_{\mathrm{wall}}. \tag{2.24}$$

This equation includes both the normal and the tangential components of the velocity. For inviscid flows, where viscous stresses are absent and the fluid can slip freely at the wall, only the normal velocity is equal to that of the wall. In some cases, for instance for flow in very small channels, it is useful to introduce a slip boundary condition for the tangential velocity component u_{t}:

$$u_{\mathrm{t}} - U_{\mathrm{wall}} = \beta \frac{\partial u_t}{\partial n}. \tag{2.25}$$

Here, β is a slip coefficient, with the dimensions of a (small) length, and $\partial/\partial n$ is the derivative in the direction normal to the wall. The slip length is the distance at which the velocity would vanish if extrapolated inside the wall (Fig. 2.2).

Frequently, we are interested in simulating a fluid domain with in- and out-flow boundaries, or we are interested in simulating only a part of a larger fluid-filled domain. In those cases it is necessary to specify in- and out-flow boundary conditions. For inflow boundaries the velocity is usually given. Realistic outflow boundaries, on the other hand, pose a challenge that unfortunately does not have a simple solution. Similar difficulties are faced by the experimentalist who must carefully design their wind tunnel to provide uniform inlet velocity and outlet conditions that have minimal influence on the upstream flow. While it is probably easier to implement uniform inlet conditions computationally than in experiments, the outlet flow is more problematic, since the solution often available to the experimentalist, simply making the wind tunnel long enough, usually requires an extensive number of grid

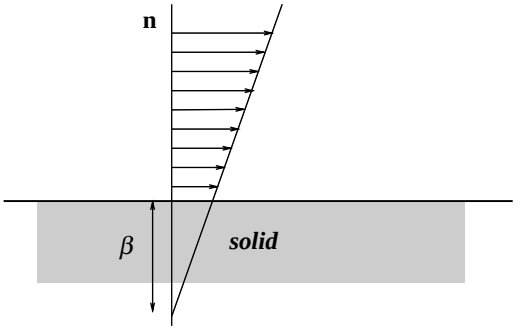


Fig. 2.2. In some models the fluid is allowed to slip on the solid surface. The velocity vanishes at some distance β inside the solid, called the slip length. This model is useful when dealing with triple lines (contact lines) involving two fluids and a solid.

points. The goal is generally to truncate the domain using boundary conditions that make it "appear" longer. Conditions on the rate of change of the velocity, such as imposing a zero gradient on the velocity field, usually can be used if the out-flow boundary is sufficiently far from the region of interest. Once the boundary conditions for the velocity have been specified, conditions for the pressure can be derived at both in- and out-flow boundaries.

In many cases computations are done in domains that are periodic in one or more coordinate directions. If we assume that the x direction is periodic, then the boundary conditions for the velocity are found from $\mathbf{u}(x,y,z) = \mathbf{u}(x+L,y,z)$, where L is the length of the period. Boundary conditions for pressure and other variables are found in the same way, and similar formulae apply when the domain is periodic in y and/or z. For theoretical work, periodic boundaries are often very attractive, since it is not necessary to deal with the often difficult task of specifying appropriate in- and out-flow boundary conditions. The only new consideration is that if the domain is periodic in the direction of gravity, then it is necessary to add a body force equal to $\rho_{\text{av}}\mathbf{g}$, where ρ_{av} is the average density of the whole domain, to prevent a uniform acceleration of the fluid. This force gives rise to a hydrostatic pressure gradient, but since it results in the conservation of momentum in the domain – not the average velocity – it does not correspond exactly to a container with a rigid bottom.

2.3 Interfaces: description and definitions

Following the motion of a deformable interface separating different fluids or phases is at the center of accurate predictions of multiphase flows. Describing the interface location and how it moves can be accomplished in several ways. The simplest case

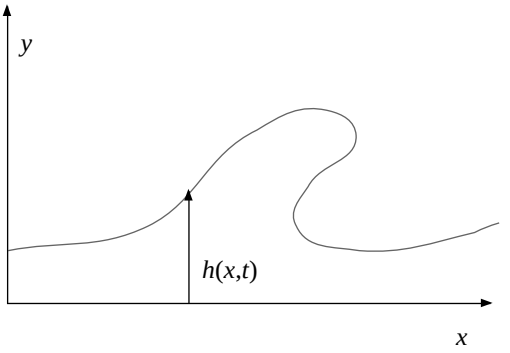


Fig. 2.3. A simple way to parameterize an interface is through the equation $y = h(x)$. However, overhangs in the curve may make the height function h multivalued. Other definitions then become necessary.

is when the location of the interface can be described by a single-valued function of one (in two dimensions) or two (in three dimensions) coordinates. For a thin liquid layer, we can, for example, write

$$y = h(x) \quad \text{(two dimensions)}; \qquad z = h(x,y) \quad \text{(three dimensions)}. \tag{2.26}$$

This situation is sketched in Fig. 2.3, which also shows the limitation of this approach: if part of the interface "overhangs," h becomes a multivalued function. While there are many situations, particularly for thin films, where Equation (2.26) is the most convenient description, we generally need a more sophisticated way to describe the interface.

To handle interfaces of arbitrary shape we can *parameterize* the interface by introducing a coordinate u in two dimensions, such that the location of the interface is given by

$$\mathbf{x}(u) = \Big(x(u), y(u)\Big). \tag{2.27}$$

Figure 2.4 shows a closed contour separating fluid "1" from fluid "2," described in this way. For a curve in two dimensions, described by Equation (2.27), $\mathrm{d}\mathbf{x}/\mathrm{d}u$ yields a vector tangent to the curve. Although u can be any parameterization of the curve, the simplest case is when u is taken to be the arc length s, where $\mathrm{d}s^2 = \mathrm{d}x^2 + \mathrm{d}y^2$. The tangent vector $\mathbf{t} = \mathrm{d}\mathbf{x}/\mathrm{d}s$ is then the unit tangent and $|\mathrm{d}\mathbf{x}/\mathrm{d}s| = 1$. The unit normal $\mathbf{n}$ is perpendicular to the curve, such that $\mathbf{t}\cdot\mathbf{n} = 0$. The orientation of the normal is arbitrary, and one should take the most convenient one depending on the problem. For a closed curve it is customary to let the normal point outwards.

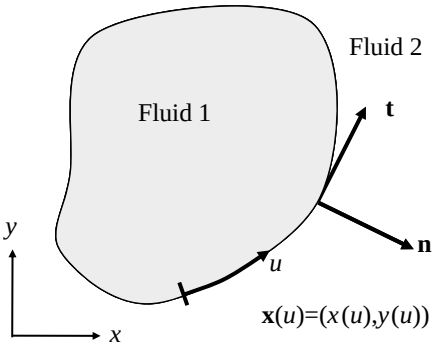


Fig. 2.4. A parameterization of the interface. The coordinate *u* follows the interface.

However, later on we will use the convention that the normal points from one of the fluids to the other. For instance, for an air–water mixture, water may be taken as the reference fluid, air being the other fluid. The normal then points from water to air. We usually label the reference fluid as fluid 1 and the other as fluid 2, and the normal points from 1 to 2.

Consider the case when the normal points outwards and the parameterization follows the curve in the trigonometric (anticlockwise) direction as in Fig. 2.4. Then, if the tangent vector is given by $\mathbf{t} = (t_1, t_2)$, the normal is given by $\mathbf{n} = (t_2, -t_1)$. The length of the unit tangent is by definition constant; so, as we move along the curve, the change of the unit tangent vector must be in the normal direction. The magnitude of the change of the tangent vector, with respect to the arc length s, is the curvature of the interface. We therefore have

$$\frac{\mathrm{d}\mathbf{t}}{\mathrm{d}s} = \kappa\mathbf{n}. \tag{2.28}$$

Similarly, it can be shown that

$$\frac{\mathrm{d}\mathbf{n}}{\mathrm{d}s} = -\kappa\mathbf{t}. \tag{2.29}$$

The two previous equations are known as an instance of the Frenet–Serret formulae. The sign of the curvature may be a vexing question. It depends on one's choice of normal orientation, as shown on Fig. 2.5. With our previous conventions, for a closed circle of radius R, the curvature is negative and equal to $-1/R$.

By taking the dot product of these equations with the normal and the tangent, respectively, the curvature is found to be

$$\kappa = \mathbf{n} \cdot \frac{\mathrm{d}\mathbf{t}}{\mathrm{d}s} = -\mathbf{t} \cdot \frac{\mathrm{d}\mathbf{n}}{\mathrm{d}s}. \tag{2.30}$$

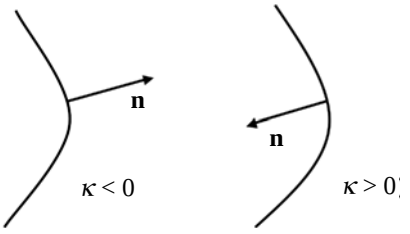


Fig. 2.5. The sign of the curvature depends on one's choice for the orientation of the normal. If the curve folds towards the normal the curvature is positive.

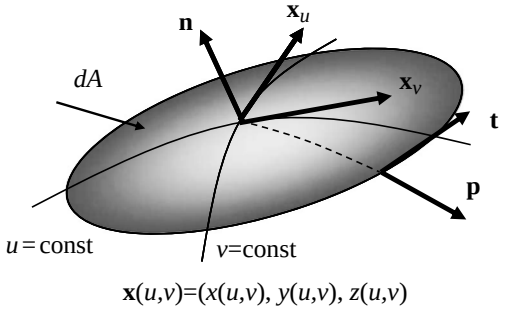


Fig. 2.6. A small surface element. The surface is parameterized by u and v. $\mathbf{p}$ is a vector perpendicular to the edge of the element but tangent to the surface.

In Appendix A we examine the properties of a two-dimensional curve in more detail and work out the above expressions in component form.

For an interface in three-dimensional space we need two independent coordinates, u and v, and the interface is described by

$$\mathbf{x}(u,v) = \Big(x(u,v), y(u,v), z(u,v)\Big), \tag{2.31}$$

as shown in Fig. 2.6. The geometry of three-dimensional surfaces is an elaborate subject and we will only need a few specific results. To avoid cluttering the discussion unnecessarily we have gathered some of the more pertinent elements of the theory in Appendix A and here we simply quote the results that we need.

In addition to knowing the location of the interface, we frequently need to compute the normal to the interface, tangent vectors, and the mean curvature. Differentiating $\mathbf{x}(u,v)$ with respect to the surface coordinates yields two vectors tangent to the surface. We will use the shorthand

$$\mathbf{x}_u = \frac{\partial \mathbf{x}}{\partial u} \qquad \text{and} \qquad \mathbf{x}_v = \frac{\partial \mathbf{x}}{\partial v}. \tag{2.32}$$

With $\mathbf{x}_u$ and $\mathbf{x}_v$ both tangent to the interface, we can find the unit normal by taking

the cross product of the tangent vectors and dividing by the length of the product:

$$\mathbf{n} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{|\mathbf{x}_u \times \mathbf{x}_v|}. \tag{2.33}$$

In Appendix A it is shown that the mean curvature can be found by taking the surface divergence of the normal vector:

$$\kappa = -\nabla_S \cdot \mathbf{n}. \tag{2.34}$$

Here, ∇_S denotes the surface divergence, defined in Appendix A. However, as also shown in the appendix, if we define a normal field that extends off the interface, then the curvature can be found by taking the usual divergence of this extended normal field:

$$\kappa = -\nabla \cdot \mathbf{n}. \tag{2.35}$$

We will also use the fact that the curvature can be found as the integral, along the edges of an infinitesimal surface element of area δA, of the interface tangent vector $\mathbf{p}$ perpendicular to the edge of the surface element (see Fig. 2.6 and a derivation in Appendix A):

$$\kappa\mathbf{n} = \lim_{\delta A \to 0} \frac{1}{\delta A} \oint \mathbf{p} \, \mathrm{d}l. \tag{2.36}$$

Here, the vector $\mathbf{p} = \mathbf{t} \times \mathbf{n}$ is tangent to the surface and perpendicular to both the normal vector $\mathbf{n}$ and the vector $\mathbf{t}$ tangent to the edge of the surface element. Thus, $\mathbf{p}$ "pulls" on the boundary of the surface element.

Instead of identifying the interface by explicitly specifying the location of every point on the interface, the interface can also be given by a marker function defined in the whole domain. Such marker functions can take many forms. We can, for example, use the characteristic function, defined as a discontinuous function by

$$H(\mathbf{x}) = \begin{cases} 1 & \text{if inside a closed interface;} \\ 0 & \text{if outside a closed interface.} \end{cases} \tag{2.37}$$

Alternatively, we may decide that $H = 1$ for fluid "1" and $H = 0$ for fluid "2." With H given, the interface is identified with the sharp change from one value to the other (Fig. 2.7). H can be constructed in different ways, but for our purpose it is convenient to express it in terms of an integral over the product of one-dimensional δ-functions. For a two-dimensional domain:

$$H(x,y) = \int_V \delta(x-x')\delta(y-y') \, \mathrm{d}v'. \tag{2.38}$$

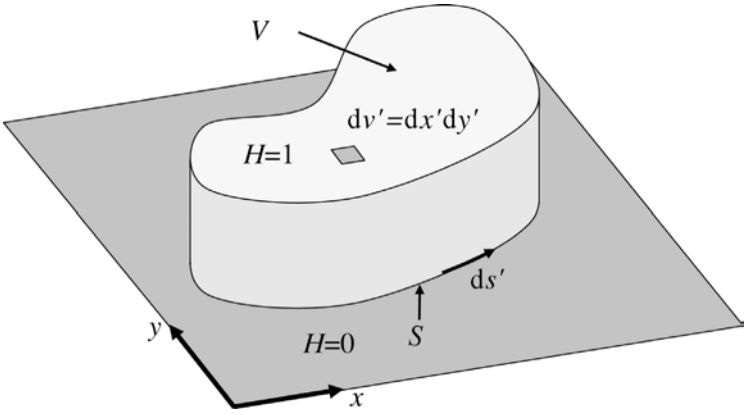


Fig. 2.7. A step function identifying the region occupied by a specific fluid.

Here, the integration is over the region bounded by the contour S and $\mathrm{d}v' = \mathrm{d}x'\,\mathrm{d}y'$ (Fig. 2.7). Obviously, $H = 1$ if the point (x,y) is inside the contour, where it will coincide with (x',y') during the integration, and zero if it is outside and will never be equal to (x',y') .

The properties of the step function are discussed in more detail in Appendix B, where we show that the gradient of H is given by

$$\nabla H(x,y) = -\int_S \delta(x-x')\delta(y-y')\mathbf{n}'\,\mathrm{d}s' = -\delta(n)\mathbf{n}, \tag{2.39}$$

where n is the coordinate normal to the interface in a local coordinate system aligned with the interface. We can also define a surface distribution $\delta_S(\mathbf{x}) = \delta(n)$, and write

$$\nabla H = -\delta_S \mathbf{n}. \tag{2.40}$$

In addition, the interface can be described by a smooth function F, if the interface is identified with a particular value of the function, say $F = 0$ (Fig. 2.8). Obviously, $F < 0$ on one side of the interface and $F > 0$ on the other. One usually takes the reference fluid (fluid "1") to be in the $F > 0$ region. Taking $F > 0$ inside the closed surface, the normal points outwards and is found by

$$\mathbf{n} = -\frac{\nabla F}{|\nabla F|} \tag{2.41}$$

and the curvature is given by

$$\kappa = -\nabla \cdot \mathbf{n} = \nabla \cdot \left(\frac{\nabla F}{|\nabla F|} \right). \tag{2.42}$$

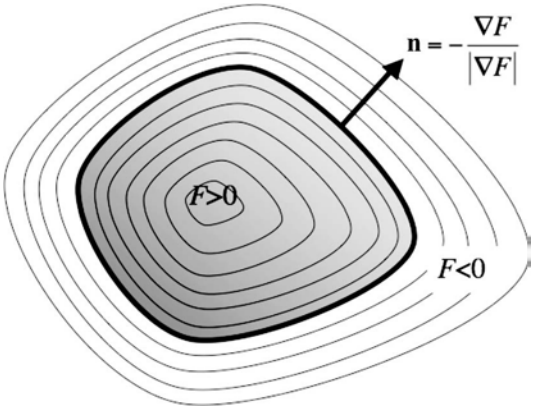


Fig. 2.8. The identification of an interface by a level-set function.

The representation of the interface as a contour with a specific value is used in *level-set* methods.

The motion of the interface S is determined by the normal velocity $V(u,v,t)$, for each point on S. This velocity may be that of the fluid itself, or may be different due to, for example, evaporation or condensation. In parametric form:

$$V(u,v,t) = \mathbf{n} \cdot \frac{\partial}{\partial t}\mathbf{x}(u,v,t). \tag{2.43}$$

To update the location of S, we note that there is no need to specify a tangential velocity, although from a physical point of view the interface is made of fluid particles that travel at some tangential as well as normal velocity. The fact that the motion of the interface is determined only by the normal velocity at the interface can be seen by considering the interface as a level-set function F. Since the function moves with the fluid velocity, its motion is described by

$$\frac{DF}{Dt} = \frac{\partial F}{\partial t} + \mathbf{u} \cdot \nabla F = 0. \tag{2.44}$$

The normal to the interface is given by (2.41), so we can rewrite the above equation as

$$\frac{\partial F}{\partial t} - \mathbf{u} \cdot \mathbf{n}|\nabla F| = \frac{\partial F}{\partial t} - V|\nabla F| = 0, \tag{2.45}$$

which involves only the normal velocity $V = \mathbf{u} \cdot \mathbf{n}$.

2.4 Fluid mechanics with interfaces

Consider the incompressible flow of two immiscible fluids filling a given domain. The domain may be decomposed into any number of subdomains filled with the

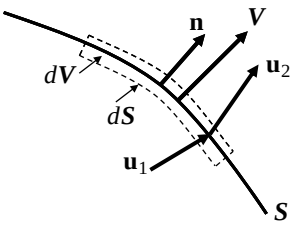


Fig. 2.9. A thin control volume δV with boundary δS including a portion of the interface S. The thickness of the control volume is taken to be zero, so no accumulation takes place.

individual phases. In any subdomain the usual Navier–Stokes equations hold, and here we derive the interface conditions that allow us to couple the fluid motion in the different fluids, across the interface. These conditions are, for the most part, derived using mass and momentum conservation. In one case (for surface tension) it is necessary to introduce additional physics.

2.4.1 Mass conservation and velocity conditions

To derive the jump conditions for the normal velocity, we apply the conservation principle used in Section 2.2.1 to the control volume shown in Fig. 2.9. Since its thickness is taken to be zero, there can be no accumulation of mass inside it, so the mass flux into the control volume must be equal to the flow out. If the velocity on one side of the interface is $\mathbf{u}_1$ and on the other it is $\mathbf{u}_2$ and the normal velocity of the interface is V, then the inflow is $\mathbf{u}_1 \cdot \mathbf{n} - V$ and the outflow is $\mathbf{u}_2 \cdot \mathbf{n} - V$. Denoting the mass flow across the interface by $\dot{m}$, we have

$$\rho_1(\mathbf{u}_1 \cdot \mathbf{n} - V) = \rho_2(\mathbf{u}_2 \cdot \mathbf{n} - V) = \dot{m}\,. \tag{2.46}$$

This is simply the Rankine–Hugoniot condition. If there is no change of phase, $\dot{m} = 0$. For incompressible flows this condition must hold for arbitrary density ratios and we therefore have

$$V = \mathbf{u}_1 \cdot \mathbf{n} = \mathbf{u}_2 \cdot \mathbf{n}. \tag{2.47}$$

Mass conservation places no restriction on the tangential velocity components. Indeed, inviscid fluids are usually assumed to slip at the interface. For very low viscosity and rarefied gases it is possible that finite slip is the appropriate interface condition, but for viscous fluid under normal operating conditions it is an experimentally observed fact – like the no-slip boundary conditions at solid walls – that no slip takes place. It is also possible to argue that if the sharp interface is assumed to come about by taking the limit of a finite interface zone as its thickness goes

to zero, then anything but no slip would result in infinitely high stresses if the interface zone consists of a fluid with a finite viscosity. Thus, if there is no phase change and the fluids are incompressible, the interfacial condition for viscous fluid is simply $\mathbf{u}_1 = \mathbf{u}_2$, or

$$[\mathbf{u}]_S = 0, \tag{2.48}$$

where we have used the *jump notation*, i.e. the notation $[x]_S = x_2 - x_1$.

2.4.2 Surface tension

From a molecular point of view, surface tension arises because the interface is not an optimal region thermodynamically: the molecules "prefer" to be at the gas or the liquid density, which minimizes the free energy. The non-optimal conditions near the interface result in an excess energy

$$\mathrm{d}E^\sigma = \sigma \,\mathrm{d}s, \tag{2.49}$$

where ds is an infinitesimal interface or surface area and σ is a material property, usually referred to as the surface tension coefficient or simply surface tension. From a mechanical point of view, however, surface tension is simply a force per unit length acting perpendicularly on any line segment in the surface. If $\mathbf{p}$ is a vector perpendicular to the line segment, the "pull" is simply $\sigma\mathbf{p}$ per unit length. These points of view are equivalent: to stretch the surface one has to pull on it. Suppose that we pull in the x direction and that the surface has extension L in the other direction. Stretching it increases the area by $\mathrm{d}s = L\,\mathrm{d}x$, at the expense of a work $\sigma L\,\mathrm{d}x = \sigma\,\mathrm{d}s$ increasing as much the interfacial free energy.

By arguments that are exactly analogous to the introduction of the stress tensor for a three-dimensional continuum (see Aris (1962), for example), it can be shown that the force on the edge of a small surface element can be written as $\mathbf{T}_S^\sigma \cdot \mathbf{p}$, where $\mathbf{T}_S^\sigma$ is the surface tension tensor and $\mathbf{p}$ is a vector normal to the edge of the element, in the tangent plane to the surface. Since

$$\mathbf{T}_S^\sigma \cdot \mathbf{p} = \sigma\mathbf{p} = \sigma\mathbf{I}_S \cdot \mathbf{p} \tag{2.50}$$

we find that $\mathbf{T}_S^\sigma = \sigma\mathbf{I}_S$, where $\mathbf{I}_S$ is the surface identity tensor.

The surface tension tensor was defined above in terms of the surface identity tensor. Frequently, it is more convenient to work with the full three-dimensional operator and use the fact that $\mathbf{I}_S$ is the tangential projection of the three-dimensional identity tensor $\mathbf{I}$. Thus, $\mathbf{I}_S = (\mathbf{I} - \mathbf{nn})$ and

$$\mathbf{T}_S^\sigma = \sigma(\mathbf{I} - \mathbf{nn}). \tag{2.51}$$

Using that $\mathbf{n} = \mathbf{t}_1 \times \mathbf{t}_2$, where $\mathbf{t}_1$ and $\mathbf{t}_2$ are orthonormal tangent vectors to the

surface, the surface tension tensor can also be written as

$$\mathbf{T}_S^\sigma = \sigma(\mathbf{t}_1\mathbf{t}_1 + \mathbf{t}_2\mathbf{t}_2). \tag{2.52}$$

The force on a surface element of area S, bounded by a contour C, is the integral of the "pull" on its edges:

$$\delta\mathbf{F}_\sigma = \oint_C \mathbf{T}_S^\sigma \cdot \mathbf{p}\, \mathrm{d}l = \int_S \nabla_S \cdot \mathbf{T}_S^\sigma \, \mathrm{d}s. \tag{2.53}$$

Taking the limit of Equation (2.53) as the surface area shrinks to a point, we define the surface force per unit area as

$$\mathbf{f}_\sigma = \nabla \cdot \mathbf{T}_S^\sigma = \nabla \cdot \sigma \mathbf{I}_S = \sigma \nabla_S \cdot \mathbf{I}_S + \mathbf{I}_S \cdot \nabla_S \sigma. \tag{2.54}$$

The first term can be shown to be $\sigma\kappa\mathbf{n}$ and the second term is simply $\nabla_S\sigma$. Thus,

$$\mathbf{f}_\sigma = \sigma\kappa\mathbf{n} + \nabla_S\sigma, \tag{2.55}$$

where the last term, the surface gradient of σ, is obviously zero for constant surface tension.

2.4.3 Momentum conservation with interfaces

Applying the conservation of momentum principle to the control volume in Fig. 2.9, moving with the interface S, we obtain

$$0 = -\oint_{\delta S} \rho\mathbf{u}(\mathbf{u}\cdot\mathbf{n} - V)\mathrm{d}s + \oint_{\delta S} \mathbf{n}\cdot\mathbf{T}\, \mathrm{d}s + \int_S \mathbf{f}_\sigma \, \mathrm{d}s. \tag{2.56}$$

Notice that the integration is around the edges of the control volume δV for the first two terms and along the interface for the last term. The first term on the right-hand side is zero for incompressible flows in the absence of phase change, since the control volume is moving with the fluid velocity. As the thickness of the control volume approaches zero, the boundary coincides with the interface, and by integrating the stresses first on one side and then on the other, the second term yields the jump in $\mathbf{T}$ across the interface. Since the results hold for any control volume that includes the interface, they must be valid at any interface point, resulting in

$$-[\mathbf{T}]_S \cdot \mathbf{n} = \sigma\kappa\mathbf{n} + \nabla_S\sigma. \tag{2.57}$$

This condition may be split into a normal and tangential stress condition, using Equation (2.55) for $\mathbf{f}_\sigma$, yielding the boxed jump conditions in Panel 2.5.

$$[\mathbf{u}]_S = 0,$$

$$-[-p + 2\mu \mathbf{n}\cdot\mathbf{S}\cdot\mathbf{n}]_S = \sigma\kappa,$$

$$-\left[2\mu \mathbf{t}^{(k)}\cdot\mathbf{S}\cdot\mathbf{n}\right]_S = \mathbf{t}^{(k)}\cdot\nabla_S\sigma,$$

$$\nabla\cdot\mathbf{u} = 0,$$

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \mathbf{f} + \mu\nabla^2\mathbf{u}.$$

Panel 2.5. The jump conditions and the partial differential equations for a Newtonian, incompressible fluid, without evaporation or condensation. The last two equations apply in the bulk of each phase. The first three are jump conditions on the interface S. There, the $\mathbf{t}^{(k)}$ are two unit tangent vectors. In the two-dimensional flow case there is a single tangent vector. Together with boundary conditions on walls or entry and exit conditions, these equations form a complete set for flows with interfaces.

$$(-p + 2\mu \mathbf{n}\cdot\mathbf{S}\cdot\mathbf{n})|_S = -p_{\text{free}} + \sigma\kappa,$$

$$2\mu\ \mathbf{t}^{(k)}\cdot\mathbf{S}\cdot\mathbf{n}\Big|_S = \mathbf{t}^{(k)}\cdot\nabla_S\sigma,$$

$$\nabla\cdot\mathbf{u} = 0,$$

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \mathbf{f} + \mu\nabla^2\mathbf{u}.$$

Panel 2.6. The boundary conditions and the partial differential equations for a Newtonian, incompressible fluid with a free surface. In this example the pressure just outside the free surface is set to p_{free}.

2.4.4 Free-surface flow

Free-surface flow is a limiting case of flows with interfaces, in which the treatment of one of the fluids is simplified. For air–water flow, for instance, we can sometimes assume that the pressure p_{free} in the air is a constant, or depends only on time, and that the viscous stresses in the air are negligible. The jump conditions then become *boundary conditions* for the liquid domain, as shown in Panel 2.6. In the case of constant σ, the tangential stresses vanish and

$$\mathbf{t}^{(k)}\cdot\mathbf{S}\cdot\mathbf{n}\Big|_S = 0. \tag{2.58}$$

This is a purely kinematic condition: it does not involve material properties at all. For this reason it has interesting consequences. The *vorticity* vector is defined in general as $\nabla \times \mathbf{u}$ and its study reveals interesting aspects of the dynamics of the flow. In two dimensions, we note $\omega = (\nabla \times \mathbf{u}) \cdot \mathbf{e}_z$, its only non-zero component. Then on the free surface

$$\omega = 2\kappa q, \tag{2.59}$$

where $q = (\mathbf{u} \cdot \mathbf{u})^{1/2}$ is the norm of the velocity, as shown in Batchelor (1970). This implies that a large vorticity will be seen on the free surface in regions of high velocity and high curvature. This is were the departure from the hypothesis of irrotational flow will be the most pronounced. Moreover, the vorticity is also seen in these regions for two-phase flows with a strong density and viscosity contrast, since these flows resemble free-surface flows. Numerical simulations show characteristic patches of vorticity in the regions of high curvature.

2.5 Fluid mechanics with interfaces: the one-fluid formulation

In contrast to the approach described in Section 2.4, where we wrote the governing equations separately for each phase and used jump conditions to couple the solutions at the fluid interface, it is possible to write one set of governing equations for the whole flow domain occupied by the various phases, without resorting to jump conditions. The various phases are treated as one fluid with variable material properties that change abruptly at the phase boundary. To account for the "extra" forces at the phase boundary, however, it is necessary to add singular terms (δ-functions) to the equations. These singular terms are the counterpart of the jump conditions of the preceding section, and it can be shown that both formulations are equivalent.

This form of the equations is often referred to as the "one-fluid" approach. Since the solution can change discontinuously across the interface, we must either interpret the governing equations in a weak sense where they are satisfied only in an integral sense, or admit solutions that include generalized functions, such as delta functions and step functions. The latter approach is taken here. The "one-fluid" formulation is the starting point for several numerical methods based on the use of fixed grids, as we will see in later chapters.

The derivation of the "one-fluid" equations is exactly the same as before (Section 2.2), except that we need to add the surface tension as a body force to the momentum equation. The mass conservation equation for the total mass has no source terms. For reacting flows, the conservation equation for each species may have localized source terms, and for flows with phase change (Chapter 11), we shall see that a localized volume source emerges naturally at the phase boundary.

For a control volume including an interface, the surface tension force is given by $\mathbf{f}_\sigma$ integrated over S, the part of the surface enclosed in the control volume. Just as we use the divergence theorem to convert surface integrals to volume integrals, we use Equation (B.9) to transform the surface integral

$$\int_S \mathbf{f}_\sigma \, \mathrm{d}s = \int_V \mathbf{f}_\sigma \delta_S \, \mathrm{d}v, \tag{2.60}$$

where $\delta_S = \delta_S(\mathbf{x} - \mathbf{x}_s)$. Adding this force to the integral form of the momentum equation, Equation (2.6), and applying the same arguments as before – that the integral can only be zero for all possible control volumes if the integrand is zero – we obtain the "one-fluid" version of the Navier–Stokes equation for incompressible, Newtonian flows with sharp interfaces:

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \nabla \cdot \mathbf{uu} = -\nabla p + \mathbf{f} + \nabla \cdot \mu [\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}}] + \mathbf{f}_\sigma \delta_S. \tag{2.61}$$

We emphasize that this equation holds for the whole flow field, even if the density field ρ and the viscosity field μ change discontinuously. For constant surface tension, the surface force is

$$\mathbf{f}_\sigma \delta_S = \sigma \kappa \mathbf{n} \delta_S. \tag{2.62}$$

For completeness, we observe that the last term can be written in several different, but equivalent forms:

$$\mathbf{f}_\sigma \delta_S = \int \mathbf{f}_\sigma \delta(\mathbf{x} - \mathbf{x}_s) \, \mathrm{d}A \quad \text{and} \quad \sigma \kappa \mathbf{n} \delta_S = \int \sigma \kappa \mathbf{n} \delta(\mathbf{x} - \mathbf{x}_s) \, \mathrm{d}A, \tag{2.63}$$

where we have used that $\delta(\mathbf{x} - \mathbf{x}_s) = \delta(x - x_s)\delta(y - y_s)\delta(z - z_s)$. Notice also that it is possible to add an arbitrary constant to the surface tension term, replacing $\sigma\kappa$ by $(\sigma\kappa + \Delta p_0)$. The constant Δp_0 is balanced exactly by a change in the pressure jump across the interface, and since the flow on either side of the interface is driven by the pressure gradient – not the absolute value of the pressure – adding the constant has no effect on the flow. Jan (1994) used this observation to subtract the average pressure inside a bubble to reduce the pressure jump across the interface in his computations of the motion of clean and contaminated axisymmetric and two-dimensional bubbles.

2.6 Nondimensional numbers

The dynamics of multifluid and multiphase flows is governed by a variety of nondimensional numbers, depending on the exact situation driving the flow. Those include the *Reynolds number*, usually defined based on the properties of one of the fluids, using an external length scale L and velocity U:

$$\mathrm{Re} = \frac{\rho L U}{\mu} = \frac{L U}{\nu}. \tag{2.64}$$

Re represents the ratio of inertial to viscous forces. If we construct a velocity scale by $\sqrt{(\Delta\rho/\rho)gL}$, then the square of the Reynolds number becomes the *Galileo number*:

$$N = \frac{g\rho\Delta\rho L^3}{\mu^2}. \tag{2.65}$$

When this number is large, gravity is balanced by inertia and turbulent dissipation; when it is small, gravity and viscous forces balance. The ratio of inertia to surface tension gives the *Weber number*, defined by

$$\mathrm{We} = \frac{\rho L U^2}{\sigma}, \tag{2.66}$$

and the ratio of viscous to capillary stresses yields the *capillary number*, given by

$$\mathrm{Ca} = \frac{\mu U}{\sigma}. \tag{2.67}$$

Eliminating the velocity U between the Reynolds and Weber numbers gives the *Ohnesorge number*:

$$\mathrm{Oh} = \frac{\mu}{\sqrt{\rho\sigma L}}. \tag{2.68}$$

This number compares viscous forces with capillary forces. A rather surprising fact is that there are two numbers (Ca and Oh) that perform this comparison, but in one case the forces are estimated using a characteristic velocity, for instance for an advancing contact line, while in the other case the forces are estimated using a characteristic length, for instance the radius of an oscillating droplet or bubble. We sometimes also work with the *Laplace number*, defined by $\mathrm{La} = 1/\mathrm{Oh}^2$. The combination

$$\mathrm{Eo} = \frac{\Delta\rho g L^2}{\sigma} \tag{2.69}$$

defines the *Eötvös number*, sometimes also called the *Bond number*, which compares gravity with capillary forces. The *Morton number* is given by

$$\mathrm{Mo} = \frac{\Delta\rho g\mu^4}{\rho^2\sigma^3}. \tag{2.70}$$

The Morton number has the attractive property that, for a given g, it depends only on the fluid properties. Working in standard (Earth) gravity, with gas–liquid flows so that the density difference is not small, the Morton number may again be used to compare viscous with capillary forces. For gas–water flows or gas–liquid–metal flows, it is usually extremely small, as are the other measures of the comparison between viscosity and surface tension. This makes these flows particularly difficult

to simulate. When inertia and gravity are the important forces, such as for surface waves, we often work with the *Froude number*:

$$\mathrm{Fr} = \frac{U^2}{gL}. \tag{2.71}$$

The numbers listed here are the most common ones for two-fluid flows. For systems with more complex physics, such as heat or mass transfer, phase change and chemical reactions, additional numbers must be defined.

2.7 Thin films, intermolecular forces, and contact lines

Topology changes in multiphase flows (such as when two droplets coalesce into one, or one droplet breaks into two) take place when thin films rupture and thin threads snap. Thin threads appear to be, by far, the easiest to deal with. There are good reasons to believe that the Navier–Stokes equations describe how their diameter becomes zero in a finite time, and simulations suggest that the overall results are relatively insensitive to how they are treated. For thin films, on the other hand, it is necessary to include additional physics. While including molecular effects cannot, in principle, be done in the framework of continuum mechanics and sharp interfaces, it is arguable that mesoscopic scales, only moderately larger than the microscopic scale, may still be described by continuum mechanics. We give a very elementary introduction to this kind of modeling in this section. More developments on breakup are given in Chapter 9.

2.7.1 Disjoining pressure and forces between interfaces

When two interfaces are less than a few hundred nanometers apart, intermolecular forces are significant. Those may be modeled as an additional singular force on the interface. The force f_{I} per unit area of the interface, directed away from the other interface, is often taken to be given by

$$f_{\mathrm{I}} = -Ah^{-3}, \tag{2.72}$$

where A is the *Hamaker constant* and h is the distance between the two interfaces; see Fig. 2.10. For two parallel interfaces the surface force in Equation (2.60) then takes the form

$$\mathbf{f}_\sigma \delta_S = -Ah^{-3}\mathbf{n}\delta_S + \sigma\kappa\mathbf{n}\delta_S + \nabla_S\sigma\delta_S, \tag{2.73}$$

where $\mathbf{n}$ is the normal oriented away from the reference phase. Such forces are expected both for a free film (Fig. 2.10) and for a solid-bounded film (Fig. 2.11).

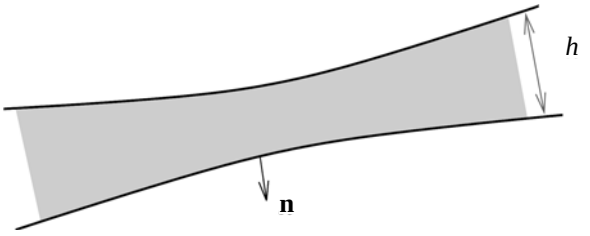


Fig. 2.10. A thin film of thickness h may be subject to intermolecular forces, creating an effective attraction or repulsion between the two interfaces.

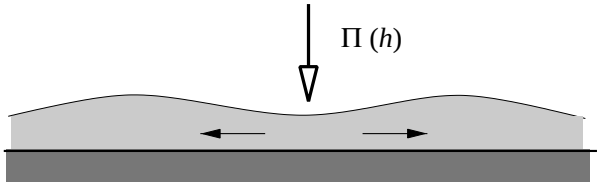


Fig. 2.11. A film near a solid wall. Positive disjoining pressure will result in the film being broken.

The equivalent of the normal stress condition on a free surface (Laplace's law in the static case) is now

$$(-p + 2\mu \mathbf{n} \cdot \mathbf{S} \cdot \mathbf{n})|_S = -p_{\text{free}} + \sigma\kappa - Ah^{-3}. \tag{2.74}$$

In the static case the viscous terms may be neglected and

$$p = p_{\text{free}} - \sigma\kappa + Ah^{-3}, \tag{2.75}$$

so the pressure is increased or decreased depending on the sign of A. Thus, $\Pi(h) = -Ah^{-3}$ is called the disjoining pressure. When $A > 0$ the disjoining pressure is negative and the interfaces attract each other. On the other hand, when $A < 0$ the disjoining pressure is positive and the interfaces repel each other. The value of A depends on the fluid, but values in the region of 10^{-20} J are common. It is well known that when $A > 0$ the film breaks, as for pure water, while when $A < 0$ the film is stable, as for water contaminated with surfactants. Indeed, positive A tends to increase the pressure in narrow regions and forces the fluid to flow out of them. Thus, a catastrophic instability occurs and the film ruptures.

In principle, Equation (2.74) can be used when solving the Navier–Stokes equation for the macroscopic flow evolution. However, the range of scales that must be resolved is enormous, and this has seldom been achieved in the literature.

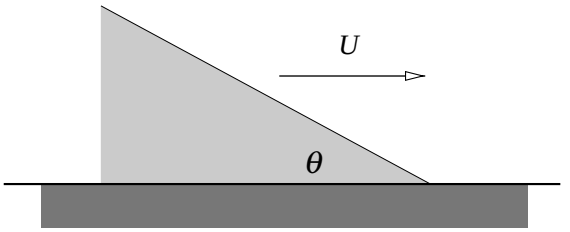


Fig. 2.12. An advancing contact line in the partially wetting ($0 < \theta < \pi$) case.

2.7.2 Contact line statics and dynamics

The interface separating two fluids may meet a solid boundary at a contact line (Fig. 2.12). In the absence of motion, the contact line angle θ is fixed by the capillary properties of the substrate and the two fluids. Consider a liquid and a gas on a solid surface. The liquid–gas surface energy per unit area is the capillary tension σ. For the liquid–solid and the gas–solid interfaces we respectively have energies per unit area $\sigma_{\rm ls}$ and $\sigma_{\rm gs}$. Moving the contact point by a small distance δx along the solid surface results in a gain of energy $\sigma_{\rm gs}\delta x$ on one side and a loss $(\sigma_{\rm ls} + \sigma\cos\theta_{\rm eq})\delta x$ on the other side. Equating these energies yields the famous Young–Laplace relation

$$\sigma_{\rm gs} = \sigma_{\rm ls} + \sigma\cos\theta_{\rm eq}, \tag{2.76}$$

where $\theta_{\rm eq}$ is the equilibrium value of the contact angle. When $\theta > \theta_{\rm eq}$, it is energetically favorable to displace the contact line to the right, and the reverse for $\theta < \theta_{\rm eq}$.

However, in experiments it is found that the situation is more complicated: the contact angle exhibits a hysteresis. The interface does not move in some range of values $\theta_{\rm a} < \theta < \theta_{\rm r}$, where $\theta_{\rm a}$ is the *advancing contact angle* and $\theta_{\rm r}$ is the *receding contact angle*. For instance, a droplet skidding on a window may at times stop: the contact line is then pinned by surface heterogeneity in an equilibrium position between the advancing and receding angles.

When the contact line moves, the energy gained is both surface energy and any potential energy due to the external forces moving the fluid: gravity or inertia. The energy lost is dissipated by viscosity. The balance of dissipation and energy gained does not lead simply to a law for the contact line motion. The difficulty comes from a famous property of the viscous dissipation: the dissipation per unit volume is singular near the contact line. Consider a perfect wedge of fluid near the contact line, so that the height function is

$$h(x,t) = \theta(x_{\rm c} - x) \tag{2.77}$$

for small θ, where x_c is the contact line position. Consider a steadily advancing contact line at velocity U. The velocity on the wall should be $\mathbf{u} = 0$, so the horizontal component should be $u = 0$. On the other hand, near the interface the velocity should be of order U to advance the interface, so that velocity gradients are of order

$$\frac{\partial u}{\partial y} \sim \frac{U}{h}. \tag{2.78}$$

Moreover, the energy dissipated up to a distance a from the contact line is

$$\varepsilon = \int_{x_c}^{x_c+a} \int_0^{h(x,t)} \mu \left(\frac{\partial u}{\partial y} \right)^2 \mathrm{d}y \, \mathrm{d}x. \tag{2.79}$$

Using (2.77) and the estimate (2.78) for the velocity gradient, this becomes

$$\varepsilon = \int_0^a \mu \frac{U^2}{\theta x} \, \mathrm{d}x, \tag{2.80}$$

which diverges logarithmically near $x = 0$. This divergence leads to a paradox: in the framework of continuum mechanics, any motion of the contact line would dissipate infinite energy and thus the contact line cannot move. One way out of the paradox is to use partial-slip boundary conditions, such as Equation (2.25). This cuts off the divergence and the dissipation becomes

$$\varepsilon = \frac{\mu U^2}{\theta} \ln\left(\frac{a}{\beta}\right). \tag{2.81}$$

2.8 Notes

2.8.1 Fluid and interface mechanics

A derivation or even a textbook presentation of the equations governing fluid interfaces is not the primary purpose of this monograph. The interested reader may consult basic fluid mechanics treatises such as Batchelor (1970), who devotes ample space to issues involving interfaces, and for a more systematic description of interface geometry the reader is referred to Weatherburn (1927), as well as other references, such as Aris (1962).

While the constitutive assumption given by Equation (2.49) allows for variable surface tension, due to contaminants or temperature variations, for example, we have excluded any stresses due to surface shear or dilation. More complex constitutive equations can be introduced to account for such effects; see Scriven (1960), Aris (1962), and Edwards *et al.* (1961) for discussions.

2.8.2 Thin films and contact lines

Intermolecular forces, our Equation (2.72), can be modeled in a variety of ways, depending on the polar nature of the interface, on electric charges, and on the presence of large molecules on the interface. Other forms of the disjoining pressure may be found in Oron *et al.* (1997), which also provides a general introduction to the dynamics of thin films. The thin-film equations are in general of the form

$$\frac{\partial h}{\partial t} = \frac{\partial}{\partial x}\left[f(h)\frac{\partial h}{\partial x} + g(h)\frac{\partial^3 h}{\partial x^3} + k(h)\right], \tag{2.82}$$

where $f(h)$, $g(h)$, and $k(h)$ are some functions dependent on the exact problem being solved. For Hamaker's law and surface tension on a solid substrate (Equation (2.74)) the film breaks, forming a self-similar solution for the thickness h that goes to zero as $t^{1/5}$ (Lister and Zhang, 1999).

The slip-length condition (2.25) first appeared in Navier (1823) in the very paper that gave the viscous flow equations. It acquired renewed interest as Huh and Scriven (1971) proposed it to remove the contact line dissipation paradox. The contact line paradoxes are discussed in detail in Dussan (1979) and issues related to static and dynamic wetting in general are discussed in de Gennes (1985), de Gennes *et al.* (2003), Oron *et al.* (1997), Pomeau (2002), and Bonn *et al.* (2009).

Experimental observations yield a *mobility relation* of the form

$$\theta(a) = f(\mathrm{Ca}, a/\beta, \ldots), \tag{2.83}$$

relating the velocity of the contact line to the apparent contact angle $\theta(a)$ at a distance a from the contact line. The length β is some microscopic scale, for instance the slip length in Equation (2.25). The theoretical form of the function f is still debated. For a perfectly wetting fluid ($\theta_{\mathrm{eq}} = 0$) and small angles, several approximations and theories (Bonn *et al.*, 2009) yield

$$\theta(a) \simeq [9\mathrm{Ca}\ln(a/\beta)]^{1/3}. \tag{2.84}$$

The logarithmic dependence may be seen as arising from the logarithmic singularity in Equation (2.81). This equation was justified by the Navier slip condition (2.25), but other microscopic theories have been proposed. A generalized Navier slip model has been proposed by Shikmurzaev (1997), who also noticed that the Navier slip, while removing the dissipation singularity, did not remove a singularity in the pressure. Intermolecular forces have also been invoked (Hervet and de Gennes, 1984; de Gennes, 1985), as well as phase-field (or Cahn–Hilliard, or second gradient, or diffuse interface) models (Pomeau, 2002) or interface relaxation theories (Blake and Shikmurzaev, 2002). For a recent discussion, see Bonn *et al.* (2009).

In computational studies it is natural to fix the contact angle in the first cell near the wall. The grid size (assuming a square grid $\Delta x = \Delta y$) is the minimum distance from the tip that is "seen" by the numerical method. Thus, an expression useful as a sort of "effective boundary condition" for numerics should be of the form

$$\theta_{\text{num}} = f(\text{Ca}, \Delta x/\beta, \ldots), \tag{2.85}$$

as, for example, discussed in Afkhami *et al.* (2009). However, this approach has not often been used and many authors have instead fixed the dynamic contact angle in the numerical method without dependence on the grid size, or have tried to use a grid scale smaller than the slip length to avoid the grid dependence.

3

Numerical solutions of the Navier–Stokes equations

The one-field formulation of the Navier–Stokes equations described in Chapter 2, where a single set of equations is used to describe the motion of all the fluids present, allows us to use numerical methods developed for single-phase flows. There are, however, two complications: the material properties (usually density and viscosity) generally vary from one fluid to the other and to set these properties we must construct an indicator function that identifies each fluid. We must usually also find the surface tension at the interface. The advection of the indicator function is the topic of Chapters 4 to 6 and finding the surface tension will be dealt with in Chapter 7. In this chapter we discuss numerical methods to solve the Navier–Stokes equations, allowing for variable density and viscosity. We will use the finite-volume method and limit the presentation to regular Cartesian grids. Since the multiphase flows considered in this book all involve relatively low velocities, we will assume incompressible flows.

For any numerical solution of the time-dependent Navier–Stokes equations it is necessary to decide:

(i) how the grid points, where the various discrete approximations are stored, are arranged;
(ii) how the velocity field is integrated in time;
(iii) how the advection and the viscous terms are discretized;
(iv) how the pressure equation, resulting from the incompressibility condition, is solved; and
(v) how boundary conditions are implemented.

These tasks can be accomplished in a variety of ways, but the approach outlined here has been widely used for multiphase flow simulations and results in a reasonably accurate and robust numerical method.

The governing equations are the Navier–Stokes equations for incompressible, Newtonian flow (Panel 2.4) including the incompressibility condition (Equation

(2.20) or (2.21), derived in the previous chapter. As discussed there, the equations can be written in a number of equivalent forms. So, to keep the discussion as general as possible, we will write the momentum equation symbolically as

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \mathbf{A} = -\nabla p + \mathbf{D} + \mathbf{f}, \tag{3.1}$$

where $\mathbf{A} = \nabla \cdot (\mathbf{uu})$ is the advection term, $\mathbf{D} = \nabla \cdot \mu (\nabla \mathbf{u} + \nabla \mathbf{u}^{\mathrm{T}})$ the diffusion term, and $\mathbf{f}$ denotes all other forces, such as gravity and surface tension. Although we have selected a specific form for the advection term here, obviously any of the forms shown in Equation (2.22) could be used instead.

3.1 Time integration

For incompressible flows, where the pressure must be adjusted to give a divergence-free velocity field at the end of each time-step, the governing equations are usually solved by the so-called projection method. In this method, a temporary velocity field is first found by ignoring the pressure gradient. This velocity field is generally not divergence free and in the second step the velocity is corrected by adding the appropriate pressure gradient. Formally, the second step is a projection onto a space of divergence-free velocity fields. Hence the name. Although it is likely that the early MAC method for incompressible flows was programmed in exactly this way, the concept was introduced by Chorin (1968) and Yanenko (1971).

The integration in time is usually done using a second-order (or higher) method, but for simplicity we start by describing a first-order method. A few different ways of achieving second-order accuracy are then outlined. Using a first-order, explicit, forward-in-time discretization of the time derivative, the momentum equation, Equation (3.1), becomes

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} + \mathbf{A}_h^n = \frac{1}{\rho^n} \left(-\nabla_h p + \mathbf{D}_h^n + \mathbf{f}^n \right). \tag{3.2}$$

Here, Δt is the size of the time step. The superscript n denotes a quantity evaluated at the beginning of the step and $n+1$ identifies the end of the step. We have introduced the notation $\mathbf{A}_h$ and $\mathbf{D}_h$ for numerical approximations of the advection and diffusion terms, respectively and ∇_h stands for a numerical approximation of the gradient or the divergence operators. The velocity at the end of the time step must be divergence free:

$$\nabla_h \cdot \mathbf{u}^{n+1} = 0. \tag{3.3}$$

Since the momentum equation includes a time derivative for the velocity field, it can be used to find the velocity at the next time step, once the pressure is known.

The continuity equation, however, is a condition that must be satisfied by the velocity at the end of the step, and although the pressure must be adjusted in such a way that it is satisfied, there is no explicit equation for the pressure in the formulation described by Equations (3.2) and (3.3). It is necessary, therefore, to devise a strategy to solve the equations in the right order, where all the pieces of information are available when they are needed. This is where the projection method comes in. The momentum equation, (3.2), is split into two parts: the first is a predictor step, where a temporary velocity field ($\mathbf{u}^*$) is found by ignoring the effect of the pressure:

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -\mathbf{A}_h^n + \frac{1}{\rho^n}\left(\mathbf{D}_h^n + \mathbf{f}^n\right). \tag{3.4}$$

In the second step, the projection step, the pressure gradient is added to yield the final velocity at the new time step:

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} = -\frac{1}{\rho^n}\nabla_h p. \tag{3.5}$$

Adding those two equations yields exactly Equation (3.2).

To find the pressure, such that the velocity at the new time step is divergence free, we take the divergence of (3.5) and use (3.3) to eliminate $\mathbf{u}^{n+1}$, resulting in a Poisson equation for the pressure:

$$\nabla_h \cdot \left(\frac{1}{\rho^n}\nabla_h p\right) = \frac{1}{\Delta t}\nabla_h \cdot \mathbf{u}^*. \tag{3.6}$$

Once the pressure has been found, Equation (3.5) can be used to find the projected velocity at time $n+1$.

The scheme described above is completely explicit and the time step must, therefore, be small enough to ensure stability. The advection term $\mathbf{A}$ and the diffusion term $\mathbf{D}$ both impose restrictions on Δt. The diffusion term is almost always discretized using second-order, centered, finite-difference approximations, which require

$$\frac{\mu \Delta t}{\rho h^2} \leq \frac{1}{4} \tag{3.7}$$

for two-dimensional flow. In three dimensions, the right-hand side is replaced by $1/6$. Here, h is the smallest grid spacing (Δx, Δy, or Δz). If we use a second-order centered approximation for the advection terms, the explicit approximation in Equation (3.4) results in a scheme that is unstable in the absence of diffusion. The scheme is stabilized by the viscous terms, if Δt is limited by

$$(\mathbf{u} \cdot \mathbf{u})\frac{\rho \Delta t}{\mu} \leq 2. \tag{3.8}$$

Notice that this condition does not involve the grid spacing h.

Temporal and spatial discretizations that result in explicit but stable schemes for the advection terms (see below) are usually subject to the Courant, Friedrichs, and Lewy (CFL) condition

$$\frac{u_{\max}\Delta t}{h} \leq 1, \tag{3.9}$$

where $u_{\max}$ is the maximum velocity. Equation (3.9) states that Δt must be sufficiently small so that a material point travels less than one grid spacing during a time step. This condition is derived using one-dimensional flow. For multidimensional schemes the limitation can be more restrictive, but advanced methods for the advection terms are frequently applied using splitting – where advection in each spatial direction is done sequentially – and then the one-dimensional limitations hold.

The time-integration scheme outlined by Equations (3.4), (3.5) and (3.6) is only first order, as already observed. The simplest way to generate a higher order method for the integration in time is by a second-order predictor–corrector method where the first-order explicit method described above is used to find a temporary velocity $\mathbf{u}^{\text{tmp}}$. The temporary velocity is then used to compute an approximate right-hand side (**RHS**) of Equation (3.2) at time $n+1$ so that a second-order approximation for the new velocity can be found by the trapezoidal rule

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} = \frac{1}{2}\Big(\mathbf{RHS}^n + \mathbf{RHS}^{\text{tmp}}\Big). \tag{3.10}$$

Here, $\mathbf{RHS}^n$ is computed using $\mathbf{u}^n$ and $\mathbf{RHS}^{\text{tmp}}$ is found using $\mathbf{u}^{\text{tmp}}$. For actual programming, it is easier to take two first-order steps and then average $\mathbf{u}^n$ and $\mathbf{u}^{n+2}$. Doing this is easily shown to be equivalent to the above scheme and makes it particularly simple to extend a first-order scheme to second order. This method is generally subject to the same stability limits as the first-order explicit method and is, in particular, subject to the diffusion limitation imposed by Equation (3.7). For low Reynolds numbers, explicit treatment of the viscous terms results in impractically small time steps and it is better to treat those terms implicitly.

A large number of methods have been developed to solve the Navier–Stokes equations with second-order accuracy in time. Many of the most popular versions treat the viscous terms implicitly and the advection terms explicitly. We outline below the method of Kim and Moin (1985), extended to flows with variable material properties. The Kim and Moin method is a two-step projection method where in the first step the advection terms are computed explicitly using an Adams–Bashforth scheme and half of the viscous terms are implicit in the temporary velocity:

$$\frac{\mathbf{u}^*-\mathbf{u}^n}{\Delta t} = -\Big(\frac{3}{2}\mathbf{A}_h^n - \frac{1}{2}\mathbf{A}_h^{n-1}\Big) + \frac{1}{2\rho^n}\Big(\mathbf{D}_h^n + \mathbf{D}_h^*\Big). \tag{3.11}$$

In the second step the velocity is corrected by adding the gradient of a pressure-like variable Φ:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^*}{\Delta t} = -\frac{\nabla\Phi}{\rho^n}. \tag{3.12}$$

An equation for Φ is derived by combining (3.12) and (3.3), resulting in an equation that looks like (3.6) but with p replaced by Φ. To see exactly what Φ is we add Equations (3.11) and (3.12), yielding

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} = -\left(\frac{3}{2}\mathbf{A}_h^n - \frac{1}{2}\mathbf{A}_h^{n-1}\right) + \frac{1}{\rho^n}\left[\frac{1}{2}(\mathbf{D}_h^n + \mathbf{D}_h^{n+1}) + \frac{1}{2}(\mathbf{D}_h^* - \mathbf{D}_h^{n+1}) - \nabla\Phi\right].$$

Here, we have added and subtracted $\mathbf{D}_h^{n+1}$ to allow a comparison with an Adams–Bashforth/Crank–Nicholson method, where the viscous terms are evaluated using the velocities at the new time $\mathbf{u}^{n+1}$ instead of the temporary velocities $\mathbf{u}^*$. From this we see that Φ is related to p by

$$\frac{1}{2}(\mathbf{D}^* - \mathbf{D}^{n+1}) - \nabla\Phi = -\nabla p. \tag{3.13}$$

Kim and Moin used a staggered grid, second-order spatial differences, and solved (3.11), which is implicit, by splitting.

The projection method is not the only way of integrating the Navier–Stokes equations in time. Essentially any standard method to solve the Navier–Stokes equations for homogeneous flows, such as the artificial compressibility method of Chorin (1967), the SIMPLE method of Patankar (1980), the PISO method of Issa (1985), and many others, can be adapted to multiphase flows, when the one-fluid formulation is used.

For flows where there are strong capillary effects, additional stability restrictions may apply. Usually, those are taken to be analogous to the CFL conditions and a capillary wave is allowed to move only one grid spacing during time Δt. The phase velocity of a capillary wave is

$$c = \left(\frac{\sigma k}{\rho_1 + \rho_2}\right)^{1/2}, \tag{3.14}$$

where k is the wavenumber. The smallest resolved wave has $k = \pi/h$, and using this wavenumber and replacing $u_{\max}$ in equation (3.9) by c, we find that the time step is limited by

$$\Delta t \leq \left[\frac{(\rho_1 + \rho_2)h^3}{\pi\sigma}\right]^{1/2}. \tag{3.15}$$

3.2 Spatial discretization

To discretize the momentum equations, we use the finite-volume approach, where the conservation principles of mass and momentum are applied to a small control volume. While the governing equations in integral form, as shown in Panel 2.1, are often taken as the starting point for the derivation of the discrete approximations, we can also simply integrate the differential form, as in Panel 2.4, over the control volume. In either case, the velocity at the center of the control volume $\mathbf{u}_c$, is approximated by the average value

$$\mathbf{u}_c = \frac{1}{V}\int_V \mathbf{u}(\mathbf{x})\,\mathrm{d}v. \tag{3.16}$$

Here, V is the volume of the control volume and, for a two-dimensional rectangular control volume, $V = \Delta x \Delta y$. To find numerical approximations for the advection and the diffusion terms, we first define the average value over a control volume

$$\mathbf{A}_c = \frac{1}{V}\int_V \nabla\cdot(\mathbf{uu})\,\mathrm{d}v = \frac{1}{V}\oint_S \mathbf{u}(\mathbf{u}\cdot\mathbf{n})\,\mathrm{d}s \tag{3.17}$$

and

$$\mathbf{D}_c = \frac{1}{V}\int_V \nabla\cdot\mathbf{T}^v\,\mathrm{d}v = \frac{1}{V}\oint_S \mathbf{n}\cdot\mathbf{T}^v\,\mathrm{d}s, \tag{3.18}$$

where $\mathbf{T}^v$ is the viscous stress tensor for incompressible flow:

$$\mathbf{T}^v = \mu[\nabla_h\mathbf{u} + (\nabla_h\mathbf{u})^T]. \tag{3.19}$$

Here, we have used the divergence theorem to rewrite the volume integrals as surface integrals over the boundary S of a control volume. Numerical approximations for these terms, $\mathbf{A}_h$ and $\mathbf{D}_h$, are obtained by evaluating the integrals numerically. The average value of the pressure gradient is

$$(\nabla p)_c = \frac{1}{V}\int_V \nabla p\,\mathrm{d}v = \frac{1}{V}\oint_S p\mathbf{n}\,\mathrm{d}s \tag{3.20}$$

and the average value of the body force is

$$\mathbf{f}_c = \frac{1}{V}\int_V \mathbf{f}(\mathbf{x})\,\mathrm{d}v. \tag{3.21}$$

The continuity equation is approximated in the integral form (Equation (2.21)) at time $n+1$ by evaluating

$$\oint_S \mathbf{u}^{n+1}\cdot\mathbf{n}\,\mathrm{d}s = 0. \tag{3.22}$$

Before we derive discrete approximations to the Navier–Stokes equations by approximating the integrals above, we have to decide what the control volumes look like and where the various variables, or rather the discrete approximations

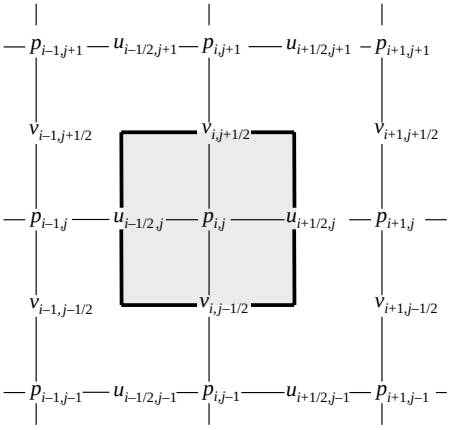


Fig. 3.1. The notation used for a standard staggered MAC mesh. The pressure is assumed to be known at the center of the control volume outlined by a thick solid line. The horizontal velocity components u are stored at the middle of the left and right edges of this control volume and the vertical velocity components v are stored at the middle of the top and bottom edges.

to these variables, are located with respect to each other. Generally, the discrete variables are put at the nodes of a regular grid, and while it may seem most natural to store the pressure, the velocity, and the material properties of the fluid at the same grid node (usually referred to as collocated grids), experience shows that for incompressible flows it is simpler to use a staggered grid where each of these variables is located on a separate grid. This is especially true if the grid lines are straight. For simplicity, we shall limit the discussion in the rest of this chapter to two-dimensional flows. The extension to three dimensions is straightforward, but makes the exposition more cumbersome.

The staggered grid was introduced by Harlow and Welch (1965) for the MAC method and has been used extensively in computational fluid mechanics ever since. Fig. 3.1 shows the structure of the grid and the location of each variable for two-dimensional flow. The starting point is a control volume around the pressure points (outlined by a thick line). The rôle of pressure for incompressible flows is to force the divergence of the velocity field to be zero: the pressure must be raised if there is a net inflow into this control volume and lowered if there is net outflow. For the computation of the net flow in or out of the control volume around the pressure node we need the horizontal velocity components u at the vertical boundaries and the vertical velocity components v on the horizontal boundaries. It is natural to locate the velocity components at the middle of the boundaries, where we need them, and the grid for the horizontal velocity is therefore displaced half a mesh to the right from the pressure node and the vertical velocity grid is displaced half a mesh upward (see Fig. 3.2). It is customary to identify the pressure nodes by the indices (i, j) and to refer to the location of the u-velocity component by $(i+1/2, j)$

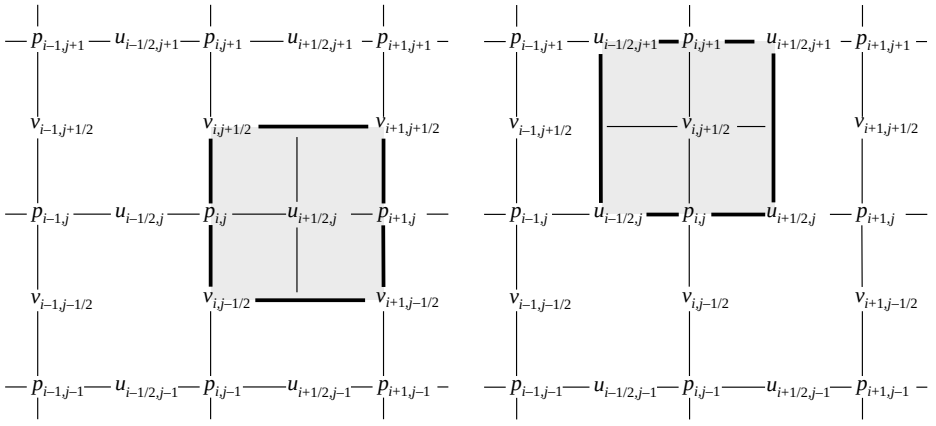


Fig. 3.2. The control volumes for the u and the v velocity components are displaced half a grid cell to the right (horizontal velocities) and up (vertical velocities). Here the indices show the location of the stored quantities. Thus, half indices indicate those variables stored at the edges of the pressure control volumes.

and the location of the v-velocity component by $(i, j+1/2)$. In an actual computer code the grids are, of course, simply shifted and each component referenced by an integer. The density and other material properties are usually stored at the pressure nodes.

On the staggered grid shown in Fig. 3.1, the discrete form of the continuity equation is obtained by writing the integral in Equation (3.22) for each side of the control volume

$$\begin{aligned} 0 &= \frac{1}{\Delta x \Delta y} \oint_S \mathbf{u}^{n+1} \cdot \mathbf{n} \, \mathrm{d}s \qquad (3.23) \\ &= \frac{1}{\Delta x \Delta y} \left\{ \int_{i+1/2,j} u^{n+1} \, \mathrm{d}y - \int_{i-1/2,j} u^{n+1} \, \mathrm{d}y + \int_{i,j+1/2} v^{n+1} \mathrm{d}x \right. \\ &\qquad \left. - \int_{i,j-1/2} v^{n+1} \, \mathrm{d}x \right\}, \end{aligned}$$

and then by approximating the integrals by the midpoint rule:

$$\frac{1}{\Delta x \Delta y} \left[\left(u^{n+1}_{i+1/2,j} - u^{n+1}_{i-1/2,j} \right) \Delta y + \left(v^{n+1}_{i,j+1/2} - v^{n+1}_{i,j-1/2} \right) \Delta x \right] = 0. \qquad (3.24)$$

Rewriting this equation slightly, the final result is

$$\frac{u^{n+1}_{i+1/2,j} - u^{n+1}_{i-1/2,j}}{\Delta x} + \frac{v^{n+1}_{i,j+1/2} - v^{n+1}_{i,j-1/2}}{\Delta y} = 0, \qquad (3.25)$$

which could also have been obtained by discretizing Equation (3.3) with finite differences.

Using the grid in Fig. 3.2 and the notation introduced above, the discrete approximations for the x and the y component of the predicted velocities (Equation (3.4)) are

$$u^*_{i+1/2,j} = u^n_{i+1/2,j} + \Delta t \left\{ \left(-A_x\right)^n_{i+1/2,j} + \left(f_{bx}\right)^n_{i+1/2,j} + \frac{1}{\frac{1}{2}(\rho^n_{i+1,j} + \rho^n_{i,j})} \left[\left(D_x\right)^n_{i+1/2,j} + \left(f_{\sigma x}\right)^n_{i+1/2,j} \right] \right\} \tag{3.26}$$

and

$$v^*_{i,j+1/2} = v^n_{i,j+1/2} + \Delta t \left\{ \left(-A_y\right)^n_{i,j+1/2} + \left(f_{by}\right)^n_{i,j+1/2} + \frac{1}{\frac{1}{2}(\rho^n_{i,j+1} + \rho^n_{i,j})} \left[\left(D_y\right)^n_{i,j+1/2} + \left(f_{\sigma y}\right)^n_{i,j+1/2} \right] \right\}. \tag{3.27}$$

The equations for the projected velocities (Equation (3.5)) are

$$u^{n+1}_{i+1/2,j} = u^*_{i+1/2,j} - \frac{\Delta t}{\frac{1}{2}(\rho^n_{i+1,j} + \rho^n_{i,j})} \frac{p_{i+1,j} - p_{i,j}}{\Delta x} \tag{3.28}$$

and

$$v^{n+1}_{i,j+1/2} = v^*_{i,j+1/2} - \frac{\Delta t}{\frac{1}{2}(\rho^n_{i,j+1} + \rho^n_{i,j})} \frac{p_{i,j+1} - p_{i,j}}{\Delta y}. \tag{3.29}$$

We note that some of the variables in the equations above, such as the density, are at locations where they are not defined. For those values, we use linear interpolation (by taking the average).

There are three main reasons why the staggered grid is so widely used instead of collocated grids where all the variables are located at the same point. One is accuracy. Since the pressure gradient is computed as the difference between adjacent points, versus points two Δx or Δy apart when a collocated grid is used, the mesh is in effect finer. However, since other derivatives are evaluated using grid points further apart the results are not as accurate as if a twice as fine collocated grid were used. The second reason is that it is relatively simple to produce conservative methods when working on a staggered grid. The third, and perhaps the most important advantage of a staggered grid, is that it results in a tighter coupling between the variables than if they were all located at the same grid node. The continuity equation, when written on a collocated grid, links every other grid point and it is possible to satisfy continuity exactly, yet have highly fluctuating velocities. Although there are ways to overcome this problem for collocated grids (see Rhie and Chow (1983), for example) the staggered grid provides a simpler approach.

3.3 Discretization of the advection terms

In the original MAC method, centered differencing was used for all spatial variables and the time integration was done by the simple explicit first-order projection method described above. Later implementations used first-order upwind or the so-called donor-cell method for the advection terms. Here, we will first present the second-order centered scheme and then discuss other alternatives.

The discrete approximation of the x-component of the advection term is found by approximating the integral in Equation (3.17) by the midpoint rule:

$$(A_x)_{i+1/2,j} = \frac{1}{\Delta x \Delta y}\Big\{\big[(uu)_{i+1,j} - (uu)_{i,j}\big]\Delta y + \big[(uv)_{i+1/2,j+1/2} - (uv)_{i+1/2,j-1/2}\big]\Delta x\Big\}; \quad (3.30)$$

and the y-component is approximated by

$$(A_y)_{i,j+1/2} = \frac{1}{\Delta x \Delta y}\Big\{\big[(uv)_{i+1/2,j+1/2} - (uv)_{i-1/2,j+1/2}\big]\Delta y + \big[(vv)_{i,j+1} - (vv)_{i,j}\big]\Delta x\Big\}. \quad (3.31)$$

The velocities are given at the center of the respective control volumes (Fig. 3.2), but the momentum fluxes in (3.30) and (3.31) are computed at the boundaries of the control volumes. The different way of finding the velocities for the momentum-flux calculations differentiates between the different numerical schemes.

In the centered, second-order scheme, the velocities at the boundaries of the velocity control volumes are found by linear interpolation and the momentum fluxes computed using these interpolated values. The x-component at $(i+1/2, j)$ is then

$$\begin{aligned}(A_x)^n_{i+1/2,j} = {} & \frac{1}{\Delta x}\left[\left(\frac{u^n_{i+3/2,j} + u^n_{i+1/2,j}}{2}\right)^2 - \left(\frac{u^n_{i+1/2,j} + u^n_{i-1/2,j}}{2}\right)^2\right] \\ & + \frac{1}{\Delta y}\left[\left(\frac{u^n_{i+1/2,j+1} + u^n_{i+1/2,j}}{2}\right)\left(\frac{v^n_{i+1,j+1/2} + v^n_{i,j+1/2}}{2}\right)\right. \\ & \left. - \left(\frac{u^n_{i+1/2,j} + u^n_{i+1/2,j-1}}{2}\right)\left(\frac{v^n_{i+1,j-1/2} + v^n_{i,j-1/2}}{2}\right)\right]. \quad (3.32)\end{aligned}$$

The y-component, at the $(i, j+1/2)$ point, is

$$(A_y)^n_{i,j+1/2} = \frac{1}{\Delta x}\left[\left(\frac{u^n_{i+1/2,j}+u^n_{i+1/2,j+1}}{2}\right)\left(\frac{v^n_{i,j+1/2}+v^n_{i+1,j+1/2}}{2}\right)\right.$$
$$\left.-\left(\frac{u^n_{i-1/2,j+1}+u^n_{i-1/2,j}}{2}\right)\left(\frac{v^n_{i,j+1/2}+v^n_{i-1,j+1/2}}{2}\right)\right]$$
$$+\frac{1}{\Delta y}\left[\left(\frac{v^n_{i,j+3/2}+v^n_{i,j+1/2}}{2}\right)^2-\left(\frac{v^n_{i,j+1/2}+v^n_{i,j-1/2}}{2}\right)^2\right]. \quad (3.33)$$

The centered second-order scheme is more accurate than any non-centered second-order scheme and usually gives the best results for fully resolved flows. However, it has two serious shortcomings. The first problem is that for flows that are not fully resolved it can produce unphysical oscillations that can degrade the quality of the results. The second problem is that the centered second-order scheme is unconditionally unstable for inviscid flows when used in combination with the explicit forward-in-time integration given by Equation (3.2). It is only the addition of the diffusion terms that makes the scheme stable, and if diffusion is small, the time-step must be small. At high Reynolds numbers this usually results in excessively small time steps.

The velocity at the boundaries of the velocity control volumes $u_{i,j}$, for example, can be found in many other ways. Each approach leads to a new scheme whose properties are different from other schemes. The simplest way to generate a robust scheme that is stable in the limit of zero viscosity is to use upwinding to find the edge velocity:

$$u_{i,j} = \begin{cases} u_{i-1/2,j}, & \text{if } \frac{1}{2}(u_{i-1/2,j}+u_{i+1/2,j}) > 0; \\ u_{i+1/2,j}, & \text{if } \frac{1}{2}(u_{i-1/2,j}+u_{i+1/2,j}) < 0. \end{cases} \quad (3.34)$$

While very robust, upwinding is only first-order accurate and leads to excessive numerical diffusion, as we will see in Chapter 4. The accuracy is improved significantly by using a third-order upwind-biased polynomial:

$$u_{i,j} = \begin{cases} (1/8)\left(3u_{i+1/2,j}+6u_{i-1/2,j}-u_{i-3/2,j}\right), & \text{if } \frac{1}{2}(u_{i-1/2,j}+u_{i+1/2,j}) > 0; \\ (1/8)\left(3u_{i-1/2,j}+6u_{i+1/2,j}-u_{i+3/2,j}\right), & \text{if } \frac{1}{2}(u_{i-1/2,j}+u_{i+1/2,j}) < 0. \end{cases} \quad (3.35)$$

This is the so-called QUICK (quadratic upstream interpolation for convective kinematics) scheme of Leonard (1979). QUICK and its variants are not completely free of "wiggles" for steep-enough gradients, but they are much more robust than the centered difference scheme and much more accurate than the first-order upwind method. Even more robust, the ENO (essentially non-oscillating) scheme of Shu and Osher (1989) has been used by a number of authors. In this scheme the edge

value is extrapolated from the known values at the center of the control volumes by

$$u_{i,j} = \begin{cases} u_{i-1/2,j} + s_i\left(\frac{\Delta x}{2}\right), & \text{if } \frac{1}{2}(u_{i-1/2,j} + u_{i+1/2,j}) > 0; \\ u_{i+1/2,j} - s_{i+1}\left(\frac{\Delta x}{2}\right), & \text{if } \frac{1}{2}(u_{i-1/2,j} + u_{i+1/2,j}) < 0. \end{cases} \tag{3.36}$$

The slope s_i is the smallest (in terms of absolute value) of

$$\begin{aligned} s_i^+ &= (u_{i+1/2,j} - u_{i-1/2,j})/\Delta x, \\ s_i^- &= (u_{i-1/2,j} - u_{i-3/2,j})/\Delta x. \end{aligned} \tag{3.37}$$

This selection of the slopes in the second-order ENO scheme is the same as that used in the MINMOD limiter (Sweby, 1984). A variant of this approach, the Bell–Colella–Glaz (BCG; Bell *et al.*, 1989) scheme (see also Martin (1998)) is used in the Gerris code (see Popinet (2003, 2008, 2009)) used for several of the simulations shown in this book. While the edge velocities found using either QUICK or ENO can be used with the conservative discretization in Equation (2.1), they are also often used with a discretization of the non-conservative version of the advection terms (Panel 2.4). Both QUICK and ENO greatly improve the robustness of the computation of the advection terms for low viscosities, while yielding accuracy that is comparable to the centered difference scheme for viscosities where both give reliable results.

3.4 The viscous terms

We approximate the integral of the viscous fluxes (Equation (3.18)) at the boundaries of the velocity control volumes shown on Fig. 3.3 by the midpoint rule. This results in

$$(D_x)^n_{i+1/2,j} = \frac{T^{\mathrm{v},xx}_{i+1,j} - T^{\mathrm{v},xx}_{i,j}}{\Delta x} + \frac{T^{\mathrm{v},xy}_{i+1/2,j+1/2} - T^{\mathrm{v},xy}_{i+1/2,j-1/2}}{\Delta y} \tag{3.38}$$

and

$$(D_y)^n_{i,j+1/2} = \frac{T^{\mathrm{v},xy}_{i+1/2,j+1/2} - T^{\mathrm{v},xy}_{i-1/2,j+1/2}}{\Delta x} + \frac{T^{\mathrm{v},yy}_{i,j+1} - T^{\mathrm{v},yy}_{i,j}}{\Delta y}. \tag{3.39}$$

The velocity derivatives that appear in the viscous stress tension $\mathbf{T}^{\mathrm{v}}$ are found using the standard second-order centered differences, resulting in

$$\begin{aligned} T^{\mathrm{v},xx}_{i,j} &= 2\mu^n_{i,j}\frac{u^n_{i+1/2,j} - u^n_{i-1/2,j}}{\Delta x}, \\ T^{\mathrm{v},xy}_{i+1/2,j+1/2} &= \mu^n_{i+1/2,j+1/2}\left(\frac{u^n_{i+1/2,j+1} - u^n_{i+1/2,j}}{\Delta y} + \frac{v^n_{i+1,j+1/2} - v^n_{i,j+1/2}}{\Delta x}\right), \\ T^{\mathrm{v},yy}_{i,j} &= 2\mu^n_{i,j}\frac{v^n_{i,j+1/2} - v^n_{i,j-1/2}}{\Delta y}. \end{aligned} \tag{3.40}$$

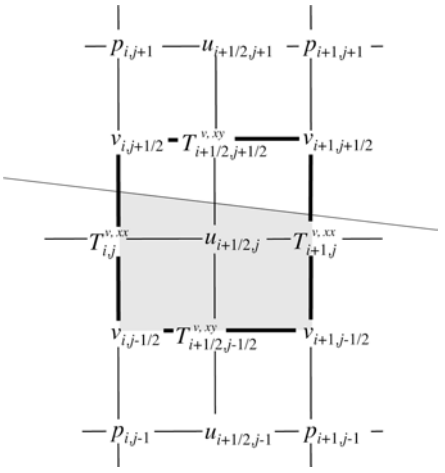


Fig. 3.3. Computation of the viscous stresses on the staggered grid. A control volume for the horizontal velocity $u_{i+1/2,j}$ is drawn. The components T^{xy} and T^{xx} must be found at locations (i, j) and $(i+1/2, j+1/2)$ respectively. An interface cutting the control volume is represented. Gray shading indicates the location of fluid 1 in the control volume.

In the equations in (3.40) we account for the fact that the viscosity coefficient μ may be space dependent. If the viscosity is constant in each fluid, but changes discontinuously across the interface, then it is, in principle, given by

$$\mu(\mathbf{x}) = \mu_1 H(\mathbf{x}) + \mu_2[1 - H(\mathbf{x})], \tag{3.41}$$

where H is the Heaviside function defined in Section 2.3. In many cases we set the viscosity using this equation, with H replaced by a marker function approximating it. In a VOF method, for instance, one would compute the viscosity at location (i, j) as

$$\mu_{i,j} = \mu_1 C_{i,j} + \mu_2(1 - C_{i,j}), \tag{3.42}$$

where the VOF color function $C_{i,j}$ approximates $H(\mathbf{x})$. In other methods, typically front-tracking or level-set methods, one would first construct a smooth indicator function and then use it as an approximation for H. For small viscosity differences this approach generally works well. However, for larger differences this approach can result in an error, since a discontinuous viscosity implies, by virtue of the equations in Panel 2.5, a discontinuous velocity derivative, so that the approximation of the derivatives in (3.40) is not consistent.

For higher accuracy we need to do something else. One option is to use one-sided approximations of the velocity derivatives that use only the data on a given side of the interface. A few authors have implemented this idea. Another option is

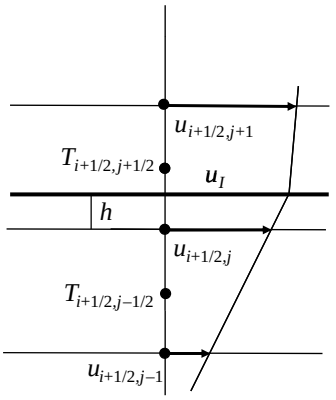


Fig. 3.4. A simple two-phase parallel shear flow with a horizontal interface. Because of the viscosity jump, the velocity gradient is discontinuous.

to use harmonic means as advocated by Patankar (1980) in the relatively simpler case of heat diffusion, or by Coward *et al.* (1997) for viscous shear flows. To explain the idea behind the harmonic mean we consider the simple two-phase parallel shear flow of Fig. 3.4. For simplicity the interface is assumed to be horizontal. We need to compute the shear stress at points $(i+1/2, j-1/2)$ and $(i+1/2, j+1/2)$. The horizontal velocity is given at levels $j-1$, j, and $j+1$. Because the region between the two levels $j-1$ and j is completely filled with a single phase, say fluid 1, the stress located at the intermediate level $j-1/2$ can easily be computed to be

$$\tau_{j-1/2} = \mu_1 \frac{u_j - u_{j-1}}{\Delta y}, \tag{3.43}$$

where we use the compact notation $\tau_{j-1/2} = T^{\mathrm{v},xy}_{i+1/2,j-1/2}$ and $u_j = u_{i+1/2,j}$. On the other hand, the stress at level $j+1/2$ must be computed using an effective viscosity $\mu_{i+1/2,j+1/2}$, so that

$$\tau_{j+1/2} = \mu_{i+1/2,j+1/2} \frac{u_{j+1} - u_j}{\Delta y}. \tag{3.44}$$

What it the correct value of $\mu_{i+1/2,j+1/2}$? First, we notice that in a shear flow the stress is not only continuous, but constant: $\tau_{j-1/2} = \tau_{j+1/2} = \tau$. Second, it is useful to introduce the velocity $u_\mathrm{I} = u(y_\mathrm{I})$ on the interface at $y_\mathrm{I} = y_j + h$, where h is the height of the interface above the grid line j. Then

$$\tau = \mu_2 \frac{u_{j+1} - u_\mathrm{I}}{\Delta y - h} = \mu_1 \frac{u_\mathrm{I} - u_j}{h}. \tag{3.45}$$

Hence,

$$u_{j+1} - u_j = u_{j+1} - u_{\mathrm{I}} + u_{\mathrm{I}} - u_j = \tau \frac{\Delta y - h}{\mu_2} + \tau \frac{h}{\mu_1} \tag{3.46}$$

and using (3.44)

$$\mu_{i+1/2,j+1/2} = \frac{\tau \Delta y}{u_{j+1} - u_j} = \Delta y \left(\frac{h}{\mu_1} + \frac{\Delta y - h}{\mu_2} \right)^{-1}. \tag{3.47}$$

Notice that $h/\Delta y$ is the fraction of phase 1 in the control volume centered around $(i+1/2, j+1/2)$ (see Fig. 3.4), so we can write it using a color function notation $C_{i,j+1/2} = C_{i+1,j+1/2} = C_{j+1/2}$, where we are consistent with the hypothesis of a horizontal interface and assume that there is no dependence on index i. Then the mixed-cell viscosity becomes

$$\mu_{i+1/2,j+1/2} = \left(\frac{\phi}{\mu_1} + \frac{1-\phi}{\mu_2} \right)^{-1}, \tag{3.48}$$

a typical harmonic mean, where $\phi = C_{j+1/2}$. However, the VOF color function $C_{i,j}$ is defined at pressure grid points (i, j) (see Chapter 5). Thus, we may use the second-order approximation $\phi = h/\Delta y = C_j + C_{j+1} - 1/2$. The interface is actually located between j and $j+1$, as we have assumed, whenever $1/2 \le C_j + C_{j+1} \le 3/2$, in other cases the single-phase formulas, for instance (3.43), must be used. An example of the use of the harmonic mean is given in Chapter 9.

In general, the problem is more complex than the simple case above where the interface is aligned both with the grid axis and the flow. The harmonic mean is not a panacea; for instance, if the horizontal interface in Fig. 3.4 is replaced by a vertical interface, the arithmetic mean (3.42) becomes exact. A consistent second-order method for this problem has not yet been published.

In practice, interfaces tend to be aligned with the flow direction in a shear flow, so the harmonic mean is more accurate. It may, however, be less robust: in flows with large density differences, the arithmetic mean "favors" the largest viscosity, while the harmonic mean "favors" the smallest viscosity. The arithmetic mean thus displaces the "effective" interface from its true position towards the small viscosity region. It thus creates a larger viscosity region around the interface, which may have the effect of "protecting" the interface against its destruction by short-wavelength physical or numerical instabilities.

In this section we have assumed that the viscous terms are computed explicitly. For an implicit treatment, the superscript n must be replaced by $n+1$.

3.5 The pressure equation

To derive a discrete Poisson equation for the pressure, we substitute Equations (3.28) and (3.29), for the corrected velocity components at the new time, into

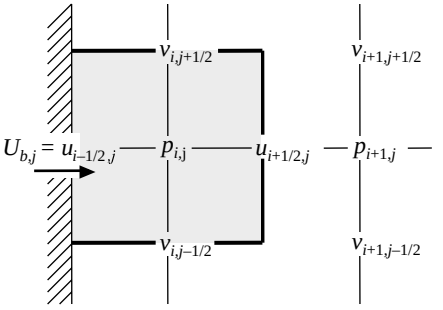


Fig. 3.5. The pressure equation next to a vertical boundary is derived using the continuity equation for a control volume where one side (the wall) has a given normal velocity.

Equation (3.25) for the continuity of the new velocities. This yields

$$\frac{1}{\Delta x^2}\left(\frac{p_{i+1,j}-p_{i,j}}{\rho^n_{i+1,j}+\rho^n_{i,j}}-\frac{p_{i,j}-p_{i-1,j}}{\rho^n_{i,j}+\rho^n_{i-1,j}}\right)+\frac{1}{\Delta y^2}\left(\frac{p_{i,j+1}-p_{i,j}}{\rho^n_{i,j+1}+\rho^n_{i,j}}-\frac{p_{i,j}-p_{i,j-1}}{\rho^n_{i,j}+\rho^n_{i,j-1}}\right)$$
$$=\frac{1}{2\Delta t}\Big(\frac{u^*_{i+1/2,j}-u^*_{i-1/2,j}}{\Delta x}+\frac{v^*_{i,j+1/2}-v^*_{i,j-1/2}}{\Delta y}\Big). \tag{3.49}$$

The pressure equation must be solved using the appropriate boundary conditions and the staggered mesh makes the derivation of the appropriate boundary conditions particularly straightforward. Consider the vertical boundary in Fig. 3.5, on the left side of the domain and passing through node $(i-1/2, j)$, where the horizontal velocity is $U_{\mathrm{b},j}$. The continuity equation for the cell next to the boundary, and surrounding the pressure node (i, j), is

$$\frac{u^{n+1}_{i+1/2,j}-U_{b,j}}{\Delta x}+\frac{v^{n+1}_{i,j+1/2}-v^{n+1}_{i,j-1/2}}{\Delta y}=0. \tag{3.50}$$

Since one of the velocities is known, we only substitute the equations for the correction velocities, (3.28) and (3.29), for the three unknown velocities, through the top, bottom, and right edges. Thus, we have

$$\frac{1}{\Delta x^2}\left(\frac{p_{i+1,j}-p_{i,j}}{\rho^n_{i+1,j}+\rho^n_{i,j}}\right)+\frac{1}{\Delta y^2}\left(\frac{p_{i,j+1}-p_{i,j}}{\rho^n_{i,j+1}+\rho^n_{i,j}}-\frac{p_{i,j}-p_{i,j-1}}{\rho^n_{i,j}+\rho^n_{i,j-1}}\right)$$
$$=\frac{1}{2\Delta t}\Big(\frac{u^*_{i+1/2,j}-U_{b,j}}{\Delta x}+\frac{v^*_{i,j+1/2}-v^*_{i,j-1/2}}{\Delta y}\Big) \tag{3.51}$$

for the pressure node next to the boundary. Similar equations are derived for the pressure next to the other boundaries and for each corner point. Notice that it is not necessary to impose any new conditions on the pressure at the boundaries and that simply using incompressibility yields the correct boundary equations.

Since the value of the pressure is not specified at the boundaries, Equations (3.49) and (3.51) do not determine the pressure uniquely and we can add an arbitrary constant to the solution. For many iterative techniques this constant is set by the average value of the initial guess and the iteration can be carried out without explicitly enforcing the constant. In other cases, the average pressure, or the pressure at one point, must be specified.

We also note that when only inflow or no-through-flow boundary conditions are used it must be verified that no net flow enters the domain, so that $\sum \mathbf{U}_b \cdot \mathbf{n} = 0$, where the sum is over boundary nodes. Otherwise the pressure solver will never converge.

The boundary conditions given by Equation (3.51) apply to solid walls and boundaries where the inflow is specified. For outflow boundaries, see Section 3.7.

Solving the pressure Equation (3.49), with the appropriate boundary conditions, (3.51), can be done in many ways. During code development and for preliminary runs, it is usually more convenient to use a simple successive overrelaxation (SOR) iteration method. To do so, Equation (3.49) is rearranged to isolate $p_{i,j}$ on the left-hand side. The pressure is then updated iteratively by substituting on the right-hand side the approximate values p^α of the pressure from the previous iteration and by taking a weighted average of the updated value and the pressure $p^\alpha_{i,j}$ from the last iteration to compute the new approximation $p^{\alpha+1}_{i,j}$:

$$
\begin{aligned}
&p^{\alpha+1}_{i,j} \qquad\qquad (3.52)\\
&= \beta \left[\frac{1}{\Delta x^2} \left(\frac{1}{\rho^n_{i+1,j} + \rho^n_{i,j}} + \frac{1}{\rho^n_{i,j} + \rho^n_{i-1,j}} \right) + \frac{1}{\Delta y^2} \left(\frac{1}{\rho^n_{i,j+1} + \rho^n_{i,j}} + \frac{1}{\rho^n_{i,j} + \rho^n_{i,j-1}} \right) \right]^{-1} \\
&\quad \times \left[\frac{1}{\Delta x^2} \left(\frac{p^\alpha_{i+1,j}}{\rho^n_{i+1,j} + \rho^n_{i,j}} + \frac{p^\alpha_{i-1,j}}{\rho^n_{i,j} + \rho^n_{i-1,j}} \right) + \frac{1}{\Delta y^2} \left(\frac{p^\alpha_{i,j+1}}{\rho^n_{i,j+1} + \rho^n_{i,j}} + \frac{p^\alpha_{i,j-1}}{\rho^n_{i,j} + \rho^n_{i,j-1}} \right) \right. \\
&\quad \left. + \frac{1}{2\Delta t} \left(\frac{u^*_{i+1/2,j} - u^*_{i-1/2,j}}{\Delta x} + \frac{v^*_{i,j+1/2} - v^*_{i,j-1/2}}{\Delta y} \right) \right] + (1-\beta) p^\alpha_{i,j}.
\end{aligned}
$$

Here, β is a relaxation parameter and for overrelaxation we must have $\beta > 1$. For stability reasons we must also have $\beta < 2$. Taking $\beta = 1.2-1.5$ is usually a good compromise between stability and accelerated convergence. $p_{i,j}$ is also isolated in the same way in the pressure equations for boundary and corner points. Initially, the pressure can be set to zero, but once the time integration has started, the value of the pressure from the last time step usually provides a good first estimate. For vector computers the SOR process needs to be modified slightly, resulting in the so-called black and red SOR.

The chief merit of the SOR iteration is its simplicity. The solution of the pressure equation is generally the most time-consuming part of any simulation of incompressible flows, and the SOR converges too slowly for resource-intensive computations. Thus, while we recommend the use of SOR during code development (or when a small problem has to be solved infrequently), for production runs it is necessary to use more advanced methods.

The development of efficient solution methods for elliptic equations is a highly developed field, and a large number of methods have been proposed. Advanced techniques include fast Poisson solvers based of the discrete fast Fourier transform or cyclic reduction, multigrid methods, and advanced iterative Krylov-subspace methods. Fast Poisson solvers were immensely popular in the eighties, in part due to the freely available and popular routines in FISHPAK (Swarztrauber and Sweet, 1979), but these methods are limited to simple boundary conditions and separable elliptic equations. They cannot, therefore, be used when the density changes, as in Equation (3.49).

Multigrid methods, developed in the late eighties (Briggs, 1987; Wesseling, 1992), have been used by many authors to solve the pressure equation for multifluid flow simulations. The fundamental idea behind multigrid solvers is perhaps best understood by looking at the iterative solution of the Poisson equation as an explicit "pseudo" time integration of a diffusion equation to a steady-state solution. To obtain convergence (steady state) as quickly as possible, we would like to use the maximum allowable time step. Elementary analysis shows that the maximum allowable time step is proportional to Δx^2 (or Δy^2), so more time steps must be used for fine grids. If we write the difference between the initial conditions and the final steady state as a Fourier series, it is easily seen that the decay rate is proportional to the square of the wavenumber. The high-wavenumber components of the solution die out quickly and the time needed to reach steady state is controlled by the decay rate of the low-wavenumber modes. Thus, if we use a fine grid, we are forced to take small steps to accurately resolve wavenumbers that do not exist after a few steps! This suggests that the convergence rate can be improved considerably by using a coarser grid, where large time steps can be used, for the slowly decaying low-wavenumber modes. In multigrid methods we generally start on a fine grid and iterate until the high-wavenumber components of the error have decayed and the rate of convergence becomes slow. The approximate solution is then transferred to a coarser grid and an equation for the correction iterated until the convergence rate slows down. The process is repeated on coarser and coarser grids until the grid is so coarse that the equations can be solved exactly. The corrections are then added to the solution found on the finer grids and the equations iterated again to eliminate errors introduced by the transfer between the grids. Once all the corrections have been added to the results on the finer grids, we should only need a few iterations

to converge fully. In practice, it is usually more efficient to adopt more complex strategies where the solution is transferred more frequently between the coarse and the fine grids.

Although multigrid methods have been used for a large number of simulations of multiphase flows, it is often found that multigrid solvers have difficulty converging for very high density ratios. Another option, therefore, is to use advanced Krylov methods to solve the pressure equation. Moreover, several researchers have recently been using a combination of advanced Krylov methods and multigrid, where several multigrid iterations are used as a preconditioner for the Krylov method. Krylov methods seek to minimize the residual iteratively using only the multiplication of a matrix with a vector. The SOR method is an example of a stationary Krylov method, but in nonstationary methods the search direction is determined based on the current residual. Practical implementations of the Krylov methods almost always involve the use of a preconditioner where the original operator is replaced by a simpler operator that in some way preserves the properties of the original operator. Early Krylov methods include the conjugate-gradient method, but more recent methods are generally based on the generalized-minimal-residual method (GMRES). For each iteration, GMRES requires the results from every previous iteration to produce a new approximation to the solution and the required memory and work therefore increases with the number of iterations. Nevertheless, it has been used succesfully in multiphase calculations (Francois *et al.*, 2006). A large number of methods have been developed to reduce the memory and work requirement, but as of this writing it looks like there is no "best" method for all problems. While it is possible that there is a method that is optimum for the rather special situation that we are interested in – a Poisson equation with coefficients that change discontinuously – the optimal method has yet to be discovered. Methods that have been used successfully, however, include the bi-conjugate gradient stabilized (BiCGSTAB) method of van der Vorst (1992, 2003), used for example by Gerlach *et al.* (2006). Takahira *et al.* (2004) claim that density ratios as large as 1:10 000 may be reached using BiCGSTAB.

The ease by which the pressure equation is solved depends generally on the density jump. While an SOR method with a low enough overrelaxation parameter always allows us to solve the pressure equation, it can be very slow and more advanced methods can fail to converge. Furthermore, small errors in the indicator or marker function can lead to negative densities that in turn usually cause convergence difficulties. These problems, however, are eliminated relatively easily by minor filtering. For many problems, such as for bubbles, the exact density difference plays a relatively minor rôle, once it is large enough, and it is possible to speed up the calculations by approximating the density of the bubbles by a value that is much higher than the real one. For droplets subject to aerodynamic forces, as in

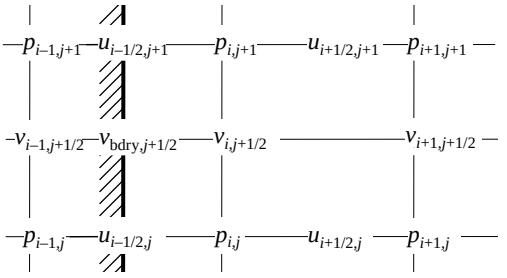


Fig. 3.6. The tangential velocity at a boundary is enforced using ghost points outside the computational domain. The velocity at a ghost point is specified so that linear interpolation gives the desired tangent velocity at the wall.

atomization, droplet impact or wave-breaking problems, the exact density ratios must be used, however.

3.6 Velocity boundary conditions

Although the treatment of the pressure boundary conditions, as well as the normal velocities, simplifies considerably on a staggered mesh, the tangential velocity boundary conditions become a little more complicated. When $v_{i,j+1/2}$ in Fig. 3.6 (next to a left boundary) is updated, the velocity at the node to the left, $v_{i-1,j+1/2}$, is needed. This node, however, is outside the boundary. The easiest way to implement the boundary conditions for the tangent velocity is by the use of ghost points. The value of the tangential velocity at the ghost point is obtained by treating it as a regular point and using the fact that for no-slip boundary conditions the fluid velocity at the boundary is equal to the wall velocity. A linear interpolation gives

$$v_{\text{bdry},j+1/2} = \frac{1}{2}(v_{i,j+1/2} + v_{i-1,j+1/2}), \tag{3.53}$$

where $v_{\text{bdry},j+1/2}$ is the tangential velocity of the wall. Solving for the velocity at the ghost point yields

$$v_{i-1,j+1/2} = 2v_{\text{bdry},j+1/2} - v_{i,j+1/2}, \qquad \text{no slip boundary.} \tag{3.54}$$

Full-slip boundary conditions are usually implemented by simply putting the velocity at the ghost point equal to the velocity inside the domain. The derivative of the tangential velocity is then zero, assuming that we use a centered finite difference approximation. For the situation shown in Fig. 3.6:

$$v_{i-1,j+1/2} = v_{i,j+1/2}, \qquad \text{full-slip boundary.} \tag{3.55}$$

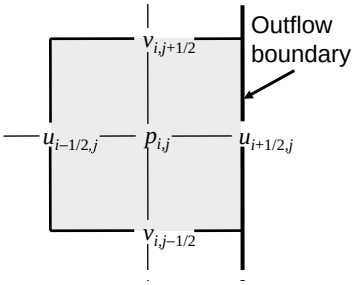


Fig. 3.7. At an outflow boundary it is generally desirable to apply conditions that disturb the upstream flow as little as possible. If the outflow is perpendicular to the boundary, assuming that the normal velocity is constant across the boundary results in reasonably "soft" outflow conditions.

3.7 Outflow boundary conditions

Often, we would like to simulate only the part of a flow field that is of interest to us, even though the domain is in reality much larger. In other cases, we are interested in a region with in- and out-flow. While it is often realistic to specify the inflow velocity exactly, the goal of the outflow boundary conditions is to let fluid leave the computational domain in realistic ways.

The simplest situation arises if it is physically justifiable to specify the outflow velocity at each grid point on the outflow boundary. Then the situation is exactly the same as for the inflow boundaries described above and the implementation is straightforward. This condition, however, is too restrictive in most cases.

Although we stop following the flow computationally at an outflow boundary, the flow usually continues beyond it. To justify truncating the domain, we have to assume that whatever the flow does downstream, it has little impact upstream of the boundary. This usually implies that the downstream flow is smooth and uniform. Thus, the outflow boundary conditions should constrain the outflow as little as possible and let the fluid "flow freely" out of the domain.

One possibility is to assume that the streamlines at the outflow are perpendicular to the boundary. If the outflow takes place at the right boundary of a rectangular domain (Fig. 3.7), this means that $v = 0$ at the boundary.

From continuity it follows that $\partial u/\partial x = 0$ and for the grid shown in Fig. 3.7 the velocity of the boundary is therefore

$$u_{i+1/2,j} = u_{i-1/2,j}. \tag{3.56}$$

Thus, we can sometimes simply use (3.56) along with $p_{i,j} =$ constant and $v_{i,j+1/2} = 0$. This approach results in a relatively "soft" outflow condition and does not, in particular, require the u velocity to be uniform across the outflow boundary. In

principle, u can even change sign, allowing some inflow. However, the location of the outflow boundary will generally impact the computed solution if there is a significant variation in the outflow velocity.

Many other boundary conditions have been proposed in the literature to account for outflows in a physically plausible, yet computationally efficient way. A discussion of outflow boundary conditions can be found in Wesseling (2001: section 6.5) and in Johansson (1993), for example.

In many cases it is convenient to make one or more coordinate directions periodic. If the domain is periodic in the x direction and the length of the period is L, then $a(x,y,z) = a(x \pm mL,y,z)$, where a is any variable and m is any positive or negative integer. If the domain is discretized by N cells in the x direction and we have one ghost grid point on either side, then the total number of grid points is $N+2$. If the ghost point on the left is given by $i = 1$ and the one on the right by $i = N+2$, then the value of a at these points is given by $a(1, j,k) = a(N+1, j,k)$ and $a(N+2, j,k) = a(2, j,k)$ respectively.

3.8 Adaptive mesh refinement

For many multiphase flow problems (as for single-phase flows) the resolution requirements vary greatly from one point to another. Allocating computational resources according to need can, at least in principle, result in significant savings compared with using a single grid with equal size cells. In spite of major efforts to develop fluid solvers based on unstructured, often arbitrarily shaped, control volumes, several decades of experience have shown that, in terms of accuracy and computational efficiency, it is hard to beat regular structured grids. The obvious solution is to use structured grids with curved grid lines that remain logically rectangular but whose grid lines can be pushed and pulled to regions of the computational domain where they are needed. This obvious and attractive approach has been used extensively for single-phase flow in complex domains (but less so for multiphase flows; see Muradoglu and Kayaalp (2006) for an exception), but it is now understood that, since the number of grid points in this approach is fixed, the achievable adaptivity is limited and the method is only suitable for relatively simple problems.

Currently, the main choice for multiphase flows are techniques based on adding grid points by refining a regular structured grid locally. A single grid cell is usually split into four (in two dimensions) or eight (three dimensions) smaller cells, either by embedding a patch of finer grid into the coarser grid or by splitting each cell individually. In the latter case, which appears to be more popular, an octree data structure (quadtree in two dimensions) is used to organize the grid and to transfer information between cells. In most cases, the small cells can be split again, for increased resolution. This approach to grid adaptivity is usually referred to

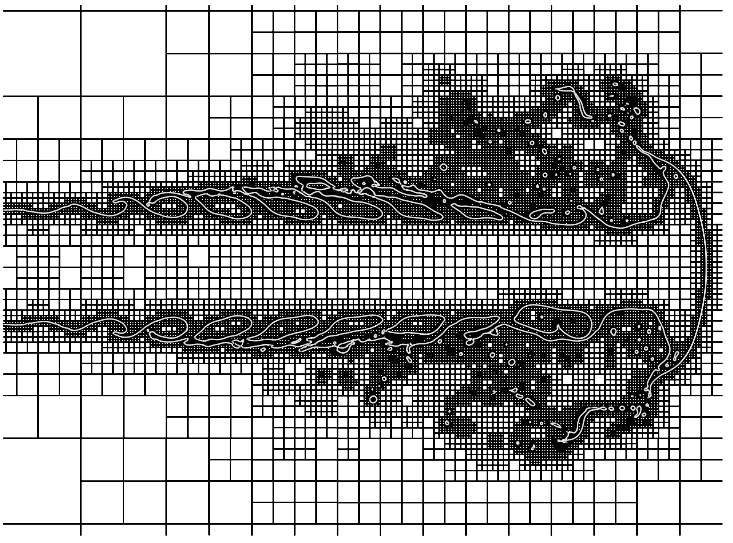


Fig. 3.8. One example of the use of adaptive mesh refinement (AMR) with a VOF simulation of atomization. Computations using the Gerris code. Here the density ratio is $\rho_l/\rho_g = 28$, the Weber number is $\mathrm{We}_l = 11\,600$ and the Reynolds number is $\mathrm{Re}_l = 5800$. The jet diameter is $1/12$ of the domain width and by using seven levels of grid refinement it is possible to reduce the required number of grid cells from over a hundred million if a uniformly fine grid is used to about three million cells.

as adaptive mesh refinement (AMR), although the name could be used for many other strategies for local refinement of the grid. Early examples of the use of octree/quadtree AMR include front-tracking simulations of biological cells by Agresar *et al.* (1998). More recent uses include simulations of underwater explosions by Kadioglu and Sussman (2008), done by a level-set/VOF method and Hua and Lou (2007) who used a front-tracking method in conjunction with a general purpose AMR code (PARAMESH, see MacNeice *et al.* (2000)) to simulate the rise of bubbles. In Fig. 3.8 we show one frame from a simulation of the atomization of a fuel jet using the open-source flow solver Gerris (see Popinet (2003, 2008, 2009)) where a VOF interface-tracking scheme is coupled with an octree AMR. The use of AMR for multiphase flow simulations involves essentially no major issues beyond those encountered for single-phase flow simulations, and we refer the reader to the already voluminous literature for more details.

3.9 Summary

In this chapter we have outlined a very standard numerical method to solve the Navier–Stokes equations numerically for flows with variable density and viscosity.

Combined with the techniques described in the next chapters to advect the phase boundary (and, therefore, the density and viscosity fields) and to compute the surface tension, this results in a method capable of simulating reasonably accurately a wide variety of multifluid flows. The accuracy, robustness, and efficiency of the method can, however, be improved in a number of ways, and we have indicated briefly a few ways to do so.

While the equations presented in this chapter are reasonably complex, they generally do not result in a very lengthy code. More advanced treatment, particularly of the advection terms, adds some complexity, but frequently it is not necessary to start completely "from scratch" when developing a new code. The equations of this chapter have been implemented by a large number of investigators and many codes and code "snippets" are available as open source codes on the Internet. Using those as a starting point can result in a considerably shorter development time.

3.10 Postscript: conservative versus non-conservative form

To gain some insight into the differences between the discretization of the conservative and non-conservative form of the Navier–Stokes equations (Panel 2.2 versus Panel 2.3) we will consider the uniform one-dimensional motion of a fluid with a sharp density interface. The pressure gradient and the viscous terms can be taken to be zero, so we only need to solve

$$\frac{\partial \rho u}{\partial t} + \frac{\partial \rho u^2}{\partial x} = 0 \tag{3.57}$$

for the momentum and

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \tag{3.58}$$

for the density. We assume a collocated grid, so that u_i and ρ_i are given at the same physical location. A simple forward in time, centered in space scheme[1] for the momentum equation lets us write

$$u_i^{n+1} = \frac{1}{\rho_i^{n+1}} \left[\rho_i^n u_i^n - \frac{\Delta t}{2h} ((\rho u^2)_{i+1}^n - (\rho u^2)_{i-1}^n) \right]. \tag{3.59}$$

The initial velocity field is constant, $u_i = U$, so we can write this as

$$u_i^{n+1} = \frac{U}{\rho_i^{n+1}} \left[\rho_i^n - \frac{U \Delta t}{2h} (\rho_{i+1}^n - \rho_{i-1}^n) \right] \tag{3.60}$$

and since the velocity should remain constant, the right-hand side should be U which requires the term in parentheses to be equal to ρ_i^{n+1}. If we apply the same

[1] This is actually an unstable scheme, but that is immaterial for the argument being made here.

scheme to the advection equation for the density, we find that

$$\rho_i^{n+1} = \rho_i^n - \frac{U\Delta t}{2h}(\rho_{i+1}^n - \rho_{i-1}^n), \tag{3.61}$$

using that the velocity is constant. The right-hand side of this equation is exactly the same as the bracket on the right-hand side of Equation (3.60), showing that the velocity remains constant, as it should.

If the density, however, is advected in a different way, say by an upwind scheme, such that

$$\rho_i^{n+1} = \rho_i^n - \frac{U\Delta t}{h}(\rho_i^n - \rho_{i-1}^n), \tag{3.62}$$

then the same cancelation will obviously not take place and the velocity will not remain exactly equal to U. While the error is generally small, it is not exactly zero, particularly if the density jump is large. The same considerations hold for a staggered grid. Equation (3.57) can, however, be written in a non-conservative form as

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = 0 \tag{3.63}$$

using Equation (3.58). Any discretization of this form will preserve the constant velocity.

For methods where the density is generated from a marker function that is advected differently from how the momentum is updated, we are likely to encounter the situation described above and the non-conservative form of the equations is therefore generally preferred. Another way to avoid the problem is to first advect the density using a scheme consistent with the advection of the momentum and then re-advect the density using the more sophisticated scheme at the end of the time step.

4

Advecting a fluid interface

When the governing equations are solved on a fixed grid, using one set of equations for the whole flow field, the different fluids must be identified in some way. This is generally done by using a marker function that takes different values in the different fluids. Sometimes a material property, such as the fluid density for incompressible fluids, can serve as a marker function, but here we shall assume that the rôle of the marker function is only to identify the different fluids. As the fluids move, and the boundary between the different fluids changes location, the marker function must be updated. Updating the marker function accurately is both critical for the success of simulations of multiphase flows and also surprisingly difficult. In this chapter we discuss the difficulties with advecting the marker function directly and the various methods that have been developed to overcome these difficulties.

The VOF method is the oldest method to advect a marker function and – after many improvements and innovations – continues to be widely used. Other marker function methods include the level-set method, the phase-field method, and the CIP method. Instead of advecting the marker function directly, the boundary between the different fluids can also be tracked using marker points, and the marker function then reconstructed from the location of the interface. Methods using marker points are generally referred to as “front-tracking” methods to distinguish them from “front-capturing” methods, where the marker function is advected directly. In addition to the difference in the way the marker function is updated, surface tension can be included in several different ways, as explained in Chapter 7.

4.1 Notations

To identify whether a given fluid i is present at a particular location $\mathbf{x}$, we use a Heaviside (step) function H, defined by

$$H_i(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} \text{ is in fluid } i; \\ 0, & \text{if } \mathbf{x} \text{ is not in fluid } i. \end{cases} \tag{4.1}$$

For a two-fluid system, $i = 1,2$ and $H_2 = 1 - H_1$, so it is sufficient to work with $H = H_1$. As the interface moves, the shape of the region occupied by each fluid changes, but each fluid particle retains its identity. Thus, the material derivative (following the motion of a fluid particle) of H is zero, or

$$\frac{DH}{Dt} = \frac{\partial H}{\partial t} + \mathbf{u} \cdot \nabla H = 0. \tag{4.2}$$

Once H is known, the material properties of each fluid can be found, and the velocity updated as described in the last chapter. The accurate advection of H, or numerical approximations to H, is the central topic of this and the next two chapters.

When following the flow computationally, we have to work with an approximation of H. Several approximations are possible. The color function C is defined as the average value of H in each computational cell. For a rectangular two-dimensional cell

$$C_{i,j} = \frac{1}{\Delta x \Delta y} \int_V H(x,y) \, \mathrm{d}x \, \mathrm{d}y. \tag{4.3}$$

Cells away from the interface are either full, $C = 1$, or empty, $C = 0$, but if an interface is located somewhere in a given cell, C for that cell has a fractional value. Often, we shall find it useful to work with smoother versions of the color function, denoted here by the "indicator function" I. It is convenient to think of I as a smoothed version of H, obtained by

$$I(x,y) = \int G(x-x',y-y')H(x',y') \, \mathrm{d}x' \, \mathrm{d}y', \tag{4.4}$$

where G is a smoothing kernel. Here, we have assumed a two-dimensional domain. In practice, I is obtained by smoothing the color function or by constructing a smooth function directly from the location of the interface. Yet another possibility is to give up any attempts to link the shape of the marker function to the step function and simply use a smooth function F, where $F > 0$ in one fluid and $F < 0$ in the other. The interface is therefore identified by the level-set $F = 0$. Various other approximations to H are in common use, and while these are generally similar to C and I described above, there are often minor differences in their properties, their names, or their physical interpretation. With this in mind, we will use c for the

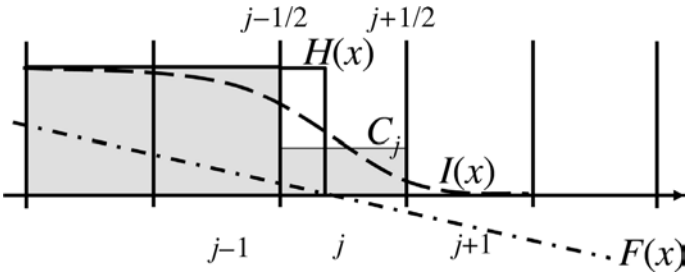


Fig. 4.1. Different representations of an interface on a grid. The solid line represents a step function H, the shaded function is the color function C_j, and the dashed line is a smooth indicator function I. The dash–dot line is a level-set function whose zero contour marks the interface.

phase-field variable in Section 4.6 and f for the CIP marker function in Section 4.7.

Figure 4.1 shows a few different ways to identify the interface. The Heaviside function $H(x)$ is the discontinuous function shown by a solid line that changes abruptly from $H = 1$ to $H = 0$ in cell j, exactly where the interface is located. The color function C_j, the average value of the marker in each cell, is shown by the shaded area, and the smooth approximation $I(x)$ is shown by the dashed line. Notice that H and C are identical everywhere except in the interface cell j, but that I approaches a uniform value more slowly. The level-set function $F(x)$, defined such that $|F| = d$, where d is the distance from the interface, is shown by a dashed–dot line. While H, I, and F obviously vary with x, C_j is inherently identified with cell j. In the discussion below we will, however, sometimes talk about C as a continuous function approximated by the C_j volumes in each cell.

4.2 Advecting the color function

It is perhaps somewhat counterintuitive that the seemingly simple problem of pushing a piecewise-constant function around by a prescribed velocity field, using the linear first-order advection equation, should be one of the hard problems in computational science. The difficulty is even more surprising given the deceptive simplicity of the one-dimensional problem. Yet, the one-dimensional problem can serve as both an introduction to the difficulties and as a motivation for the various techniques that have been designed to overcome the difficulties. We shall therefore examine the one-dimensional problem in some detail here. To simplify our task, we assume that the flow is incompressible so that the velocity is constant,

$u = U > 0$. The advection of the color function is therefore governed by

$$\frac{\partial C}{\partial t} + \frac{\partial F}{\partial x} = 0, \tag{4.5}$$

where $F = UC$ is the flux function. As the interface moves to the right, C flows into cell j through the left boundary and out through the right boundary. For cell j we denote the left and the right boundaries by $j-1/2$ and $j+1/2$, respectively, and the fluxes by $F_{j-1/2}$ and $F_{j+1/2}$. If the value of C in cell j at time level n is denoted by C_j^n and the value at the end of a time step Δt by C_j^{n+1}, then we can write

$$C_j^{n+1} = C_j^n - \frac{1}{\Delta x}\int_t^{t+\Delta t} (F_{j+1/2} - F_{j-1/2})\, \mathrm{d}t. \tag{4.6}$$

Or, in words, the change in C_j is the difference between what flows in and what flows out of cell j. Here, $F_{j-1/2} = UC_{j-1/2}$ and $F_{j+1/2} = UC_{j+1/2}$, and the key challenge is to accurately estimate $C_{j-1/2}$ and $C_{j+1/2}$.

If we take C in each cell to be a constant, then the integration of the fluxes over time is simple. The value of C that crosses the $j-1/2$ boundary is simply the value in the cell to the left, C_{j-1}, and the value of C crossing the $j+1/2$ boundary is C_j. Thus, as long as we limit the size of each time step to $\Delta t U \leq \Delta x$, we can update C_j by

$$C_j^{n+1} = C_j^n - \frac{U\Delta t}{\Delta x}(C_j - C_{j-1}). \tag{4.7}$$

In Fig. 4.2 we show the advection of C using this scheme. The initial condition is a blob of constant C that should move to the right by a constant velocity without changing its shape. The exact solution is shown at time $t = 0$, $t = 140\Delta t$, and $t = 280\Delta t$, along with the solution advected using the scheme above (Equation (4.7) using 200 equal-sized control volumes for a domain of length 1, blob width of $20\Delta x$, and $\Delta t = 0.5\Delta x$). Obviously, scheme (4.7) does a terrible job. Even though the blob has only moved twice its length it has diffused excessively and it continues to diffuse as it moves further.

What went wrong? We used the *exact* fluxes, so the problem cannot be there. The culprit is the assumption that the correct way to compute the flux of C is to represent the distribution of C in each cell by the cell average C_j. Since C is supposed to approximate H we do not want any C to flow out of cell j until it is full. Thus, if cell j is initially half full, so the interface is in the middle, and we take a time step $\Delta t = U\Delta x/4$ so that the interface moves to the right boundary, then cell j should be full and cell $j+1$ should still be empty at the end of the step. The problem, therefore, is that we computed the fluxes by assuming that C is equal to its average, everywhere in each cell.

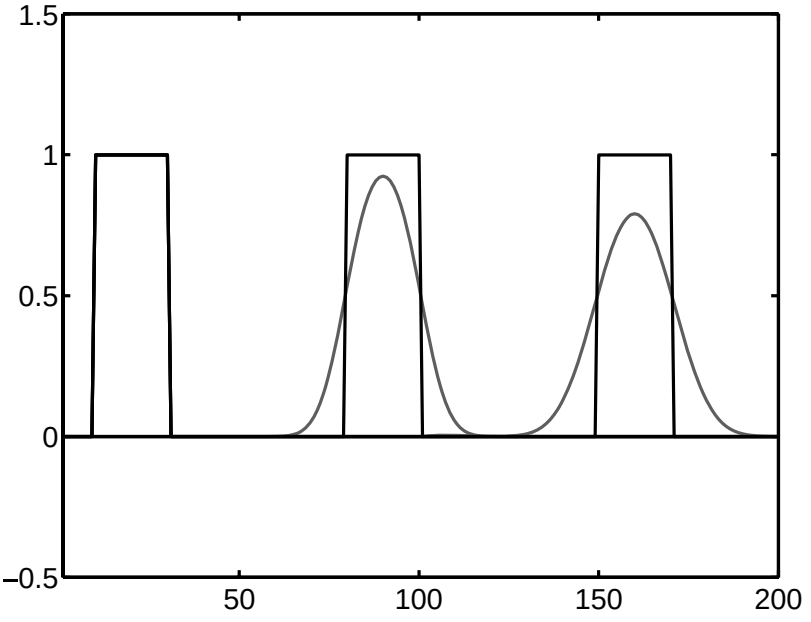


Fig. 4.2. Evolution of a slug of material using the first-order upwind method. The exact solution shows the slug moving unchanged downstream (black line) but the solution found using the upwind method (gray line) rapidly becomes excessively diffuse. The domain is resolved by 200 equal-sized control volumes for a domain of length 1, blob width of $20\Delta x$, and $\Delta t = 0.5\Delta x$. The solution is plotted at $t = 0$, $t = 140\Delta t$, and $t = 280\Delta t$.

The obvious solution to this smearing is to represent the distribution of the marker in each cell by a higher order function that recognizes that it is distributed over the cell in such a way that the left-hand side is full and the right-hand side is empty. If the distribution in each cell is given by a linear function with slope s_j, a little bit of geometry, using Fig. 4.3, shows that during time interval Δt the amount that flows into cell j is $[C^n_{j-1} + (\Delta x - U\Delta t)s_{j-1}/2]U\Delta t$, and the amount that flows out is $[C^n_j + (\Delta x - U\Delta t)s_j/2]U\Delta t$. Thus, the amount of C in cell j at time $n+1$ is given by

$$C^{n+1}_j = (1-\lambda)C^n_j + \lambda C^n_{j-1} - \frac{\Delta x}{2}(1-\lambda)\lambda(s_j - s_{j-1}) \tag{4.8}$$

where $\lambda = U\Delta t/\Delta x$. The slope in each cell can be estimated in several ways. Define the left and the right slopes for cell j by

$$s^+_j = \frac{C^n_{j+1} - C^n_j}{\Delta x} \quad \text{and} \quad s^-_j = \frac{C^n_j - C^n_{j-1}}{\Delta x}. \tag{4.9}$$

The weighted average is

$$s_j = \frac{1-\kappa}{2}s^-_j + \frac{1+\kappa}{2}s^+_j, \tag{4.10}$$

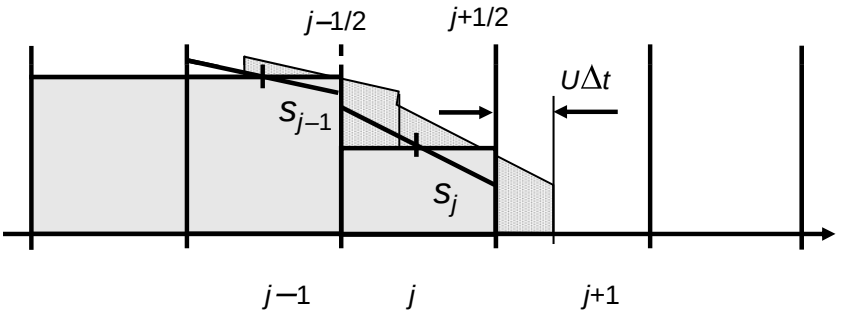


Fig. 4.3. The advection of a color function using a higher order scheme. The dark gray area is the initial solution taking the value in each cell equal to the average cell value. After the slopes in each cell have been constructed, using the values in adjacent cells, the fluxes can be computed exactly by geometric considerations. The light gray area shows the solution at the new time, before averaging.

where κ is an arbitrary constant. Different selections of κ give different schemes. $\kappa = 1$, for example, gives the Lax–Wendroff scheme.

The advection of a blob using Equation (4.8) is plotted in Fig. 4.4 for $\kappa = -1,\ 0,\ +1$, at the same times and using the same resolution and time step as in Fig. 4.2. The good news is that the results are not as diffusive as for the first-order upwind method (Equation (4.7) and Fig. 4.2), but the bad news is that the solutions oscillate around the interface for all values of κ. Scheme (4.8), therefore, does not really capture the sharp interface any better than the upwind scheme in Fig. 4.2. Unfortunately, using a higher order function, such as a parabola, only produces more oscillations.

This problem – the fact that using a first-order scheme to advect a discontinuity results in excessive diffusion and that higher order methods yield oscillations – has been the subject of extensive research by a large number of investigators. The remedies are all similar in nature. The solution is advected by a high-order scheme nearly everywhere, but around the discontinuity the solution is carefully modified to eliminate oscillations. This smoothing amounts to adding *artificial viscosity* and results in a formal reduction in the order of the scheme near the discontinuity. Artificial viscosity was originally introduced by von Neuman (see Richtmyer and Morton (1967)) to damp out oscillations around shocks for gas dynamics simulations, but the modern way of looking at artificial viscosity originated with the high-order Godunov method of van Leer (1979) and the flux-corrected transport method (Boris and Book, 1973). While modern shock-capturing methods have reached a high degree of sophistication and do, in particular, capture shocks in compressible flows very well, they are usually not entirely satisfactory for the advection of a passive marker function, particularly for a long time. Shocks in compressible flows

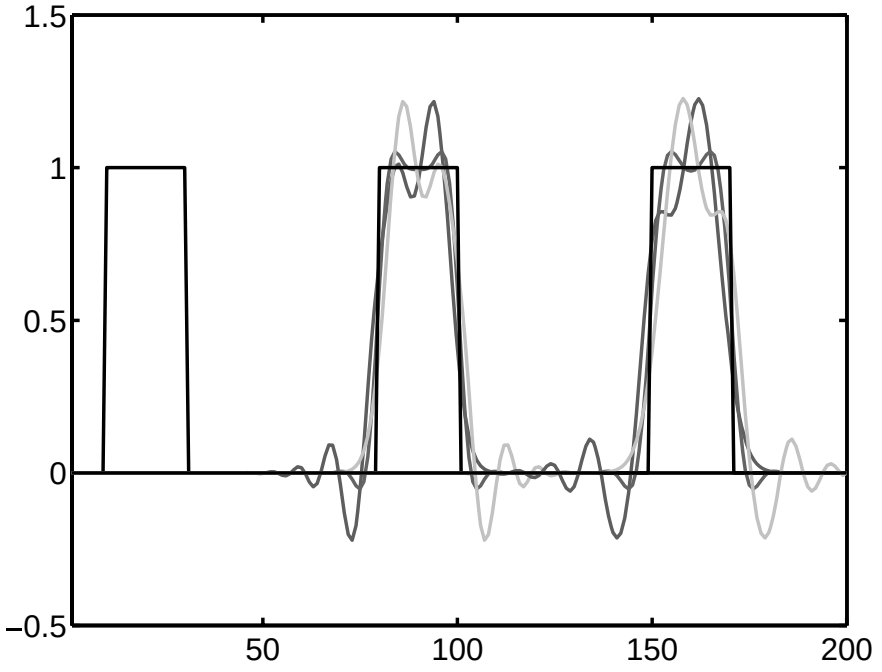


Fig. 4.4. Evolution of a slug of material using higher order methods. The exact solution is shown by a solid black line and the results by three different higher order methods are shown by lines of different shades of gray. The solution is shown at the same times as in Fig. 4.2 and all parameters are the same.

generally "want" to remain sharp[1] and slight smoothing of the shock is immediately repaired. For a passive marker there is no inherent mechanism that sharpens a smoothed interface and the flow does not "know" the difference between an absolutely sharp interface and a smoother one. Thus, shocks are generally captured better than contact discontinuities in simulations of compressible flows. For the advection of a marker function, advanced monotonicity-preserving methods for shock capturing work well for short times, but different approaches are needed for longer times.

4.3 The volume-of-fluid (VOF) method

While high-order monotonicity-preserving methods have been a great success for computations of flows with shocks, their success in solving Equation (4.5) is mixed at best – although we should acknowledge that, for relatively short times, the best of these methods do just fine. For the long time evolution of a marker function we

[1] Since characteristics flow into the shock.

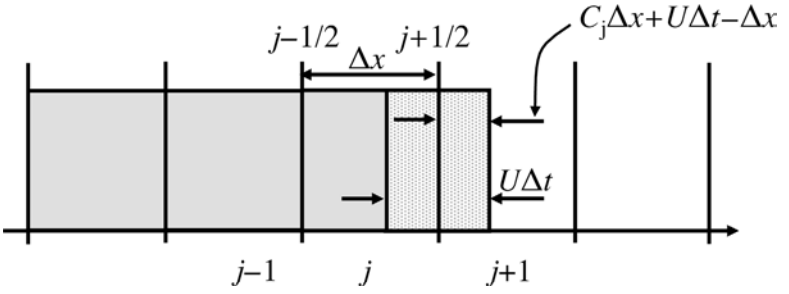


Fig. 4.5. One-dimensional advection by the VOF method. Given the value of the color function in the interface cell j, and the side where the full cell is, the location of the interface can be found and the fluxes computed exactly.

have to step back, take a fresh look at the problem of advecting the color function C, and seek a different way of doing so.

In one dimension, as depicted in Fig. 4.5, the answer is embarrassingly simple. C_j is either 0 or 1, except in an interface cell, so the value of C_j in the interface cell yields immediately the exact location of the interface. If $C_{j-1} = 1$, $C_j = 1/3$, and $C_{j+1} = 0$ then the interface is $\Delta x/3$ from the left boundary. If $C_j = 1/2$, then the interface is in the middle of the cell, and so forth. If $U \geq 0$, then the flux through the left boundary is always U and we can compute exactly the flux through the right boundary, since we know the exact location of the interface. If $U\Delta t < (1-C_j)\Delta x$, then the flux is zero; and if $U\Delta t > (1-C_j)\Delta x$, then the flux is first zero and then U after the interface reaches the right boundary. Thus,

$$\int_t^{t+\Delta t} F_{j+1/2}\,\mathrm{d}t = \begin{cases} 0, & \Delta t \leq (1-C_j)\Delta x/U; \\ (C_j-1)\Delta x + U\Delta t, & \Delta t > (1-C_j)\Delta x/U. \end{cases} \tag{4.11}$$

This expression was used to compute the exact motion of the blob in Figs. 4.2 and 4.4.

Thus, for one-dimensional problems the advection of the marker function is essentially trivial. Not so in higher dimensions! The first attempt to extend the advection described by Equation (4.11) to higher dimensions was the simple line interface calculation, or SLIC, method of Noh and Woodward (1976). In this approach the marker function is advected by time splitting, where we advect first in one coordinate direction and then in the other (assuming two-dimensional flow). For advection in the horizontal direction, an interface cell is divided by a vertical line into a full part and an empty part, with the decision of which side is empty and which is full depending on the volume fraction in the cells to the left and the

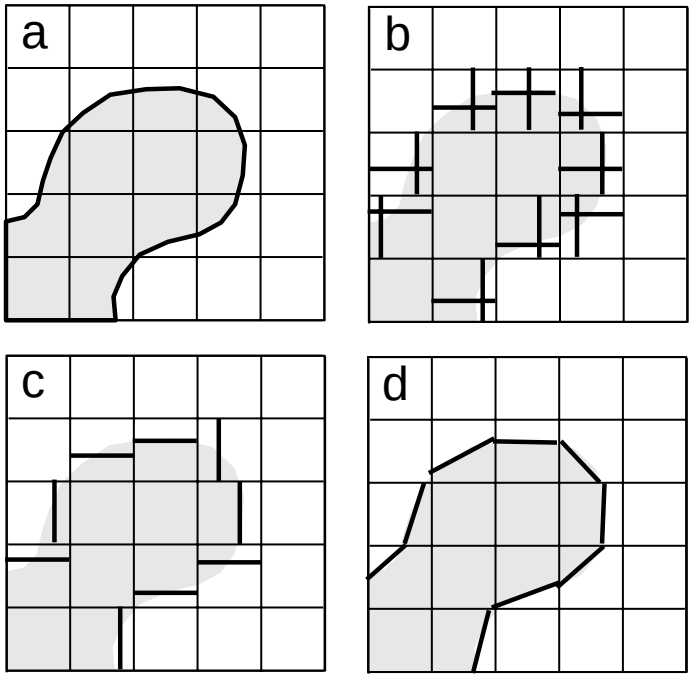


Fig. 4.6. VOF reconstruction of the solution for advection in two dimensions. (a) The original interface. (b) The original SLIC reconstruction. (c) The Hirt and Nichols reconstruction. (d) PLIC reconstruction. Figure adapted from Rudman (1997).

right. Once the location of the interface has been determined, the time integration of the fluxes is done using Equation (4.11). The interface is then advected in the vertical direction by dividing the cell by a horizontal line into a full and an empty part. In three dimensions this process obviously must also be repeated for the third coordinate direction. Hirt and Nichols (1981) proposed a slightly different method where the interface was still approximated by straight lines, parallel to the coordinate axis, but the same orientation was used for the advection in the different coordinate directions. To determine whether the interface should be horizontal or vertical, Hirt and Nichols found the normal to the interface, using values of C in the neighboring cells, and selected the orientation of the interface depending on whether the normal was more closely aligned with the horizontal or the vertical axis. Although perhaps more appealing than the original SLIC method, tests by Rudman (1997) suggest that the Hirt and Nichols method is not significantly more accurate. In addition to distorting the interfaces, both methods generally generate considerable amount of "floatsam" and "jetsam," where pieces of the interface break away in an unphysical way.

Although the method of Hirt and Nichols perhaps did not improve significantly on the SLIC approach, it nevertheless suggested that the key to improving the behavior of the advection scheme was the *reconstruction* of the interface in each cell, using the value of the marker function in each cell, along with the value in the neighboring cells. In the piecewise linear interface calculation (PLIC) method introduced by DeBar (1974) and Youngs (1982), the interface is approximated by a straight-line segment in each cell, but the line can be oriented arbitrarily with respect to the coordinate axis. The orientation of the line is determined by the normal to the interface, which is found by considering the value of C in both the cell under consideration and in the adjacent cells. Once the interface in each cell has been constructed, the fluxes from one cell to another are computed by geometric considerations. The result of the advection generally depends on the accuracy of the interface reconstruction; finding the normal accurately, therefore, becomes critical for PLIC methods. Several methods have been proposed to do so. Given C in each cell and the normals, the exact location of the interface can be determined. In two dimensions the line segment can cross any of two adjacent or opposite cell faces, so there are two basic interface configurations. In three dimensions there are many more possible configurations, adding considerably to the complexity of the method.

Figure 4.6, adapted from Rudman (1997), shows the main difference between the interface reconstruction using SLIC, the Hirt and Nichols VOF method, and PLIC. While the linear reconstruction captures slowly varying sections of the interface very well, it usually does less well for interfaces that are changing rapidly. Notice that the line segments are not continuous across cell boundaries.

In the next chapter we discuss modern VOF methods in detail, and later in the book we will show several examples of computations done using VOF methods.

4.4 Front tracking

Instead of advecting a marker function by reconstructing the location of the interface in a partially full cell, for one-dimensional problems we can, of course, simply use a marker point that is moved with the imposed velocity. If the interface is located at x_f, then the amount of C_j in the interface cell is given exactly by $C_j = (x_\mathrm{f} - x_{j-1/2})/\Delta x$. We can also construct a smoother approximations to the step function by setting the cell values as a function of the distance to the interface. In one dimension both approaches are equally straightforward. For two- and three-dimensional problems, we need to use many marker points that are connected to represent a curve (in two dimensions) or a surface (in three dimensions). Figure 4.7 shows schematically the representation of an interface using connected marker points, in two dimensions.

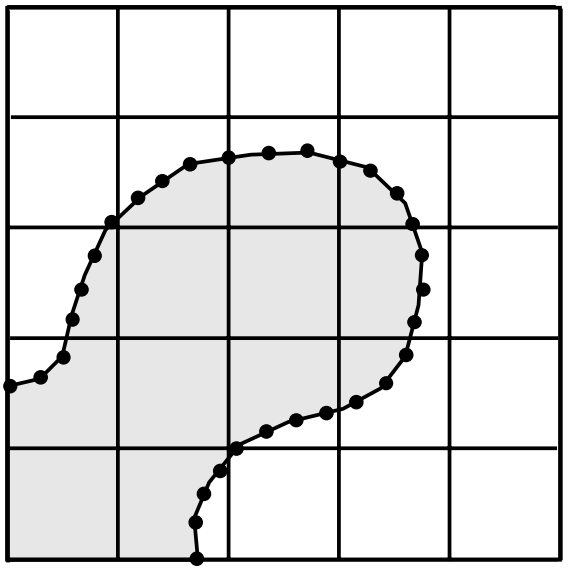


Fig. 4.7. When using a front-tracking method the fluid interface is represented by connected marker particles that are advected by the fluid velocity, interpolated from the fixed grid.

The use of connected marker points in simulations of the motion of a fluid interface in viscous fluids goes back to Daly (1969b), who used the points to compute surface tension in MAC simulations of the Rayleigh–Taylor instability. The use of marker points to track shocks was discussed by Richtmyer and Morton (1967), who appear to have introduced the term "front tracking," but they neither showed nor referenced any implementations. Glimm and collaborators (Glimm *et al.*, 1981; Glimm and McBryan, 1985; Chern *et al.*, 1986) developed the preliminary ideas of Richtmyer and Morton (1967) into a general methodology for flows with shocks and interfaces.

Connected marker points can be used to track the boundary between different fluids or phases in several different ways. In most cases, however, it is necessary to decide how the front is represented and managed as it stretches and deforms; how the interface is advanced in time; how the interface interacts with the underlying grid used to solve the equations governing the fluid flow; and how the topology of the interface is changed when fluid blobs merge and break apart. With the exception of two-dimensional flows, where any data structure can be made to work relatively easily, essentially all front-tracking methods use triangulated unstructured grids to represent the interface. This allows grid points to be added and deleted as the grid stretches and compresses. Unstructured triangulated grids have been

used extensively for finite element methods and for finite-volume computations of inviscid flows in aerospace applications. Extensive literature is therefore available for both the generation and management of such grids (see Thompson *et al.* (1998), for example). For multiphase flow simulations, where the conservation equations governing the fluid flow are solved on a fixed grid, the effort required to manage the front is usually small compared with the overall effort required for the simulation. Furthermore, the information carried by the grid is usually fairly simple and the demand on grid quality is not as high as when the triangulated grid is used to solve the full fluid equations. The ease of writing the front part of a simulation code does, however, depend sensitively on using the most appropriate data structure for the front.

The way in which the front interacts with the underlying fixed grid is what distinguishes between the various front-tracking methods. Several issues must be decided, including how information is transmitted between the front and the grid and how the update of variables next to the front is accomplished. The simplest approach is to take the front to represent a smooth transition between the different fluids. Since the interface is accounted for by singular source terms in the governing equations, this corresponds to approximating the singular functions by smooth distributions on the fixed grid. The origin of this approach can be found in several particle-in-cell methods for fluid flow and plasma simulations (Hockney and Eastwood, 1981; Birdsall and Langdon, 1985), the vortex-in-cell algorithm of Christiansen (1973), and the immersed-boundary method of Peskin (1977). Since the rôle of the front is exclusively the advection of the marker function, this approach is very much like a front-capturing method with a perfect (or nearly perfect) advection scheme. While information must be passed between the front and the grid, the smoothing of the interface alleviates the need for any modification of the solution method for the fluid equations near the interface (except that the solver must be written for material properties that vary spatially). For finite-Reynolds-number multiphase flows, Unverdi and Tryggvason (1992) developed a method where the front is used both to update the marker function and to include surface tension. Their method has been used to study a wide variety of multiphase flows, and we describe it in more detail in Chapter 6.

To avoid smoothing the interface, but still use a fixed grid for the solution of the conservation equations, it is necessary to modify the numerical approximations near the front. In Glimm's front-tracking method, the field variables at the front are extrapolated to grid points on the other side of the front, allowing the use of regular finite-difference discretizations for grid points next to an interface. A similar approach is used in the ghost-fluid method of Fedkiw *et al.* (1999), although the interface tracking is done using a level-set method instead of connected marker particles. As discussed by Glimm *et al.* (2001), the extrapolation can be done in

several different ways, but in all cases the contribution from the ghost points results in the equivalent of source terms in the discrete governing equations. This is also the case in the immersed-interface method of Lee and LeVeque (2003), where the jump conditions at the front are incorporated directly into the numerical approximations for gradients evaluated near the front, on the fixed grid.

Another approach to keep the interface sharp is the modification of the fixed grid near the front in such a way that the grid lines coincide with the interface. This generally leads to irregularly shaped control volumes near the interface and often involves merging some of the control volumes to eliminate very small ones. The origin of this approach is the "cut cell" methods initially developed for aerospace applications (see Powell (1998), for example) but its application to multiphase flow simulations was originated by Udaykumar *et al.* (1997). See also Udaykumar *et al.* (1999, 2001), Liu *et al.* (2005), and Marella *et al.* (2005). Since the jump conditions can be imposed directly at the cell boundary, this method is, in many ways, similar to using two separate grids for each phase. We do, however, mention it here as most of the grid is left unchanged.

One of the main objections to the use of front-tracking methods is that changes in topology, where fluid regions merge or break up, are not handled automatically, as in methods where the marker function is advected directly. Changing the connection of the front points can obviously be accomplished, but at the cost of increased code complexity. When two interfaces come close together, the film between them can sometimes become very thin and rupture. However, it may or may not be appropriate to fuse interfaces together when the film is of the order of one grid spacing. In front tracking the interfaces never fuse together unless something special is done, and sometimes the added level of control provided by the tracking is desirable. The question of how interfaces merge is still not a completely resolved issue, as discussed in Chapter 2, and we will return to how to handle it computationally at the end of Chapter 6.

4.5 The level-set method

In the level-set method the different fluid regions are identified by a smooth marker function $F(\mathbf{x},t)$, which is positive in one fluid and negative in the other. The boundary between the fluids is identified by the $F(\mathbf{x},t)=0$ level curve. The level-set function moves with the fluid and, therefore, evolves according to

$$\frac{\partial F}{\partial t}+\mathbf{u}\cdot\nabla F=0. \tag{4.12}$$

The motion of the zero level-set curve depends only on the normal velocity component. The normal can be found by

$$\mathbf{n} = -\frac{\nabla F}{|\nabla F|}. \tag{4.13}$$

Substituting for ∇F in Equation (4.12) and using that $\mathbf{u}\cdot\mathbf{n} = V_n$ results in

$$\frac{\partial F}{\partial t} - V|\nabla F| = 0, \tag{4.14}$$

which is an alternate expression for the advection of F.

Unlike the C function in the VOF method, the level-set marker function is smooth and it can, therefore (at least in principle), be advected using any standard method for hyperbolic equations. Most authors, however, have used high-order schemes, such as the ENO method of Osher and Shu (1991).

To reconstruct the material properties of the flow (density and viscosity, for example), a marker function is constructed from F

$$I(F) = \begin{cases} 0, & \text{if } F < -\alpha\Delta x; \\ \frac{1}{2}(1 + (F/\alpha\Delta x) + \frac{1}{\pi}\sin(\pi F/\alpha\Delta x)), & \text{if } |F| \le \alpha\Delta x; \\ 1, & \text{if } F > \alpha\Delta x. \end{cases} \tag{4.15}$$

Here, Δx is the size of a grid cell and α is an empirical coefficient, often taken to be equal to three, giving an interface thickness of about six cells. Thus, I changes from zero to one over only a few cells, describing a smooth transition zone from one fluid to the next. Once I has been constructed, the various material properties can be assigned. Given $I(\mathbf{x})$, we can generate a smoothed grid delta function at the interface by taking its gradient:

$$\delta = \nabla I = \frac{\mathrm{d}I}{\mathrm{d}F}\nabla F. \tag{4.16}$$

The mapping in Equation (4.15) requires the use of the values of F off the $F = 0$ contour (or level set). For the thickness of the transition zone to remain approximately constant, F must have the same shape near the interface for all times. If F is simply advected by the fluid, this is not usually the case. Where the interface is stretched, the gradient of F becomes steeper, and where the interface is compressed the gradient becomes smaller. To deal with this problem, Sussman *et al.* (1994) introduced a reinitialization procedure where they modified F by solving

$$\frac{\partial F}{\partial \tau} + \mathrm{sgn}(F_0)(|\nabla F| - 1) = 0 \tag{4.17}$$

at each time step. Here, sgn is the sign function, F_0 is the un-reinitialized level-set function, and τ is a pseudo-time. Equation (4.17) is integrated to steady state, thus enforcing $|\nabla F| = 1$. This makes F a distance function and ensures that the slope

near the interface is always the same. For slowly moving interfaces the level-set function only needs to be reinitialized every once in a while, but for more rapidly moving interfaces it may have to be reinitialized at every time step. The reinitialization can result in an artificial motion of the interface which can contribute to poor mass conservation. Several improvements have therefore been proposed, including approximating the sgn function using a smoother function. Peng *et al.* (1999) proposed replacing $\mathrm{sgn}(F_0)$ in Equation (4.17) by

$$S(F) = \frac{F}{\sqrt{F^2 + |\nabla F|^2 \Delta x^2}} \tag{4.18}$$

and applying (4.17) at every time step. As long as the level-set function takes large positive or negative values far away from the interface, the reconstruction only needs to be done near the $F = 0$ contour. Since the solution to Equation (4.17) propagates outward from the interface, where F is given, the solution is first corrected there and it is usually sufficient to take only a few steps in pseudo-time. The reconstruction of the level-set function as a distance function was critical in making level sets work for fluid dynamics simulations. As Osher and Fedkiw (2001) point out, reinitialization is still an active area of research.

For fluid dynamics problems the F function is generally advected by the fluid velocity. In other applications, such as the motion of a solidification front, the velocity is only defined at the interface and must be extended off the interface to advect the level-set function. The extension of the velocities off the interface has been the subject of several papers, including Chen *et al.* (1997b), who showed that the velocity field known at the interface could be extended into the region around the interface by using an upwind method to solve

$$\frac{\partial S}{\partial \tau} + S(F)\mathbf{n} \cdot \nabla S = 0 \tag{4.19}$$

for each velocity component (taking $S = u$, v, or w). This process results in a velocity field that is constant in a direction normal to the interface. As for the reinitalization of the level-set function as a distance function (Equation (4.17)), the velocity field only needs to be constructed in the neighborhood of the $F = 0$ contour.

The level-set method, at least in its original embodiment, has the appeal of simplicity. For a fluid mechanics problem one only has to solve one additional partial differential equation and there are essentially no additional complex steps, such as the reconstruction of an interface in the VOF method or the addition of new computational objects such as when front tracking is used. This simplicity has, however, come at some cost, and early implementations had considerable problems with mass conservation. While many of the early difficulties have been overcome,

efforts to generate more accurate methods generally result in added complexity. This has eroded the main appeal of level-set methods and made them more comparable to other approaches, both in terms of complexity and performance.

For further discussions of the level-set method, the reader is referred to Sussman *et al.* (1998), who proposed a way to improve the mass conservation, to Sussman and Puckett (2000), who introduced a hybrid VOF/level-set method, sometimes known by its acronym CLSVOF, and to Fedkiw *et al.* (1999), who developed a "ghost fluid" method based on assigning fictious values to grid points on the other side of a fluid discontinuity. For general reviews of the level-set method, see Osher and Fedkiw (2002) and Sethian (2001).

4.6 Phase-field methods

In the phase-field method, the interface is kept relatively sharp by a modification of the governing equations. The interface is assumed to be of a finite thickness and described by thermodynamically consistent conservation laws. In actual implementations, however, the thickness of the transition is much larger than it is in real systems and it is not clear whether keeping the thermodynamic conditions correct in an artificially thick interface has any advantages over methods that model the behavior in the transition zone in other ways. The phase-field approach has found widespread use in simulation of solidification, but its use for fluid dynamic simulations is more limited.

To update the phase function c, which identifies the different fluids, a nonlinear diffusion term is added to the advection equation, resulting in the so-called Cahn–Hilliard equation with advection. The diffusion term smears out an interface that is becoming thinner due to straining but an antidiffusive part prevents the interface from becoming too thick, such as if the interface is compressed. The Navier–Stokes equations are also modified by adding a term that results in surface tension in the interface zone. The key to the modification is the introduction of a free-energy function, and by selecting the function in the proper way – including the appropriate values for the various adjustable coefficients – it is possible to ensure that the thickness of the interface remains of the same order as the grid spacing.

For a two-fluid system where the two fluids have similar densities (so the Boussinesq approximation can be used) and same viscosity and mass diffusion coefficient, Jacqmin (1999) solved

$$\rho_0 \frac{D\mathbf{u}}{Dt} = -\nabla S + \mu \nabla^2 \mathbf{u} - c\nabla\phi + \mathbf{g}\rho(c) \tag{4.20}$$

for the fluid velocity and updated the phase-field variable by

$$\frac{Dc}{Dt} = \kappa \nabla^2 \phi. \tag{4.21}$$

Here, κ is a diffusion coefficient and the potential ϕ is derived from the free energy of the fluid by

$$\phi = \beta \frac{\mathrm{d}\psi(c)}{\mathrm{d}c} - \alpha \nabla^2 c, \tag{4.22}$$

where $\psi(c) = (c+1/2)^2(c-1/2)^2$. The coefficients α and β describe the relative importance of the "gradient" energy and bulk energy density. It can be shown that the thickness of the phase boundary is $O(\alpha/\beta)$ and surface tension is proportional to $\sqrt{\alpha\beta}$. S is a pressure-like variable used to enforce incompressibility. For any given grid spacing Δx, we must select the appropriate diffusion coefficient κ as well as α and β. For the ψ used above, Jacqmin (1999) showed that the surface tension is $\sigma = \sqrt{\alpha\beta/18}$ and that the interface thickness ε, defined as the 90% variation of c, is given by $\varepsilon \approx 4.164\sqrt{\alpha/\beta}$. Thus, if surface tension and interface thickness are given (and we take ε to be equal to two or three grid spacings Δx), then the coefficients are determined. Jacqmin also showed that the diffusion coefficient for the phase-field variable c must be selected such that $\kappa = O(\varepsilon^\delta)$, where $\delta < 2$. To advect c, Jacqmin introduced a fourth-order method that allowed the interface thickness to be two or three grid spacings.

Jacqmin (1999) analyzed the method in some detail and showed examples of computations of a two-dimensional Rayleigh–Taylor instability in the Boussinesq limit. Other applications of the phase-field method to two-fluid simulations include Jamet *et al.* (2001), who focused on phase-change problems, Verschueren *et al.* (2001), who examined thermocapillary flow instabilities in a Hele–Shaw cell, and Jacqmin (2000), who studied the contact-line dynamics of a diffuse fluid interface. More recent progress can be found in Yang *et al.* (2006a) and Ding *et al.* (2007), for example.

Results for two-fluid problems produced by the phase-field method suggest that it is comparable to other methods that use a fixed grid to solve the one-field formulation of the governing equations. The smoothing of the transition zone and the use of front capturing (rather than explicit tracking) is likely to make the phase-field method similar to the level-set approach. However, it is important to note that the methods are derived based on fundamentally different assumptions and that the phase-field method can, at least in principle, be used to study small-scale phenomena, such as contact-line motion, for which tracking methods that start from the sharp interface hypothesis are unsuitable.

4.7 The CIP method

The CIP method is designed to reduce dispersive error in the advection of a function f by fitting a cubic polynomial to the nodal values of f and its derivatives.

While the initials CIP have stayed constant since the introduction of the method (Takewaki *et al.*, 1985; Takewaki and Yabe, 1987), the name of the method has evolved. CIP initially stood for *cubic interpolated pseudo-particle*, then for *cubic interpolated propagation* (Nakamura and Yabe, 1999), and most recently for the *constrained interpolation profile* method (Yabe *et al.*, 2001).

The fundamental idea of the CIP method is that, in addition to solving an advection equation for a conserved marker function f,

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0, \tag{4.23}$$

we also solve an advection equation for the derivative of f, $g = \partial f/\partial x$. This advection equation is easily constructed by differentiating Equation (4.23):

$$\frac{\partial g}{\partial t} + u\frac{\partial g}{\partial x} = 0. \tag{4.24}$$

Here, we have assumed that u is a constant, so the right-hand side of both equations is zero. If the velocity field is not constant, the advection equation for g can be split into two parts that are treated sequentially. In the first part we solve the advection part, assuming that the right-hand side is zero, and in the second part we add the effect of the right-hand side. The discussion below applies to the first part. Equations (4.23) and (4.24) state that f and its derivative g are advected without change of shape. Thus, the solution at time t is simply the solution at $t-\Delta t$.

To advect f and g, we introduce a cubic polynomial, $P(x) = ax^3 + bx^2 + cx + d$ and determine the coefficients in such a way that P matches f and g at grid points j and $j-1$ (for $u > 0$, when $u < 0$ we use points j and $j+1$). Taking $x_{j-1} = 0$ and $x_j = \Delta x$, we easily find that

$$\begin{aligned} a_j &= \frac{2}{\Delta x^3}\left(f_{j-1} - f_j\right) + \frac{1}{\Delta x^2}\left(g_{j-1} + g_j\right), \\ b_j &= \frac{3}{\Delta x^2}\left(f_j - f_{j-1}\right) - \frac{1}{\Delta x}\left(g_j + 2g_{j-1}\right), \\ c_j &= g_{j-1}, \\ d_j &= f_{j-1}. \end{aligned} \tag{4.25}$$

The exact solution of Equations (4.23) and (4.24) can be found by simply translating the profiles, so that $f(t,x) = f(t-\Delta t, x-u\Delta t)$ and $g(t,x) = \partial f(t-\Delta t, x-u\Delta t)/\partial x$. Therefore, if $\xi = u\Delta t$, the values of f_j^{n+1} and g_j^{n+1} are

$$\begin{aligned} f_j^{n+1} &= a_j\xi^3 + b_j\xi^2 + g_{j-1}^n\xi + f_{j-1}^n, \\ g_j^{n+1} &= 3a_j\xi^2 + 2b_j\xi + g_{j-1}^n. \end{aligned} \tag{4.26}$$

Figure 4.8 shows the advection of a square hump by the CIP method described above. The grid used, the initial conditions, and the time step are the same as in

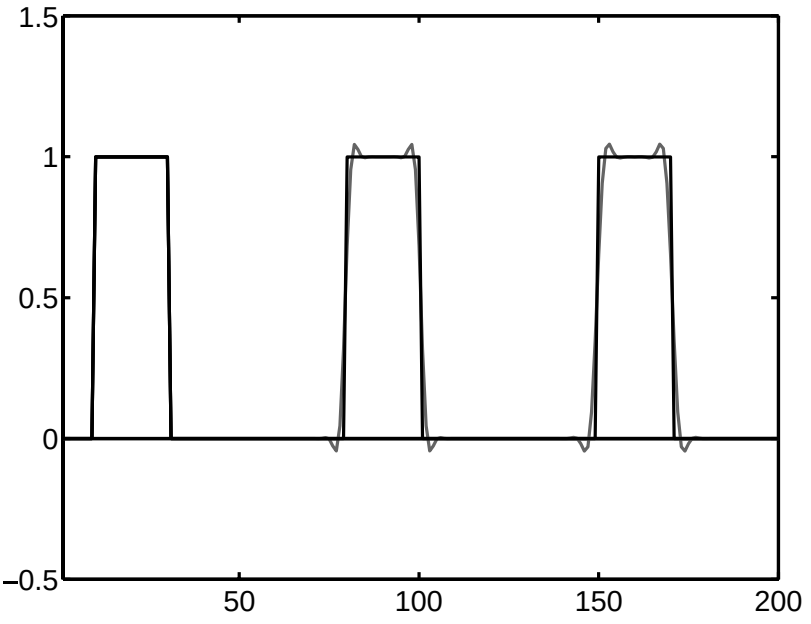


Fig. 4.8. Advection by the CIP method. All parameters are the same as in Fig. 4.2.

Figs. 4.2 and 4.4. To generate the initial conditions for g, we took the derivative of the initial f using centered finite differences. Although the discontinuity is preserved very well, small oscillations still appear. We note that the original CIP method was not invented specifically for discontinuous solutions and that its remarkably good performance in Fig. 4.8 is due to the very low dispersive error of the method.

For multidimensional flows it is, of course, possible to use time splitting and do the advection separately in each direction. More advanced implementations, however, introduce a multidimensional polynomial that is then shifted with the local velocity. For a discussion of the CIP method, see Yabe *et al.* (2001).

4.8 Summary

The use of a single set of governing equations to describe the flow of two or more fluids separated by a common interface requires the accurate tracking of the interface. This is usually done by advecting a marker function which takes one value in one fluid and another value in the other fluid. Advecting the marker function in such a way that it remains sharp at the interface is one of the most challenging problems of modern computational fluid dynamics. Several methods have been developed to do this, and here we have given a brief overview of some of the

main alternatives. In the remainder of the book we will focus on two of those, the VOF method and the front-tracking method of Unverdi and Tryggvason (1992). The VOF method is the most widely used method to directly advect the marker function. While early implementations may not have been able to capture interfaces accurately, major improvements have been made and these methods are now capable of producing solutions comparable to the best of the more recent alternatives. Although some of the alternatives, such as level sets, have gained popularity because of their apparent simplicity, efforts to increase their accuracy generally result in added complexity, thus reducing their appeal when compared with the VOF method. The most advanced methods to directly advect a marker function on a fixed grid are currently capable of producing solutions of a quality comparable to early implementations of front-tracking methods. However, the accuracy of front-tracking methods can also be improved, and it is probably fair to say that, for comparable computational effort, the use of interface markers will always yield superior results. Such methods are, however, more complex, and it is likely that methods where the marker function is advected on a fixed grid will remain more popular for the foreseeable future.

5

The volume-of-fluid method

In Chapter 4 we saw several families of methods for locating and advecting the interface. Here, we focus on one of them: the VOF method. Historically, it was used for free surface flows with only one "fluid," although it is now routinely used for two-fluid flows. In the VOF method the marker function is represented by the fraction of a computational grid cell which is occupied by the fluid assumed to be the reference phase.

A very large number (probably dozens) of VOF methods have been proposed. When we choose the method we try to strike a balance between several qualities: accuracy, simplicity and volume conservation.

5.1 Basic properties

The volume fraction or color function C is the discrete version of the characteristic function H; see Equation (4.3). We will be considering only two-phase or free-surface flows, so that the C data represent the fraction of each grid cell occupied by the reference phase. Furthermore, we restrict our analysis to Cartesian grids with square cells of side $h = \Delta x = \Delta y$.

The function C varies between the constant value one in full cells to zero in empty cells, while mixed cells with an intermediate value of C define the transition region where the interface is localized.

Low-order VOF methods do not need to specify the location of the interface in the transition region, but a geometrical interpretation of these methods shows that in two dimensions the interface line in each mixed cell is represented by a segment parallel to one of the two coordinate axes. The interface is clearly not continuous across the cell boundary, and the jump usually is of order h, $\mathcal{O}(h)$, as seen in Fig. 5.1b. Higher order methods reconstruct the interface in various ways. The standard one is the PLIC reconstruction, where the interface in each mixed cell is represented by a segment perpendicular to the local gradient, $\mathbf{m} = -\nabla C$, of

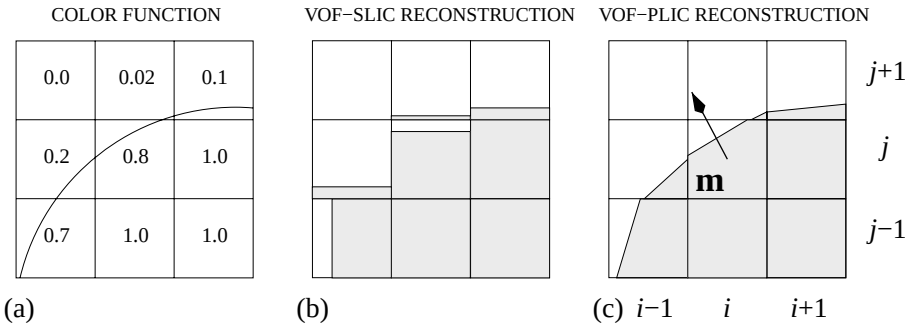


Fig. 5.1. The basic principle of the VOF methods: (a) a portion of the interface line and the color function value in each cell; (b) the SLIC reconstruction, where each segment is parallel to one coordinate axis; (c) a PLIC reconstruction with unconnected segments across each cell; the normal vector $\mathbf{m} = -\nabla C$ is pointing outwards from the reference phase.

the scalar function C (as in the previous chapters, we are assuming that the normal vector is pointing outwards from the reference phase; the reader should be aware that in many articles the opposite is true). However, there is still no requirement that the interface be continuous at the cell boundary, but now the interface discontinuity is usually much smaller and in general it is a function of the grid spacing h and of the local interface curvature κ.

A VOF method proceeds in two steps:

(i) Reconstruction of the interface shape: from the knowledge of the volume fraction in each cell, one has to build an approximation of the interface. The problem is illustrated in Fig. 5.2. The key issue for a PLIC reconstruction is to find the local normal vector.

(ii) Advection of the reconstructed interface in a given velocity field. This amounts to exchanging reference phase volumes across the boundary of neighboring cells.

We describe these two steps in the following sections. Several qualities are expected from a VOF method. It preserves mass in a natural way provided the advection method is adequate. Topology changes, such as those occuring during reconnection or breakup, are implicit in the formulation. The extension from two-dimensional to three-dimensional Cartesian geometry is relatively straightforward (although less so than in a level-set method). The C data structure is a static matrix on a fixed grid and does not require any dynamical adjustment, as opposed to a marker approach. Furthermore, the algorithms are local, in the sense that only the C values of the neighboring cells are needed to update the C value in a given cell.

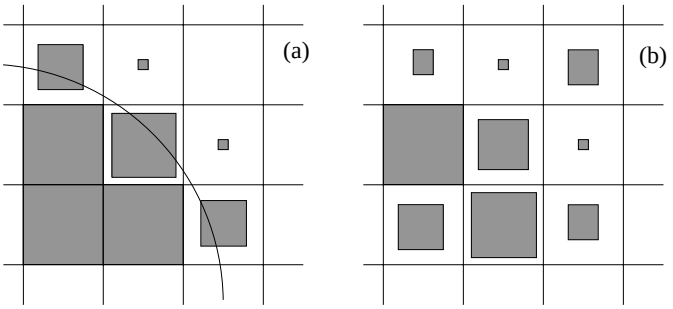


Fig. 5.2. Representing the color function as shaded squares of size proportional to the fractional volumes helps to understand the reconstruction problem. (a) A well-behaved distribution of the color function; it corresponds to a smooth circular arc. (b) A more confused distribution.

For this reason, it is relatively simple to implement these algorithms in parallel, in the framework of domain decomposition techniques.

Of all these qualities, the mass conservation is specific to the VOF technique. Let us consider the multidimensional version of the advection equation for the marker function H for an incompressible flow

$$\frac{\partial H}{\partial t} + \nabla \cdot (\mathbf{u} H) = 0 \,. \tag{5.1}$$

We first integrate this equation over the square cell (i, j) of side h of a Cartesian two-dimensional grid and use the definition (4.3) of the color function C:

$$h^2 \frac{\partial C_{i,j}(t)}{\partial t} + \int_\Gamma \mathbf{u} \cdot \mathbf{n} H(\mathbf{x}, t) \, \mathrm{d}l = 0 \,, \tag{5.2}$$

where Γ is the cell boundary line and $\mathbf{n}$ the outgoing unit normal. Finally, we integrate (5.2) in the time $\Delta t = t^{n+1} - t^n$ to get

$$h^2 (C_{i,j}^{n+1} - C_{i,j}^n) = -(\Phi_{x:i+1/2,j}^n - \Phi_{x:i-1/2,j}^n) - (\Phi_{y:i,j+1/2}^n - \Phi_{y:i,j-1/2}^n) \,, \tag{5.3}$$

where the flux $\Phi_{x:i+1/2,j}^n$ denotes the reference phase area crossing the right side of the cell in the time Δt. By summing Equation (5.3) over all the grid cells with appropriate boundary conditions, the internal fluxes cancel out in pairs and we get

$$\sum_{ij} C_{i,j}^{n+1} = \sum_{ij} C_{i,j}^n, \tag{5.4}$$

which leads to conservation of the total area. Notice that in writing expression (5.3) we have implicitly assumed that the numerical algorithm does not generate unphysical overshoots, $C > 1$, or undershoots, $C < 0$, in the volume fraction and that it does not flux some area twice across the cell boundary. We will come back to these issues later on in the chapter.

5.2 Interface reconstruction

For PLIC methods the reconstruction is basically a two-step procedure. In any given cell the normal $\mathbf{m}$ (the notation $\mathbf{n}$ is used for the unit normal, i.e. $\mathbf{n} = \mathbf{m}/|\mathbf{m}|$) is first determined from the knowledge of the color function in this cell and in the neighboring ones. The equation of the interface segment is then written as

$$\mathbf{m} \cdot \mathbf{x} = m_x x + m_y y = \alpha. \tag{5.5}$$

Geometrically, the interface line (5.5) is moved along the normal direction, i.e. the parameter α is adjusted, until the area under the interface equals $h^2 C_{i,j}$.

5.2.1 Convergence order of a reconstruction method

The numerical estimate of the integral of a given function $f(x)$ geometrically requires the evaluation of the area comprised between the graph of the function and the coordinate x-axis. When we reconstruct an interface from the volume fraction data we are actually solving the inverse problem: we know the area under the interface line and we want to find the function.

To integrate a function $y = f(x)$ we can subdivide the range $a \leq x \leq b$ into n equal subintervals of size $h = \Delta x = (b-a)/n$. In particular, in the trapezoidal rule we approximate the function with a piecewise linear interpolation and in each subinterval the area under the function is given by the area of a right trapezoid, as shown in Fig. 5.3. The truncation error in the evaluation of the area spanned by the function, which is supposed to have continuous second derivatives, is proportional to $f''(\xi)h^2$, with $a \leq \xi \leq b$. Hence, the trapezoidal rule is a second-order method, $\mathcal{O}(h^2)$, and a straight line, by having a zero second derivative, is reproduced correctly. In Fig. 5.3 we see that the line arc is approximated differently by the trapezoidal rule and a typical VOF-PLIC reconstruction. However, it is reasonable to require that a second-order reconstruction algorithm should have a similar truncation error. Reconstruction techniques that do not reproduce an arbitrary straight line exactly are lower than second order, i.e. they are $\mathcal{O}(h^n)$ with $n < 2$. A method that approximates interfaces as circle arcs could be third order, which is very high for a VOF method. In our opinion it is not practical to attempt to develop VOF methods with polynomials of such an order, in particular in three dimensions. However, partial use of second-order polynomials has proven useful, in part in connection with surface tension (see Section 7.4.2).

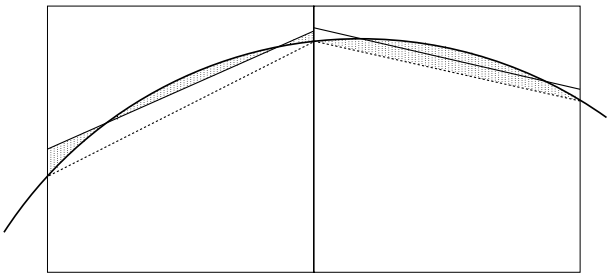


Fig. 5.3. A line cutting two grid cells (thick solid line). In the trapezoidal rule the piecewise linear approximation connects consecutive points on the line and it is continuous across the cell boundary (dashed lines). The VOF-PLIC reconstruction is not continuous across the cell boundary but satisfies the area conservation constraint (thin solid lines). The dotted regions represent the area error for a VOF-PLIC reconstruction (left) and the trapezoidal rule (right).

5.2.2 Evaluation of the interface unit normal

The choice of the method used to calculate the interface normal is independent of the other steps: it can be based on a finite-difference approximation of the volume-fraction gradient ∇C or satisfy some other minimizing criteria. All the algorithms described in this section use a 3×3 block of cells to determine the approximate interface in the central cell of the block. In Fig. 5.1c this central cell is denoted by the two indices (i, j).

5.2.2.1 Youngs' finite-difference method

In this method, developed by Youngs (1984) and independently by Li (1995), the normal $\mathbf{m}$ of (5.5) is estimated as a gradient,

$$\mathbf{m} = -\nabla_h C\,, \tag{5.6}$$

with finite differences. We first evaluate the normal vector $\mathbf{m}$ at the four corners of the central cell (i, j); for example, the components of $\mathbf{m}$ on the top-right corner are given by

$$m_{x;i+1/2,j+1/2} = -\frac{1}{2h}(C_{i+1,j+1} + C_{i+1,j} - C_{i,j+1} - C_{i,j})\,, \tag{5.7}$$

$$m_{y;i+1/2,j+1/2} = -\frac{1}{2h}(C_{i+1,j+1} - C_{i+1,j} + C_{i,j+1} - C_{i,j})\,, \tag{5.8}$$

and similarly for the other three corners. The cell-centered vector is finally obtained by averaging the four cell-corner values

$$\mathbf{m}_{i,j} = \frac{1}{4}(\mathbf{m}_{i+1/2,j+1/2} + \mathbf{m}_{i+1/2,j-1/2} + \mathbf{m}_{i-1/2,j+1/2} + \mathbf{m}_{i-1/2,j-1/2})\,. \tag{5.9}$$

This finite-difference scheme does not reproduce any straight line exactly. This may be shown by working out the expressions (5.7), (5.8), and (5.9) in a particular case, for instance by considering the line $y = 2x/3 + h$ in the 3×3 block of cells of side h with the origin in the bottom-left corner. The calculations are left to the reader. However, the results of numerical tests on linear interfaces will point out this weakness of the method.

5.2.2.2 Centered-columns difference method

In the same block of cells the volume fractions can be added column-wise along the vertical direction to define the height function $y = f(x)$, or row-wise for the width function $x = g(y)$. For example, the height y_{i-1} at the abscissa x_{i-1}, placed in the center of the column as shown in Fig. 5.4a, is given by the expression $hy_{i-1} = h^2 \sum_{k=-1}^{1} C_{i-1,j+k}$, while in Fig. 5.4b the width x_{j+1} at the ordinate y_{j+1} is $hx_{j+1} = h^2 \sum_{k=-1}^{1} C_{i+k,j+1}$. The height y_{i-1} is placed exactly on a linear interface only if the straight line cuts the two vertical sides of the column (see again Fig. 5.4a,c for two different cases). We approximate the height function $y = f(x)$ in the central cell of the block with the linear equation

$$\mathrm{sgn}(m_y)\, y = -m_x x + \alpha' \tag{5.10}$$

and compute the slope of the straight line with a centered scheme, $m_x = m_{\mathrm{xc}}$,

$$m_{\mathrm{xc}} = -\frac{1}{2h}(y_{i+1} - y_{i-1}) = -\frac{1}{2}\sum_{k=-1}^{1}(C_{i+1,j+k} - C_{i-1,j+k})\,. \tag{5.11}$$

Since $m_y = -\partial C/\partial y$, we compute the sign of the variation of C along the y-direction with centered finite differences. We must calculate explicitly the sign of m_y, because by adding C column-wise we lose information about which phase is on the top or on the bottom of the block of cells.

We can also describe the interface line with the width function $x = g(y)$ and approximate it linearly with

$$\mathrm{sgn}(m_x)\, x = -m_y y + \alpha'' \tag{5.12}$$

and similarly compute m_y as

$$m_{\mathrm{yc}} = -\frac{1}{2h}(x_{j+1} - x_{j-1}) = -\frac{1}{2}\sum_{k=-1}^{1}(C_{i+k,j+1} - C_{i+k,j-1})\,. \tag{5.13}$$

It is evident from Fig. 5.4 that when a linear interface cuts two opposite sides of the block of cells, one of the two representations gives the correct slope, m_{xc} in case (a) and m_{yc} in case (b). In case (c), where the interface cuts two consecutive sides, the centered-columns scheme does not compute the correct slope. As a matter of fact, a wider stencil with more columns or rows should be used; however, backward and

forward difference schemes can also be considered, as discussed in the next section. Since the centered-columns scheme cannot reconstruct any straight line exactly, it is not second order. Furthermore, the interface shape is not actually known, so we need a strategy to select one between the two numerically computed values m_{yc} and m_{xc}. To this aim we define a simple criterion based on the reconstruction of a linear interface. We consider the two linear equations $y = m_x x + \alpha'$ and $x = m_y y + \alpha''$ and notice the following conditions between the two coefficients m_x and m_y: $|m_x| = |m_y| = 1$ for a straight line with a 45-degree slope, $|m_x| = 1/|m_y| < 1$ for the line shown in Fig. 5.4a and $|m_y| = 1/|m_x| < 1$ in Fig. 5.4b. Therefore, to reconstruct exactly the straight line in the two cases (a) and (b), we must select the angular coefficient with the minimum absolute value

$$|m^*| = \min(|m_{xc}|, |m_{yc}|)\,. \tag{5.14}$$

In the centered-columns method, we have considered a centered scheme to compute the angular coefficient, but one can also use forward or backward finite differences, as in the efficient least-squares VOF interface reconstruction algorithm (ELVIRA) method. A different criterion should be applied when we have two or more estimates for the same angular coefficient, say m_x. In Fig. 5.4c, y_{i-1} overestimates the local value of the height function and the absolute value of m_{xc}, based on the centered scheme, is smaller than the angular coefficient of the straight line. Hence, if we have two different estimates for the same angular coefficient, say m_{x1} and m_{x2}, we now choose

$$|m^*| = \max(|m_{x1}|, |m_{x2}|)\,. \tag{5.15}$$

With these two simple criteria we can reconstruct any linear interface, but they may not be very efficient in the case of a curved interface. In the next section we discuss a method that selects an optimal angular coefficient among a set of candidates by minimizing an area error function.

5.2.2.3 The ELVIRA method

The ELVIRA method was introduced by Pilliod Jr. and Puckett (2004). We consider again the height function $y = f(x)$, and for the slope m_x we compute the centered scheme (5.11) and also the backward and forward estimates $m_{x\mathrm{b}}$ and $m_{x\mathrm{f}}$, which are given by the following expressions:

$$\begin{aligned} m_{x\mathrm{b}} &= -\frac{1}{h}(y_i - y_{i-1}) = -\sum_{k=-1}^{1}(C_{i,j+k} - C_{i-1,j+k})\,, \\ m_{x\mathrm{f}} &= -\frac{1}{h}(y_{i+1} - y_i) = -\sum_{k=-1}^{1}(C_{i+1,j+k} - C_{i,j+k})\,, \end{aligned} \tag{5.16}$$

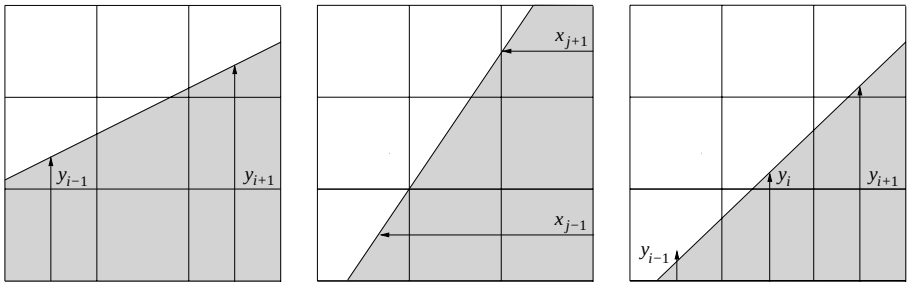


Fig. 5.4. In the centered-columns scheme, volume fractions are added column-wise, heights y_{i-1} and y_{i+1}, for case (a) and row-wise, widths x_{j-1} and x_{j+1}, for case (b), to get the correct slope of the linear interface. In case (c), an off-centered scheme, with heights y_i and y_{i+1}, should be used.

and similarly we compute m_{yb} and m_{yf} for the angular coefficient m_y of the width function (5.12). There are now six different choices and a criterion has to be designed in order to select the "best" linear approximation. In the ELVIRA strategy the choice is done by minimizing a measure of the error between the volume fractions given by the true and approximate interfaces. For each of the six normal vectors $\mathbf{m}_n$, with $1 \leq n \leq 6$, the corresponding value of the line constant α_n is first determined by area conservation in the central cell. Then the approximate linear interface is drawn across the whole 3×3 block of cells, defining a new volume fraction value $\tilde{C}$ in each of the surrounding eight cells. The normalized area in the cell (k,l) under the line corresponding to the normal $\mathbf{m}_n$ is $h^2\tilde{C}_{k,l}(\mathbf{m}_n)$, and the area error in L_2 is

$$E(n) = h^2 \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} (\tilde{C}_{k,l}(\mathbf{m}_n) - C_{k,l})^2 \tag{5.17}$$

and in L_∞ it is

$$E(n) = h^2 \max \left| (\tilde{C}_{k,l}(\mathbf{m}_n) - C_{k,l} \right| . \tag{5.18}$$

The selected value of $\mathbf{m}_n$ among the six angular coefficients is the one that minimizes E. It is easy to verify that there is always at least one selection that reproduces correctly a straight line even when it cuts two consecutive sides of the block of cells, as in Fig. 5.4c, since the line is constrained to go across two consecutive rows or columns at least. For a tessellation of the plane with rectangular cells of equal size, this algorithm requires a 3×3 block of cells to reconstruct any linear interface; however, a wider stencil is required when the cell size is not constant

across the computational domain. The ELVIRA method has the drawback of being relatively slow, especially in three dimensions, where the number of evaluations of the predicted volume fractions $\tilde{C}_{k,l,p}(\mathbf{m}_n)$ becomes large.

5.2.2.4 The least-squares fit method

The least-squares fit class of methods introduced by Scardovelli and Zaleski (2003) relies on the minimization of a distance functional starting from a preliminary guess. The method is as follows:

(i) A "preliminary" reconstruction is performed in the whole domain, using one of the fast methods for normal estimation, either Youngs' finite-difference or the centered-columns scheme. The value of α is also estimated everywhere, yielding a segment in each cell.

(ii) A few points are selected in each cell; for example, at each end and in the center of the segment. The rationale about this choice is that a good linear approximation has to cut a curved interface twice in each cell, as in Fig. 5.5a, and the fit will be in this way determined by points lying on both sides of the interface. A set of n_p interface points (x_i, y_i) is then constructed.

(iii) One of the two functionals

$$I_1 = \sum_{i=1}^{n_p} [\omega_i (y_i + m x_i + \alpha)^2] \quad \text{or} \quad I_2 = \sum_{i=1}^{n_p} [\omega_i (x_i + m y_i + \alpha)^2] \tag{5.19}$$

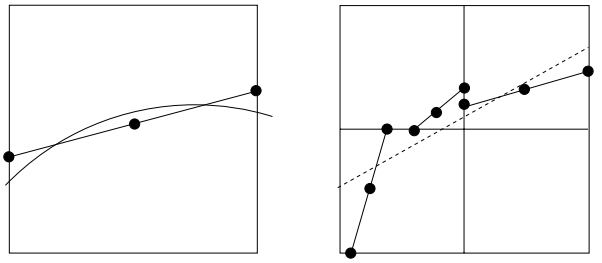
is minimized, depending on the slope of the preliminary reconstruction. The weights ω_i can be a function of the distance from the central cell of the block, but here are all equal to one. The resulting 2×2 linear algebraic system is solved only for the angular coefficient m.

(iv) The value of α in the central cell is obtained from area conservation.

This technique can be applied iteratively by using the linear fit reconstruction as the "preliminary" one, then any straight line is reconstructed exactly in a few iterations. However, for most applications just one linear fit is sufficient, as shown in the reconstruction accuracy tests of Section 5.3.2.

5.2.2.5 A short survey of other methods and extension to three dimensions

A detailed list and presentation of the VOF reconstruction algorithms developed in the last several years would require several pages; thus, we conclude this section with only a few remarks. An alternative approach to ELVIRA is the LVIRA algorithm by Puckett (1991), where the number of candidates is not limited to six; instead, a minimum, either local or absolute, of the area error, Equation (5.17) or (5.18), is searched by changing the normal $\mathbf{m}$. A smoother variation of the normal along the interface line is obtained by López *et al.* (2004): they first reconstruct



(a) (b)

Fig. 5.5. A schematic picture of the least-squares fit reconstruction. (a) The segment, where a number of points are selected, is the preliminary reconstruction of the interface line. (b) The set of points used in the least-squares fit is shown together with the line resulting from the fit (dashed line). For clarity, only a subset of the 3×3 block is shown.

the interface with Youngs' method, collect the midpoints of the segments in an ordered list, and finally interpolate the points with a cubic spline as a function of the arc length s. The local normal is found by differentiating the parametric curve $x = x(s)$, $y = y(s)$. A reconstruction with two consecutive segments in each cell has been implemented by Scardovelli and Zaleski (2003) to better represent high-curvature regions.

Youngs' and the centered-columns schemes can be directly extended to three dimensions in Cartesian grids. A three-dimensional version of ELVIRA has been implemented by Miller and Colella (2002) with the number of candidate normal vectors varying from 72 to 144. To reduce the number of calculations, the minimization of the three-dimensional extension of the error function (5.17) is performed with respect to the local height function, defined over columns of five cells, and not directly the C data. As a matter of fact, a $3 \times 3 \times 3$ block of cells is not sufficient to compute with finite differences the normal components of an arbitrary plane and a wider stencil with $5 \times 5 \times 5$ cells is usually adopted. The three-dimensional version of the least-squares fit method has been carried out by Aulisa *et al.* (2007).

5.2.3 Determination of α

Once the normal $\mathbf{m}$ has been calculated, the non-homogeneous term α of (5.5) is determined by enforcing area conservation. With reference to Fig. 5.6, for a more general rectangular cell, of sides Δx and Δy, it can be computed with an iterative procedure that requires an estimate of α, the determination of the area $A(\alpha)$ of the polygon ABFGD, and its comparison with the value $\Delta x \Delta y C$, until the difference between the two areas is below some prescribed tolerance. In other terms, we have

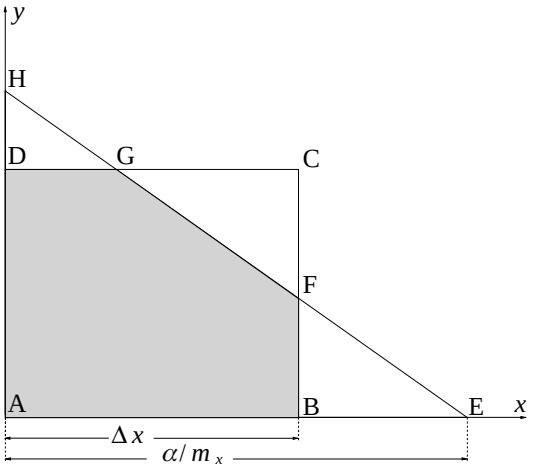


Fig. 5.6. The "cut area" is the gray region inside the rectangular cell ABCD and below the straight line EH.

to find a zero of the nonlinear function

$$g(\alpha) = A(\alpha) - \Delta x \Delta y C \tag{5.20}$$

with a root-finding algorithm. The area of ABFGD can be found by collecting in counterclockwise order the coordinates (x_i, y_i) of its vertices and by using the following formula for the area A of a n-sided polygon:

$$A = \frac{1}{2} \sum_{k=1}^{n} (x_k y_{k+1} - x_{k+1} y_k), \tag{5.21}$$

where vertex $n+1$ coincides with vertex 1. We describe an alternative approach which is not iterative and that relies heavily on the symmetry of a Cartesian cell so that, with proper mirror reflections about the coordinate axes, Equation (5.5) has both coefficients m_x and m_y positive, as shown in Fig. 5.6. In this case the area A of the polygon ABFGD is given by the expression

$$A = \frac{1}{2m_x m_y} \left[\alpha^2 - F_2(\alpha - m_x \Delta x) - F_2(\alpha - m_y \Delta y) \right], \tag{5.22}$$

where $F_2(z) = z^2$ when $z > 0$ and zero otherwise. The three contributions to A represent the areas of the similar triangles AEH, BEF and DGH. The expression (5.22) is a quadratic function of α when the interface line cuts two consecutive cell sides, then one of the two cut figures is a triangle, and it is linear when the two intersections with the cell boundary are on opposite sides. Furthermore, it is a continuous, strictly monotonically increasing function of α and it can be easily

inverted. In Appendix C we discuss briefly both the direct problem – that is, to find the cut area or the cut volume occupied by the reference phase given α and the normal $\mathbf{m}$ – and the inverse relation for rectangular and right hexahedral cells.

5.3 Tests of reconstruction methods

We now examine the accuracy and convergence properties of the reconstruction methods described in Section 5.2. We consider stationary interfaces, then no advection is performed and there is no error due to time discretization.

5.3.1 Errors measurement and convergence rate

Let $H(\mathbf{x},t)$ be the characteristic function associated with a given fluid body and $\widetilde{H}_h(\mathbf{x},t)$ its estimate associated with a partition of the plane with square cells of side h and with a reconstruction method. A natural measure of the difference between the exact and reconstructed interfaces is the error E in L_1:

$$E = \frac{1}{B}\int\int |H(\mathbf{x},t) - \widetilde{H}_h(\mathbf{x},t)|\,\mathrm{d}x\,\mathrm{d}y\,, \tag{5.23}$$

where B is a normalization constant. If $B = 1$, then the error is given by the sum of the areas comprised between the real and reconstructed interface lines inside each mixed cell, as shown in Fig. 5.3; if B is the length L of the exact interface line, then the error E can be envisioned as the width of a small ribbon along the interface, representing the average displacement from the real interface; finally, B can be equal to the total area A of the fluid body and E is not a dimensional number. Similar error measures can be defined in L_2 and L_∞. The order of convergence $\mathscr{O}$ of a reconstruction method can be numerically calculated by considering the errors E_h, obtained with grid spacing h, and $E_{h/2}$ with $h/2$ and by using the following definition:

$$\mathscr{O} = \frac{\ln(E_h/E_{h/2})}{\ln(h/(h/2))} = \frac{\ln(E_h/E_{h/2})}{\ln(2)}\,. \tag{5.24}$$

Clearly, it does not depend on the normalization constant B, but it is a function of the fluid-body shape, its position, and orientation with respect to the grid lines and obviously of the grid spacing h. The asymptotic order of convergence is found in the limit as the grid spacing h goes to zero, $h \to 0$, and it can only be extrapolated numerically.

5.3.2 Reconstruction accuracy tests

We limit the number of tests to two configurations: a straight line and an ellipse. In the first test we check how the reconstruction algorithms approximate a straight

Table 5.1. Reconstruction error and convergence rate in approximating a straight line for different grid resolutions.

Grid	Youngs $E/\mathscr{O}$	Centered $E/\mathscr{O}$	Linear fit-1st $E/\mathscr{O}$
10×10	8.21e-4	3.29e-5	1.78e-6
	0.97	0.45	1.21
20×20	4.18e-4	2.40e-5	7.67e-7
	1.01	0.65	0.81
40×40	2.07e-4	1.52e-5	4.36e-7
	1.00	0.81	0.85
80×80	1.03e-4	8.62e-6	2.42e-7
	1.00	1.05	1.08
160×160	5.16e-5	4.15e-6	1.14e-7
	1.01	0.96	0.96
320×320	2.56e-5	2.13e-6	5.88e-8

line, since we expect a second-order method to reconstruct exactly any straight line. In the second case we consider a curve with a smoothly varying curvature and we address in particular the issue of the asymptotic convergence rate. We calculate the error (5.23) with $B = 1$ by first finding the points where the exact and reconstructed interface lines cross each other and the cell edges and then evaluate the integrals between these points analytically. We average the error (5.23) over many different cases to stabilize its value, since in some instances the error can be rather small because of a fortuitous alignment of the interface with the grid lines. We first consider 1000 different straight lines where we randomly position a point of the line near the center $(0.5, 0.5)$ of a unit square with a random angle between the interface line and the horizontal coordinate line. The results are presented in Table 5.1 for different grid resolutions. The ELVIRA scheme reconstructs exactly any straight line, the other three methods only for a limited number of slopes. In the linear fit we have used the centered-columns scheme as a preliminary reconstruction. If we apply it iteratively on top of the previous reconstruction, the error decreases by several orders of magnitude at each iteration, so that at the second iteration it is below machine accuracy. Both Youngs' and the centered-columns methods are roughly first order, even if the latter has a lower error.

The second interface line is an ellipse described by the equation $x^2/a^2 + y^2/b^2 = 1$, with $a^2 = 0.12$ and $b^2 = 0.02$, and the ratio between the maximum and minimum radius of curvature is about 15. With a random number generator we position the center of the ellipse near the central point of the unit box and choose the angle between the major axis of the ellipse and the horizontal coordinate line. We now

Table 5.2. Reconstruction error and convergence rate in approximating an ellipse for different grid resolutions.

Grid	Youngs $E/\mathscr{O}$	Centered $E/\mathscr{O}$	ELVIRA $E/\mathscr{O}$	Linear fit-1st $E/\mathscr{O}$
10×10	3.90e-3	4.25e-3	4.76e-3	3.30e-3
	2.34	2.47	2.42	2.41
20×20	7.68e-4	7.65e-4	8.87e-4	6.23e-4
	1.84	2.27	2.33	2.15
40×40	2.14e-4	1.59e-4	1.77e-4	1.41e-4
	1.38	2.02	2.10	2.07
80×80	8.23e-5	3.93e-5	4.12e-5	3.41e-5
	1.16	1.89	2.04	2.02
160×160	3.69e-5	1.06e-5	1.00e-5	8.36e-6
	1.06	1.73	2.00	2.01
320×320	1.77e-5	3.20e-6	2.49e-6	2.07e-6

need 150 different cases to stabilize the results up to the third significant digit. The results are presented in Table 5.2 for different grid resolutions. Youngs' reconstruction is asymptotically a first-order method even if at very low resolution it performs very well; in other words, when the local radius of curvature is comparable to the grid spacing, it is competitive with the other methods. ELVIRA is clearly second-order accurate. By considering that a grid resolution with 320×320 cells is indeed very high for a single fluid body with an elliptical shape, we can say that for practical purposes the centered-columns scheme has a convergence rate intermediate between 1 and 2, while the linear fit is second-order accurate without the need of any further iteration.

5.4 Interface advection

In PLIC methods the interface is first reconstructed by a discontinuous piecewise linear line and then advected in a given velocity field up to the next discrete time. This evolution may be approximated with *geometrical* methods, by advecting the end points of each segment of the interface, or with fluxing methods, by computing the reference phase fluxes across the cell boundary.

In the geometrical approach, a deep insight on interface advection can be gained by expressing one-dimensional advection schemes along the x-direction as different mappings of the plane onto itself. The mappings possess the remarkable property that the consistency condition $0 \leq C \leq 1$ is satisfied after advection in each grid cell.

5.4.1 Geometrical one-dimensional linear-mapping method

We discuss two different linear mappings. In the first one we advect in a given one-dimensional velocity field each grid cell and then we mark out the deformed cells in the original grid to update the volume fraction data; hence the name *out-of-cell*. By contrast, in the *onto-cell* mapping we compute the area that will be advected into each grid cell.

5.4.1.1 Operator splitting

In one-dimensional advection we follow the interface as it is advected by a discretized velocity field $u(x)$. In two dimensions, one-dimensional advections in the two directions are performed in sequence. To reduce possible asymmetries induced by the splitting we can consider first a motion along the x-direction and then along y on odd time steps and vice versa on even time steps. For three-dimensional Cartesian grids we need three one-dimensional sweeps and a more complex sequence of sweep direction orders.

5.4.1.2 The out-of-cell explicit linear mapping T^{E}

Consider the case in which we perform the advection along the x-axis first. In the linear mapping method the reconstructed interface at time step n is mapped on the interface at time t^{n+1} by a linearized velocity field. Each end point of the interface segments is moved with an *interpolated velocity*. Consider the motion $x(t)$ of a particle in the velocity field $u(x)$. The equation of motion is $\mathrm{d}x/\mathrm{d}t = u(x)$. An explicit first-order scheme for its integration is

$$x(t^{n+1}) = x(t^n) + u[x(t^n)]\,(t^{n+1} - t^n)\,. \tag{5.25}$$

Before proceeding further, it is useful to rescale space and time in grid spacing and time step units. Using dimensions of h and $\Delta t = t^{n+1} - t^n$, the new nondimensional physical variables are $x' = x/h$, $t' = t/\Delta t$, and $u' = u\Delta t/h$. Notice that the x-component of the velocity vector is now a CFL number. The grid cells are mapped to unit squares and the origin of the local coordinate system is in the lower-left corner. From now on, we drop the primes when using the cell nondimensional variables; then:

$$x(t^{n+1}) = x(t^n) + u[x(t^n)]\,. \tag{5.26}$$

To complete the integration of the equation of motion we need an approximation of u. We label with Σ the unit area of the cell (i, j) and consider a linear interpolation of the u velocity between the two values $u_{i-1/2,j}$, on the left edge of the MAC cell, and $u_{i+1/2,j}$, on its right edge:

$$u(x) = u_{i-1/2,j}\,(1-x) + u_{i+1/2,j}\,x\,. \tag{5.27}$$

Combining (5.26) and (5.27) we see that the updated position of the point $\mathbf{x}' = \mathbf{x}(t^{n+1})$ is a function of the old coordinates

$$\begin{cases} x' = bx + u_{i-1/2,j} \\ y' = y \end{cases}, \tag{5.28}$$

with $b = 1 + u_{i+1/2,j} - u_{i-1/2,j}$. The system (5.28) describes a linear mapping $\mathbf{x}' = T_x^{\mathrm{E}}(\mathbf{x})$. The mapping of the whole computational domain is piecewise linear, each piece transforming a square cell Σ of the grid onto a rectangle Γ_x:

$$\Gamma_x = T_x^{\mathrm{E}}(\Sigma). \tag{5.29}$$

In Fig. 5.7 we show the case where the velocity field compresses the square, but depending on the velocity values the square may also be expanded and shifted to the left or to the right as well. Because the mapping is linear, straight lines are transformed into straight lines and the reconstructed interface is easily advected and then remapped onto the original mesh. In the case depicted in Fig. 5.7 the new volume fraction in the central cell Σ is obtained by adding the three area contributions from the central and the two adjacent cells after their transformation by T_x^{E}:

$$C_{i,j}^{n+1} = D + E + F. \tag{5.30}$$

The three-dimensional extension is straightforward, except that D, E, and F are now polyhedra obtained by cutting right hexahedra by the transported planar interface elements. Their volume is computed by the technique described in Appendix C. It is important to stress that the entire plane is partitionned in a *tessellation* made of image rectangles such as Γ_x (Fig. 5.8). In other words, the Γ_x rectangles do not overlap or leave empty space, which ensures that area is properly advected. The method has the important feature that the updated volume fraction C satisfies $0 \leq C \leq 1$. This is because the volume fraction is computed as the sum of the three areas D, E, and F.

It is now possible to construct a two-dimensional advection method with two consecutive out-of-cell explicit linear mappings:

(i) reconstruct the interface at time step t^n;
(ii) perform the advection along the x axis by using the mapping T_x^{E} to move the two interface end points in each cut cell;
(iii) from the updated position of the end points, compute the areas with the expressions given in Appendix C and update an intermediate volume fraction field C^*:

$$C_{i,j}^* = D_x + E_x + F_x; \tag{5.31}$$

(iv) reconstruct the interface from the C^* data;

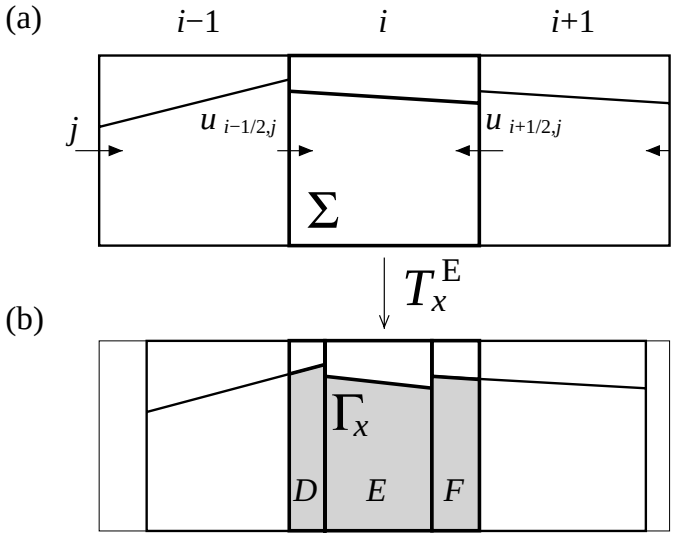


Fig. 5.7. (a) The horizontal one-dimensional mapping T_x^{E} transforms the square cell Σ onto the rectangle Γ_x, by transporting the cell edges with the one-dimensional flow. (b) Three consecutive cells are transformed by the piecewise linear mapping T_x^{E} onto three rectangles, which are projected back to the original grid to calculate the three contributions D, E, and F to the square cell Σ.

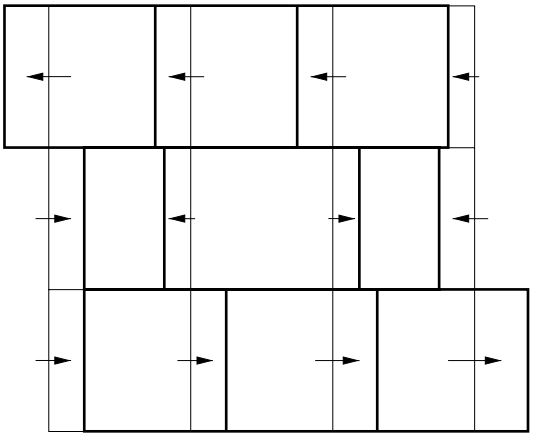


Fig. 5.8. The images of the square cells by the mapping T_x^{E} form a tessellation of the plane by rectangles (thick solid lines). These may be either expanded or compressed and shifted to the left or to the right with respect to the original square cells (thin solid lines), depending on the local value of the velocity field.

(v) perform the advection along the y axis by using the mapping T_y^{E} to compute in a similar way

$$C_{i,j}^{n+1} = D_y + E_y + F_y \,. \tag{5.32}$$

The extension to three dimensions requires three reconstructions and three one-dimensional advections. This method has a major disadvantage: it does not preserve volume and mass exactly. To achieve mass conservation it must be combined with a different one-dimensional advection scheme.

5.4.1.3 The onto-cell implicit linear mapping T^{I}

To construct this new mapping instead of the explicit method (5.26), we consider an implicit first-order scheme to integrate the equation of motion

$$x(t^{n+1}) = x(t^n) + u[x(t^{n+1})] \,, \tag{5.33}$$

but notice that the x-position at time t^{n+1} is calculated with the velocity field u at time t^n. By using (5.33) and (5.27) we find that the action of the mapping $\mathbf{x}' = T_x^{\mathrm{I}}(\mathbf{x})$ reads in components

$$\begin{cases} x' = ax + au_{i-1/2,j} \\ y' = y \end{cases} , \tag{5.34}$$

where $a = 1/(1 - u_{i+1/2,j} + u_{i-1/2,j})$. Consider now the pre-image Γ_x of the central square Σ of Fig. 5.9 by T_x^{I}; then

$$\Sigma = T_x^{\mathrm{I}}(\Gamma_x) \,. \tag{5.35}$$

We compute the new volume fraction in the central cell by considering the area of the reference phase in Γ_x *sent onto* Σ

$$C_{i,j}^{n+1} = D + E + F \,, \tag{5.36}$$

where the three area contributions are defined in Fig. 5.9. For clarity, we point out a few differences between the two mappings T_x^{E} and T_x^{I}. In particular, the width of the area D is $u_{i-1/2,j}$ in Fig. 5.7 and $a\,u_{i-1/2,j}$ in Fig. 5.9. Furthermore, the slope of the advected interface changes in a different way, since the coefficient of the linear mapping is b for T_x^{E} and a for T_x^{I}. Finally, it is important to stress that, as in the explicit mapping, the whole computational domain is partitionned in a tessellation made of pre-image rectangles such as Γ_x. This ensures that no area is "lost" or fluxed twice; however, the sequence of two consecutive onto-cell implicit mappings in the x- and y-directions does not conserve the area.

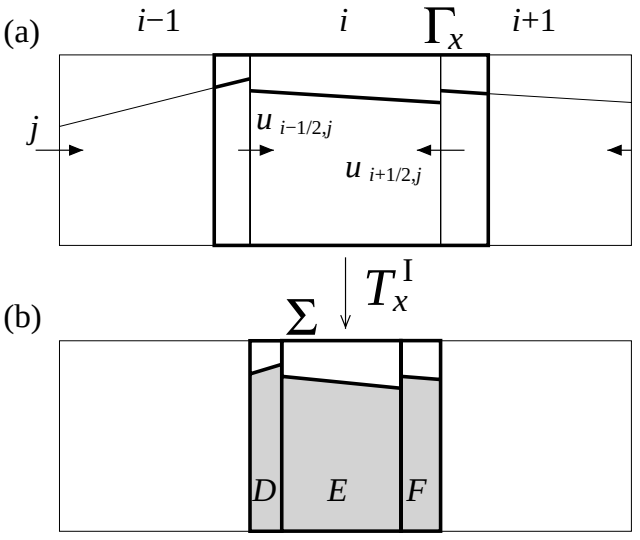


Fig. 5.9. (a) The horizontal one-dimensional mapping T_x^{I} transforms the rectangle Γ_x onto the square cell Σ by using the velocity at the cell sides. The velocity field u is the same as Fig. 5.7. (b) The contributions to Γ_x of three consecutive cells are transformed by the linear mapping T_x^{I} onto D, E, and F of the central cell Σ.

5.4.1.4 Combined linear mapping

A remarkable conservation property is achieved when we combine the two mappings. Indeed, the mapping sequence T_x^{I} followed by T_y^{E} conserves area/mass exactly. When the initial rectangle Γ_x is mapped onto Σ, the reference phase area is compressed (or expanded) by the factor $a = 1/(1 - u_{i+1/2,j} + u_{i-1/2,j})$. Then, when Σ is mapped onto a new rectangle Γ_y, it is expanded (or compressed) by the mapping T_y^{E}, which reads

$$\begin{cases} x' = x \\ y' = by + v_{i,j-1/2} \end{cases}, \tag{5.37}$$

with $b = 1 + v_{i,j+1/2} - v_{i,j-1/2}$. The combination of the two mappings leads to the following two-dimensional advection scheme:

(i) reconstruct the interface at time step t^n;
(ii) perform the advection along the x-axis by using the onto-cell mapping T_x^{I} and get the provisional scalar field C^*:

$$C_{i,j}^* = D_x + E_x + F_x; \tag{5.38}$$

(iii) reconstruct the interface from the C^* data;

(iv) perform the advection along the y-axis by using the out-of-cell mapping T_y^{E} and get

$$C_{i,j}^{n+1} = D_y + E_y + F_y \,. \tag{5.39}$$

The area of $\Gamma_y = T_y^{\mathrm{E}}(\Sigma) = T_y^{\mathrm{E}}\, T_x^{\mathrm{I}}(\Gamma_x)$ is the area of Γ_x multiplied by

$$a\,b = \frac{1 + v_{i,j+1/2} - v_{i,j-1/2}}{1 - u_{i+1/2,j} + u_{i-1/2,j}}, \tag{5.40}$$

but for an incompressible flow the divergence-free condition on a staggered MAC grid with square cells reads

$$u_{i+1/2,j} - u_{i-1/2,j} + v_{i,j+1/2} - v_{i,j-1/2} = 0; \tag{5.41}$$

then $a\,b = 1$ and $\Gamma_y = \Gamma_x$. Thus, for an incompressible flow, the area is conserved exactly and the consistency condition $0 \leq C \leq 1$ is always satisfied. To minimize asymmetries, the combined mapping $T_y^{\mathrm{E}}\, T_x^{\mathrm{I}}$ at one time step should be alternated with $T_x^{\mathrm{E}}\, T_y^{\mathrm{I}}$ at the next time step. Finally, notice that if we consider the mapping $T_y^{\mathrm{I}}\, T_x^{\mathrm{E}}$ instead of $T_y^{\mathrm{E}}\, T_x^{\mathrm{I}}$, the velocity components of different cells appear in the two coefficients a and b; then, (5.41) does not apply and the area is not conserved.

An extension to three dimensions has been proposed by Aulisa *et al.* (2007). The velocity field $\mathbf{u} = (u, v, w)$ may be split in the three fields $\mathbf{u}_1 = (u_1, v_1, 0)$, $\mathbf{u}_2 = (u_2, 0, w_2)$, $\mathbf{u}_3 = (0, v_3, w_3)$, where each field $\mathbf{u}_i$ is assumed to be incompressible, $\nabla \cdot \mathbf{u}_i = 0$, and $\mathbf{u}_1 + \mathbf{u}_2 + \mathbf{u}_3 = \mathbf{u}$. There are six scalar equations for the six unknowns $(u_1, v_1, u_2, w_2, v_3, w_3)$, but only five of them are independent. For example, if $\mathbf{u}_1$ and $\mathbf{u}_2$ satisfy $\nabla \cdot \mathbf{u}_1 = \nabla \cdot \mathbf{u}_2 = 0$, then $\mathbf{u}_3 = \mathbf{u} - \mathbf{u}_1 - \mathbf{u}_2$, but in this case $\nabla \cdot \mathbf{u}_3 = \nabla \cdot (\mathbf{u} - \mathbf{u}_1 - \mathbf{u}_2) = 0$ is automatically satisfied. Therefore, it is possible to exploit this degree of freedom to specify one of the six unknowns and, starting from the boundary where the velocity field is given, to compute on a staggered MAC grid the velocity field components which are still unknown by solving the divergence-free equations.

5.4.2 Related one-dimensional advection methods

The terminology we have employed in the previous section is not the most widespread one. Instead, for historical reasons one often refers to Eulerian and Lagrangian methods. The two linear mappings have been naturally derived from an explicit or implicit Lagrangian point of view. In Lagrangian methods, particles or markers are transported in space by the velocity field. The velocity is defined at the grid nodes and needs to be interpolated to the points where markers are located. If a fixed grid is used, then a Lagrangian method requires the interface after its advection to be projected back to the underlying grid to calculate local field values.

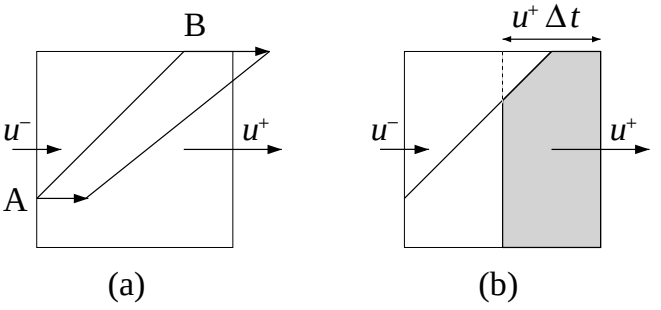


Fig. 5.10. (a) In the Lagrangian method, the end points A and B of the interface segment are advected by the flow. The local velocity may be interpolated linearly between the velocities u^- and u^+ at the cell faces. (b) In the Eulerian method, the total area flux through the right side of the cell is the area of the rectangle with width $u^+\,\Delta t$, while the reference phase flux is the shaded portion of this rectangle.

On the other hand, in Eulerian methods the field values are updated at the local grid points or cells. For the volume fraction field this is performed by exchanging area fluxes between adjacent cells. The difference between the two methods is illustrated in Fig. 5.10.

5.4.2.1 One-dimensional advection equation for the volume fraction

We consider again the advection equation (5.1), for the marker function for an incompressible flow:

$$\frac{\partial H}{\partial t} + \nabla \cdot (\mathbf{u} H) = H \nabla \cdot \mathbf{u} = 0\,. \tag{5.42}$$

For operator split schemes we consider the one-dimensional version of (5.42), integrate it over the square cell (i, j), and consider for the term $\partial u/\partial x$ an average value across the cell:

$$h^2 \frac{\partial C_{i,j}(t)}{\partial t} + \int_\Gamma \mathbf{u} \cdot \mathbf{n} H(\mathbf{x}, t)\, \mathrm{d}l = h^2 C_{i,j}(t) \frac{\partial u}{\partial x}\,, \tag{5.43}$$

where the term on the right-hand side represents a one-dimensional compression or expansion that may differ from zero even if the multidimensional flow is incompressible, as pointed out by Rider and Kothe (1998). We integrate this equation in time, approximate the velocity derivative with a centered finite-difference scheme, and use again nondimensional variables:

$$C_{ij}^{n+1} = C_{ij}^n - \widetilde{\Phi}_{i+1/2,j} + \widetilde{\Phi}_{i-1/2,j} + \widetilde{C}_{ij}(u_{i+1/2,j} - u_{i-1/2,j})\,. \tag{5.44}$$

There are different choices for $\widetilde{C}_{ij}$, but once we have selected one we have to compute properly the fluxes $\widetilde{\Phi}_{i+1/2,j}$ and $\widetilde{\Phi}_{i-1/2,j}$. The geometrical method will help us to calculate correctly the fluxes.

5.4.2.2 The Lagrangian explicit scheme

We set $\widetilde{C}_{ij} = C^n_{ij}$ in (5.44), then the scheme is explicit and we get

$$C^{n+1}_{ij} = bC^n_{ij} - \widetilde{\Phi}_{i+1/2,j} + \widetilde{\Phi}_{i-1/2,j}, \tag{5.45}$$

where $b = (1 + u_{i+1/2,j} - u_{i-1/2,j})$ is the contraction/expansion coefficient of the out-of-cell linear mapping (5.28). With reference to Fig. 5.7, where the cell area Σ is mapped onto Γ_x, the term bC^n_{ij} is the value of the area E. Therefore, the two fluxes $\widetilde{\Phi}$ of (5.45) must be equal to the two areas D and F of Fig. 5.7. This "Lagrangian" scheme for interface advection was first introduced by Li (1995). If the two fluxes are not calculated after the interface line has been advected by the mapping (5.28), minor inconsistencies may arise; see, for example, Rider and Kothe (1998) and Puckett *et al.* (1997). They are usually seen as small unphysical overshoots, $C > 1$, or undershoots, $C < 0$, in the volume fraction, but also as little "holes" generated in the bulk of the reference phase or as some level of small debris outside it, which are usually called "wisps." These minor inconsistencies should not be confused with the so-called "floatsam" and "jetsam," which are bigger in nature and mainly due to low-order reconstruction methods, such as SLIC, where the interface discontinuity at the cell boundary is $\mathscr{O}(h)$.

5.4.2.3 The Eulerian implicit scheme

This scheme is equivalent to the onto-cell linear mapping. We set $\widetilde{C}_{ij} = C^{n+1}_{ij}$ in (5.44); the scheme is now implicit and we get

$$C^{n+1}_{ij} = a\,(C^n_{ij} - \widetilde{\Phi}_{i+1/2,j} + \widetilde{\Phi}_{i-1/2,j}), \tag{5.46}$$

where $a = 1/(1 - u_{i+1/2,j} + u_{i-1/2,j})$ is the contraction/expansion coefficient of the onto-cell linear mapping (5.34). Equation (5.46) and Fig. 5.9 suggest an alternative way to update the volume fraction that does not require the mapping of the interface segments that are inside the area Γ_x to the cell Σ. We calculate the volume fraction contributions inside Γ_x that will be mapped to D, E, and F in Fig. 5.9, and we multiply their sum by the coefficient a. This scheme is also called "Eulerian" because the fluxes $\widetilde{\Phi}$ are calculated as in the simple fluxing scheme of Fig. 5.10b.

5.4.3 Unsplit methods

In this section we describe two simple unsplit methods and then discuss some of the major issues that need to be addressed in the development of a Lagrangian or

an Eulerian unsplit method. Several other methods have been proposed, but they are rather complex and their description is beyond the scope of this book.

5.4.3.1 A naive unsplit Eulerian advection

A naive (but easy to implement) Eulerian scheme which satisfies exactly the divergence-free condition (5.41) is as follows:

(i) Reconstruct the interface at time step t^n.

(ii) Compute the reference phase fluxes leaving each grid cell. In the cell (i, j) of Fig. 5.11, these are the two fluxes $\Phi^n_{y:i,j+1/2}$ and $\Phi^n_{x:i+1/2,j}$ corresponding to the shaded area inside the two rectangles of thickness $v_{i,j+1/2}$ and $u_{i+1/2,j}$.

(iii) Update the volume fraction field as in Equation (5.3) (but with nondimensional variables):

$$C^{n+1}_{i,j} = C^n_{i,j} - (\Phi^n_{x:i+1/2,j} - \Phi^n_{x:i-1/2,j}) - (\Phi^n_{y:i,j+1/2} - \Phi^n_{y:i,j-1/2}).$$

This scheme can be viewed as a concurrent split method and, as shown in equation (5.4), it conserves mass to machine accuracy. However, this statement deserves some comment, since the scheme may not satisfy in a few cells the consistency condition: $0 \leq C \leq 1$. To see why, consider for instance the situation in Fig. 5.11 where the reference phase is located near the top-right corner of the cell and covers a rectangular area of size $\delta = C^n_{i,j} = u_{i+1/2,j} \times v_{i,j+1/2}$. This area is fluxed out twice, $\Phi^n_{x:i+1/2,j} = \Phi^n_{y:i,j+1/2} = \delta$, and if we assume that no reference phase is fluxed in from the bottom and left cells, the volume fraction value at the next time step becomes negative, $C^{n+1}_{i,j} = -\delta$. Therefore some mass has to be removed or added arbitrarily after advection: whenever $C > 1$ mass is removed or locally redistributed to let $C = 1$ and similarly when $C < 0$. On the other hand, if $C^n_{i,j} + \Phi^n_{x:i-1/2,j} + \Phi^n_{y:i,j-1/2} \geq 2\delta$, then $C^{n+1}_{i,j} \geq 0$, but still the same area is advected twice. The concurrent split method has another defect: at each time step mass flows from the cell (i, j) to the adjacent cells horizontally and vertically, but no mass is transferred to the diagonal cell $(i+1, j+1)$, as shown in Fig. 5.11. As a result if a smooth interface is advected even by a simple divergence-free flow, such as a uniform translation or a solid-body rotation, the interface line rapidly develops spurious oscillations. In a dynamical situation these oscillations may become unstable and give rise to numerical breakup of the interface.

5.4.3.2 Unsplit geometrical linear mapping method

The two linear mappings (5.34) and (5.37) can be applied without an intermediate reconstruction and the method becomes unsplit by defining the combined mapping

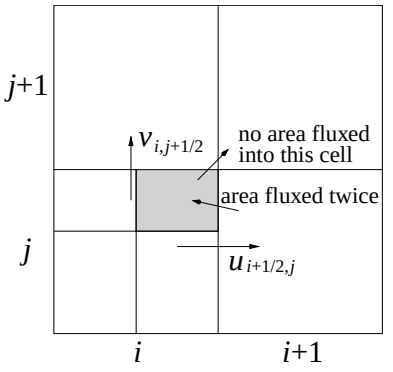


Fig. 5.11. In a concurrent split advection method, the rectangular area near the top-right corner is fluxed simultaneously to the right and upwards; moreover, there is no flux in the diagonal direction.

$\Pi_{xy} = T_x^{\mathrm{I}} \cdot T_y^{\mathrm{E}}$:

$$\begin{cases} x' = a\,x + a\,u_{i-1/2,j} \\ y' = b\,y + v_{i,j-1/2} \end{cases}, \tag{5.47}$$

where $a = 1/(1 - u_{i+1/2,j} + u_{i-1/2,j})$ and $b = (1 + v_{i,j+1/2} - v_{i,j-1/2})$. Because the transformation is linear, it still transforms straight lines into straight lines and allows an easy computation of the remapped interface.

The procedure is depicted in Fig. 5.12, and we remark the fact that the scheme may require the application of the transformation to three consecutive interface segments at most, since the pre-image of the central square in the implicit step may contain a portion of the two lateral cells as well. In Fig. 5.13 we show the unsplit mapping at work in a clockwise rotation. Only one reconstruction is needed at each time step, but the fluxing algorithm is somewhat more complex. There is still the need to alternate the starting sweep direction, namely Π_{xy} with the dual Π_{yx}, otherwise one coordinate direction is always taken as the implicit one. With some extra computations the scheme can be made more symmetric by a linear combination of the two transformations within the same time step, $\Pi = (\Pi_{xy} + \Pi_{yx})/2$, as described by Aulisa *et al.* (2003b).

5.4.3.3 Unsplit Lagrangian methods

In a Lagrangian method we advect points along the characteristic lines in the time step Δt by solving the equation of motion $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{u}(\mathbf{x}, t)$, where $\mathbf{x}$ is the point position and $\mathbf{u}$ the local velocity, usually obtained by interpolating the velocity field defined at the grid points. In Fig. 5.14, the four vertices 1234 of the cell (i, j)

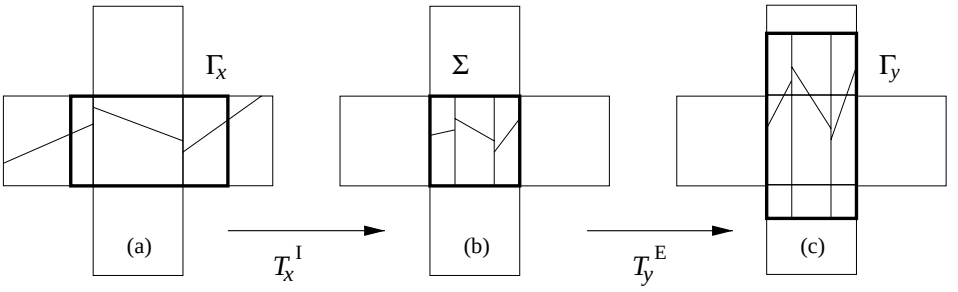


Fig. 5.12. A schematic view of the unsplit geometrical method: (a) initial configuration; (b) mapping of the rectangle Γ_x onto the unit square Σ; (c) mapping of the unit square Σ onto the rectangle Γ_y.

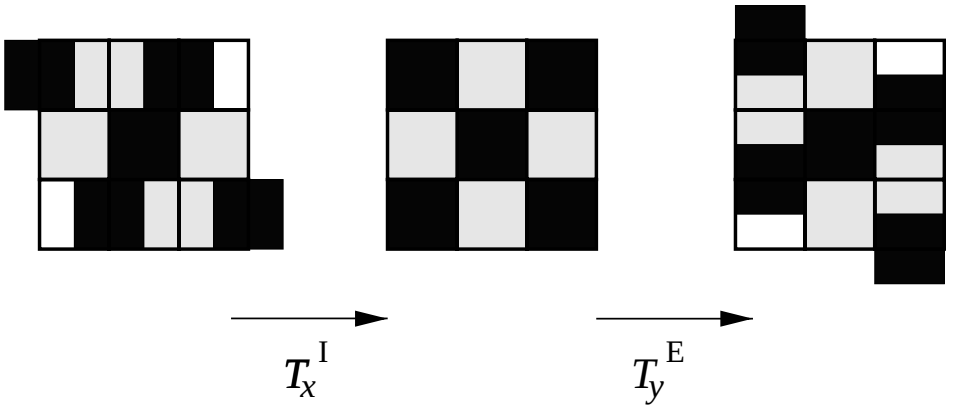


Fig. 5.13. The set of all mappings Π_{xy}, one for each computational cell, may be viewed as transforming an initial tessellation of rectangles aligned horizontally into the square grid and then into another tessellation of rectangles aligned vertically. As an example, we show the pre-image and the image rectangles in a solid body rotation of the plane, where $u = \omega y$ and $v = -\omega x$.

are advected, but alternatively only the vertices 12AB4 of the polygon containing the reference phase could be advanced in the given velocity field. Second-order or even higher order methods, including predictor–corrector or Runge–Kutta integration schemes, have been used in the literature to integrate the equation of motion. Next we have to distribute the mass in the advected polygons onto the cells of the fixed grid. This is a geometrical problem that requires for each of the four hatched regions of the deformed cell (i, j) of Fig. 5.14 the calculation of the coordinates of its vertices and that of its area by using Equation (5.21). By adding the contributions of all the grid cells, we update the volume-fraction field at the next discrete time. In theory, the method should conserve the mass exactly because the flow is incompressible; in practice, this is not true for several reasons:

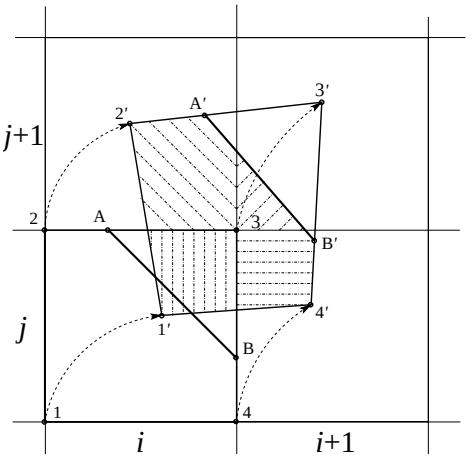


Fig. 5.14. Unsplit Lagrangian advection. In this example the whole cell (i, j) is moved by advecting its four vertices along the characteristic lines. The four hatched areas represent the contribution of the advected cell to the updated value of the volume fraction in the four cells of the fixed grid.

(i) The discrete velocity field computed by a flow solver is not exactly divergence free.
(ii) The local velocities of the advected points are usually interpolated from the grid-point values with numerical schemes that do not preserve the divergence-free condition.
(iii) The numerical integration of the equation of motion is also affected by errors.
(iv) A segment which is advected with a numerical scheme of order greater than one is deformed into a continuous, piecewise differentiable curve. The derivatives of the velocity field, and thus the slopes of the advected curve, jump when crossing the cell boundary.

As a result, the area of the deformed cell after advection is not that of the starting Cartesian cell. The area excess or defect can in principle be redistributed among the four hatched areas of Fig. 5.14 with some "ad hoc" criteria, but in the process there is no guarantee that the condition $0 \leq C \leq 1$ is satisfied everywhere. This may require another local redistribution algorithm to bring the volume-fraction data within their meaningful range of variation.

5.4.3.4 Unsplit Eulerian methods

In an Eulerian method we compute the reference phase fluxes Φ through the cell boundary. This is done geometrically by defining the material area that in the time

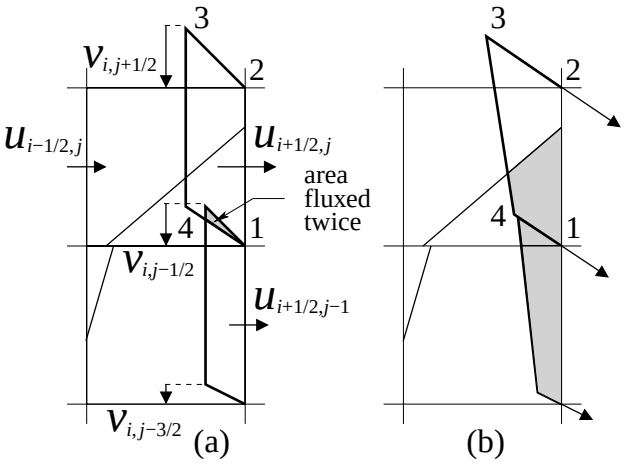


Fig. 5.15. Unsplit Eulerian advection: Two different flux polygon constructions at a cell side. (a) In the method by Rider and Kothe (1998), the velocity components on a staggered MAC grid are used to determine the trapezoid 1–2–3–4. (b) In the scheme by López *et al.* (2004), the velocity field is first interpolated to the cell vertices 1 and 2 to compute the slope of sides 2–3 and 1–4. Enforcement of area conservation and spatial variation of the velocity field determine the position and slope of side 3–4.

Δt will cross each cell side. In a split scheme the total fluid flux is given by the rectangle of area $u^+ \Delta t\, h$ shown in Fig. 5.10b, while the flux Φ is the gray portion of this rectangle filled by the reference phase. Several unsplit methods have been developed; here, we illustrate only two of them which approximate the fluid flux through a cell side as a trapezoidal polygon. In the flux polygon proposed by Rider and Kothe (1998), shown in Fig. 5.15a, the rectangular area of thickness $u_{i+1/2,j}$ of the split algorithm has been changed to a trapezoid by including two triangular areas to take into account the multidimensionality of the flow. The height of the triangle added on the top cell side is $v_{i,j+1/2}$; that of the triangle removed from the bottom side is $v_{i,j-1/2}$. As a result, the fluid area fluxing is not equal to $u_{i+1/2,j}$; hence, the total flux through the cell boundary may not satisfy the discrete divergence-free condition of the flow (5.41) and the incoming material area in the cell may be smaller or bigger than the outgoing one. Another problem with this scheme is that some area may be fluxed twice (as shown in Fig. 5.15). To reduce the negative effects of these issues the authors actually solve the two-dimensional version of (5.42) and of its discrete form (5.44)

$$C_{ij}^{n+1} = C_{ij}^{n} - \widetilde{\Phi}_{i+1/2,j} + \widetilde{\Phi}_{i-1/2,j} - \widetilde{\Phi}_{i,j+1/2} + \widetilde{\Phi}_{i,j-1/2} + \frac{C_{ij}^{n+1} + C_{ij}^{n}}{2} \nabla \cdot \mathbf{u},$$

where $\widetilde{\Phi}_{i+1/2,j}$ is the portion of the trapezoid 1234 of Fig. 5.15a which is occupied by the reference phase and that in the time Δt will flow through the right side 1–2 of the cell. In this scheme the term $\nabla \cdot \mathbf{u}$ represents the net balance between the incoming and outgoing fluid fluxes. Finally, the volume fraction is redistributed locally whenever $C > 1$ or $C < 0$.

In the scheme proposed by López *et al.* (2004), the velocity field is first interpolated to the cell vertices 1 and 2 of Fig. 5.15b (for example, on a MAC grid $u_1 = u_{i+1/2,j-1/2} = (u_{i+1/2,j} + u_{i+1/2,j-1})/2$) to define the direction of the sides 2–3 and 1–4 of the trapezoidal flux polygon. The slope of side 3–4 is given by

$$\left(\frac{\mathrm{d}x}{\mathrm{d}y}\right)_{3-4} = \frac{x_2 - x_1 - (u_2 - u_1)\Delta t}{y_2 - y_1 - (v_2 - v_1)\Delta t}\,.$$

Side 3–4 is then moved parallel to this direction until the area of the trapezoid is equal to $u_{i+1/2,j}\,\Delta t\,h$. By construction, no area is fluxed twice and the incoming material area is always equal to the outgoing one as long as the discrete velocity field is divergence free.

5.4.3.5 Redistribution algorithms

We have previously pointed out a number of inaccuracies in the integration scheme that may cause undershoots, $C < 0$, or overshoots, $C > 1$, of the volume fraction data. The easiest way to get around the problem is simply to ignore it. This is conveniently done when the over/undershoots are rather small by letting

$$C_{i,j} = \min(1, \max(C_{i,j}, 0))\,.$$

On the other hand, if the total fluid volume must be conserved exactly, we need to redistribute the C data locally as in the following algorithm proposed by Harvie and Fletcher (2000). Suppose that $C_{i,j} > 1$, then define $\varepsilon = C_{i,j} - 1$ and look for the maximum C value, strictly less than 1, in the cells (i', j') surrounding (i, j). Let this value be $C_{i',j'} < 1$ and let $C_{\mathrm{T}} = C_{i',j'} + \varepsilon$, then redefine the C value in the two cells (i', j') and (i, j) as

$$C_{i',j'} = \min(C_{\mathrm{T}}, 1)\,; \quad C_{i,j} = \max(C_{\mathrm{T}}, 1)\,.$$

If $C_{i,j} \leq 1$ the procedure stops, otherwise the redistribution algorithm will search for the next neighboring cell with the maximum C value. A similar procedure is called when $C_{i,j} < 0$.

5.5 Tests of reconstruction and advection methods

Typical advection tests involve simple translations and solid body rotations and they are particularly useful to evaluate the basic properties of different reconstruction and advection algorithms, since a fluid body immersed in these flows should

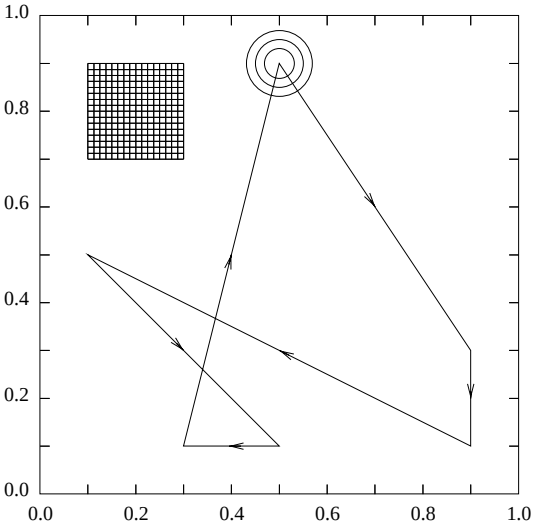


Fig. 5.16. Translation test: three different circles are advected along a continuous, piecewise linear line. A portion of the computational grid with 80×80 cells is also shown.

preserve its form. More demanding tests involve flows with a nonuniform vorticity field that deform and stretch a fluid body as it moves in the computational domain to such an extent that the interface may break up or coalesce. In the following tests we consider a circular fluid body at time $t = 0$ that returns to its initial position at $t = T$, and we estimate the geometrical error E with an L_1 norm defined as

$$E = \sum_{i,j} h^2 |C^T_{i,j} - C^0_{i,j}| \,. \tag{5.48}$$

In the next two tests, a simple translation and a more complex single vortex test, we use the ELVIRA reconstruction algorithm and the unsplit geometrical advection scheme that ensures an exact mass conservation.

5.5.1 Translation test

For this test we consider a unit square computational domain with 80×80 square cells of side $h = 1/80$. Three circles with different radius r, $r/h = 2.5, 4, 5.5$, are centered at point $(0.5, 0.9)$. The circles move along a continuous, piecewise linear line as shown in Fig. 5.16, to remove the effects of possible favourable or unfavourable alignments of the streamlines with the grid lines. A constant velocity field is assumed along each straight section of the line and a constant CFL number, based on the maximum velocity component of each section, is used in the whole simulation. Figure 5.17 shows the initial reconstruction of the three circles with

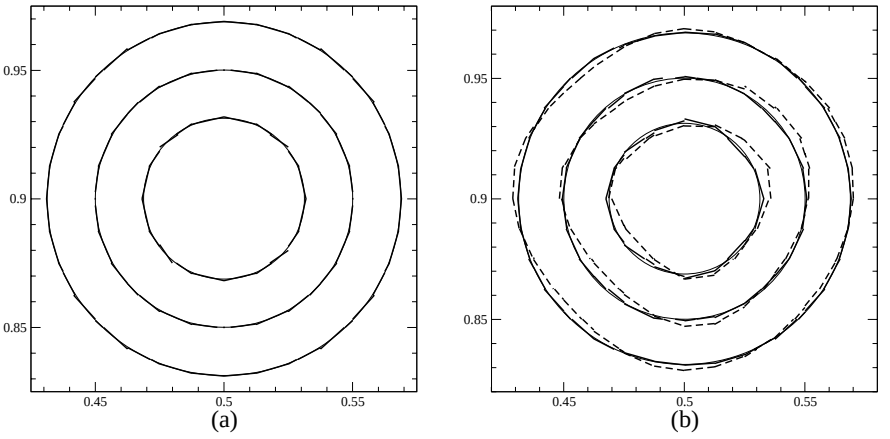


Fig. 5.17. Translation test of Fig. 5.16: (a) initial reconstruction of the three circles with a different radius r, $r/h = 2.5, 4, 5.5$; (b) reconstruction at the end of the translation test with CFL = 1 (solid segments) and CFL = 0.1 (dashed segments).

the ELVIRA and their final position with a value of the CFL number equal to 1 and 0.1. The initial and final reconstructions at CFL $= 1$ are rather good, but at CFL $= 0.1$ the final configurations are rather distorted, in particular at small radii. In actual dynamical simulations the CFL number is limited to a value smaller than one because of the numerical stability of the discrete approximations of the Navier–Stokes equations. Therefore, a ratio of at least 4 between the local radius of curvature and the grid spacing can be considered as an indicative value for a well-resolved two-phase flow simulation based on VOF methods with a single segment reconstruction in each mixed cell.

The translation test is well suited to study the reconstruction error alone in moving conditions, since the fluxing areas are exact for this simple test. Notice that as the fluid body is advected with CFL $= 1$, say along the x-direction, the C field is simply shifted by one cell to the left or to the right, but its distribution does not change. Hence, the error (5.48) is zero. As we lower the CFL number we need more reconstructions to cover the same distance, and we add a small amount of error to the simulation each time we reconstruct. As a result, we expect the error (5.48) to increase with the number of time steps towards a saturation level, i.e. the error remains bounded, when the interfaces for two consecutive time levels are very close to each other. This behavior is shown in Fig. 5.18 for the circle with $r/h = 5.5$ in the grid with 80×80 cells, as a function of the Courant number and the grid resolution. Notice that in the asymptotic region at a fixed CFL number the convergence rate is about first order at the lowest resolution, and it slowly increases towards second order with grid resolution.

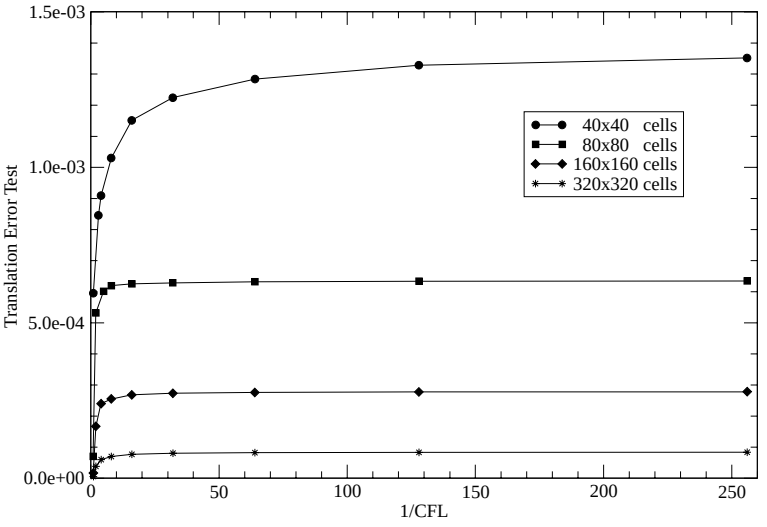


Fig. 5.18. Translation errors as a function of the CFL number and grid resolution, with the ELVIRA reconstruction algorithm and the unsplit geometrical linear-mapping advection method.

5.5.2 Vortex-in-a-box test

This test was introduced by Bell *et al.* (1989) and Rider and Kothe (1998) and is particularly suited to study the combined performance of the reconstruction and advection algorithms. The velocity field is determined by the stream function

$$\psi(x,y,t) = \frac{1}{\pi}\cos(\pi t/T)\sin^2(\pi x)\sin^2(\pi y). \tag{5.49}$$

The computational domain is a unit box partitioned with square cells of side $h = 1/n$, where n is the number of cells along each coordinate direction. The stream function value is computed in the vertices of the cell and the values of the velocity components on a staggered MAC grid are found by the centered finite-difference approximation of $u = \partial\psi/\partial y$ and $v = -\partial\psi/\partial x$. The flow rotates in the clockwise direction, as shown in Fig. 5.19, and the divergence-free condition of the velocity field is satisfied to machine accuracy. The stream function ψ is modulated in time by the factor $\cos(\pi t/T)$, where T is the period. A circular fluid body of radius $r = 0.15$ and center at $(0.5, 0.75)$ is positioned in this flow at time $t = 0$ and it is stretched and spirals around the center of the box. It reaches a maximum deformation at time $t = T/2$; then it returns to its initial position at $t = T$. In the results of Fig. 5.20 we have considered the period $T = 2$ and CFL$= 1$ at the initial time. The "exact" interface has been obtained by connecting with segments an ordered list of

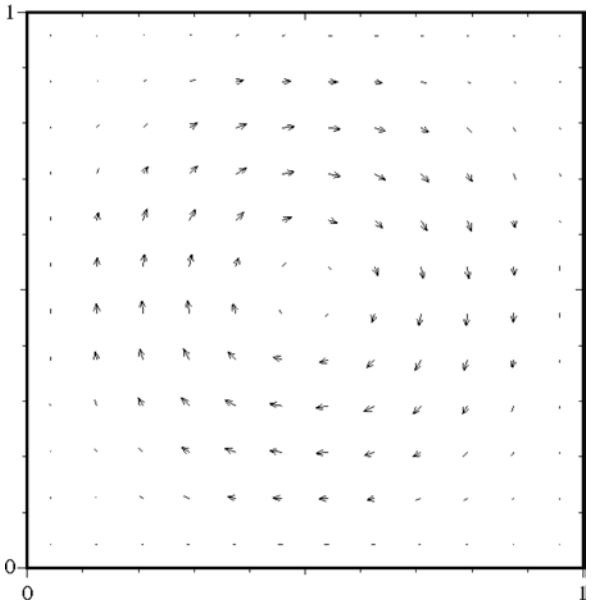


Fig. 5.19. Vortex-in-a-box velocity field.

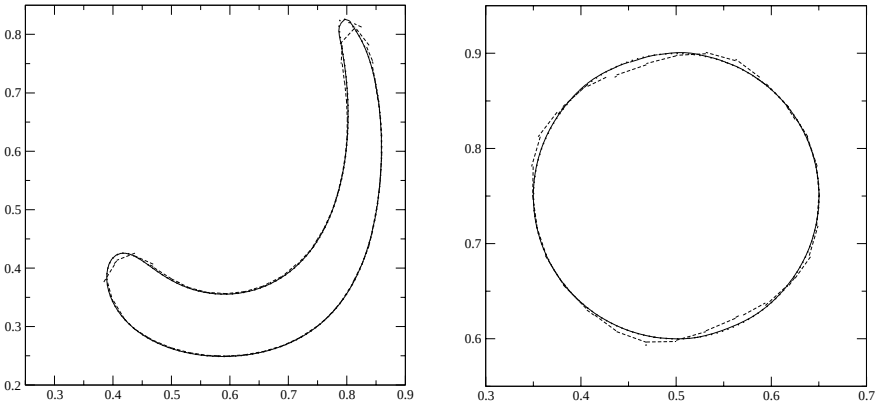


Fig. 5.20. Vortex-in-a-box test with period $T = 2$ and with $CFL = 1$ at the initial time. The exact and reconstructed interfaces are given at $t = T/2$ (left) and $t = T$ (right). In both figures the dotted line is the "exact" interface described by interface markers, the dashed segments reconstruct the interface on a 32^2 grid, and the solid segments on a 128^2 grid.

interface markers that have been advected numerically along the flow streamlines with a fourth-order Runge–Kutta integration scheme. The reconstruction and advection of the interface on a 32^2 grid cannot follow the interface shape correctly when the local radius of curvature is comparable to the grid spacing. This clearly

Table 5.3. The geometrical error E (5.48) and the order of convergence $\mathscr{O}$ (5.24) for the vortex-in-a-box test with period $T = 2$, for different grid resolutions.

	32×32		64×64		128×128
E	2.520e-3		6.224e-4		1.351e-4
$\mathscr{O}$		2.018		2.203	

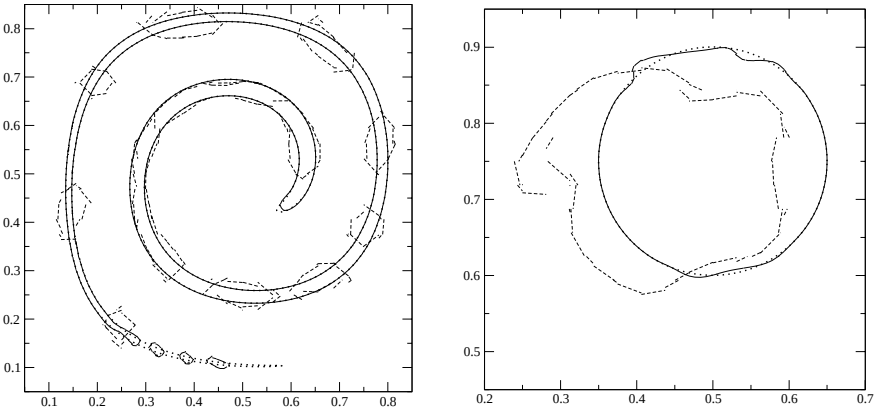


Fig. 5.21. Vortex-in-a-box test with period $T = 8$ and with CFL$= 1$ at the initial time. The exact and reconstructed interfaces are given at $t = T/2$ (left) and $t = T$ (right). In both figures the dotted line is the "exact" interface described by interface markers, the dashed segments reconstruct the interface on a 32^2 grid, and the solid segments on a 128^2 grid.

happens at $t = T/2$ near the head and the tail of the filament, which correspond respectively to the bottom and top regions of the circle at the end of the simulation, where most of the final error is found. As we increase the grid resolution to $n = 128$, the error decreases and it can be seen only in a small area near the tip of the tail, at the coarse resolution of Fig. 5.20. The geometrical error (5.48) and the order of convergence (5.24) of the ELVIRA reconstruction combined with the unsplit geometrical advection for $T = 2$ is given in Table 5.3 for different grid resolutions. A second-order convergence rate is found for this set of data. As we increase the period from $T = 2$ to $T = 8$ the reconstruction algorithm, which is limited to a single segment per cell, cannot resolve a filament of the order of or even thinner than the grid spacing h. The interface is progressively destroyed and the algorithm acts effectively as a numerical surface tension by collecting small debris into small rounded structures. The reconstructed interface at halftime and at the end of the simulation is shown in Fig. 5.21 for the two resolutions $n = 32, 128$.

5.6 Hybrid methods

The VOF method has been recently coupled to other techniques in order to take advantage of the strengths of each method. In this section we only review the way in which the coupling can be done, but we do not discuss simulation results. We first consider the coupling of VOF and level-set methods. In an incompressible flow, the conservative form of the advection equation

$$\frac{\partial G}{\partial t} + \nabla \cdot (\mathbf{u} G) = 0 \tag{5.50}$$

can be integrated in time for both the color function, $G = C$, and the level-set function, $G = F$, with an operator split technique. A geometric approach is usually used for the color function and high-order upwind and weighted ENO schemes have been considered for the level-set function. After the advection step, the coupling between the two scalar functions F and C occurs at two different instances. First, when the interface is reconstructed, the interface normal $\mathbf{m}$ can be computed directly as $\mathbf{m} = -\nabla F$. Another interesting way to compute the normal vector was suggested by Sussman and Puckett (2000). In the square cell of side h and center (x_i, y_j) the interface is approximated as the linear function described by the equation $\widetilde{F}(x,y) = a(x - x_i) + b(y - y_j) + c = 0$, then this straight line is extended to the surrounding 3×3 block of cells, as in the ELVIRA reconstruction. The discretized error function

$$E_{i,j}(a,b) = \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} \omega_{k,l} H'_{\varepsilon}(F_{k,l}) \left(F_{k,l} - \widetilde{F}(x_k, y_l) \right)^2 \tag{5.51}$$

is minimized only with respect to a and b, since the value of the constant c comes from area conservation. In expression (5.51), $\omega_{k,l}$ represents a discrete weight, $F_{k,l}$ is the level-set value at (x_k, y_l), and $H'_{\varepsilon}(F)$ is a smoothed delta function with size $\varepsilon = \sqrt{2}h$ derived from the following smoothed Heaviside function $H_{\varepsilon}(F)$:

$$H_{\varepsilon}(F) = \begin{cases} 0, & \text{if } F < \varepsilon; \\ \frac{1}{2}\left[1 + \frac{F}{\varepsilon} + \frac{1}{\pi}\sin\left(\frac{\pi F}{\varepsilon}\right)\right], & \text{if } |F| \le \varepsilon; \\ 1, & \text{if } F > \varepsilon. \end{cases} \tag{5.52}$$

The second exchange of information between the color function and the level-set function occurs after the interface reconstruction when F is reinitialized to the signed distance from the interface. In Fig. 5.22a the intersection point D between the interface line and its normal through point A is inside the segment 1–2 and the value of F at point A is the signed length of AD (if the normal $\mathbf{m}$ points outwards from the reference phase, then F is positive in the region occupied by the reference phase). If the point D is outside the interface segment 1–2, then it is necessary

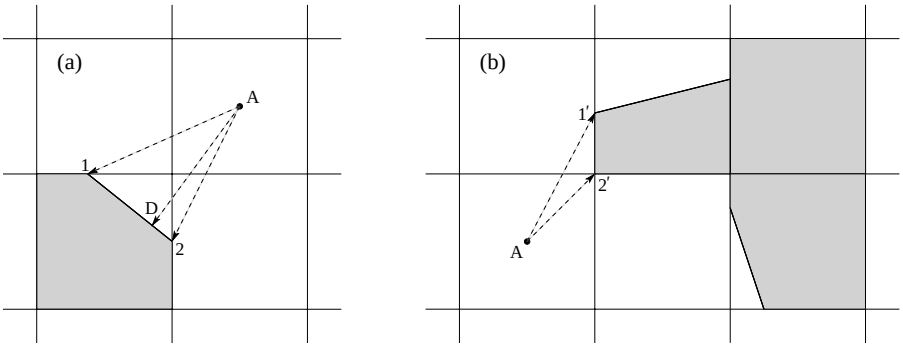


Fig. 5.22. (a) The level-set function F at point A is initialized to the signed distance AD from the interface segment 1–2. If the point D is outside the segment, the end points 1 and 2 are also considered. (b) In order to have a continuous function F it is also necessary to consider parts of the cell boundary.

to consider also its end points and $|F(\mathrm{A})| = \min(|\mathrm{A}-1|, |\mathrm{A}-2|)$. The interface reconstruction with the VOF method is, in general, not continuous across the cell boundary and the calculation of the level-set function is actually somewhat more complicated, since it is also necessary to consider portions of the cell boundary in order to have F continuous, as shown in Fig. 5.22b. The level-set function is usually reinitialized to the exact signed distance only inside a small ribbon centered on the interface line with a thickness $K \leq 8h$; outside this ribbon F is initialized to a constant value, say $|F| = (1 + K/2)h$. With a few minor differences in its implementation, this coupled technique has been used by many authors to study a large number of physical problems. We mention only the extension to adaptive unstructured triangular grids by Yang *et al.* (2006b). The advection equation for F is solved with a finite element method, while an unsplit Lagrangian technique is adopted for the color function C, similar in principle to that described in Section 5.4.3.3 and in Fig. 5.14, but with triangular grid cells. As before, at each time step the level-set function is reinitialized in the nodes of the fixed grid to the signed distance from the reconstructed interface, while the interface is reconstructed in each advected cell with the normal $\mathbf{m}$ given by $\mathbf{m} = \nabla\widetilde{F}$. The function $\widetilde{F}$ is a quadratic fit to the values of the level-set function in the surrounding cells sharing at least one vertex with the advected cell, $\widetilde{F}(x',y') = ax'^2 + bx'y' + cy'^2 + dx' + ey' + g$. In the last expression, the origin of the local Cartesian system of coordinates x' and y' is in the center of the advected cell where the normal $\mathbf{m}$ is computed, while the coefficients a, b, c, d, e and g are determined by a least-squares method.

In the hybrid method by Aulisa *et al.* (2003a) the VOF method is coupled to a front-tracking method. The interface is defined by a piecewise-linear, continuous line obtained by connecting with segments an ordered set of grid intersection and

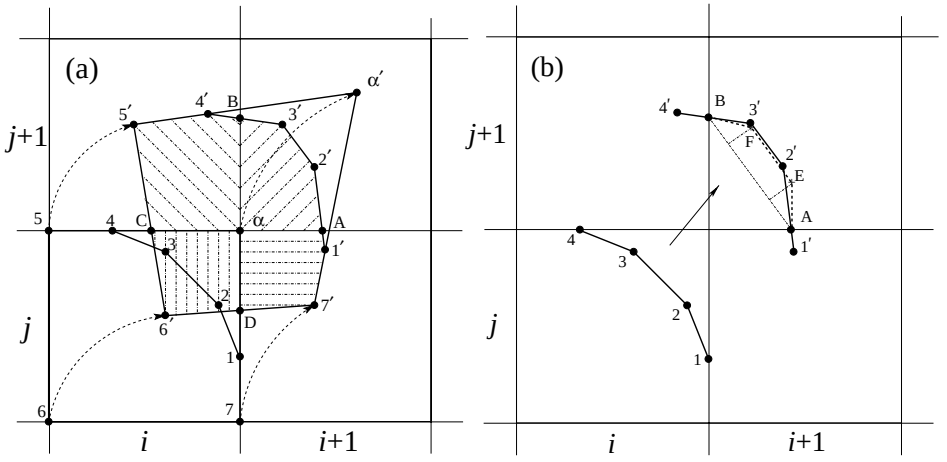


Fig. 5.23. (a) Lagrangian advection of the whole cell (i,j) and its reference phase redistribution in the neighboring cells. (b) Area conservation markers E,F replace markers $2',3'$ by conserving the area of the two polygons A$2'3'$B and AEFB.

area conservation markers. In each cell cut by the interface the reference phase is bounded by a polygon. The vertices of the polygon are collected in counterclockwise order and in the example of Fig. 5.23a, to be compared with Fig. 5.14, there are seven vertices: points $5,6,7$ are cell vertices, points $1,4$ are intersection markers, where the interface line cuts the grid lines, and points $2,3$ are area conservation markers. The coordinates (x_i,y_i) of these seven vertices are stored in normalized form, i.e. $0 \leq x_i,y_i \leq 1$, with respect to a local system of coordinates. The four vertices $\alpha,5,6,7$ of the cut cell are then advected along the flow streamlines with a fourth-order Runge–Kutta integration scheme to points $\alpha',5',6',7'$. Given the position of the advected vertices and the stored normalized coordinates, the position of the points $1',2',3',4'$ is readily computed with a bilinear interpolation. The polygon vertices are connected with straight lines and the intersections A,B,C, and D with the grid lines are determined. Clearly, only the two points A,B are intersection markers. The reference phase inside the donor cell (i,j) can be displaced over several neighboring acceptor cells after advection. The contribution to the volume fraction of each acceptor cell is one of the four differently hatched regions of Fig. 5.23a, and its area can be easily computed from the coordinates of its vertices with expression (5.21). As the simulation goes on, markers may locally concentrate or spread apart; therefore, at each time step, markers are redistributed homogeneously along the interface line, eventually by adding or removing a few of them. This simple geometrical procedure is illustrated in Fig. 5.23b, where the two inner points $2',3'$ are first replaced by E,F along the segment BA with $|\mathrm{BF}| = |\mathrm{EA}| = |\mathrm{FE}|/2$. The two new area conservation markers E,F are then equally displaced along the

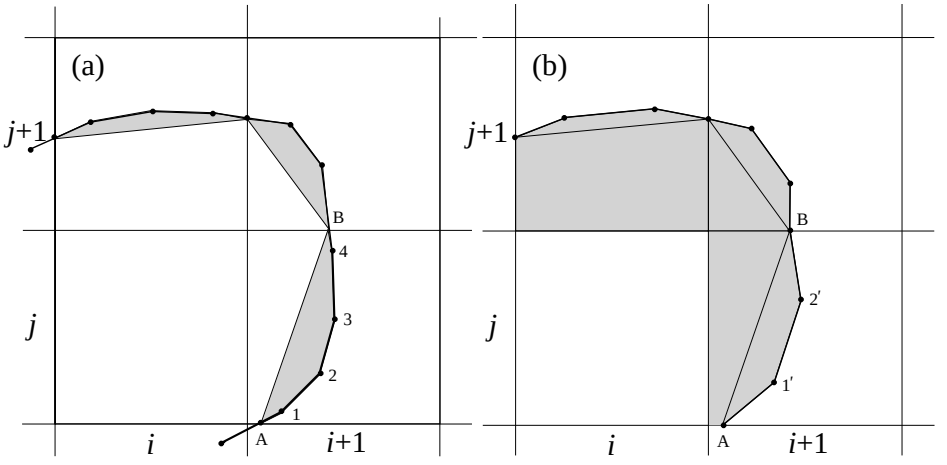


Fig. 5.24. (a) After the Lagrangian advection of the interface the area involved in the redistribution algorithm is a small ribbon (shaded area) along the interface. (b) The area conservation markers $1'$,$2'$ replace the inner markers 1,2,3,4 by conserving the spanned area. The color function is updated after the redistribution algorithm by collecting markers and cell corners in couterclockwise order.

normal direction until the two polygons A$2'3'$B and AEFB have the same area. In the first version of this hybrid method there was a strong coupling between the Lagrangian advection of the reference phase and the color function C, as both full and mixed cells are advected in the manner shown in Fig. 5.23a to update the C data. This technique also presented some difficulty in the case of multiple intersections and of clustering of markers in the same cell, and a reduction algorithm was implemented to limit their number.

In a more recent version of the method (Aulisa *et al.*, 2004) the area conservation constraint is applied on a small ribbon along the interface line, see Fig. 5.24a,b, where the points A,$1',2'$,B replace A,$1,2,3,4$,B by conserving the spanned area. The new algorithm is much faster, as only mixed cells are involved in the Lagrangian advection, and more precise, as markers are added or removed locally where this is needed, with no artificial removal of intersection markers. On the other hand, the coupling is not as strong as before, since the color function is now computed only at the end of the redistribution by collecting in couterclockwise order all intersection and conservation markers and cell corners; see Fig. 5.24b. The technique has been extended to three-dimensional Cartesian grids (Aulisa *et al.*, 2004) and to unstructured quadrangular grids (Aubert *et al.*, 2006). Figure 5.25 shows how the method performs in the vortex-in-a-box test with a counterclockwise flow and should be compared with Fig. 5.21. However, with these new

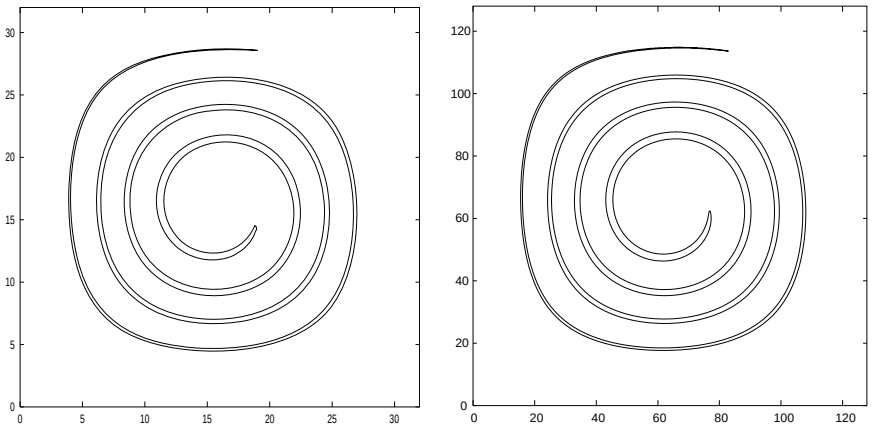


Fig. 5.25. Vortex-in-a-box test on a grid with 32^2 (left) and 128^2 (right) cells. The period is $T = 16$ and the rotation is counterclockwise.

features the method is basically a front-tracking method, which is the topic of the next chapter.

6

Advecting marker points: front tracking

Instead of advecting a marker function identifying the different fluids directly, as in the VOF method, the boundary between the fluids can be represented by connected marker points that are moved by the fluid. This approach is usually called *front tracking*, and in Chapter 4 we discussed the basic idea briefly and gave a short historical overview. In this chapter we describe front tracking in more detail.

The use of connected marker points to track the motion of a complex and deforming fluid interface can lead to several different methods, depending on the details of the implementation. Generally, however, front tracking involves the following considerations:

(i) *The data structure used to describe the front.* Although the use of marker particles simplifies many aspects of the advection of a fluid interface, other aspects become more complex. We use *front* to refer to the complete set of computational objects used to represent the interface. In addition to the marker points, the front often includes information about the connectivity of the points, as well as a description of the physics at the interface. The management of the front can be greatly simplified by the use of the appropriate data structure. There is a fundamental difference in the level of complexity between fronts in two dimensions and in three dimensions and, in general, any data structure can be made to work reasonably efficiently in two dimensions. For three-dimensional flows, the proper data structure can determine the difference between success or failure.

(ii) *The maintenance of a front as it is deformed, stretched, and compressed by the flow.* Generally, it is necessary to update the representation of the front by adding or deleting front-objects dynamically. Sometimes it is also necessary to reset the connectivity of the marker points and on occasion it

is desirable to smooth the front. All these operations depend critically on the flexibility of the data structure.

(iii) *The front interaction with the fixed grid, where the equations governing the fluid motion are solved.* In most cases the front is moved by a velocity field computed by solving the Navier–Stokes equations on a fixed grid and as the front moves it is necessary to update the grid values of the fluid properties on the fixed grid. Surface tension is usually also computed on the front and must be added to the fluid equations on the fixed grid. The most widely used way to accomplish this transfer of information between the moving front and the fixed grid is by approximating the front by a smooth distribution that can be represented on the fixed grid. The smoothing method, however, is not the only approach to transfer information between the front and the fixed fluid grid. The main alternatives are methods that attempt to maintain a sharp interface by modifying the numerical approximations used near the front – including extrapolating the field on one side of the front across it to create "ghost" values there so that all variables can be updated using regular stencils – and methods where the fixed grid is modified near the front. In this chapter we focus on the first approach.

(iv) *Topology changes.* When two blobs of the same fluid coalesce or one fluid blob breaks up into two or more blobs, the front must be restructured to reflect the changes in the topology of the interface. This has long been considered the main problem with explicit front-tracking, and we will end the chapter with a short discussion of the main issues.

When the fluid interface is tracked by marker points we work with two grids: the fixed grid, where the equations governing the fluid flow are solved, and the lower dimensional grid that tracks the interface. In Fig. 6.1, where one frame from a simulation of a buoyant bubble is shown, the unstructured triangulated front grid is clearly visible. One cross-sectional plane of the regular structured fluid grid, where the velocity field is plotted, is also shown. The front grid is used for the accurate advection of the fluid interface and the computations of surface tension. The evolution of the front is discussed in this chapter and the computation of the surface tension is discussed in Chapter 7.

6.1 The structure of the front

The representation of a two-dimensional front by marker points is fairly simple, and any data structure can generally be made to work efficiently. In three dimensions, however, the representation of the front is more critical for the successful implementation of the various operations required. Here, we first discuss briefly

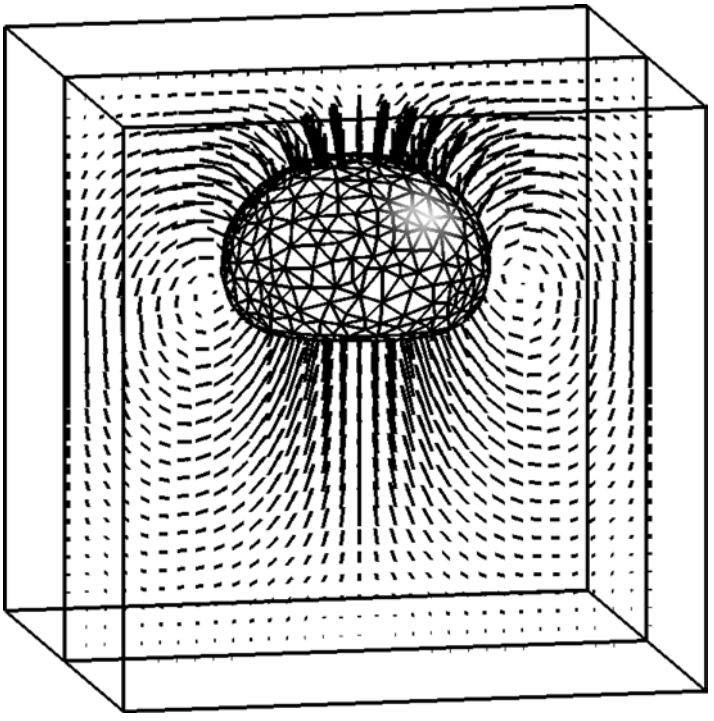


Fig. 6.1. Front-tracking simulation of a buoyant bubble. The bubble surface is tracked by an unstructured triangular grid, but the fluid equations are solved on a regular structured grid.

the representation of a two-dimensional front by an ordered array of points and then we present a more general approach using unstructured grids.

6.1.1 Structured two-dimensional fronts

A two-dimensional interface can be represented by an ordered array of marker points. The location of the points with respect to each other is given by their order, and computation of point separation, for example, is straightforward. Figure 6.2 shows a two-dimensional front constructed in this way. The only information that needs to be stored is the point coordinates

$$\mathbf{x}_i = (x_i, y_i), \quad \text{where} \quad i = 1, \ldots, N \tag{6.1}$$

and N is the total number of points used to represent the interface. Since the points are ordered, the points next to point i are points $i \pm 1$. If the simulations include more than one interface, then it is necessary to include information about

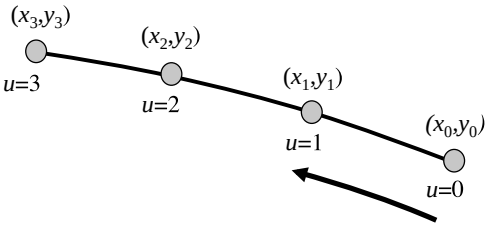


Fig. 6.2. The layout of an ordered string of material points used to represent a front in two dimensions.

where one interface ends and another one begins, or to use separate arrays for each interface.

To find the tangent to the front, or to insert a new point into the front, it is necessary to make some assumptions about the shape of the interface between the points. In preliminary work, or where a large number of points is used, linear interpolation, where the front is assumed to consist of straight-line segments, can be used. Linear interpolation, particularly when used to insert new points, may, however, lead to poor mass conservation and curvature fluctuations that can be undesirable if the surface tension is high. A higher order interpolation function is therefore often preferable.

Generally it is desirable to use a relatively "local" interpolation function that only depends on the points around the part of the curve where we are working. A simple Legendre interpolation usually works well. By numbering the points as shown in Fig. 6.2, an *n*th-order polynomial gives the position of any point on the curve by

$$\mathbf{x}^n(u) = \sum_{i=0}^{n} \prod_{j=0, j\neq i}^{n} \frac{u-u_j}{u_i-u_j}\mathbf{x}_i. \tag{6.2}$$

Here we have used an interpolation parameter u that takes integer values at each marker point, as shown in the figure. For a curve going through four points ($u = [0,1,2,3]$), this expression gives a cubic polynomial:

$$\begin{aligned}\mathbf{x}^3(u) \;=\; & -\frac{1}{6}(u-1)(u-2)(u-3)\mathbf{x}_0 + \frac{1}{2}u(u-2)(u-3)\mathbf{x}_1 \\ & -\frac{1}{2}u(u-1)(u-3)\mathbf{x}_2 + \frac{1}{6}u(u-1)(u-2)\mathbf{x}_3.\end{aligned} \tag{6.3}$$

The coordinates of the midpoint between points 1 and 2, at $u = 3/2$, are

$$\mathbf{x}^3(3/2) = \frac{1}{16}\Big[-\mathbf{x}_0 + 9(\mathbf{x}_1+\mathbf{x}_2) - \mathbf{x}_3\Big], \tag{6.4}$$

for example, and a tangent at $u = 3/2$ is easily found to be given by

$$\mathbf{t}^3(3/2) = \frac{\partial \mathbf{x}}{\partial u} = \frac{1}{24}\Big[-\mathbf{x}_0 + 27(\mathbf{x}_1 - \mathbf{x}_2) + \mathbf{x}_3\Big]. \tag{6.5}$$

Notice that $\mathbf{t}$ as computed by Equation (6.5) is not a unit tangent vector. The superscript "3" on the left-hand side in Equations (6.3), (6.4), and (6.5) denotes that we have used a third-order interpolation.

For two-dimensional fronts that do not stretch and deform much, the use of ordered points for the interface is particularly straightforward. The limitation of this approach becomes apparent when the front deforms and some points move too far apart and other parts of the front become crowded with points. To maintain accuracy, points must either be added and deleted or redistributed to maintain adequate resolution. Preventing the separation of the points from becoming too large by adding new ones is obviously the most important consideration. Deletion of points, however, is also important, both to reduce the total number of points and to prevent the formation of small-scale "wiggles" that sometimes arise if an interface is resolved much more finely than the underlying velocity field.

There are essentially two strategies available to maintain the resolution of a front consisting of ordered points: redistribute the points evenly in arc length, or insert (delete) points where the separation between the points becomes too large (small). When the points are redistributed evenly we need an interpolation parameter that represents the distance between the points. Ideally, this would be the arc length of the curve, but since it is fairly involved to find the exact arc length, we usually approximate it by straight-line segments between the points, so that

$$s_i = \sum_{j=1}^{i-1} \left[(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2)\right]^{1/2} \tag{6.6}$$

is most frequently used. If the total length is s_N then the distance between the points is $\Delta s = s_N/N$, assuming a closed interface. The locations of the new points are found by $\mathbf{x}_i^{\text{new}} = \mathbf{x}(i\Delta s)$, using the interpolation function that we have selected, such as (6.2). When points are inserted only as needed and other points stay put, the new points are usually put halfway between the old points. For a Legendre interpolation, a point between points 1 and 2 in Fig. 6.2 would be inserted at the location given by Equation (6.4). Similarly, when the separation between two points is so small that it is desirable to delete one of the points, the least disruptive action is usually to merge the two points into one, at a location determined by Equation (6.4). When points are added to or deleted from an ordered array, the coordinates of points with higher indices must be shifted to maintain the ordering. This can be tedious if many points are added or deleted at once.

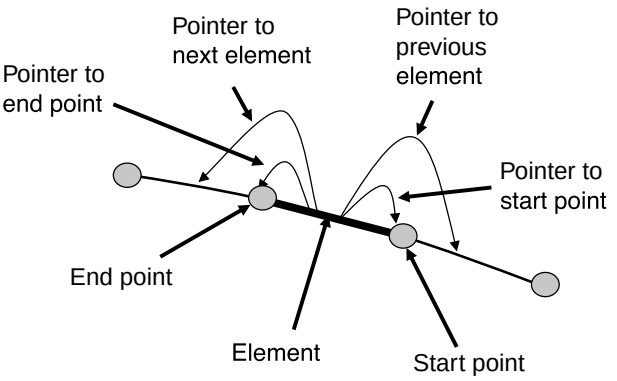


Fig. 6.3. The structure of a two-dimensional front. The points only "know" their coordinates. The elements, on the other hand, carry all information about the structure of the front, including pointers to the corner points and to the adjacent elements.

The representation of a curve in two dimensions is a highly developed subject and much more accurate representations are easily constructed. When relatively few points are used and high accuracy is required, a cubic spline polynomial, which has continuous first and second derivatives everywhere, is frequently a good choice. The polynomial is defined locally by

$$\mathbf{x}(s) = \mathbf{a}_i\, s^3 + \mathbf{b}_i\, s^2 + \mathbf{c}_i\, s + \mathbf{d}_i, \tag{6.7}$$

where the coefficients in the intervals between each pair of marker points are determined by requiring the function and the first and second derivatives to be continuous at the marker points. Usually, we choose the arc length (or an approximation, such as Equation (6.6)) as the interpolation parameter and find a separate set of coefficients for $x(s)$ and $y(s)$ by solving a system of tridiagonal linear equations. A detailed discussion of cubic splines and other high-order interpolation functions can be found in standard textbooks (Press *et al.*, 1993; Hammerlin and Hoffmann, 1991). Although high-order interpolation functions have been used to compute highly accurate solutions (Popinet and Zaleski, 1999), they are generally limited to two-dimensional flows and relatively simple problems. We will, therefore, not discuss them further here.

6.1.2 Unstructured fronts

The use of an ordered array to represent the front is generally impractical for three-dimensional surfaces, unless they remain essentially flat. For fronts that represent complex interfaces it is better to use unstructured triangular grids where the front consists of points connected by triangular elements. Although using a front

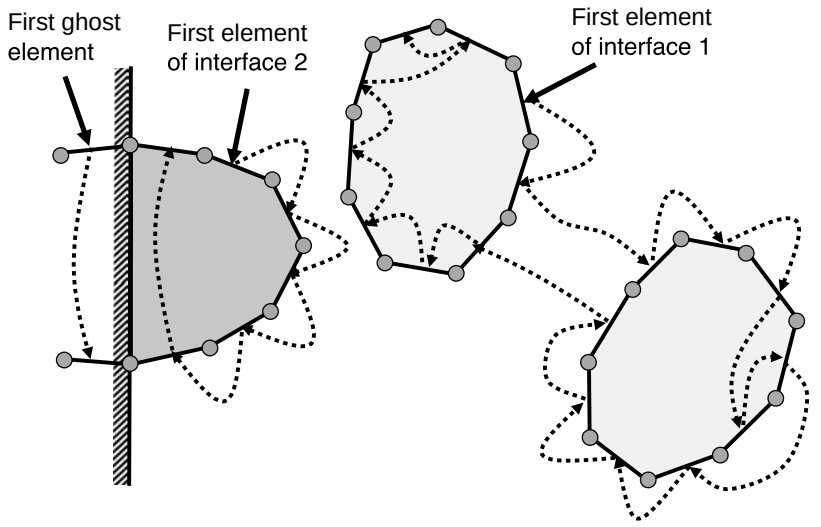


Fig. 6.4. The linked-list connections for two-dimensional front elements. The front consists of two interfaces and one set of ghosts elements. In principle the connections have no relations to the physical ordering of the elements.

consisting of both points and elements is something of an overkill for the two-dimensional case, it is easier to visualize the front structure when the elements connect only two points. We therefore start by examining one particular implementation of an unstructured two-dimensional front. The key variables are shown in Fig. 6.3. The front consists of points that are connected by elements. For each point, the only information stored is its coordinates. The elements, on the other hand, contain most of the front information. Each element knows about the points that it is connected to, as well as the local structure of the front, including its neighboring elements. The elements also contain information about the physical properties associated with the interface, such as the surface tension, change in the value of the marker function across the front, and any other quantities that are needed for a particular simulation. The elements are given a direction that defines the "outside" and the "inside" of the interface. For a given interface, all the elements must obviously have the same direction.

Since we need to add and delete front objects (points and elements) in the course of a simulation, it is easiest to store the objects in a linked list. In a linked list each object contains, in addition to its intrinsic properties such as coordinates for the points and information about the front structure for the elements, a pointer to the next object in the list. A linked list is particularly easily implemented as a structure in C or C++. In programming languages that do not include structures it

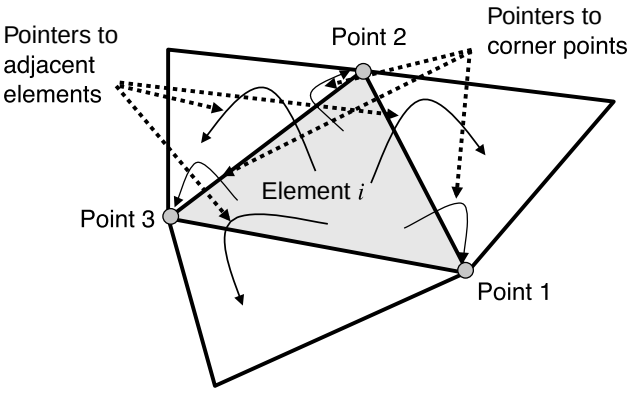


Fig. 6.5. The structure of a three-dimensional front. The points only "know" their coordinates. The elements, on the other hand, carry all information about the structure of the front, including pointers to the corner points and to the adjacent elements.

is necessary to maintain a separate array for each variable that is associated with the object, including the pointers to the next objects. Although the elegance of a single structure is lost, the actual work involved is only marginally increased. The order within the arrays is completely arbitrary, but one object must be designated as the first object in the front. The total number of objects in the front is stored and any operation involving the front is done by starting with the first object. The pointer to the next object in the linked list is then used to move to the next object and so on, until all the objects have been visited. The unused objects in the array are also linked and a pointer to the first unused object is stored. The use of a linked list makes the addition and removal of objects particularly simple, and while it is not absolutely necessary to maintain a pointer to the previous object also (a doubly linked list), deleting objects is simpler if we do.

The use of a linked list to store the front objects makes it particularly straightforward to manage more than one interface by simply adding pointers to the first object (selected arbitrarily) of each interface. We note that, generally, there is no need to treat physically distinct interfaces, such as those of two or more bubbles, as distinct computational structures, even if their physical properties, such as surface tension coefficients, are different. In some cases the interfaces may, however, require fundamentally different treatments, or we may want to introduce new fronts for computational reasons. For interfaces intersecting an external boundary, we do, for example, usually introduce ghost points and elements. In this case we put a ghost point outside the boundary and connect it to the point on the wall by a ghost element. These ghost objects are not included in the front that describes the fluid interface, but the front element connected to the wall point sees the ghost element

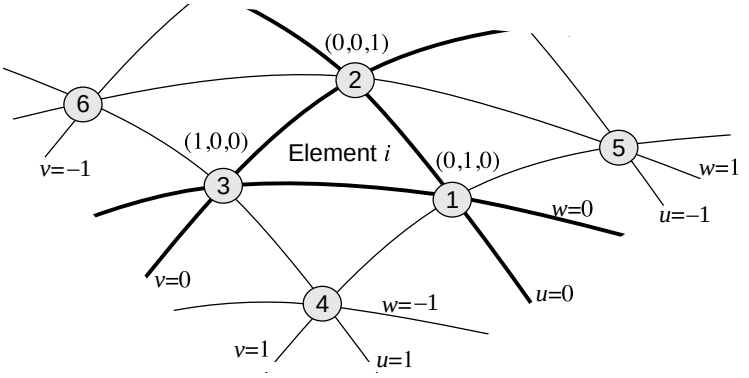


Fig. 6.6. Barycentric coordinates for a three-dimensional element. The edges of element i are defined by the $u = 0$, $v = 0$, and $w = 0$ lines.

as its neighbor. The ghost objects form a (usually small) linked list that allows us to access them in the same way as the regular front objects. The position of the ghost point is adjusted in such a way that the front tangent at the wall point has the desired value. In Fig. 6.4 we show a two-dimensional front used to represent two blobs of the same material and another blob whose properties are sufficiently different so that it must be treated as a different interface. The second interface intersects a domain boundary and the boundary conditions are enforced using ghost elements.

Three-dimensional fronts are built in the same way as two-dimensional fronts, except that three points are now connected by a triangular element. The points, again, only know their coordinates, but the elements know about their corner points and the elements that share their edges. Each element has an "outside" and an "inside" and all elements on a given interface must be oriented in the same way. The corners of each element and the edges are numbered counterclockwise (or clockwise – it is the consistency that matters!) when viewed from the "outside." Figure 6.5 shows the key variables that are stored for a three-dimensional front.

As for two-dimensional fronts, it is often necessary to assume the shape of the interface between the points. And, as for two-dimensional fronts, linear interpolation is sometimes acceptable. In most cases, however, more accuracy is necessary and a higher order interpolation must be used. To generate an interpolation function, using the three corner points of each element and the additional three corner points of the elements that share its edges, we can use the barycentric coordinates (u, v, w) defined in Fig. 6.6. The barycentric coordinate system is constructed locally for each element by laying down grid lines that go through the corner points of the element that we are considering and the elements that share an edge with that element.

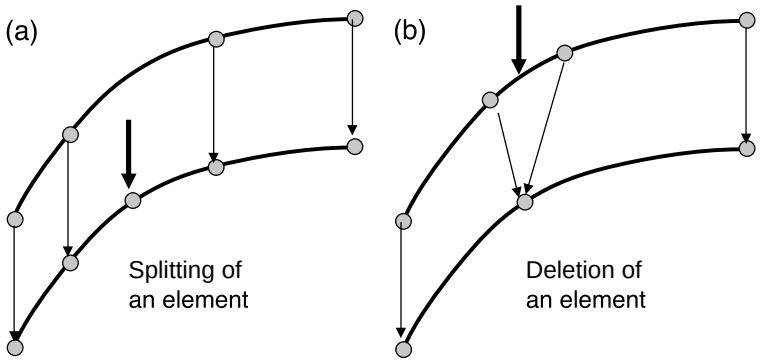


Fig. 6.7. Adding and deleting front elements in two dimensions. Here an element is (a) split by adding a point at the center of a long element and (b) a short element is deleted by merging its endpoints.

In the figure, points 1, 2, and 3 define element i and points 4, 5, and 6 belong to the adjacent elements. The edges of element i are given by the $u = v = w = 0$ lines. Notice that the coordinates are not independent and we must have

$$u + v + w = 1. \tag{6.8}$$

Any interpolated quantity p is given by

$$\begin{aligned} p(u,v,w) = {} & \frac{1}{2}(1-u)\left[-up_5 + (1-v)p_3 + (1-w)p_2\right] \\ & + \frac{1}{2}(1-v)\left[(1-u)p_3 - vp_6 + (1-w)p_1\right] \\ & + \frac{1}{2}(1-w)\left[(1-u)p_2 + (1-v)p_1 - wp_4\right] \end{aligned} \tag{6.9}$$

where p_1, p_2, ... are the values at the points. The centroid of the marked element is at approximately $u = v = w = 1/3$, or

$$\mathbf{x}(1/3,1/3,1/3) = \frac{1}{9}\Big[4(\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3) - (\mathbf{x}_4 + \mathbf{x}_5 + \mathbf{x}_6)\Big], \tag{6.10}$$

where we have taken $p = \mathbf{x}$ to be the point coordinates. The point halfway between corner points 1 and 2 is given by $u = v = 1/2$ and $w = 0$ or

$$\mathbf{x}(1/2,1/2,0) = \frac{1}{2}(\mathbf{x}_1 + \mathbf{x}_2) + \frac{1}{4}\mathbf{x}_3 - \frac{1}{8}(\mathbf{x}_5 + \mathbf{x}_6). \tag{6.11}$$

The coordinates of the midpoints of the other edges are found in the same way.

6.2 Restructuring the fronts

In general, an interface will stretch and deform as a result of the fluid motion. Stretching results in an increased separation of the marker points and eventually it is necessary to insert new points to resolve the interface adequately. When the interface is compressed, the points are crowded together, and although it is, in principle, not necessary to remove points, in practice it is generally better to do so. The refining of an unstructured grid is a highly developed subject, particularly for aerospace applications (e.g. Thompson *et al.*, 1998), but since the primary purpose of a front identifying a fluid interface is to carry information (as opposed to providing a grid on which the conservation equations are solved) the restructuring can be fairly simple.

For unstructured grids, where points are connected by elements, the size of an element is directly related to the separation of the front points and thus determines whether the grid must be refined or coarsened. Since the elements carry all connectivity information, it is therefore easiest to work with them. Thus, the addition or deletion of points is a result of splitting or removing elements. Figure 6.7 shows schematically a simple restructuring for a two-dimensional unstructured front. In (a) a large element is split by adding a point and in (b) an element is removed. The new point can be put at the average location of the end points of the element that is being split or removed, but (as for the structured fronts discussed above) this generally results in poor mass conservation and artificial pressure perturbations for high surface tension. It is better, therefore, to account for the curvature of the element and use Equation (6.3) to determine the location of the new point. Once the new point has been inserted, the various pointers are adjusted and the connectivity in the linked list reset.

For three-dimensional flows, the restructuring operations are more complicated. Not only can the addition and deletion of elements be done in a variety of ways, but the detection of when an action is needed can also be based on several different criteria. Figure 6.8 shows a relatively straightforward restructuring approach based on splitting an edge of an element to refine the front and collapsing an edge to coarsen it. The splitting of an edge results in the addition of one new front point and two new front elements and the collapse of an edge removes one point and two elements from the front. Sometimes it is also beneficial to reconnect the points by swapping edges to make the elements better shaped, leaving the number of points and elements unchanged. While the addition of elements is relatively straightforward, the deletion of elements can sometimes result in a mesh where two of the neighboring elements share an edge. This results in a grid that is generally unacceptable, since two of the points used for the interpolation described by Equation (6.11) are now the same. In those cases it is usually best to leave the element

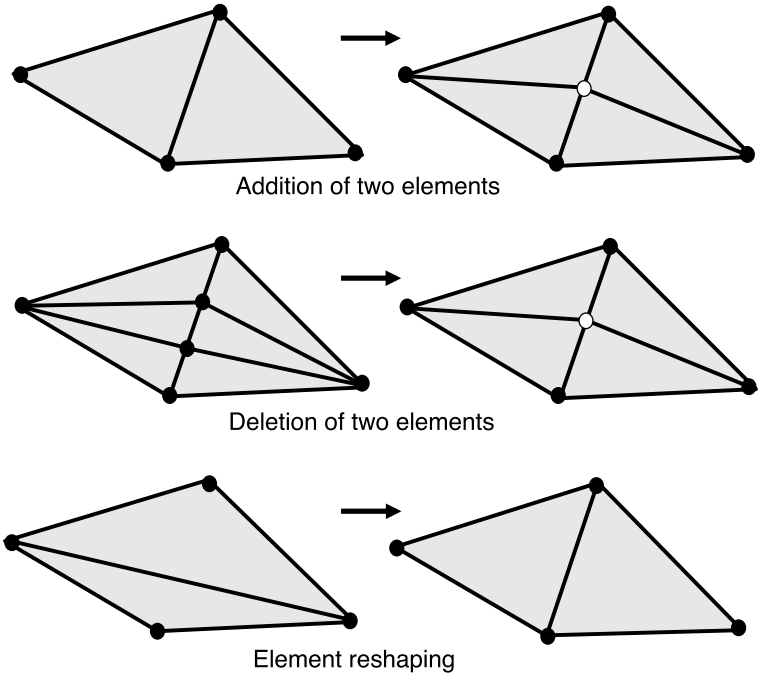


Fig. 6.8. The restructuring of a three-dimensional front. To increase the resolution the long edge of an element is split, thus adding one point and two new elements. To coarsen the grid the short edge of two elements is collapsed, eliminating one point and two elements. Element reshaping does not change the number of front objects.

under consideration untouched, or delete a different element. For relatively flat interfaces, the reshaping of elements by edge swapping shown in Fig. 6.8 usually results in improved fronts. In some cases, however, this is not the case. Consider, for example, a narrow ridge described by long and narrow elements whose longest edge runs parallel to the ridge. If the edges are swapped, then the edges will be perpendicular to the ridge, converting a smooth ridge into a jagged one. In addition to sometimes resulting in undesirable results, we get a comparable reshaping by the addition of elements, followed by element deletion. Explicit edge swapping, therefore, is frequently not needed.

For two-dimensional flows, the length of an element determines when elements are added or deleted. Usually, a minimum and a maximum element size is prescribed and an action is taken if the size of an element exceeds these limits. The limits are selected such that the resolution of the front is comparable to the fixed grid used to resolve the flow. Generally, two to four elements per grid mesh is a good rule of thumb. To determine whether it is necessary to add or delete an element for a three-dimensional front, it is possible to use the size of each element,

the length of its edges, or its shape. The shape can be measured by the "aspect ratio," defined as the length of the perimeter squared divided by the area of the element, normalized by $\sqrt{3} \times 12$, which is the aspect ratio of an equilateral triangle. Using the length of the edges and the aspect ratio to control the restructuring works well: if the length of the longest edge is longer than a preset value, it is split and two new elements added; if the shortest edge is shorter than a minimum value, it is collapsed and two elements deleted. If the aspect ratio of an element is larger than a minimum value and its shortest edge is larger than the minimum size, two elements are also added. A minimum edge length of a third of the grid spacing, a maximum length of a grid spacing, and a maximum normalized aspect ratio of $3/2$ usually works well. When edge swapping is included, it is done based on the aspect ratio.

Since the addition and deletion of elements (or front restructuring) generally results in changes to the neighboring elements, it is best first to identify all the elements that need to be modified and then do the modifications after all "bad" elements have been found. When the number of elements requiring modification is very large, sometimes it is worthwhile to examine the front again, after the first round of updating has been done. The time needed for the management of the front is generally a small fraction of the total time required for a multiphase flow simulation, so iterating twice is not a major time sink.

6.3 The front-grid communications

Since the front is advected by the velocity field computed on a fixed grid, and the location of the front specifies where the material properties of the fluid change, it is necessary to transfer information from the fixed grid to the front and from the front to the fixed grid. This can be done in a variety of ways. Here we discuss in detail the methods where the interface marks a smooth transition zone between one fluid and the other and no modification is required in the numerical approximations used to solve the fluid flow for grid points next to the front. "Sharp" interface methods where the flow solver is modified next to the front usually require a more elaborate communication scheme and are not discussed here.

Before delving into the details of the different methods, we first need to address a simple but extremely important operation. Namely, how to identify which grid points are close to a given front point.

6.3.1 Locating the front on the fixed grid

When the fixed grid is structured and regular, locating the grid point closest to an arbitrary front point is relatively straightforward. If we denote the total number of

grid points in one direction by N_x, the total length by L_x, and we take the grid point where $i = 0$ to correspond to $x = 0$, then the grid point to the left of front point x_f is given by:

$$i = \text{FLOOR}(x_f \times N_x / L_x) \tag{6.12}$$

where FLOOR stands for an operation rounding its operand *down* to its closest integer value. If the left-hand side of the grid is denoted by $i = 1$, instead of 0, or if the grids are staggered, small modifications are obviously needed.

In many cases we wish to simulate periodic domains where the front can move out of the domain on one side and reappear in through the other side. This can be done in a very simple way by recognizing that there is no need for the front to occupy the same period as the fixed grid. All that is needed is to correctly identify the grid point that corresponds to a given front position. A slight modification of the operation above accomplishes this:

$$i = \text{FLOOR}\left(\text{MOD}(x_f, L_x) \times N_x / L_x\right). \tag{6.13}$$

Here, MOD(*A*,*B*) is an operation that gives the remainder after *A* has been divided by *B*. For closed fronts, such as those representing the surface of a bubble or a drop, nothing else needs to be changed. For periodic fronts, the end point in one period is connected to the first point in the next period, but only one period is actually computed. When computing the length of such elements, or a curve is fitted through the end points, it is necessary, therefore, to correct for the positions of the points in the next period.

The strategy for identifying the fixed grid point closest to a given front point works also on irregular grids, as long as they are logically rectangular and can be mapped into a rectangular domain where the grid lines are straight. In those cases we simply store the mapped coordinates of each front point and use those when we need to communicate between the fixed grid and the moving front.

While locating a grid point closest to a front point is easy, going the other way – finding the front point closest to a given grid point – is much harder. Usually, it is possible to avoid doing so altogether by always going from the front to the grid. If, however, finding a front point closest to a given grid point is absolutely necessary, the elementary way is to examine all front points and find the one closest to the given grid point. For N_f front points it obviously takes $O(N_f)$ operations to do so for each point on the fixed grid. If we need to do this for every grid point (or most of them) it is much cheaper first to loop over the front and generate a (usually short) list of front points or elements closest to each grid point. When looking for front points closest to a given grid point, the search can be limited to the local list.

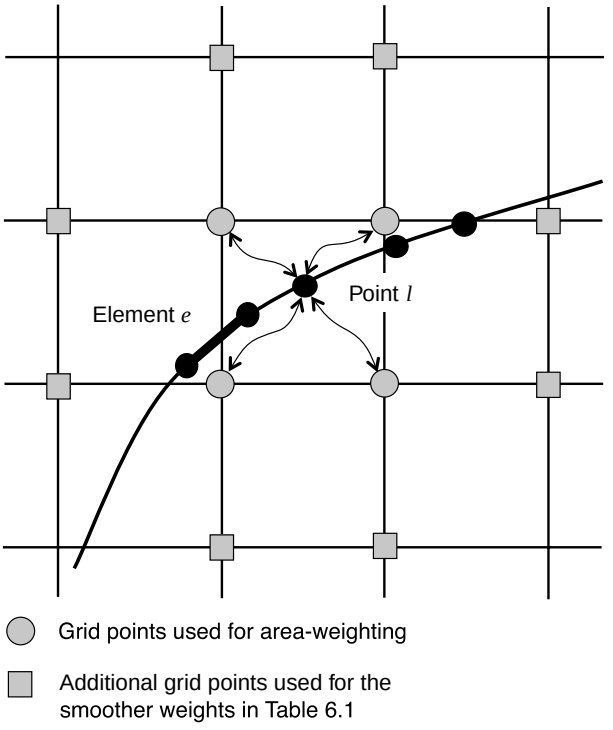


Fig. 6.9. A front and a fixed grid. Here we represent the front using elements connecting the front markers. Information is passed between the front points and the fixed grid.

6.3.2 Interpolation and smoothing

After the grid points that are closest to a given location on the front have been identified, information can be passed between the front and the fixed grid. In the smoothed interface approach, the transition zone between one fluid and the other is assumed to be sufficiently smooth so that when variables available on the fixed grid are needed on the front, they can simply be interpolated:

$$\phi_{\mathrm{f}}^{l} = \sum_{ij} w_{i,j}^{l} \phi_{i,j}. \tag{6.14}$$

Here, ϕ_{f}^{l} is a quantity on the front at location l, $\phi_{i,j}$ is the same quantity on the grid, $w_{i,j}^{l}$ is the weight of each grid point with respect to location l, and the summation is over the points on the fixed grid that are close to the front point. Figure 6.9 shows the location of the front with respect to the fixed grid. The velocity of a front point is found in this way, for example, with ϕ denoting the different velocity components.

The weights can be selected in several different ways, but are generally constrained to satisfy certain conditions. For interpolation it is reasonable to require

that the interpolated front value is bounded by the grid values and that the front value is the same as the grid value if a front point coincides with a grid point. The weights must also sum up to one:

$$\sum_{i,j} w_{i,j}^{l} = 1. \tag{6.15}$$

In addition to interpolating the velocities and other quantities from the fixed grid, it is often necessary to transfer quantities that exist on the front, such as surface tension, to the fixed grid. Since the front represents a δ-function, the transfer corresponds to the construction of an approximation to this δ-function on the fixed grid. This "smoothing" can be done in several different ways, but in many cases the transferred quantity is conserved and it is important to preserve the integrated value when moving from the front to the fixed grid.[1] The interface quantity ϕ_{f} is usually expressed in units per unit area (or length in two dimensions), but the grid value $\phi_{i,j,(k)}$ should be given in terms of units per unit volume. To ensure that the total value is conserved in the smoothing, we must require that

$$\int_{\Delta s} \phi_{\mathrm{f}}(s)\, \mathrm{d}s = \int_{\Delta v} \phi_{i,j,(k)}(\mathbf{x})\, \mathrm{d}v. \tag{6.16}$$

where Δs is the area (length in two dimensions) of the interface that is smoothed to volume (area in two dimensions) Δv. For two-dimensional flow, the discrete form of the grid value is therefore given by

$$\phi_{i,j} = \sum_{l} \phi_{\mathrm{f}}^{l} w_{i,j}^{l} \frac{\Delta s_l}{\Delta x \Delta y}, \tag{6.17}$$

where Δs_l is the area of element l and Δx and Δy are the grid spacings.

Although it is not necessary to do so, the same weighting function is usually used to interpolate values from the fixed grid to the front and to smooth front values onto the fixed grid. The number of grid points used in the interpolation depends on the particular weighting function selected. Since the weights have a finite support, there is a relatively small number of front elements that contribute to the value at each grid point. In actual implementations of the algorithm for transfer of quantities from the front to the grid, the process generally involves looping over the interface elements and adding the front quantity to the grid points that are near the front.

The quality of the transfer of information between the front and the fixed grid can depend on the interpolation weights selected. In early simulations using moving points and fixed grids the point value was occasionally simply assigned to the nearest grid point, but this is generally too crude. Using the four nearest points

[1] For the case when the transferred quantity is not conserved, see Chapter 7.

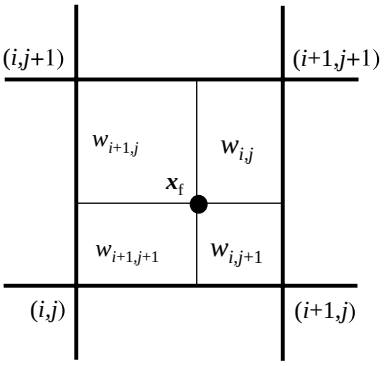


Fig. 6.10. The area weighting used to interpolate grid quantities to the front and to smooth front information onto the grid.

for two-dimensional situations (eight in three-dimensions) often works well, but smoother interpolation functions are sometimes needed.

In most cases the weighting functions are written as a product of one-dimensional functions. In two dimensions, for example, the weight for the grid point (i, j) for interpolation to $\mathbf{x}_{\rm f}^l = (x_{\rm f}^l, y_{\rm f}^l)$ is written as

$$w_{i,j}^l \left(\mathbf{x}_{\rm f}^l\right) = d(r_x)\, d(r_y) \tag{6.18}$$

where r_x is the scaled distance between $x_{\rm f}^l$ and the grid line located at x_i and r_y is the scaled distance between $y_{\rm f}^l$ and the grid line located at y_j. Here, and in what follows, we assume that the grid lines are all straight. If we assume a fixed grid spacing Δx and that $i = 0$ corresponds to $x = 0$, then $r_x = (x_{\rm f}^l - i\Delta x)/\Delta x$. r_y is given by a similar expression. The simplest interpolation scheme is the area (volume) weighting given by

$$d(r) = \begin{cases} 1-r, & 0 < r < 1, \\ 1+r, & -1 < r < 0, \\ 0, & |r| \geq 1\,, \end{cases} \tag{6.19}$$

The weights can be interpreted as the area fractions shown in Fig. 6.10.

Although area weighting works well in a large number of cases, sometimes it is desirable to use smoother functions. Smoother weighting functions have been developed in the context of at least three different applications: in the particle-in-cell method (PIC) originated by Harlow (1964), in the immersed-boundary method developed by Peskin (1977), and for vortex "blob" methods (see Nordmark (1991), for example). Table 6.1 shows three commonly used smoothing functions. The first two were introduced by Peskin and the third one, the cubic B-spline, has been used in PIC simulations (Brackbill and Ruppel, 1986). While there is some difference

Table 6.1. Smooth weighting functions for the transfer of information between the front and the fixed grid.

1	$d(r)=\begin{cases}(1/4)(1+cos(\pi r/2)), & \lvert r\rvert<2\\ 0, & \lvert r\rvert\geq 2\end{cases}$	Peskin (1977)
2	$d(r)=\begin{cases}d_1(r), & \lvert r\rvert\leq 1\\ 1/2-d_1(2-\lvert r\rvert), & 1<\lvert r\rvert<2\\ 0, & \lvert r\rvert\geq 2\end{cases}$ where $d_1(r)=\frac{3-2\lvert r\rvert+\sqrt{1+4\lvert r\rvert-4r^2}}{8}$	Peskin and McQueen (1989)
3	$d(r)=\begin{cases}\frac{2}{3}-\left(\lvert r\rvert\right)^2+\frac{1}{2}\left(\lvert r\rvert\right)^3, & 0\leq\lvert r\rvert\leq 1\\ \frac{1}{6}\left(2-\lvert r\rvert\right)^3, & 1\leq\lvert r\rvert\leq 2\\ 0, & \text{otherwise}\end{cases}$	Brackbill and Ruppel (1986)

between results of multifluid simulations using area weighting and the smoother distribution functions, the differences between the smoother ones are usually minimal. Although it is, at least in principle, desirable to have a grid approximation that is as compact as possible, a very narrow support, obtained by using only a few grid points close to the front, results in an increased grid effect. While area weighting works very well in most cases, the functions proposed by Peskin are obviously smoother and may sometimes yield better results. The area weighting, however, involves only two grid points in each direction and, therefore, is more efficient, particularly in three dimensions, where it requires values from eight grid points versus 27 for Peskin's interpolation functions. The compactness of the area weighting also results in both higher efficiency and simpler treatment near the boundaries of the computational domain, as compared with smoother distribution functions.

6.4 Advection of the front

Once the velocity of each front point has been found, its new position can be found by integration. If a simple first-order explicit Euler integration is used, then the new location of the points is given by

$$\mathbf{x}_{\mathrm{f}}^{n+1}=\mathbf{x}_{\mathrm{f}}^{n}+\mathbf{v}_{\mathrm{f}}^{n}\Delta t \tag{6.20}$$

where $\mathbf{x}_{\mathrm{f}}$ is the front position, $\mathbf{v}_{\mathrm{f}}$ is the front velocity, and Δt is the time step. Superscript n denotes the current time and $n+1$ denotes the end of the time step. In practice, higher order integration is generally used.

Unlike the VOF method, where the new cell averages of the marker function are found from the fluxes through the boundaries of the cells, advection of the marker

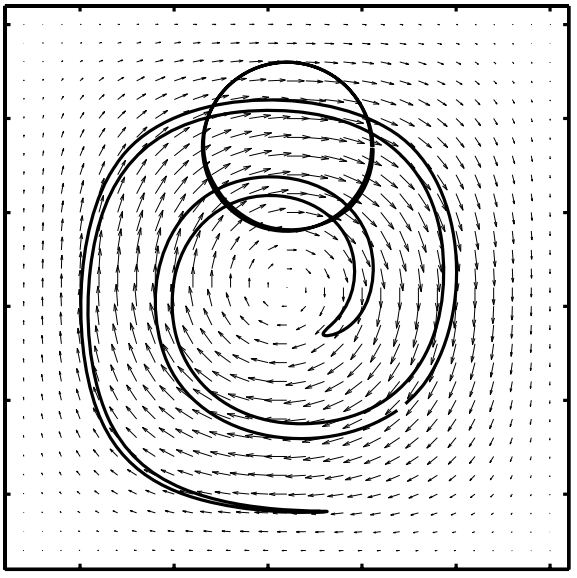


Fig. 6.11. Accuracy test for the advection of a front. The circular front is advected in a vortical flow where it is deformed. The flow is then reversed and the front is returned to its original shape.

function by integrating the motion of interface points does not guarantee its conservation. Accurate advection of the front points does, however, minimize this error. In some cases, particularly for very long runs with many bubbles or drops where the resolution of each particle is relatively low, the total mass change is unacceptably high. In these cases the size of the bubbles can be corrected every few time steps. Since the correction is very small at each time, the effect on the result is negligible. The inaccuracy in the advection of the front is due to errors coming from the interpolation of the velocities and the integration scheme. Increasing the accuracy of the front advection by using a higher order time-stepping method is straightforward. The error due to the interpolation comes from the fact that although the discrete velocity field may be divergence free (for incompressible flows), the interpolated velocity field is not necessarily divergence free. An interpolation scheme that produces a divergence-free velocity at the front points has been developed by Peskin and Printz (1993). The result, however, is a more complex pressure equation.

To test how well mass is conserved in actual simulations, we show in Fig. 6.11 the result of applying the test introduced by Bell *et al.* (1989). Here, a string of marker particles is advected in a unit square by the following velocity field derived from the stream function (5.49):

$$u = 2\cos(\pi t/T)\sin^2(\pi x)\sin(\pi y)\cos(\pi y),$$
$$v = -2\cos(\pi t/T)\sin^2(\pi y)\sin(\pi x)\cos(\pi x).$$

At the end of the period T, which is taken as 8 here, the marker particles should be exactly where they started, so the difference gives an indication of the accuracy of the advection. The markers initially form a circle of radius 0.15, located at $(x,y) = (0.5, 0.75)$. In Fig. 6.11, the original circle, the shape of the marker points at the maximum deformation, and the circle after the flow field has been reversed are shown for advection done using a second-order predictor–corrector scheme with $\Delta t = 0.005$. The velocity field is given on a 32^2 regular grid. Obviously, the differences are barely visible. The VOF method, as Fig. 5.21 shows, needs a much finer grid to produce results that are similar to those in Fig. 6.11.

Once the interface has been moved, the marker function is found as described in Section 6.5 and the various material properties can be set as functions of the marker function.

6.5 Constructing the marker function

When front tracking is used, the fluid properties, such as the density and viscosity, are not advected directly; instead, the boundary between the different fluids is moved. It is necessary, therefore, to reset these quantities at every time step. Generally, we first construct a marker function $I(\mathbf{x})$ that is a constant in each fluid and then use the marker function to set other properties. There are three main ways to construct a marker function from the location of the interface:

(i) If the marker function is treated as a point value (rather than the average in a cell) we can loop over the interface points and set the marker value on the fixed grid as a function of the shortest normal distance from the interface. Since the interface is usually restricted to move less than the size of one fixed grid mesh, this update can be limited to the grid points in the immediate neighborhood of the interface.

(ii) The gradient of the jump in the marker function across the interface can be distributed onto the fixed grid and the marker function recovered by integrating the gradient.

(iii) The front location can be used to construct the volume fraction of the cells through which it crosses.

The first approach is fairly straightforward, but does have one major drawback: when two interfaces are very close to each other, or when an interface folds back on itself, such that two front segments are between the same two fixed grid points, then the property value on the fixed grid depends on which interface segment is being considered. Since this situation is fairly common, a more general method is necessary and we will not discuss the first case further. The second approach is fairly well developed and has been used extensively, so we shall describe it in some detail. The same goes for the last procedure, although, as will become clear, it can still be developed further, particularly for three-dimensional flows.

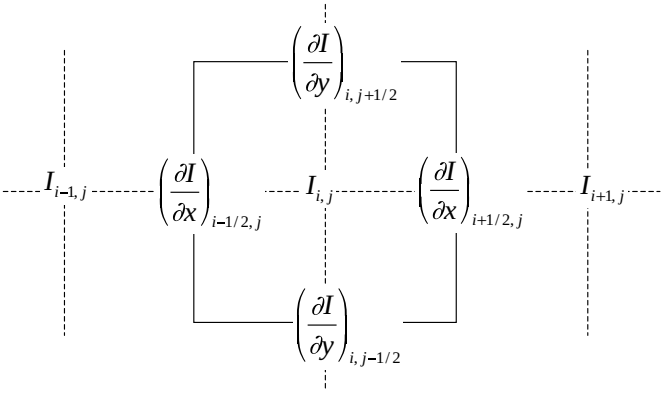


Fig. 6.12. The construction of a marker function from its gradient on a staggered grid.

6.5.1 Constructing the marker function from its gradient

To develop a method that sets the marker function correctly even when two interfaces lie close to each other, we use the fact that the front marks the jump in the marker function and that this jump is translated into a steep gradient on the fixed grid. If two interfaces are close to each other, the grid gradients simply cancel. The gradient can be expressed as

$$\nabla I = \int \Delta I \mathbf{n} \delta(\mathbf{x} - \mathbf{x}_\mathrm{f}) \, \mathrm{d}s. \tag{6.21}$$

where ΔI is the jump in the value of the marker function across the interface (usually a constant for each interface). The grid value of the components of the gradient are given by Equation (6.17) as described in the previous section:

$$(\nabla I)_{i,j} = \sum_l (\Delta I \mathbf{n})_l w^l_{i,j} \frac{\Delta s_l}{\Delta x \Delta y}. \tag{6.22}$$

Notice that the fixed grid can be different for the different components of the gradients. We can, in particular, use a staggered grid where the grid for the y component is shifted half a cell width upward and the grid for the x component is shifted half a cell to the right from grid point (i, j).

Once the grid-gradient field has been constructed, the marker field must be recovered. The simplest approach is to integrate the marker gradient directly from a point where the marker is known. If we distribute the gradient on a staggered grid, the x gradient is available at points $(i-1/2, j)$ and so on (see Fig. 6.12 for the two-dimensional case) and if the marker at $(i-1, j)$ is known, then

$$I_{i,j} = I_{i-1,j} + \Delta x \left(\frac{\partial I}{\partial x}\right)_{i-1/2,j}. \tag{6.23}$$

By a repeated application of this equation we can construct the marker value everywhere, provided we start at a point where the marker field is known. Although this operation only has to be performed at points near the front, since the marker away from the front has not changed, we need to find a point from which to integrate. The marker field can also depend slightly on the direction in which the integration is done, and errors can accumulate if we integrate over several grid points, so that the value of the marker function on the other side of the interface may not be correct.

To get a more symmetric expression for the marker at (i, j) we can integrate the density gradient from the other three grid points, $(i+1, j)$, $(i, j+1)$, and $(i, j-1)$, and average the results. The results can be rearranged to obtain

$$\frac{I_{i+1,j} + I_{i-1,j} - 2I_{i,j}}{\Delta x^2} + \frac{I_{i,j+1} + I_{i,j-1} - 2I_{i,j}}{\Delta y^2} = \frac{(\partial I/\partial x)_{i+1/2,j} - (\partial I/\partial x)_{i-1/2,j}}{\Delta x} + \frac{(\partial I/\partial y)_{i,j+1/2} - (\partial I/\partial y)_{i,j-1/2}}{\Delta y}, \quad (6.24)$$

which needs to be solved for $I_{i,j}$. Notice that Equation (6.24) is a discretization of

$$\nabla^2 I = \nabla \cdot (\nabla I), \quad (6.25)$$

where the Laplacian has been approximated by the standard second-order centered difference approximation and the divergence of the I gradient is found using the values at the half-points that came directly from the front.

While Equation (6.24) can be solved by any standard Poisson equation solver, it is usually not necessary to do that. The change in the marker function is confined to the cells that the interface crosses through, so it is sufficient to solve for I in those cells and their neighbors. The solution of Equation (6.24) in a narrow band around the interface can be accomplished by a three-step process. First, the interface cells are marked by looping over the front points and setting a flag for those cells through which the interface crosses. A one-directional linked list of interface cells is then generated by running through all the grid cells and linking marked cells. Finally, the marker function is found by solving Equation (6.24) iteratively in the marked cells, using the linked list to move from one marked cell to the next. The band of cells containing the front is usually only a few cells across and the solution generally converges rapidly. Since the interface moves, but no more than a cell width, it is usually also necessary to flag points next to the cells that contain the interface (see Fig. 6.13). To find the marker function near the boundary of the computational domain, we note that since Equation (6.24) is derived from Equation (6.23), the simplest way is not to include the wall direction for points next to the boundary.

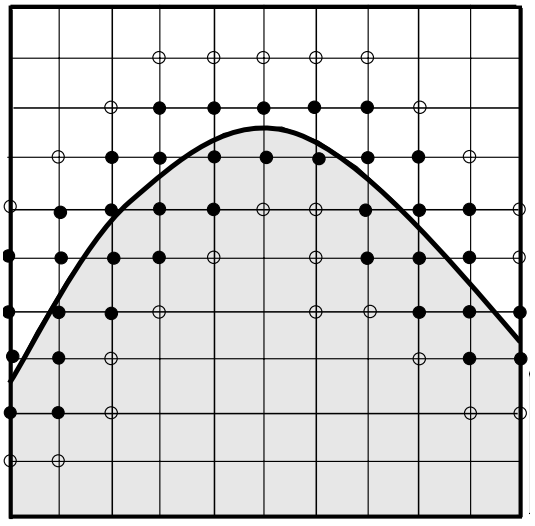


Fig. 6.13. The solution of Equation (6.24) using a narrow band around the front. The gradient has been smoothed onto the grid points marked by the black dots (assuming area weighting), but the points marked by open circles are usually also included to account for the motion of the front.

Figure 6.14 shows the construction of the marker field in a domain containing a circular drop. The first frame on the left shows the marker function gradient in the x-direction, after it has been distributed onto a 32^2 grid by the first distribution function in Table 6.1. The second frame shows the gradient in the y-direction. Finally, the marker function is recovered by solving Equation (6.24). We note that since the gradient at the front is distributed to the fixed grid from discrete front points, a large separation of the front points may result in some grid points not being assigned the correct amount of front gradient and the gradient may be slightly different than a gradient resulting from a dense distribution of front points. This can result in slight over- and under-shoots near the interface, particularly for compact distribution functions like the area weighting. If the marker function is reconstructed by solving Equation (6.24) everywhere in the computational domain, instead of only in a narrow band near the interface, we sometimes find a slight drift in the mean value of the marker function. This can be corrected by a post-processing step, if needed.

6.5.2 Construction of the volume fraction from the front location

Instead of reconstructing the indicator function from the location of the interface by integrating the smoothed gradient as described in the previous section, it is possible to use the location of the interface to construct the volume fraction in each

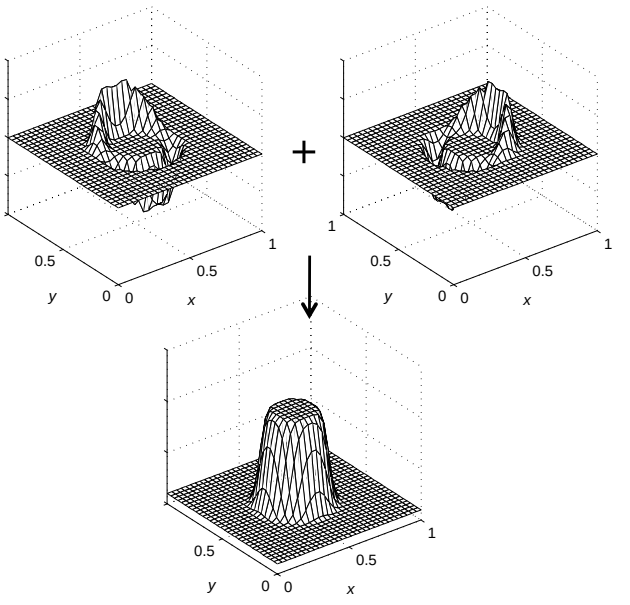


Fig. 6.14. The generation of a marker function (bottom) from the x and y gradients (top).

cell crossed by the interface. This, however, is not a completely trivial operation, although with enough programming it is clear that it can be done – at least for a two-dimensional flow and for an interface that crosses each cell at most once. The goal, however, is a method that is completely general, that allows an arbitrary number of crossings of each cell by the interface, and that requires minimal assumptions about the interface configuration in neighboring cells.

A front crossing a grid cell divides the cell into one or more regions where the marker function has one value and one or more regions where the marker has another value. In the following discussion we will take the value on the right of the front to be unity and the value on the left to be zero. A front consisting of straight-line segments along with the boundaries of the cell form a polygon, and the area of the polygon divided by the total area of the cell is the volume fraction in the cell under consideration. As Fig. 6.15 shows, the front can cross each cell more than once and the front can cross through any boundary in any direction. If the polygon is known, then the area is easily found. The area of any closed region is given by

$$\text{Area} = \int_\Omega \mathrm{d}v = \int_\Omega \nabla \cdot \mathbf{A}\, \mathrm{d}v = \oint \mathbf{A} \cdot \mathbf{n}\, \mathrm{d}s, \tag{6.26}$$

where we have introduced a vector $\mathbf{A}$, with the property that $\nabla \cdot \mathbf{A} = 1$, to allow us to write the integral as a contour integral. Several choices are possible for $\mathbf{A}$, the

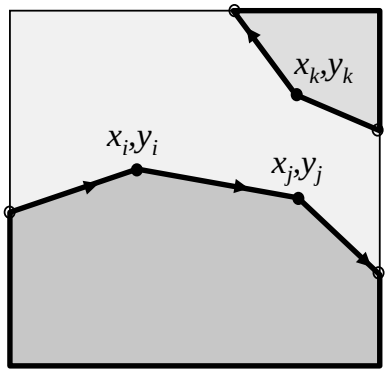


Fig. 6.15. The construction of a marker function in a cut cell. Given the location of the front, the volume fraction of the cell can be constructed as described in the text.

simplest being $\mathbf{A} = (x,0)$, $\mathbf{A} = (0,y)$, or $\mathbf{A} = (1/2)(x,y)$. These different choices, using $\mathbf{n} = (-\partial y/\partial s, \partial x/\partial s)$, result in

$$\text{Area} = \oint y\frac{\partial x}{\partial s}\,\mathrm{d}s = -\oint x\frac{\partial y}{\partial s}\,\mathrm{d}s = \frac{1}{2}\oint\left(y\frac{\partial x}{\partial s} - x\frac{\partial y}{\partial s}\right)\mathrm{d}s. \tag{6.27}$$

If we assume that the interface consists of linear elements, then the integrals can be evaluated using the midpoint rule:

$$\begin{aligned}\text{Area} &= \frac{1}{2}\sum_i (y_{i+1}+y_i)(x_{i+1}-x_i) = -\frac{1}{2}\sum_i (x_{i+1}+x_i)(y_{i+1}-y_i)\\ &= \frac{1}{4}\sum_i \left[(y_{i+1}+y_i)(x_{i+1}-x_i) - (x_{i+1}+x_i)(y_{i+1}-y_i)\right].\end{aligned} \tag{6.28}$$

The first expression in (6.28) is the sum of the areas below each element, the second form is the sum of the areas to the left of the elements, and the last form is the average of these areas.

The actual implementation of (6.28) into a general-purpose computer code is only simple for ordered arrays of points in two dimensions that cross each cell only once. In that case we can identify when the front enters a given grid cell and when it leaves, thus allowing us to determine which parts of the cell boundary belong to the polygon enclosed by the front and to compute the area. For interfaces that cross multiple times, the identification of where the cells cross the cell boundaries and in which order can become complex, as Fig. 6.15 shows. The problem of identifying where the interface crosses the cell boundaries and in what order is more complex when we work with an unstructured front and, of course, in three dimensions. While the algorithmic complexities are significant, some progress has been made, as discussed in the last chapter for the hybrid VOF–front-tracking method and in Note 2 in Section 6.7. Instead of attempting to find the intersection

of each triangular element with horizontal and vertical planes, we can divide each element into much smaller elements and take each such sub-element to lie fully within the cell where its centroid falls. This does obviously lead to a small error, not only in the interface cell, but also in all cells below it; but as the size of the sub-elements decreases, the error does also.

Once the volume fraction function has been constructed, the fluid properties can be set as a function of the volume fraction. Usually, a linear relationship is used, but sometimes it is better to use the harmonic mean for the viscosity, as discussed in Chapter 3.

6.6 Changes in the front topology

In general, numerical simulations of multiphase flow must account for topology changes of fluid interfaces when, for example, drops or bubbles coalesce or break up. When the interface is explicitly tracked by connected marker points, such changes must be accounted for by modifying the front in the appropriate way. The complexity of this operation is often cited as the greatest disadvantage of front-tracking methods. In methods that follow the interface by a marker function, topology changes take place whenever two interfaces, or different parts of the same interface, come closer than about one grid spacing. When front-tracking methods are used, no change takes place unless it is explicitly done. While automatic coalescence can be very convenient in some cases, particularly if the topology change does not need to be treated accurately, this is also a serious weakness of such methods. Coalescence is usually strongly dependent on how quickly the fluid between the coalescing parts drains, and simply connecting parts of the interface that are close may give the incorrect solution.

Topology changes in multifluid flows can be divided into two broad classes:

- *Films that rupture.* If a large drop approaches another drop or a flat surface, the fluid in between must be "squeezed" out before the drops are sufficiently close so that the film becomes unstable to attractive forces that can rupture it.
- *Threads that break.* A long and thin cylinder of one fluid will generally break by Rayleigh instability where one part of the cylinder becomes sufficiently thin so that surface tension "pinches" it in two.

The exact mechanisms governing how threads snap and films break are still not well understood. There are good reasons to believe that threads can become infinitely thin in a finite time and that their breaking is "almost" described by the Navier–Stokes equations. Films, on the other hand, are generally believed to rupture due to short-range attractive forces once they are a few hundred angstroms

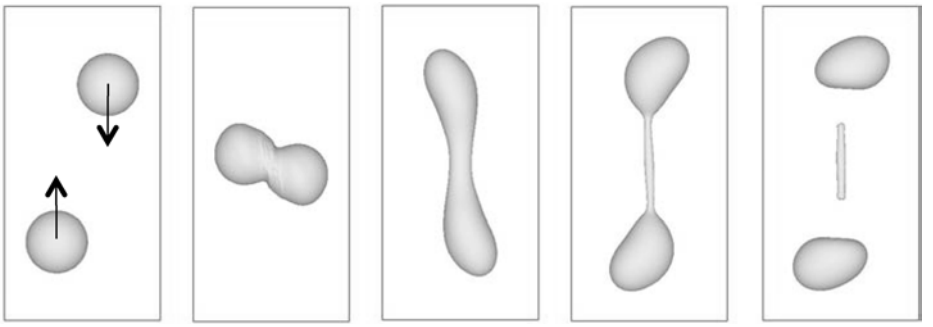


Fig. 6.16. The off-centered collision of two drops using a front-tracking method with a topology-change algorithm that allows the drops to first coalesce and then break up again. Time goes from left to right and the arrows in the first frame show the initial velocity of the drops. Figure courtesy of Dr. S. Mortazavi.

thick. These forces are usually not included in the continuum description, as discussed in Sections 2.8.2 and 9.2.2.

Accomplishing topology changes in a front-tracking code is a two-step process. First, the part of the front that should undergo topology change must be identified and then the actual change must be done. For simple problems, the region where a change should take place is often obvious and no special search technique is needed. In general, however, rupture or coalescence can take place anywhere and it is necessary to search the whole front to find where two fronts or two parts of the same front are close to each other. The simplest, but least efficient, way to conduct this search is to compute the distance between the centroids of every front element. For N front elements this is an $O(N^2)$ operation, but by dividing the computational domain into small subregions and looking only at the elements within each region, the efficiency of the search can be increased considerably. Another way is to use the fact that the grid gradients of the marker function due to close parallel fronts cancel. Thus, if we first construct a marker function from the front using the method described in Section 6.5 and then interpolate the marker function back onto the front, a value significantly different from 0.5 (assuming that the marker function takes the values 0 and 1) indicates that the front is close to another front. Once close fronts have been identified, we must decide if a topology change should take place. In principle, this should be determined based on whether a thread will snap or a film will rupture. In practice, the thread and the film are underresolved, and unless their evolution is accounted for by subgrid models of the type discussed briefly in Section 11.3 we are unlikely to have an accurate estimate of the thickness of the film or the diameter of the thread. In most front-tracking simulations involving topology changes, rupture, therefore, has been initiated when fronts are

less than a grid space apart. While this makes the results similar to what is seen when the marker function is advected directly (by VOF or level-set methods), the presence of the front makes it possible to change the rupture criteria and examine how sensitive the solution is to its exact value.

For two-dimensional flows, where the interface is simply a string of connected marker particles, the actual change in the topology of the interface is relatively easy. For three-dimensional flows the challenges are considerably greater. Such modifications are possible, however. Nobari and Tryggvason (1996) did, for example, develop an algorithm to coalesce two colliding drops, but the method was not very general and did not include the possibility of "pinching" thin threads (see Fig. 10.1). A more general topology-change algorithm was presented by Du *et al.* (2006) and implemented in the FronTier code. A different strategy was introduced by Shin and Juric (2002), who chose to abandon any effort to modify the front directly and instead recreated it periodically from a level surface of the grid marker function. This approach was used by Esmaeeli and Tryggvason (2004a) in their simulations of three-dimensional film boiling. In Fig. 6.16 we show several frames from a simulation of the collision of two drops that first coalesce and then break apart again, done using a generalization of the algorithm used by Nobari and Tryggvason (1996). Although topology changes are clearly possible in front-tracking simulations of fully three-dimensional flows, it is still a relatively unexplored field, and efficient and simple ways to do so would be desirable.

6.7 Notes

(1) Although the interpolation functions are usually constructed by a repeated multiplication of one-dimensional functions, it is also possible to use functions that are inherently multidimensional. Many such functions have been introduced as "blobs" for vortex methods where a singular point vortex is regularized by using a smooth approximation to the delta function. Cortez and Minion (2000), for example, used

$$\phi(r) = \frac{1}{\pi}(2 - r^2)\mathrm{e}^{-r^2} \tag{6.29}$$

and

$$\phi(r) = \frac{1}{6\pi}(24 - 36r^2 + 12r^4 - r^6)\mathrm{e}^{-r^2}. \tag{6.30}$$

(2) The construction of the volume fraction from the location of a front has recently been addressed by Aulisa *et al.* (2003a, 2004). For an alternative approach to construct the indicator function, see Ceniceros and Roma (2005).

7

Surface tension

The accurate computation of the surface tension is perhaps one of the most critical aspects of any method designed to follow the motion of the boundary between immiscible fluids for a long time. In methods based on the "one-fluid" formulation, where a fixed grid is used to find the motion, surface tension is added as a body force to the discrete version of the Navier–Stokes equations. Finding the surface force depends on whether the fluid interface is tracked by a direct advection of a marker function (VOF) or whether discrete points are used to mark the interface (front tracking). Here, we describe how to find surface tension for both VOF and front-tracking methods. At the end of the chapter we examine the performance of the various methods and the challenges in computing surface tension accurately and robustly.

7.1 Computing surface tension from marker functions

A *marker function* such as the color function C of the VOF method or the level-set function F is a function that indicates (marks) where the interface is. A whole family of methods for surface tension have been developed for the use of marker functions; however, these methods may also be used in connection with front-tracking methods. The standard approach is termed the continuous surface force (CSF) method. We also describe a variant that conserves momentum exactly, the continuous surface stress (CSS) method.

7.1.1 Continuous surface force method

For simplicity, we consider first the case where σ is constant. The surface tension force added to the Navier–Stokes equations is then from (2.62)

$$\mathbf{f}_\sigma \delta_S = \sigma \kappa \mathbf{n} \delta_S. \tag{7.1}$$

In the CSF method, the δ_S distribution is approximated by $|\nabla_h C|$ (in what follows the index h indicates a finite difference or numerical approximation). This approximation seems natural, since the color or marker function C approximates the Heaviside function H and we have as in (2.40)

$$\nabla H = -\delta_S \mathbf{n}. \tag{7.2}$$

Thus, we add to the velocity nodes, identified by the indices (i, j), the following force $\mathbf{f}_{i,j}$:

$$\mathbf{f}_{i,j} = \sigma \kappa^h |\nabla_h C| \mathbf{n}^h, \tag{7.3}$$

where κ^h is an approximation (to be found) of the local curvature and $\mathbf{n}^h$ is an approximation of the normal. As we shall see below, it is sometimes necessary to smooth this approximation of the δ-function. However, a simple (apparently even naive) method results without any smoothing. Using an elementary finite-difference estimate of the normal similar to that in Section 5.2.2.1 yields

$$\mathbf{n}^h = -\frac{\nabla_h C}{|\nabla_h C|}, \tag{7.4}$$

with properly centered finite differences (see below), and thus

$$\mathbf{f}_{ij} = -\sigma \kappa^h \nabla_h C. \tag{7.5}$$

This unsmoothed version allows a remarkable exact balance between pressure and surface tension in a special case. On a spherical droplet or round cylinder, $\kappa = \kappa_0$ is a constant. In the static case ($\mathbf{u} = 0$ everywhere) without gravity, the pressure forces balance exactly the surface tension around the droplet. The static equilibrium equation is

$$-\nabla p + \mathbf{f}_\sigma \delta_S = 0. \tag{7.6}$$

Under the condition that κ^h is replaced by a constant κ_0, the discretization of Equation (7.6) yields

$$-\nabla_h p - \sigma \kappa_0 \nabla_h C = 0. \tag{7.7}$$

Using indices:

$$-\frac{p_{i+1,j} - p_{i,j}}{\Delta x} - \frac{\sigma \kappa_0}{\Delta x}(C_{i+1,j} - C_{i,j}) = 0, \tag{7.8}$$

$$-\frac{p_{i,j+1} - p_{i,j}}{\Delta y} - \frac{\sigma \kappa_0}{\Delta y}(C_{i,j+1} - C_{i,j}) = 0. \tag{7.9}$$

This can be integrated to

$$p_{i,j} = p_2 - \sigma \kappa_0 C_{i,j}, \tag{7.10}$$

where p_2 is the pressure in the $C = 0$ region. Equation (7.10) approximates nicely the continuum solution $p(\mathbf{x}) = p_2 - \sigma\kappa_0 H(\mathbf{x})$, yielding $p_{i,j} = p_2 - \sigma\kappa_0$ inside the droplet and $p_{i,j} = p_2$ outside it as per Laplace's law (recall that with the reference phase, fluid "1," inside the droplet and the normal pointing outwards the curvature κ_0 is negative; see Fig. 2.5). The three-dimensional case is just as simple.

An approximation for κ remains to be found. There are many suggestions, but a particularly simple one is to use the curvature–divergence relation of Appendix A, Equation (A.32), and obtain

$$\mathbf{f}_{ij} = \sigma(\nabla_h \cdot \mathbf{n}^h)\nabla_h C, \tag{7.11}$$

where $\mathbf{n}^h$ is an approximation of the normal. Beyond VOF, a CSF method may be constructed with any interface scheme. Any indicator function I, for instance the level-set function, may be used instead of the color function C. This yields trivially the normal

$$\mathbf{n}^h = -\frac{\nabla_h I}{|\nabla_h I|}. \tag{7.12}$$

It is often practical to use the density ρ to identify the different fluids. In that case it is useful to notice that

$$\rho = \rho_1 H + \rho_2(1-H) = \rho_2 - [\rho]_S H, \tag{7.13}$$

where $[\rho]_S = \rho_2 - \rho_1$ and thus

$$\nabla\rho = -[\rho]_S \nabla H. \tag{7.14}$$

A discretization of $\mathbf{f}_\sigma \delta_S$ is therefore

$$\mathbf{f}_{ij} = -\sigma \frac{\nabla_h \rho}{[\rho]_S} \nabla_h \cdot \mathbf{n}^h. \tag{7.15}$$

Below is a detailed implementation of this naive CSF method. We need $\mathbf{n}^h$ at the faces of the cells and thus we compute a *non-unit* normal with components

$$m_{x:i+1/2,j} = -\frac{C_{i+1,j} - C_{i,j}}{\Delta x} \quad \text{and} \quad m_{y:i,j+1/2} = -\frac{C_{i,j+1} - C_{i,j}}{\Delta y}. \tag{7.16}$$

The components on faces where they are not available are found by interpolation, as is standard in the MAC method. For example:

$$m_{x:i,j+1/2} = \frac{1}{4}\left(m_{x:i+1/2,j} + m_{x:i+1/2,j+1} + m_{x:i-1/2,j} + m_{x:i-1/2,j+1}\right). \tag{7.17}$$

The unit normal is computed as

$$\mathbf{n}_{i+1/2,j} = \frac{\mathbf{m}_{i+1/2,j}}{|\mathbf{m}_{i+1/2,j}|}, \tag{7.18}$$

where

$$|\mathbf{m}_{i+1/2,j}| = (m^2_{x:i+1/2,j} + m^2_{y:i+1/2,j})^{1/2}, \tag{7.19}$$

$$|\mathbf{m}_{i,j+1/2}| = (m^2_{x:i,j+1/2} + m^2_{y:i,j+1/2})^{1/2}. \tag{7.20}$$

These expressions cannot be used in cells where $\mathbf{m}_{i+1/2,j} = 0$, which will happen at some distance from the interface. A simple workaround is to use

$$\mathbf{n}_{i+1/2,j} = \frac{\mathbf{m}_{i+1/2,j}}{|\mathbf{m}_{i+1/2,j}| + \varepsilon_{\text{tiny}}}, \tag{7.21}$$

where $\varepsilon_{\text{tiny}}$ is a tiny floating-point number. Better methods would distinguish between cells where ∇C is nonzero and this computation of the normal makes sense, and cells where $\nabla C = 0$ and this computation of the unit normal has to be avoided.

Finally, the curvature in the cell center is computed as the discrete divergence of the unit normal:

$$\kappa_{i,j} = -\left[\frac{1}{\Delta x}(n_{x:i+1/2,j} - n_{x:i-1/2,j}) + \frac{1}{\Delta y}(n_{y:i,j+1/2} - n_{y:i,j-1/2})\right]. \tag{7.22}$$

The face-centered values of the curvature are found as averages from the cell-centered values. The surface tension is then computed at the cell faces as required for the staggered grid:

$$f_{x:i+1/2,j} = -\frac{\sigma}{2\Delta x}(C_{i+1,j} - C_{i,j})(\kappa_{i+1,j} + \kappa_{i,j}) \tag{7.23}$$

$$f_{y:i,j+1/2} = -\frac{\sigma}{2\Delta y}(C_{i,j+1} - C_{i,j})(\kappa_{i,j+1} + \kappa_{i,j}). \tag{7.24}$$

7.1.2 Continuous surface stress method

Instead of discretizing the force representation of the surface tension, Equation (2.62), one may start from Equation (2.51):

$$\mathbf{f}_\sigma \delta_S = \nabla \cdot \mathbf{T}^\sigma_S \delta_S = \nabla \cdot [\sigma(\mathbf{I} - \mathbf{nn})\delta_S]. \tag{7.25}$$

The expression is valid even when σ varies. The CSS is the discretization of (7.25) without any smoothing:

$$\mathbf{f}_{ij} = \nabla_h \cdot (\mathbf{T}^\sigma_S |\nabla_h C|) = \nabla_h \cdot [\sigma(\mathbf{I} - \mathbf{n}^h \mathbf{n}^h)|\nabla_h C|], \tag{7.26}$$

where $\mathbf{n}^h$ is a properly centered numerical approximation of $\mathbf{n}$. The advantage is then that we have a conservative method, and that it is not necessary to compute a curvature. A disadvantage, however, is that a simple static solution of the form (7.10) cannot be found.

Let us give the detailed implementation. The gradient of C is computed now on cell corners

$$\frac{\partial}{\partial x}C_{i+1/2,j+1/2} = \frac{C_{i+1,j} - C_{i,j} + C_{i+1,j+1} - C_{i,j+1}}{2\Delta x}, \tag{7.27}$$

$$\frac{\partial}{\partial y}C_{i+1/2,j+1/2} = \frac{C_{i,j+1} - C_{i,j} + C_{i+1,j+1} - C_{i+1,j}}{2\Delta y}, \tag{7.28}$$

with similar expressions for the other corners. The unit normal is computed as in (7.18). Then the stress tensor at cell corners is

$$\mathbf{T}^{\sigma}_{S:i+1/2,j+1/2} = \sigma(\mathbf{I} - \mathbf{n}_{i+1/2,j+1/2}\mathbf{n}_{i+1/2,j+1/2})|\nabla_h C|, \tag{7.29}$$

where $|\nabla_h C|$ is computed from the components in Equations (7.27) and (7.28). We want to compute the divergence of $\mathbf{T}^{\sigma}_{S}$ on cell faces; thus, we need the off-diagonal component $T^{\sigma}_{S:xy}$ on cell corners and the diagonal components $T^{\sigma}_{S:xx}$ and $T^{\sigma}_{S:yy}$ on cell centers. These are computed by averaging as in

$$T^{\sigma}_{S:xx:i,j} = \frac{1}{4}\left(T^{\sigma}_{S:xx:i+1/2,j+1/2} + T^{\sigma}_{S:xx:i-1/2,j+1/2} + T^{\sigma}_{S:xx:i+1/2,j-1/2} + T^{\sigma}_{S:xx:i-1/2,j-1/2}\right) \tag{7.30}$$

and

$$\begin{aligned} T^{\sigma}_{S:yy:i,j} \;=\; & \frac{1}{4}\Big(T^{\sigma}_{S:yy:i+1/2,j+1/2} + T^{\sigma}_{S:yy:i-1/2,j+1/2} + T^{\sigma}_{S:yy:i+1/2,j-1/2} \\ & + T^{\sigma}_{S:yy:i-1/2,j-1/2}\Big). \end{aligned} \tag{7.31}$$

Then the surface tension at the cell faces is

$$f_{x:i+1/2,j} = \frac{1}{\Delta x}(T^{\sigma}_{S:xx:i+1,j} - T^{\sigma}_{S:xx:i,j}) + \frac{1}{\Delta y}(T^{\sigma}_{S:xy:i+1/2,j+1/2} - T^{\sigma}_{S:xy:i+1/2,j-1/2}), \tag{7.32}$$

$$\begin{aligned} f_{y:i,j+1/2} \;=\; & \frac{1}{\Delta y}(T^{\sigma}_{S:yy:i,j+1} - T^{\sigma}_{S:yy:i,j}) \\ & + \frac{1}{\Delta x}(T^{\sigma}_{S:xy:i+1/2,j+1/2} - T^{\sigma}_{S:xy:i-1/2,j+1/2}). \end{aligned} \tag{7.33}$$

7.1.3 Direct addition and elementary smoothing in the VOF method

In the CSF and CSS methods described above, as well as in the proper representation of surface tension (PROST) method described later, components of the force are computed directly at the velocity nodes and added at the predictor step. This method, however, gives unsatisfactory results for CSF and CSS, as shown in the

tests in Section 7.3. A simple type of smoothing that somewhat improves the results is performed by taking the first neighbor points. In two dimensions the smoothed C is

$$\tilde{C}_{i,j} = \frac{1}{2}C_{i,j} + \frac{1}{8}(C_{i,j-1} + C_{i,j+1} + C_{i-1,j} + C_{i+1,j}). \tag{7.34}$$

This "five-point smoothing" will be used in the tests of CSF and CSS below.

In general, two approaches are used: either the force is first computed from the raw interface data and then distributed, or the interface data are smoothed before the force is computed. In both cases the force is then, in effect, distributed over a zone around the interface with a thickness of a few grid cells. However, neither the PROST method nor the pressure correction marker method (PCMM), discussed at the end of this chapter, require smoothing.

7.1.4 Weighted distribution in the VOF method: kernel smoothing

In the CSF method, the expression $|\nabla_h C|$ was used to approximate the δ-function. In order to smooth the spatial distribution of this approximation of the δ-function we may convolute C with a smoothing kernel. The continuous version of smoothing is the transformation

$$\tilde{H}(\mathbf{x}) = (H * K)(\mathbf{x}) = \int_V H(\mathbf{x}')K(\mathbf{x}-\mathbf{x}';\varepsilon)\,\mathrm{d}\mathbf{x}', \tag{7.35}$$

where $K(\mathbf{x};\varepsilon)$ is an integration kernel of width ε verifying $K \to \delta_S$, when $\varepsilon \to 0$. Because convolution and differentiation commute, smoothing a step function with a kernel is the same as transferring a δ-function onto a grid using weighting functions, as discussed in Chapter 6. Kernels are weighting functions and vice versa.

Many kernels have been proposed for the CSF method. The one we used in the tests of the CSF and CSS methods is

$$K_8(\mathbf{x}) = A(\varepsilon)\left[1 - \left(\frac{|\mathbf{x}|^2}{\varepsilon^2}\right)\right]^4 \quad \text{for} \quad |\mathbf{x}| < \varepsilon \tag{7.36}$$

and $K_8(\mathbf{x}) = 0$ for $|\mathbf{x}| > \varepsilon$, where ε is the width of the filter and $A(\varepsilon)$ is a normalization constant. A graphical representation of the kernel action is shown on Figure 7.1. A discrete approximation of the kernel smoothing is

$$\tilde{C}_{ij} = A(\varepsilon)\sum_m\sum_l C_{lm}\left[1 - \frac{x_{il}^2 + y_{jm}^2}{\varepsilon^2}\right]^4 h^2, \tag{7.37}$$

where $x_{il} = x_i - x_l$, $y_{jm} = y_j - y_m$ with obvious notations. The sum is over all l, m

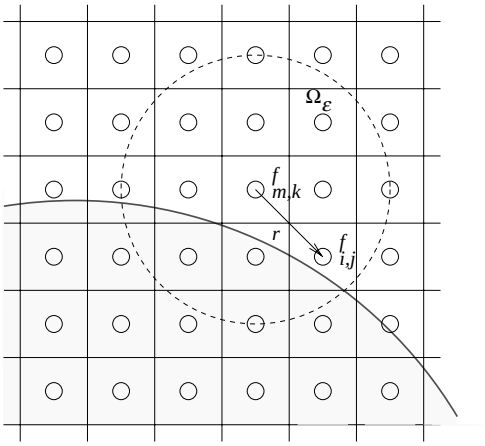


Fig. 7.1. Smoothing of the color function by a kernel.

such that (x_l, y_m) is in the disk Ω_ε of radius ε. The normalization constant $A(\varepsilon)$ is chosen so that

$$A(\varepsilon)\sum_m\sum_l\left[1-\frac{x_{il}^2+y_{jm}^2}{\varepsilon^2}\right]^4 h^2 = 1. \tag{7.38}$$

The derivatives of C needed in the CSF method may be computed by finite differences from the $\tilde{C}$ function, or directly from derivatives of the kernel. This second method has been tested by us for the gradient $\nabla\tilde{C} = (\partial\tilde{C}/\partial x, \partial\tilde{C}/\partial y)$:

$$\frac{\partial}{\partial x}\tilde{C}_{ij} = -8A(\varepsilon)h^2\sum_m\sum_l C_{lm}\left[1-\frac{x_{il}^2+y_{jm}^2}{\varepsilon^2}\right]^3\frac{x_{il}}{\varepsilon^2}, \tag{7.39}$$

$$\frac{\partial}{\partial y}\tilde{C}_{ij} = -8A(\varepsilon)h^2\sum_m\sum_l C_{lm}\left[1-\frac{x_{il}^2+y_{jm}^2}{\varepsilon^2}\right]^3\frac{y_{jm}}{\varepsilon^2}. \tag{7.40}$$

The smoothed approximation to the unit normal $\tilde{\mathbf{n}}$ follows without ambiguity. For the numerically estimated curvature we may again either differentiate analytically the kernel or use finite differences. The direct derivation of the kernel has been found too imprecise; thus, we use again the curvature–divergence relation of Equation (A.32), and

$$\kappa_{i,j} = -\left[\frac{1}{2h}(\tilde{n}_{x:i+1,j}-\tilde{n}_{x:i-1,j})+\frac{1}{2h}(\tilde{n}_{y:i,j+1}-\tilde{n}_{y:i,j-1})\right]. \tag{7.41}$$

The final result is located at cell centers. The curvature value at cell faces is obtained by averaging. With this smoothing, the normal has been approximated in

a band of width ε around the surface S, but the curvature will be approximated correctly only in a thinner band, say of thickness $\varepsilon - 2$. Thus, it is necessary to construct an approximated δ-function which is narrower than $\nabla\tilde{C}$. In our implementation we take

$$\tilde{\delta}_{S:i,j} = 6\tilde{C}_{ij}(1-\tilde{C}_{ij})|\nabla\tilde{C}_{ij}|. \tag{7.42}$$

This expression is peaked in the center of the band. Provided $\tilde{C}$ is monotonic, it satisfies the normalization condition

$$\int_{-\varepsilon}^{\varepsilon} \tilde{\delta}_S(n)\,\mathrm{d}n = \int_0^1 6\tilde{C}(1-\tilde{C})\,\mathrm{d}\tilde{C} = 1 \tag{7.43}$$

where n is a coordinate perpendicular to the interface.

7.1.5 Axisymmetric interfaces

In addition to two- and three-dimensional geometries, we often examine problems which can be assumed to exhibit axisymmetry. Although most of what has been said about two-dimensional flows applies, there are important differences. Using (A.24)

$$\sigma\kappa\mathbf{n}\delta_S = \sigma\left(\kappa_1 + \frac{r}{\cos\theta}\right)\mathbf{n}\delta_S, \tag{7.44}$$

where r is the radial coordinate in cylindrical (r,ϕ,z) coordinates and θ the angle between the normal $\mathbf{n}$ and the z axis. This can be rewritten as

$$\sigma\kappa\mathbf{n}\delta_S = \sigma\left(\nabla_2\cdot\mathbf{n} + \frac{r}{n_z}\right)\mathbf{n}\delta_S, \tag{7.45}$$

where $\nabla_2 = (\partial_r, \partial_z)$ is the two-dimensional gradient in the meridian plane. The methods described in this chapter for two-dimensional geometries can then be adapted to compute the axisymmetric force.

7.2 Computing the surface tension of a tracked front

When the front is represented by markers, the computation of the surface tension is, for the most part, a matter of using the discrete points to approximate geometric quantities. There are, however, a few different alternatives, and the transfer of the surface force to the discrete Navier–Stokes equation on the fixed grid can also be done in several ways.

We usually think of the net force due to surface tension as being proportional to the curvature of the interface. However, we actually rarely need the pointwise

value of the curvature. In most cases it is the total force on a small segment of the front, Δs, that is needed. Thus, the challenge is to find

$$\delta \mathbf{f}_e = \int_{\Delta s} \sigma \kappa \mathbf{n} \, \mathrm{d}s, \tag{7.46}$$

where σ is the surface tension coefficient, κ is the curvature for a two-dimensional front and twice the mean curvature for a three-dimensional surface, $\mathbf{n}$ is the normal to the interface, and Δs is the length of a surface element in two dimensions and the area of an element in three dimensions. A conceptually straightforward approach, where we find the curvature at a point on the front and multiply it by the area associated with that point (the front segment) has two problems. It is, first of all, relatively difficult to find a good numerical approximation to the curvature, given that the front points are usually not evenly spaced and that the front may not be perfectly represented. The second problem is that this approach does not guarantee that the net force on a closed surface with constant surface tension is zero, as it should be. This is analogous to the difference between the CSF and the CSS method for finding surface tension in VOF methods, where the latter is conservative and the former is not. Both difficulties can be overcome by rewriting the integral over the front segment as a contour integral over the boundary of the segment. It is intuitively obvious that it is possible to compute the force on the segment by examining only the boundaries. Surface tension is, by its definition, a force that pulls on the boundary of a segment cut out of a surface in a direction that is normal to the boundary but tangent to the surface. For a flat interface (and constant surface tension), the pull on the edges of a flat interface segment is, therefore, balanced out and the net force is zero. When the interface is curved, the pulls on the different edges of a surface segment do not exactly balance each other, resulting in a net force normal to the element.

Although the concept is the same for two- and three-dimensional fronts, the derivation and the numerical implementation is slightly different, so we will deal with those two cases separately.

7.2.1 Two-dimensional interfaces

We start by examining the force on a short segment of the interface shown in Fig. 7.2. We take the segment belonging to point e to consist of half of the element connecting points d and e plus half the element connecting e and f. If the points are evenly spaced, point e is the midpoint of the segment. In the figure, the end points of the segment are denoted by 1 and 2. Using the definition of the curvature of a two-dimensional line (see Appendix A), $\kappa \mathbf{n} = \partial \mathbf{t} / \partial s$, we can write the force

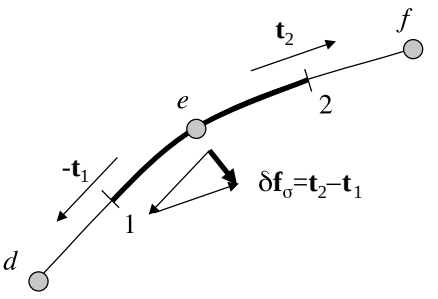


Fig. 7.2. The two-dimensional curvature calculations.

as

$$\delta\mathbf{f}_e = \int_{\Delta s} \sigma\kappa\mathbf{n}\,\mathrm{d}s = \sigma\int_{\Delta s}\frac{\partial\mathbf{t}}{\partial s}\,\mathrm{d}s = \sigma(\mathbf{t}_2-\mathbf{t}_1). \tag{7.47}$$

Therefore, instead of having to find the curvature, we only need to find the tangents of the end points of each segment. This formulation also makes the extension to variable surface tension almost trivial. A simple expansion of the partial derivative shows that

$$\frac{\partial\sigma\mathbf{t}}{\partial s} = \sigma\frac{\partial\mathbf{t}}{\partial s}+\frac{\partial\sigma}{\partial s}\mathbf{t} = \sigma\kappa\mathbf{n}+\frac{\partial\sigma}{\partial s}\mathbf{t}, \tag{7.48}$$

which is the usual expression accounting for both the normal and the tangential force. Since

$$\delta\mathbf{f}_e = \int_{\Delta s}\frac{\partial\sigma\mathbf{t}}{\partial s}\,\mathrm{d}s = (\sigma\mathbf{t})_2-(\sigma\mathbf{t})_1, \tag{7.49}$$

the force on each segment is computed by simply subtracting the product of the surface tension coefficients and the tangents at the end points of each segments for both constant and variable surface tension.

The accuracy and efficiency of the computations depends on how we find the tangent vectors. Here, we will find the tangent vector at the middle of the front elements and compute the force on an interface segment consisting of half the element on one side of a front point and half the element on the other side. See Fig. 7.2. We note, however, that we could just as well compute the force on each front element by finding the tangents at the front points. In this case the resultant would be stored at the element centroids.

The simplest approximation for the unit tangent at the center of each element is given by

$$\mathbf{t} = \frac{\mathbf{x}_2-\mathbf{x}_1}{|\mathbf{x}_2-\mathbf{x}_1|} = \frac{\mathbf{x}_2-\mathbf{x}_1}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}} = \frac{\mathbf{x}_2-\mathbf{x}_1}{\Delta s_{12}}. \tag{7.50}$$

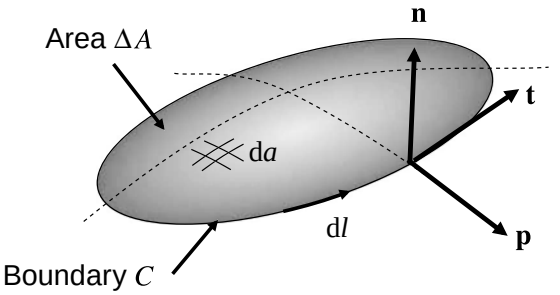


Fig. 7.3. The three-dimensional curvature calculations. The surface element ΔA is bounded by the contour C.

This simple approximation usually works well enough, but if a higher order approximation is needed, Equation (6.5) can be used to find the tangent.

Equation (7.50) can be implemented for an unstructured front grid (where the structure of the front is given by elements connecting the points) by looping over the front elements and adding the positive tangent to the start point of each element and the negative tangent to the end point.

7.2.2 Three-dimensional interfaces

For a three-dimensional interface, we use that the mean curvature of a surface can be written as

$$\kappa \mathbf{n} = (\mathbf{n} \times \nabla) \times \mathbf{n}; \tag{7.51}$$

see Equation (A.34). The force on a surface segment, therefore, is

$$\delta \mathbf{f}_e = \sigma \int_{\Delta A} \kappa \mathbf{n} \, \mathrm{d}a = \sigma \int_{\Delta A} (\mathbf{n} \times \nabla) \times \mathbf{n} \, \mathrm{d}a = \sigma \oint_C \mathbf{p} \, \mathrm{d}l, \tag{7.52}$$

where we have used the Stokes theorem to convert the area integral into a line integral along the edges of the segment. Here, $\mathbf{p} = \mathbf{t} \times \mathbf{n}$, where $\mathbf{t}$ is a vector tangent to the edge of the element, $\mathbf{n}$ is a normal vector to the surface, ΔA is the element and C is its boundary. See Fig. 7.3. The surface tension coefficient times $\mathbf{p}$ gives the "pull" on the edge and the net "pull" on the segment is obtained by integrating around its edges. If the segment is flat, the net force is zero, but if the segment is curved, then the net force is normal to it, when the surface tension coefficient is constant. As in two dimensions, this formulation ensures that the net force on a closed surface is zero, as long as the force on the common edge of two segments is the same. This formulation also extends to variable surface tension in an obvious

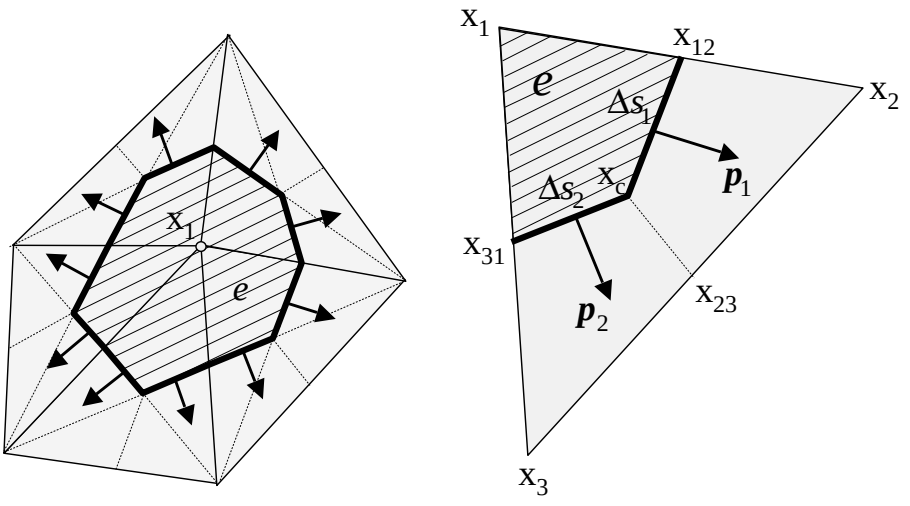


Fig. 7.4. The three-dimensional curvature calculations for front-tracking simulations. Here, the curvature is computed at the nodes by adding up contributions from the adjacent elements.

way:

$$\delta \mathbf{f}_e = \oint_C \sigma \mathbf{p} \, \mathrm{d}l. \tag{7.53}$$

As in two dimensions, the surface segments can be selected in different ways, but here we will work with a surface segment that consists of a third of all the elements connected to a given front point. The surface segment for the point whose coordinates are $\mathbf{x}_1$ is the cross-hatched area, bounded by a thick line, in the left hand side of Fig. 7.4. Consider now one of the elements, denoted by *e*, in more detail (right-hand side of Fig. 7.4). The area contributing to each point is found by identifying the centroid of the element and drawing lines from the centroid to the midpoints of the edges of the element that connect to the point in the middle. The centroid of element *e* is at

$$\mathbf{x}_\mathrm{c} = \frac{1}{3}\Big(\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3\Big). \tag{7.54}$$

The midpoints of the sides are

$$\mathbf{x}_{12} = \frac{1}{2}\Big(\mathbf{x}_1 + \mathbf{x}_2\Big), \qquad \mathbf{x}_{23} = \frac{1}{2}\Big(\mathbf{x}_2 + \mathbf{x}_3\Big), \qquad \mathbf{x}_{13} = \frac{1}{2}\Big(\mathbf{x}_1 + \mathbf{x}_3\Big). \tag{7.55}$$

The normal to the element is found by

$$\mathbf{n}_e = \frac{\Delta\mathbf{x}_{12} \times \Delta\mathbf{x}_{13}}{|\Delta\mathbf{x}_{12} \times \Delta\mathbf{x}_{13}|}, \tag{7.56}$$

where $\Delta\mathbf{x}_{12} = (\mathbf{x}_1 - \mathbf{x}_2)$ and so on. Since the force on the side connecting $\mathbf{x}_1$ and $\mathbf{x}_{12}$ is canceled by the force from the other element bordering that side, and the same is true for the side connecting $\mathbf{x}_1$ and $\mathbf{x}_{31}$, we only need to consider the lines connecting $\mathbf{x}_{12}$ to $\mathbf{x}_c$ and $\mathbf{x}_c$ to $\mathbf{x}_{31}$. The integral over the first line segment is

$$\int_{\Delta C} \mathbf{p}\, \mathrm{d}l = \mathbf{p}_1 \Delta s_1 = \mathbf{n}_e \times \frac{\mathbf{x}_c - \mathbf{x}_{12}}{\Delta s_1} \Delta s_1 = \mathbf{n}_e \times (\mathbf{x}_c - \mathbf{x}_{12}) \tag{7.57}$$

where $\Delta s_1 = |\mathbf{x}_{12} - \mathbf{x}_c|$. The net contribution to the force on the area segment surrounding point 1, from element e, is the integral over both line segments. Thus, we can write

$$\delta\mathbf{f}_{1e} = \mathbf{p}_1 \Delta s_1 + \mathbf{p}_2 \Delta s_2 = \mathbf{n}_e \times (\mathbf{x}_c - \mathbf{x}_{12}) + \mathbf{n}_e \times (\mathbf{x}_{31} - \mathbf{x}_c) = \frac{1}{2}\mathbf{n}_e \times \Delta\mathbf{x}_{32} \tag{7.58}$$

using Equation (7.55). Thus, the force on a segment around point 1 from element e can be found simply by computing half the "pull" on the edge connecting points 2 and 3. Element e, therefore, contributes the following forces to its three nodes:

$$\delta\mathbf{f}_{1e} = \frac{1}{2}\mathbf{n}_e \times \Delta\mathbf{x}_{32}, \quad \delta\mathbf{f}_{2e} = \frac{1}{2}\mathbf{n}_e \times \Delta\mathbf{x}_{13}, \quad \delta\mathbf{f}_{3e} = \frac{1}{2}\mathbf{n}_e \times \Delta\mathbf{x}_{21}. \tag{7.59}$$

As in two dimensions, the total force at each front point is the contribution from all the elements connecting to the point. Unlike two dimensions, however, an arbitrary number of elements can contribute to each node point.

The method described above works reasonably well, but if higher order approximations are needed, the tangents must be found by differentiating the surfaces obtained by fitting a larger number of points, as given by Equation (6.9).

7.2.3 Smoothing the surface tension on the fixed grid

After the surface force on a small segment of the interface $\delta\mathbf{f}_l$ has been found, it has to be transferred to the stationary grid so that it can be included in the solution of the Navier–Stokes equations. The most straightforward approach is to smooth the force directly onto the grid, following the procedure outlined in Chapter 6. For two-dimensional flows, this operation is shown in Fig. 7.5 and the force per unit volume at grid point (i, j) is given by

$$\mathbf{f}_{i,j} = \sum_l \delta\mathbf{f}_l w_{i,j}^l \frac{\Delta s_l}{h^2}. \tag{7.60}$$

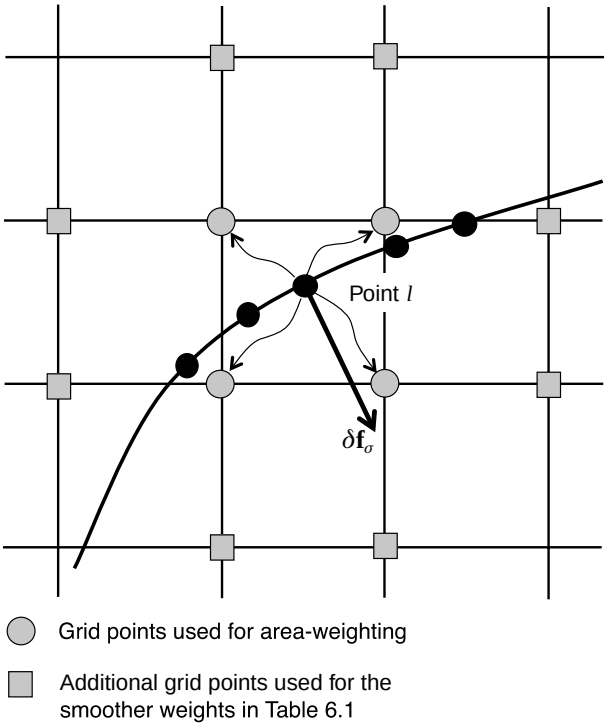


Fig. 7.5. Distribution of the force in a marker method. For an interface tracked by a chain of markers, the force on each element is computed as the difference between tangential forces on each marker.

Here, $\delta\mathbf{f}_l$ is a discrete approximation to the total surface force on the element, as computed on the front, and $\mathbf{f}_{i,j}$ is an approximation to the surface tension per unit volume on the fixed grid. Δs_l is the length of the surface segment l and the $\Delta s_l/h^2$ factor is there to ensure that the total force is conserved. For three-dimensional flow we multiply by $\Delta s_l/h^3$, where Δs_l now is the area of the surface segment. The different choices for the weighting functions $w^l_{i,j}$ were discussed in Chapter 6. The forces are then added to the discrete Navier–Stokes equations to update the velocity field. For staggered grids, the different components of the force are assigned to the grid nodes where the corresponding velocity component is computed. Notice that for programming purposes it is not necessary to add the forces from all the elements adjacent to each point. The forces for each elements, Equation (7.59), could just as well be transferred to the fixed grid as soon as they have been computed. That does, however, increase the number of quantities that must be transferred.

Distributing the surface force to the fixed grid as described above is probably the most natural way to add it to the discrete Navier–Stokes equations. It is, however, not the only way. In methods where the marker function is advected directly

(such as VOF and level-set methods) the curvature and the normal vector are found directly on the fixed grid. In front-tracking methods we reconstruct a marker function I, on the fixed grid from the location of the front, and we could therefore find the surface tension directly from the marker function, just as in the VOF method. The third possibility arises by recognizing that the surface force per unit area, as given by Equation (7.1) consists of two parts: the surface tension coefficient times the curvature $\sigma\kappa$, and the normal to the interface times a delta function $\mathbf{n}\delta_S$. Each part can, in principle, be computed either on the front and transferred to the grid, or found directly on the grid from the marker function.

As discussed in the context of Equation (7.10), using a normal found from the marker function on the fixed grid allows us to balance the discrete pressure gradient exactly. This suggests that we should take the curvature from the front, where it can presumably be found to a high degree of accuracy, but compute the normal on the grid, as done for the VOF method. Thus, we rewrite Equation (7.1) at a given gridpoint (i, j) as

$$\mathbf{f}_{i,j} = (\sigma\kappa)_{i,j}\nabla I_{i,j}. \tag{7.61}$$

Here, $\nabla I_{i,j}$ is found by taking the numerical derivative of $I(i, j)$ on the fixed grid, using standard centered differences, and the challenge is to find $(\sigma\kappa)_{i,j}$ at each grid point, given that we know the force on the front.

In the method of Shin *et al.* (2005a), the force at each grid point is first found by smoothing the surface force computed on the front, in the way described above (Equation (7.60)). The gradient of the indicator function on the front, $\mathbf{G} = -\nabla I = \mathbf{n}\delta_S$, is then also distributed to the grid:

$$\mathbf{G}_{i,j} = \sum_l \mathbf{n}_l w^l_{i,j} \frac{\Delta s_l}{h^2}. \tag{7.62}$$

If we knew the curvature at the fixed grid points, we could therefore also write the surface force on the fixed grid as

$$\mathbf{f}_{i,j} = (\sigma\kappa)_{i,j}\mathbf{G}_{i,j} \tag{7.63}$$

Taking the dot product of the force, given by Equation (7.63), and $\mathbf{G}_{i,j}$ we get

$$\mathbf{f}_{i,j} \cdot \mathbf{G}_{i,j} = (\sigma\kappa)_{i,j}\mathbf{G}_{i,j} \cdot \mathbf{G}_{i,j}. \tag{7.64}$$

Thus,

$$(\sigma\kappa)_{i,j} = \frac{\mathbf{f}_{i,j} \cdot \mathbf{G}_{i,j}}{\mathbf{G}_{i,j} \cdot \mathbf{G}_{i,j}}, \tag{7.65}$$

and once we have $(\sigma\kappa)_{i,j}$, the surface force on the grid is given by Equation (7.61).

It is possible to obtain $(\sigma\kappa)_{i,j}$ at each grid point in other ways. We can, in particular, compute the curvature at each front point and distribute $\sigma\kappa$ from the front to the fixed grid. Distributing a scalar quantity that is not conserved from a front to the fixed grid is sometimes needed in other situations, and those can be handled in a way similar to the one described below. The basic approach is to assign to each grid point the weighted value of the scalar of front points located nearby. The weighted contributions from the front points are added together at each point on the fixed grids and the weights are added together separately. By dividing the sum of the weighted scalars by the sum of the weights, we obtain a value for the scalar at the grid point. For the surface tension, it seems reasonable to weight the curvature by the length of each element, so that longer elements carry more weight than shorter ones, as well as by weights $w^l_{i,j}$ that take into account how close the front point is to a given grid point. In this case, the surface force at each grid point is given by

$$(\sigma\kappa)_{i,j} = \frac{\sum_l (\sigma\kappa)_l w^l_{i,j} (\Delta s_l / h^2)}{\sum_l w^l_{i,j} (\Delta s_l / h^2)}. \tag{7.66}$$

To find the curvature at the front points, we can discretize Equation (A.3) or its three-dimensional counterpart, or we can work with the net force on a small surface segment. To do the latter, we approximate the integral of the surface force over a small segment of the interface by the point value of $\sigma\kappa$ and the length of the segment around the point

$$\mathbf{f}_e = \int_{\Delta s} \sigma\kappa\mathbf{n}\, \mathrm{d}s \approx (\sigma\kappa\mathbf{n})_e\, \Delta s, \tag{7.67}$$

then the dot product of $\mathbf{f}_e$ with itself, at the point, gives

$$\mathbf{f}_e \cdot \mathbf{f}_e = (\sigma\kappa\Delta s)^2_e\, \mathbf{n}_e \cdot \mathbf{n}_e = (\sigma\kappa\Delta s)^2_e. \tag{7.68}$$

Thus,

$$(\sigma\kappa\Delta s)_e = \pm\sqrt{\mathbf{f}_e \cdot \mathbf{f}_e}. \tag{7.69}$$

To recover the sign we need to look at the geometry again. In two dimensions we can, for example, take the sign of the triple product $\mathbf{f}_\mathrm{p} \times \mathbf{t}_\mathrm{p} \cdot \mathbf{k} = f_x t_y - f_y t_x$, where $\mathbf{t}_\mathrm{p} = (\mathbf{t}_1 + \mathbf{t}_2)/2$ and $\mathbf{k}$ is the unit vector perpendicular to the two-dimensional plane of the flow.

If we use the same scheme to distribute $\sigma\kappa$ to the fixed grid as used for the gradient of the indicator function (area weighting, for example) we will end up with a band of $\sigma\kappa$ values around the interface that is thinner than the band where the gradient of the indicator function is nonzero. This can be fixed by using a different smoothing scheme that distributes the front value to more grid points, or the band can be made wider afterwards. To make the band wider – or to propagate the front

value off the front – we can use several approaches, but one of the more straightforward ones is to give each point near the original band a value found by averaging the value at the neighboring points inside the band. If the band needs to be made even thicker, this "Laplacian" extension can be repeated as often as needed, possibly until the front values have been extended to all points in the domain.

7.3 Testing the surface tension methods

Computing surface tension accurately for a wide range of governing parameters has been one of the major challenges in developing methods for multiphase flows. Surface tension has no counterpart in computations of homogeneous flows and testing, and validating the various methods put forward is therefore of unique importance. In this section we discuss a few such tests, first for stationary interfaces and then for dynamic ones.

7.3.1 Static case: spurious currents

The most elementary test that every method to compute the surface tension must pass is that it must predict correctly the pressure inside a stationary spherical (or circular in two dimensions) droplet, as given by Laplace's law. All the methods described here obviously predict the average rise in pressure reasonably well. The pressure does, however, generally not rise smoothly. As Fig. 7.6 shows, it exhibits fluctuations that often manifest themselves as pressure spikes around the interface. The pressure fluctuations result in velocity fluctuations that can, in extreme cases, break the droplet apart. For a spherical or circular droplet in equilibrium the velocity should, of course, be exactly zero and the velocities in Fig. 7.7 are generally referred to as spurious or parasitic currents. The results in Figs. 7.6 and 7.7 were computed using the CSS method with five-point smoothing (7.34), for which the spurious effects are moderate. For front-tracking methods the currents are generally smaller, particularly if the surface tension is transferred to the fixed grid using a smooth weighting function, such as those shown in Table 6.1. Many computations have been carried out where the spurious currents have been insignificant. However, their presence limits the range of parameters that can be covered, and considerable effort has been devoted to attempts to reduce or eliminate parasitic currents.

The magnitude of the parasitic currents (and the pressure fluctuations) varies wildly from method to method and as a function of the dimensionless numbers governing each situation. For a stationary droplet, there are three dimensionless numbers that characterize the flow: the Laplace number $\mathrm{La} = \rho_{\mathrm{liq}} d\sigma/\mu_{\mathrm{liq}}^2$ where d is the droplet diameter, and the viscosity and density ratios $\mu_{\mathrm{liq}}/\mu_{\mathrm{gas}}$ and $\rho_{\mathrm{liq}}/\rho_{\mathrm{gas}}$.

In Table 7.1 we show results from a study of parasitic currents for several test cases, using a few different methods. The cases are in order of increasing difficulty.

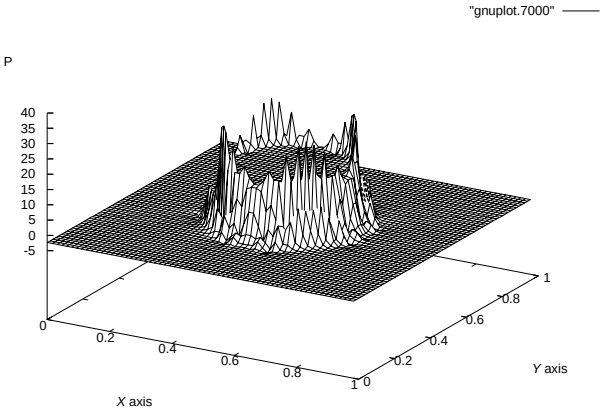


Fig. 7.6. The pressure field for a cylindrical static droplet (case F, CSS method, no smoothing). The diameter of the droplet contains 28 grid points.

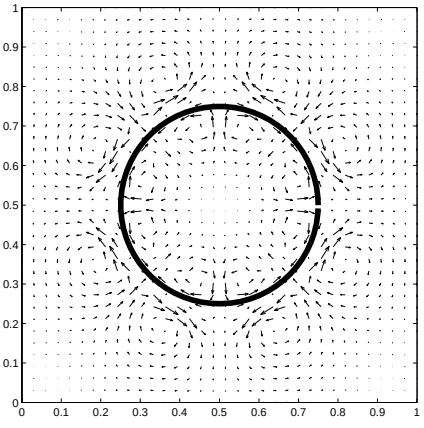


Fig. 7.7. The parasitic currents generated by a cylindrical droplet (case F, CSS method, no smoothing). The diameter of the droplet contains 28 grid points and the maximum nondimensional velocities $\mu U/\sigma$ are $O(10^{-3})$.

Cases F and A have comparable and low Laplace number; cases B, C, and D are increasingly close to the case of a small air bubble in water. The chosen Laplace number (2×10^6) corresponds to a $d = 1.3$ cm air bubble. The methods are the CSS and CSF methods described previously, together with the elementary smoothing (7.34) and the front-tracking method described in Section 7.2. The maximum velocity on the computational grid U is taken in the long time limit, because the parasitic currents often have long transients before settling to an equilibrium value. Given in the table is the capillary number $\mathrm{Ca} = \mu_{\mathrm{liq}} U/\sigma$. For small La, the convergence to equilibrium is smooth; for large La, the capillary number Ca oscillates

Table 7.1. Measurements of spurious currents. The capillary number *Ca* as computed by three simple surface tension methods.

Case	La	ρ_{liq}/ρ_{gas}	μ_{liq}/μ_{gas}	D/h	Ca		
					CSS	CSF	Front tracking
F	0.120	1	1	30	6×10^{-3}	1.4×10^{-2}	3.0×10^{-4}
A	0.357	1	1	32	3×10^{-3}	1.2×10^{-2}	3.0×10^{-4}
B	2×10^{6}	1	1	32	3×10^{-4}	5.0×10^{-4}	1.5×10^{-4}
C	2×10^{6}	10^{3}	1	32	disintegrates	3.5×10^{-3}	1.1×10^{-3}
D	2×10^{6}	10^{3}	50	32	disintegrates	4.5×10^{-3}	blows up

strongly. Most authors have performed tests at small La and for symmetric fluids, but many applications are rather different.

The results of CSS and CSF with elementary smoothing (7.34) are much worse than the front-tracking method for small La and the CSS method performs somewhat better than the CSF method. As La is increased, the situation changes. For the CSS and CSF methods, the droplet tends to disintegrate under the action of spurious currents when the smoothing described by Equation (7.34) is employed. Without any smoothing, however, the droplet survives in a perturbed shape for the CSF method. We thus report in Table 7.1 the results for CSS and CSF in cases F, A, and B with the elementary smoothing given by Equation (7.34) and results of CSF in the cases C and D without smoothing.

The results of more sophisticated methods, such as CSF with the kernel K_8 given by Equation (7.36), PROST, PCMM, and the height-function (HF), method are reported in Table 7.2. These methods perform consistently better than the elementary methods, but the performance of the CSF method with the K_8 kernel is of the same order of magnitude as that of the front-tracking method.

7.3.2 Dynamic case

Tests of rising bubbles and falling droplets show an interesting balance between surface tension and the viscous and pressure stresses that tend to deform the bubble and several authors have compared the steady deformation with experimental observations and other computations, such as those of Ryskin and Leal (1984). Generally the agreement is good. However, such tests are only partially dynamic: there is flow but usually no interface oscillations.

Table 7.2. Measurements of spurious currents: the capillary number *Ca* as computed by four advanced surface tension methods. We use the notation $r = \rho_{\text{liq}}/\rho_{\text{gas}}$ and $m = \mu_{\text{liq}}/\mu_{\text{gas}}$.

					Ca			
Case	La	r	m	D/h	CSFK8	PROST	PCMM	HF
F	0.120	1	1	30	10^{-4}	NA	10^{-6}	NA
A	0.357	1	1	32	NA	6×10^{-5}	NA	NA
E	120–12 000	1	1	32	NA	NA	NA	machine accuracy

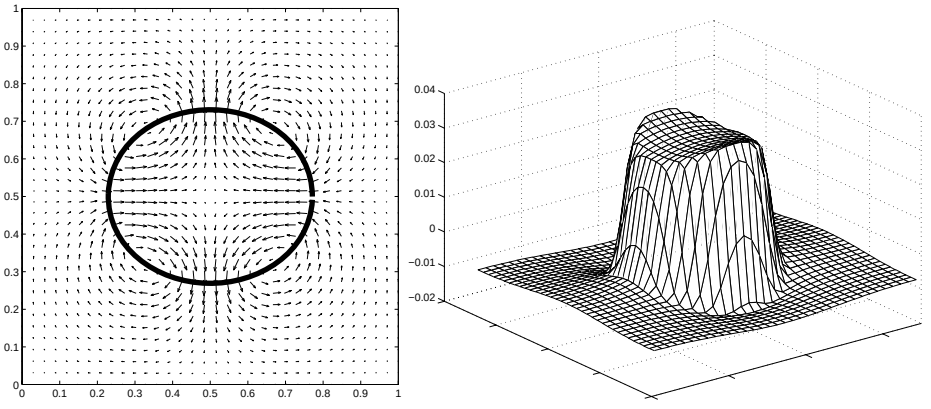


Fig. 7.8. The velocity (left) and pressure (right) for an oscillating droplet at nondimensional time $t\sigma/d\mu = 9.0925$. The computations are done using a front tracking on a 32^2 grid. The droplet is initially perturbed by 10% of its radius.

Capillary oscillations of drops and bubbles where the shape changes in time usually provide a more stringent test, and in Fig. 7.8 we show one example of a simulation of an oscillating drop. Analytical solutions exist for various situations where small-amplitude oscillations are superimposed on a "regular" interface shape such as a sphere, a cylinder, or a flat plane, and it is interesting to compare the numerical solutions with them. For a flat interface the standard ideal fluid solution gives surface oscillations of frequency $\omega_0^2 = \sigma k^3/(\rho_1+\rho_2)$, where $k = 2\pi/\lambda$ is the wavenumber. However, with viscosity ν, the frequency is shifted. The shift can be rather large, since it is proportional to $\sqrt{\nu}$. Moreover, viscous effects are important in many applications and they have to be tested as well. One thus has to find the viscous "normal-mode" solution. This solution is briefly discussed by Chandrasekhar (1961), but even with that solution there are several difficulties, as

listed below. We consider a wave of shape

$$h(x,t) = a(t)\cos(kx). \tag{7.70}$$

(i) Care must be taken to avoid box size effects. The height L_z of the box above the interface must be large compared with the wavelength λ. Corrections when kh is large are of the order $\tanh(kh) - 1$ in the inviscid limit.

(ii) Nonlinear effects will be important when a is not very small compared with λ. They will introduce an error compared with the viscous normal-mode solution.

(iii) The viscous normal-mode solution (Chandrasekhar, 1961) gives the evolution of the height $h(x,t)$, the velocity field $\mathbf{u}(\mathbf{x},t)$, and the pressure for a normal mode. To observe the normal mode, all fields must be initialized at $t = 0$. If only the interface is initialized but the initial velocity is zero, a different solution must be used. A solution for this case has been found by Cortelezzi and Prosperetti (1981; Prosperetti, 1981). Both the normal-mode solution and the Prosperetti solution are extremely complex. Their description is rather lengthy, and so the interested reader is directed to the references.

In Fig. 7.9 and in Table 7.3 we show the comparison of the observed evolution and frequency of a planar capillary wave on a plane interface, compared with the Prosperetti solution. The density and viscosity ratios are equal to one. Several methods are compared, and the results are obtained from the literature: Gueyffier *et al.* (1999) for CSS, Gerlach *et al.* (2006) for CSF/K_8 and PROST, Popinet and Zaleski (1999) for PCMM, and Popinet (2009) for the HF method. The PROST, PCMM, and HF methods are defined below. In the CSS method, agreement with the analytical solution is good, but there is no convergence beyond the 0.01 error mark (A decrease of the error is seen again for 512^2 grids, but no steady convergence). Decreasing the solution amplitude helps but does not seem to improve the error beyond 0.01. For PCMM the convergence is rapid at first then slower. For both methods the results are worse when the density and viscosity ratios are increased. The K_8 and PROST methods give better results, especially at finer resolutions. The HF method gives the best results at almost all resolutions, and is performing especially well at coarse resolution (eight points per wavelength).

7.4 More sophisticated surface tension methods

7.4.1 Direct addition with pressure correction

In the PCMM (Popinet and Zaleski, 1999), the force is computed from a marker representation of the interface. The tangents are estimated at the boundary of a

Table 7.3. Relative error for a capillary wave. The Laplace number *La* is based on the wavelength.

Method	a_0	8	16	32	64	128	La
VOF/CSS	5×10^{-3}	—	—	0.05	0.007	0.013	3000
VOF/CSFK8	10^{-2}	0.45	0.24	0.1095	0.0456	—	NA
PROST	10^{-2}	0.30	0.082	0.0069	1.8×10^{-3}	—	NA
PCMM	NA	0.30	0.08	0.013	0.01	0.007	3000
HF	10^{-2}	0.16	0.028	0.0084	1.5×10^{-3}	5.5×10^{-4}	3000

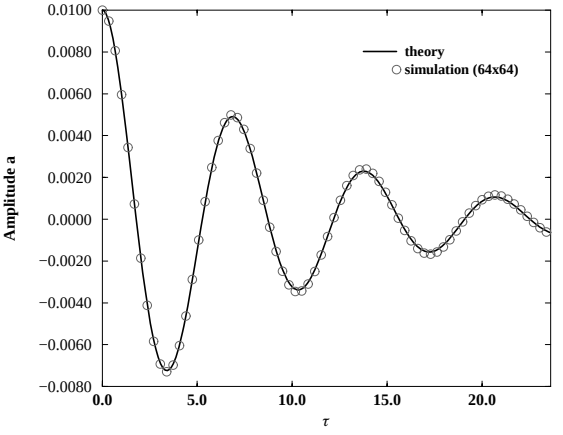


Fig. 7.9. Evolution of the absolute value of the amplitude of the wave versus non-dimensional time $\tau=\omega_0 t$, comparing the analytical solution and the numerical simulation for a box size 64×64. The density ratio is $\rho_1/\rho_2=1$.

control volume centered on a velocity node, for instance the horizontal velocity node $u_{i-1/2,j}$ in Fig. 7.10. The horizontal projection of the tensile force exerted by the two tangents is added directly, without smoothing, to the horizontal velocity node. As such, the method is too imprecise and it is necessary to correct the expression of the pressure. The pressure integrated over the control volume is (in two dimensions)

$$\mathbf{e}_x\cdot\int\nabla p\,\mathrm{d}v=-\int_{\mathrm{AC}}p\,\mathrm{d}s+\int_{\mathrm{DF}}p\,\mathrm{d}s, \tag{7.71}$$

where $\mathbf{e}_x$ is a unit vector in the x-direction. Using the pressure at the nodes $(i-1,j)$ and (i,j) is too imprecise because the pressure jumps by a finite amount on the interface. Two cases are to be considered: (i) the crossing point E on the right side

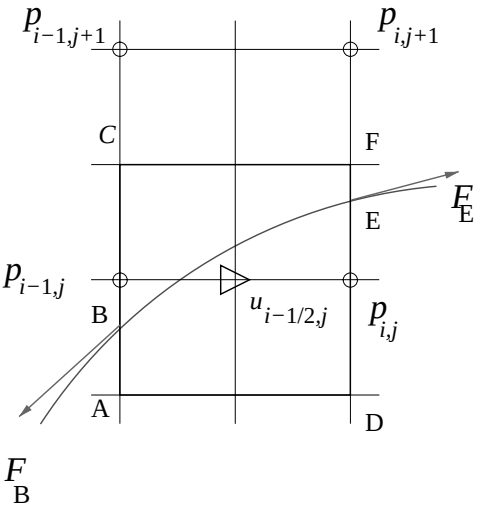


Fig. 7.10. Distribution of the force in the PCMM.

of the cell is above the pressure node $p_{i,j}$, as in Fig. 7.10, and (ii) the crossing point E is below the pressure node. In the first case the approximation of the pressure involves a *pressure correction* from the pressure node above:

$$\int_{\mathrm{DF}} p \, \mathrm{d}s = \mathrm{DE} p_{i,j} + \mathrm{EF} p_{i,j+1}. \tag{7.72}$$

In the second case the pressure node below, $p_{i,j-1}$, is used:

$$\int_{\mathrm{DF}} p \, \mathrm{d}s = \mathrm{DE} p_{i,j-1} + \mathrm{EF} p_{i,j}. \tag{7.73}$$

One of these two expressions replaces the usual centered approximation $\mathrm{DF} p_{i,j}$.

The pressure correction method has been implemented in two dimensions but not in three dimensions. Implementations were performed both with a front-tracking method (the above PCMM method) and also with a VOF reconstruction (Ashgriz *et al.*, 2005). However, it seems better adapted to a front-tracking method in which the tangents at the boundary of the control volume may be computed accurately.

7.4.2 CSF method with better curvature: PROST

The CSF method may be improved by a better curvature calculation. Indeed, as we saw in Section 7.1.1, an *exact* curvature calculation would yield an exact balance between pressure and surface tension, eliminating parasitic currents. This is the basis for the PROST method introduced by Renardy and Renardy (2002). Instead

of using expression (7.4), the curvature is computed by fitting a quadratic curve to the color function C_{ij} in a 3×3 block. The equation for the quadratic curve is

$$a+\mathbf{b}\cdot\mathbf{x}+\mathbf{x}\cdot\mathbf{C}\cdot\mathbf{x}=0, \tag{7.74}$$

where the parameters are the scalar a, the vector $\mathbf{b}$ and the symmetric matrix $\mathbf{C}$. Together they form a parameter vector $\mathbf{q}$. The error to be minimized by the fit is the difference between the color-function-defined area and the areas under a trial quadratic curve. The normalized area in cell (i,j) is noted $v_{ij}(\mathbf{q})$. The error is then

$$E=\sum_{ij}(v_{ij}(\mathbf{q})-C_{ij})^2, \tag{7.75}$$

where the sum is over a 3×3 block of cells (a $3\times 3\times 3$ block in three dimensions). The best fit is obtained by a least-squares algorithm, available in standard numerical packages. The computation of the curvature is straightforward once the best quadratic curve has been identified. This is the so-called PROST method. PROST gives good results, but it is computationally expensive because of the search for the least-squares curve. Another way to obtain accurate curvatures is described in the next section.

7.4.3 Numerical estimate of the curvature from the volume fractions: the HF method

In Chapter 5 we introduced a local height function to calculate the normal vector. We now extend this approach to the computation of the curvature in the following way (Sussman, 2003; Cummins *et al.*, 2005; Popinet, 2009):

(i) determine the coordinate direction of the maximal component of $\mathbf{n}$;
(ii) sum the volume fractions along this direction to evaluate the height function;
(iii) compute the curvature with centered finite differences.

In the example of Fig. 7.11 we have $|n_x|>|n_y|$ in the cell (i,j) and a 7×3 stencil is then used to compute the horizontal width function $x=g(y)$, in particular $x_{j-1}=h\sum_{k=-3}^{3}C_{i+k,j-1}$. The curvature for this case is calculated as

$$\kappa=\frac{g_{yy}}{(1+g_y^2)^{3/2}}, \tag{7.76}$$

where the derivatives g_y and g_{yy} are estimated with central differences, $g_y=(x_{j+1}-x_{j-1})/2h$ and $g_{yy}=(x_{j+1}-2x_j+x_{j-1})/h^2$. With this choice of the sign in equation (7.76) the curvature of the reconstructed interface line with the volume fraction distribution of Fig. 7.11 is negative, as in Fig. 2.5. Similar expressions hold for the

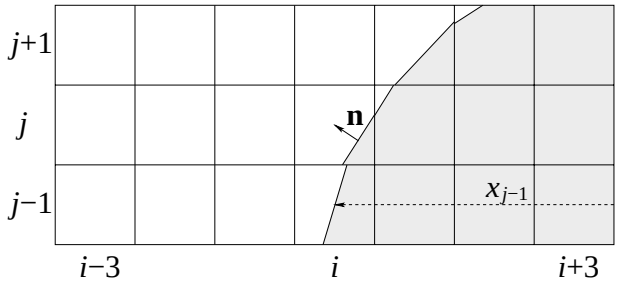


Fig. 7.11. A 7×3 stencil is used to compute the horizontal height function x in the three cells $(i, j-1)$, (i, j), and $(i, j+1)$ when $|n_x| > |n_y|$.

height $y = f(x)$. There are situations where the height function is not within the cell boundary; for example, for the cell $(i+1, j)$ of Fig. 7.11. In this case it is more accurate to calculate the local curvature by interpolating the curvature value of the neighboring cells. Finally, since the height function is a one-dimensional integral across the discontinuity of the Heaviside function, it is a more regular function than the volume fraction C. Thus, it provides a better estimate of the local curvature. For a comparison among different methods, we recall that in this chapter we have also derived an expression for the curvature κ based on a finite-difference approximation of the unit normal $\mathbf{n}$ and of its divergence in Section 7.1.1, while in Section 7.1.4 the curvature is computed with a similar expression, but with the unit normal $\widetilde{\mathbf{n}}$ calculated from a convolution of the discontinuous function C with the derivatives of the kernel (7.36). We compare the accuracy of curvature estimates of these three methods by considering the L_∞ error norm, defined as $L_\infty = \max|\widetilde{\kappa}_h - \kappa|$, where $\widetilde{\kappa}_h$ is our numerical approximation to the local value of the curvature κ, on a circular interface. A circle with radius $R = 0.15$ and curvature $\kappa = 1/R = 6.\overline{6}$ is centered at $(1/2, 1/2)$. We subdivide the unit square computational domain with n^2 square cells of side $h = 1/n$, with $n = 32, 64, 128, 256$, and in Fig. 7.12 we show the L_∞ error norms for the interface curvature. We see second-order convergence $\mathscr{O}(h^2)$ for the HF method and an approximate $\mathscr{O}(h^{-1})$ error for both the kernel convolution and the finite-difference method. In particular, we have assumed the value $\varepsilon = 2h$ for the width of the filter in the kernel (7.36). If the width ε is kept constant, i.e. it remains independent from the grid resolution h, then second-order convergence is obtained, but the method becomes computationally very expensive.

The HF method for the computation of curvature was applied by Popinet (2009) for the computation of surface tension. When the VOF marker function is too irregular to compute the curvature by the HF method, the code falls back on a least-squares fit through the VOF reconstructions. The HF method gives very good

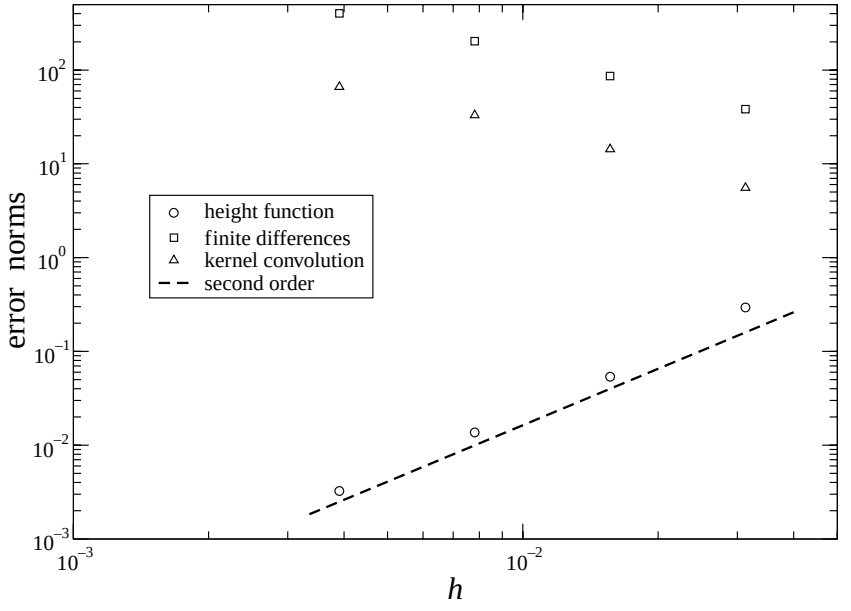


Fig. 7.12. L_∞ error norms for the curvature estimated along a circular interface with the kernel-convoluted C data, a finite-difference scheme and the height function.

results for spurious currents, which decrease to machine accuracy provided the droplet is left to relax to equilibrium over a sufficiently long time, and for capillary oscillations (see Table 7.3).

7.5 Conclusion on numerical methods

Numerical methods for direct numerical simulations of multiphase flows have now been developed to the point that they can be used to examine in detail fairly complex problems. While further method development will certainly take place (as more accuracy, efficiency, and robustness are always desirable), we believe that the most exciting development in the next several years will be the use of these methods to probe the mysteries of increasingly complex flows. Thus, we will now change the focus and in the rest of the book we will present a few examples of how DNS have been used to examine the dynamics of a few selected multiphase flows.

8

Disperse bubbly flows

8.1 Introduction

Understanding and predicting bubbly flows is of critical importance in a large number of industrial applications, including boiling heat transfer in power plants, various metallurgical processes, and in bubble columns in the chemical industry. In bubble columns, used for partial oxidation of ethylene to acetaldehyde, isobutene separation, wet oxidation of heavily polluted effluent, and the production of synthetic fuels, for example, gas is injected at the bottom and, as the bubbles rise, the gas diffuses into the liquid and reacts (Furusaki *et al.*, 2001). Bubble columns ranging from tens to hundreds of cubic meters are common in the chemical industry and up to thousands of cubic meters in bioreactors, where longer process times are needed. The absence of any moving parts and their relatively simple construction makes bubble columns particularly attractive for large-scale operations (Deckwer, 1992). Their operation, however, is usually dependent on the size of the vessel and the difficulty of scaling up small pilot models makes numerical predictions important. Similar considerations apply to other bubble systems.

Computational modeling of industrial-size multiphase flow systems must by necessity rely on models of the average flow. Such models range from simple mixture models to more sophisticated two-fluid models, where separate equations are solved for the dispersed and the continuous phase. Since no attempt is made to resolve the unsteady motion of individual bubbles, closure relations are necessary for the unresolved motion and the forces between the bubbles and the liquid. Closure relations are usually determined through a combination of dimensional arguments and correlation of experimental data. The situation is analogous to computations of turbulent flows using the Reynolds-averaged Navier–Stokes equations, where momentum transfer due to unsteady small-scale motion must be modeled. For details of two-fluid modeling, see Drew (1983), Ishii (1987), Zhang and Prosperetti

(1994), Drew and Passman (1999), and Prosperetti and Tryggvason (2007), for example.

For the turbulent motion of single-phase flows, direct numerical simulations (DNS) – where the unsteady Navier–Stokes equations are solved on fine-enough grids to fully resolve all flow scales – have had a major impact on closure modeling. The goals of DNS of multiphase flows are similar. In addition to information about how the drift Reynolds number, velocity fluctuations, and bubble dispersion change with the properties of the system, the computations should yield insight into how bubbles interact, both with each other and the liquid. The simulations should show whether there is a predominant microstructure and/or interaction mode, and whether the flow forms structures that are much larger than the size of the dispersed bubbles. Information about the microstructure is essential for the construction of models of multiphase flows and can also help to identify what approximations can be made.

The need for DNS to help with the construction of reliable closure models has been recognized for a long time, but it is only recently that it has become possible to simulate flows containing a large number of bubbles for a sufficiently long time so that reliable average quantities can be obtained. Earlier simulations generally examined either the motion of only one bubble (Ryskin and Leal, 1984), used simplified potential flow models (Sangani and Didwania, 1993; Smereka, 1993), or assumed point particles (e.g. Druzhinin and Elghobashi, 2001). While such simplified models can yield useful information in limiting cases, for non-dilute bubbly flows at intermediate Reynolds numbers it is necessary to solve the full unsteady Navier–Stokes equations. In the rest of this chapter we will review the status of two studies of disperse bubbly flows, both of which have been reported in several papers. As numerical studies of disperse bubbly flows are a rapidly evolving field, we warn the reader that these examples only serve as illustrations of what can be (and has been) achieved using the types of method described in this book and that you should consult the most recent literature for the current state of the art. The first example is the buoyant rise of a homogeneous distribution of many bubbles in an initially quiescent flow and the second example is the motion of buoyant bubbles in vertical channels.

The rise of a single buoyant bubble is governed by four nondimensional numbers (see Clift *et al.* (1978)). Two of those are the ratios of the bubble density and viscosity to the density and viscosity of the liquid, ρ_b/ρ_l and μ_b/μ_l (where the subscript l denotes the liquid and b stands for the fluid inside the bubble), and the other two can be selected in a number of ways. Usually, we use the Galileo and the Eötvös number, N and Eo respectively; see Section 2.6. In the chemical engineering literature, the Galileo number is often replaced by the Morton number, since, for a given fluid and gravity acceleration, the Morton number is a constant.

For flows with many bubbles, the void fraction α must also be specified, and often other nondimensional numbers are needed to describe the situation (the pipe Reynolds number for flows in pipes and channels, for example).

The rise of a single buoyant bubble, as a function of N and Eo, is fairly well understood (Clift *et al.*, 1978). In the limit of low Eo (high surface tension or a small bubble) a bubble remains spherical as it rises. As Eo increases, the value of N determines what happens. For low N (high viscosity), the bubble remains spherical. For higher N, the bubble first becomes ellipsoidal and then "skirted," in both cases rising steadily with a laminar wake. At even higher N the deformed bubble rises unsteadily, either "wobbling" or along a spiral path. For very high N and Eo the bubbles form a spherical cap bubble with a turbulent wake. Exactly how the bubbles transition between the various rise modes is not completely understood as of yet, and there is, for example, some evidence that a "wobbling" motion may be a transient between a steady rectilinear rise and a rise along a spiral path (Ellingsen and Risso, 2001). Real bubbles are also generally covered by surfactants of poorly understood composition that influence the boundary conditions at the fluid interface. This is a particularly severe problem for small air bubbles in water and can make comparisons with experimental observations difficult. In this chapter we consider only completely clean bubbles.

8.2 Homogeneous bubbly flows

Homogeneous bubbly flow, where many buoyant bubbles rise together in an initially quiescent fluid, is perhaps the simplest example of multiphase gas–liquid flows. Such flows can be simulated using periodic domains where the bubbles in each period interact freely but the configuration is repeated infinitely many times in each coordinate direction. In the simplest case, there is only one bubble per period, so the configuration of the bubble array does not change as it rises. While such regular arrays are unlikely to be seen in an experiment, they provide a useful reference configuration for freely evolving arrays. As the number of bubbles in each period is increased, the regular array becomes unstable and the bubbles generally rise unsteadily, repeatedly undergoing close interactions with each other. The behavior is statistically steady, however, and the average motion (averaged over long-enough times) does not change. While the number of bubbles per period clearly influences the average motion for a very small number of bubbles, the hope is that, once the size of the system is large enough, information obtained by averaging over each period will be representative of a truly homogeneous bubbly flow.

Extensive experimental studies have been done on homogeneous bubbly flows, primarily to find how the average drift velocity of the bubbles (the velocity relative to the liquid) depends on the void fraction. The classic reference for experimental

correlations is Ishii and Zuber (1979), who used a large number of experimental results, from a variety of sources, to construct correlations for the drift velocity of bubbles, droplets, and solid particles for a wide variety of flow conditions. The Ishii and Zuber correlations generally show a monotonic decrease in the drift velocity with increasing void fraction. Experiments addressing the interaction of bubbles with the liquid can be found in Lance and Bataille (1991), who examined bubbles injected into a turbulent flow, and in Cartellier and Rivière (2001) and Zenit *et al.* (2001) for initially laminar flows. Cartellier and Rivière (2001) studied nearly spherical bubbles at $\mathcal{O}(1)$ and $\mathcal{O}(10)$ Reynolds numbers and Zenit *et al.* (2001) experimented with nearly spherical bubbles at $\mathcal{O}(100)$ Reynolds numbers, attempting to experimentally test the applicability of potential flow simulations, such as those of Sangani and Didwania (1993) and Smereka (1993), to real flows. Although some aspects of the theory were confirmed, the experiments showed that for the most part the potential flow model is not applicable, even to experiments carefully constructed to operate in the parameter range where the model was believed to apply.

Tryggvason and coworkers have used the front-tracking method originally introduced by Unverdi and Tryggvason (1992) to examine several aspects of homogeneous bubbly flows. Esmaeeli and Tryggvason (1998) examined a case where the average rise Reynolds number (defined as $\mathrm{Re}_\mathrm{b} = \rho_\mathrm{l} U_\mathrm{b} d_\mathrm{e}/\mu_\mathrm{l}$, where U_b is the drift velocity of the bubble and d_e is the effective bubble diameter) of the bubbles remained relatively small, 1–2, and Esmaeeli and Tryggvason (1999) looked at another case where the Reynolds number is in the range 20–30. Most of the simulations were limited to two-dimensional flows, although a few three-dimensional simulations with up to eight bubbles were also presented. The time averages of the two-dimensional simulations were generally well converged, but exhibited a dependency on the size of the system. This dependency was stronger for the low Reynolds number case than for the moderate Reynolds number one. Although many of the qualitative aspects of a few bubble interactions are captured by two-dimensional simulations, the much stronger interactions between two-dimensional bubbles can lead to quantitative differences. Esmaeeli and Tryggvason (1996) simulated the motion of a few hundred two-dimensional bubbles at $\mathcal{O}(1)$ Reynolds number and showed that the bubble motion leads to an inverse energy cascade where the flow structures continuously increase in size. This is similar to the evolution of stirred two-dimensional turbulence, and although the same interaction is not expected in three dimensions, the simulations demonstrated the importance of examining large systems with many bubbles. Systems with a large number of three-dimensional bubbles were examined by Bunner and Tryggvason (2002a,b), who used a fully parallel version of the method used by Esmaeeli and Tryggvason. Their largest simulation followed the motion of over 200 three-dimensional

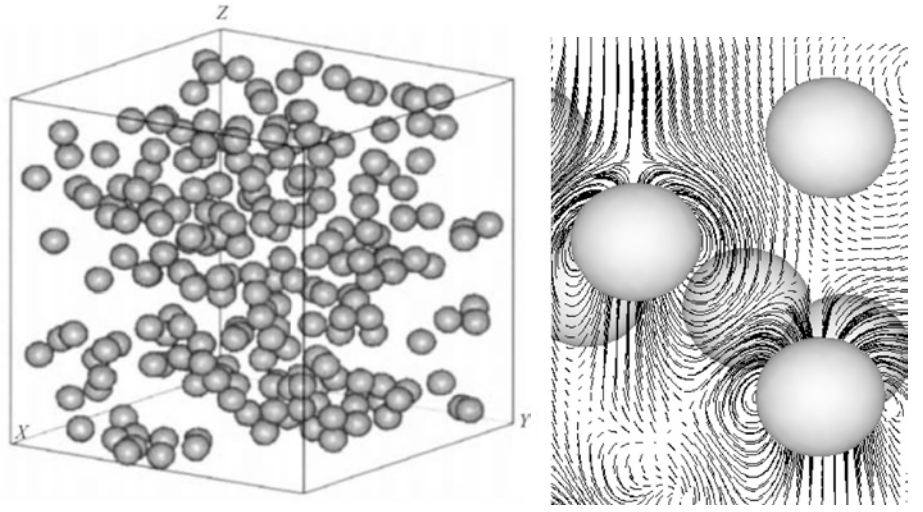


Fig. 8.1. Left: one frame from a simulation of 216 nearly spherical buoyant bubbles. Right: a close-up of the flow field around a few bubbles, from a simulation of 91 freely evolving bubbles. The governing parameters are given in the text. From Bunner and Tryggvason (2002a), 2002 ©Cambridge Journals, published by Cambridge University Press, reproduced with permission.

buoyant bubbles per periodic domain for a relatively long time. The governing parameters were selected such that the average rise Reynolds number was about 20–30 (comparable to Esmaeeli and Tryggvason (1999) but not identical), depending on the void fraction, and the deformation of the bubbles was small. Although the motion of the individual bubbles was unsteady, the simulations were carried out for a long-enough time so the average behavior of the system was well defined, as in the two-dimensional simulations of Esmaeeli and Tryggvason.

The simulations of Esmaeeli and Tryggvason (1998, 1999) and Bunner and Tryggvason (2002a,b) have resulted in a considerable amount of insight, as well as data. The first question is obviously how large a system needs to be simulated for the results to be representative of a homogeneous bubbly flow. Several simulations, starting with one bubble per period and going up to over 200 bubbles showed that for many averaged quantities, such as the average bubble rise velocity and the Reynolds stresses in the liquid, it was sufficient to consider only about 10 bubbles. The average fluctuation of the bubble velocities and the bubble dispersion converged much more slowly as the system size was increased. The bubble interactions played an important rôle for all the cases considered, and there was a significant difference between the rise velocity of a single bubble per period (a regular array) and freely interacting bubbles. While freely evolving bubbles at low Reynolds numbers rose faster than a regular array (in agreement with Stokes flow

results), at higher Reynolds numbers the trend was reversed and the freely moving bubbles rose more slowly. For both the Reynolds number ranges studied, the results showed that, for the nearly spherical bubbles, two-bubble interactions take place by the "drafting, kissing, and tumbling" mechanism of Fortes *et al.* (1987), where a bubble is drawn into the wake of a bubble in front of it, the bubbles collide, followed by a "tumbling" where the vertical bubble pair changes its orientation to horizontal, so the bubbles rise side by side. This is, of course, very different from either a Stokes flow where two bubbles do not change their relative orientation unless acted on by a third bubble, or the predictions of potential flow where two bubbles, rising along a common vertical axis, are pushed apart, not drawn together. Examination of the pair distribution function for the bubbles shows a preference for horizontal alignment of bubble pairs, independent of system size, but the distribution of bubbles remains nearly uniform. The results of Bunner and Tryggvason (2002a), combined with the more limited results in Esmaeeli and Tryggvason (1998, 1999), as well as the results of Esmaeeli and Tryggvason (2005) for even higher Reynolds numbers, show that there is an increasing tendency for the bubbles to line up side by side as the rise Reynolds number increases. This suggests a monotonic trend from the nearly no preference found by Ladd (1993) for Stokes flow, toward the strong layer formation seen in the potential flow simulations of Sangani and Didwania (1993) and Smereka (1993). The same trend was seen experimentally by Cartellier and Rivière (2001). Bunner and Tryggvason (2002b) also examined the dispersion of the bubbles and found that the probability distribution of the bubble velocity fluctuations was approximately Gaussian. Thus, while the dispersion was strongly anisotropic (the vertical dispersion was much larger than the horizontal dispersion), it should be possible to describe bubble dispersion as a diffusion process. The velocity fluctuations in the liquid phase were also examined and it was found that they scaled well with the void fraction times the slip velocity squared, as predicted by potential flow models. Unlike in potential flows, however, the velocity fluctuations were strongly anisotropic.

Since, for the most part, the number of bubbles in the simulations of Bunner and Tryggvason (2002a,b) was relatively small, the question of whether larger scale structures form is still open. Their results do, however, suggest that an initially homogeneous systems of nearly spherical bubbles will remain so for a long time. The energy spectrum for the largest simulation, in particular, quickly reached a steady state, showing no growth of modes much longer than the bubble dimensions.

In Fig. 8.1, the bubble configuration at one time from a simulation of the motion of 216 bubbles is shown on the left. The right-hand side shows a close-up of the flow around a few bubbles from a different simulation with 91 bubbles, but with all other governing parameters remaining the same. Here, $N = 900$, $\mathrm{Eo} = 1$, and the void fraction is 6%. For the larger simulation, a 256^3 uniformly spaced grid

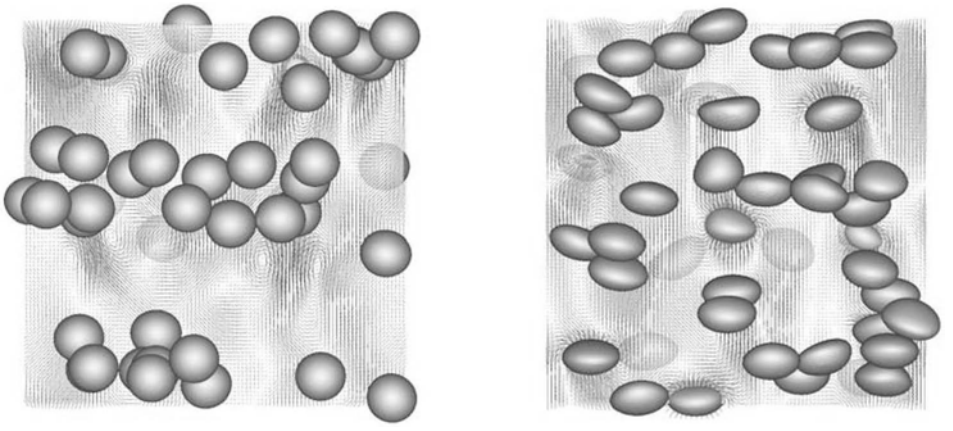


Fig. 8.2. Two frames from simulations of 48 buoyant bubbles in periodic domains. The left frame shows nearly spherical bubbles that have a tendency to line up into horizontal "rafts," whereas the more deformable bubbles on the right are more prone to form vertical columns. The governing parameters are given in the text. Reprinted with permission from Esmaeeli and Tryggvason (2005). Copyright 2005, American Institute of Physics.

was used, resulting in about 20 grid points per bubble diameter. Resolution tests, using a smaller number of bubbles, indicated that, for the governing parameters used here, this resolution yielded essentially fully converged solutions. For details, see Bunner and Tryggvason (2002a).

In the studies discussed above, the bubbles remained nearly spherical. To examine what happened if the bubbles were more deformable, Bunner and Tryggvason (2003) conducted two sets of simulations using 27 bubbles per periodic domain. In one set the bubbles were spherical and in the other set the bubbles deformed into ellipsoids. The rise Reynolds number of the bubbles generally was in the range 20–30, depending on the void fraction and the deformability of the bubbles. The nearly spherical bubbles quickly reached a well-defined average rise velocity and remained nearly uniformly distributed across the computational domain. The deformable bubbles initially behaved similarly, except that their velocity fluctuations were larger. In some cases, however, the nearly uniform distribution changed to a completely different state where the bubbles accumulated in vertical streams, rising much faster than when they were uniformly distributed. This behavior can be explained by the dependency of the lift force that the bubbles experience on the deformation of the bubbles. Suppose a large number of bubbles come together for some reason. Since they look like a large bubble to the surrounding fluid, the group will rise faster than a single bubble, drawing fluid with them. A spherical bubble rising in the shear flow at the edge of this plume will experience a lift force directed out from the plume. The lift force on a deformable bubble, on the other

hand, is directed into the plume, as explained by Ervin and Tryggvason (1997). Spherical bubbles, temporarily crowded together, will therefore disperse, but deformable bubbles will be drawn into the plume, further strengthening the bubble stream (or "chimney"). Although streaming was not always seen in simulations with deformable bubbles, it is likely that streaming would have taken place if the computations had been carried out for a long-enough time or the number of bubbles had been larger. Simulations with the bubbles initially confined to a single column showed that while the nearly spherical bubbles immediately dispersed, the deformable bubbles stayed in the column and rose much faster than uniformly distributed bubbles. A similar study, but at a considerably higher Reynolds number, was done by Esmaeeli and Tryggvason (2005), who considered 48 nearly spherical as well as more deformable bubbles. The simulations showed path oscillations of the bubbles in both cases and shape oscillations of the deformable bubbles. At quasi-steady state the distribution of the deformable bubbles was relatively uniform – although the pair distribution showed that the bubbles had a tendency to stack up vertically – but the spherical bubbles were distributed nonuniformly as a result of the formation of horizontal "rafts." For both cases, however, the probability density function of the fluctuation velocities of the bubbles was found to be approximately Gaussian. The temporal autocorrelation functions of the fluctuation velocities showed that the horizontal components became uncorrelated faster than the vertical component and the correlation time for the vertical autocorrelation for deformable bubbles was at least two times larger than that for nearly spherical ones. Figure 8.2 shows two frames from these simulations. The nearly spherical bubbles are shown on the left and the more deformable ones on the right. In these simulations, $N = 8000$ and $\alpha = 5.8\%$. For the nearly spherical bubbles $\mathrm{Eo} = 0.5$, resulting in an average rise Reynolds number of about $\mathrm{Re_b} = 92$, and for the deformable ones $\mathrm{Eo} = 4$ and $\mathrm{Re_b} = 78$. The simulations were done using a 192^3 uniform grid.

8.3 Bubbly flows in vertical channels

Bubbly flows in vertical pipes and channels are encountered in a wide variety of industrial systems and have, therefore, been the subject of several studies. The best-known early study is by Serizawa *et al.* (1975a,b), who examined experimentally the void fraction distribution and the velocity profile in turbulent air–water bubbly flows. Other experiments have been done by Wang *et al.* (1987), Liu and Bankoff (1993), Liu (1997), Kashinsky and Randin (1999), So *et al.* (2002), Kitagawa *et al.* (2004), and Guet *et al.* (2004), for example. The results show that for nearly spherical bubbles the void fraction distribution and the velocity profile in the core of the channel are relatively uniform and that a void fraction peak is generally found

near the wall for upflow but not for downflow. Sufficiently deformable bubbles, on the other hand, show exactly the opposite behavior and migrate to the center of the channel in upflow. A number of authors have also used two-fluid averaged models to predict bubbly flows in vertical channels. See, for example, Lopez De Bertodano *et al.* (1987, 1994), Kuo *et al.* (1997), Guet *et al.* (2005), and others. The models generally reproduce the experimental results reasonably well, once the free parameters have been adjusted.

While the flow is likely to be turbulent in most cases of practical interest, laminar flow is an important limiting case that can be used to explore aspects of multiphase flow modeling that do not depend on the specifics of the turbulence. This was recognized by Antal *et al.* (1991), who developed a two-fluid model for such flows and compared the model predictions with experimental results. The agreement between the model and the experiments was good, although for upflow there was a need to introduce a wall-repulsion force to keep the center of the bubbles at least a bubble radius away from the walls, and the authors observed that the results showed some dependency on the exact value of the lift coefficient used. The model of Antal and collaborators has also been examined by Azpitarte and Buscaglia (2003). Other studies of laminar flow include the experimental investigation by Song *et al.* (2001), who considered flows with both uniform and nonuniform distribution of bubble sizes, and Luo *et al.* (2003), who examined the motion of light particles. Both studies were done for upflow and both found wall-peaking.

Lu *et al.* (2006) and Lu and Tryggvason (2006, 2007, 2008) have recently examined various aspects of bubbly flows in vertical channels, using DNS. The channel is bounded by two walls and periodic in the spanwise and the streamwise directions. Gravity acts downward, so the bubbles rise due to buoyancy and an imposed pressure gradient drives the flow either downward or upward. Initially, several bubbles were distributed randomly in the channel and the flow was then simulated for a long-enough time so that it reached an approximately steady state. At steady state the wall shear balances the weight of the mixture and the imposed pressure gradient. In addition to the wall shear, other averaged quantities, such as the total flow rate, were monitored to ensure a steady state. By focusing on an infinitely long and infinitely wide (or periodic) channel at steady state, the problem is reduced to considering the various averaged properties versus the wall-normal coordinate. This simplifies both the presentation of the results and modeling of the average flow (Antal *et al.*, 1991). To get these averages by DNS, it is of course necessary to simulate fully three-dimensional channels.

Lu *et al.* (2006) focused on nearly spherical bubbles in laminar flows and found that at steady state the flow always consisted of two well-defined regions, both for upflow and downflow: a thin wall-layer and a homogeneous core, occupying most of the channel. The formation of these regions is due to lift-induced lateral motion

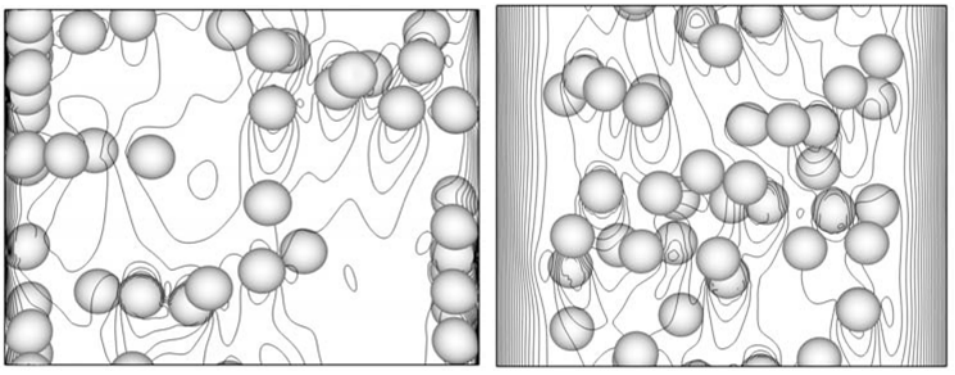


Fig. 8.3. The bubble distribution at one time for buoyant bubbles in laminar upflow (left) and downflow (right) in a vertical channel. Isocontours of the vertical velocity are also shown for a plane going through the center of the channel. Reprinted from Lu *et al.* (2006), with permission from Elsevier.

of the bubbles. For a nearly spherical bubble, rising due to buoyancy in a vertical shear, it is well known that the lift force pushes the bubble toward the side where the liquid is moving faster with respect to the bubble. Thus, in upflow a bubble near the wall is pushed toward the wall and in downflow the bubble is pushed away from the wall. The weight of the bubble–liquid mixture and the imposed pressure gradient must be balanced by a shear stress due to a velocity gradient. For upflow, the mixture, on the average, must be sufficiently light so that the imposed pressure gradient can push it upward. As bubbles are removed from the core, its average density increases until the weight is balanced exactly by the pressure gradient. The shear is then zero and the migration of the bubbles to the wall stops. For downflow the opposite happens. Bubbles move into the core and make it more buoyant, until its weight is balanced by the pressure gradient and further lateral migration is stopped. Thus, in both cases the core is in hydrostatic equilibrium and the velocity gradient is zero everywhere except in the wall layer. For upflow, where the weight of the mixture in the core is increased by pushing bubbles to the wall, the bubble-rich mixture in the wall layer is driven upward by the imposed pressure gradient. For downflow, on the other hand, bubbles must be drawn away from the wall to decrease the weight of the mixture in the core and the bubble-free wall layer is driven downward by its weight and the imposed pressure gradient. This distribution is stable in the sense that, if too many bubbles end up in the wall-layer for upflow, the core slows down with respect to the wall layer, thus generating shear that will drive the bubbles back out of the wall layer. Similarly, if too many bubbles end up in the core for downflow, its velocity is reduced and bubbles are

driven back to the wall. Figure 8.3 shows the bubble distribution and the vertical velocity at one time for both upflow and downflow.

The average void fraction is easily predicted given the considerations explained above. For both upflow and downflow the void fraction in the core is such that the weight of the mixture balances the imposed pressure gradient. For upflow, where the wall layer is about a bubble diameter thick, the void fraction is determined by how many bubbles must be removed from the core. For downflow, the thickness of the bubble-free wall layer is given by how many bubbles must be added to the core. Figure 8.4 (left frames) plots the average void fraction profile across the channel along with the predictions of the model outlined above. Obviously, the agreement is excellent. The shape of the void fraction profile in the wall layer for upflow is different from the average, but the exact shape can also be obtained by assuming that the bubbles are nearly spherical.

For downflow, where the wall layer is bubble free, the velocity profile is easily found by integrating the Navier–Stokes equations for steady laminar, parallel flow and the flow rate can be predicted analytically, with a fair degree of accuracy. For upflow, on the other hand, the presence of the bubbles makes the situation more complex and the velocity profile is not as easily found. Figure 8.4 (right frames) shows the average velocity profile across the channel for upflow and downflow. For downflow, the model predictions are also included. Since the velocity increase across the wall layer determines the liquid velocity in the core of the channel, it is critical for predicting the total flow rate. For a more detailed discussion of laminar bubbly flows in a vertical channel, see Lu *et al.* (2006).

The discussion above suggests that the flow configuration is not dependent on the flow being laminar and also not critically sensitive to the size of the bubbles, as long as they remain nearly spherical. This is indeed the case. Lu and Tryggvason (2006) studied bubbles in a turbulent downflow in some detail. The initial single-phase turbulent flow was the same as that used in Lu *et al.* (2005a), and by continuing the simulation of the single-phase turbulent flow it was confirmed that the code preserved the statistics of the turbulent flow. The pressure gradient and the viscosity used in the simulation resulted in a friction Reynolds number, using the friction velocity u^+ and the half-width of the channel H equal to $\mathrm{Re}^+ = u^+ H/\nu_\mathrm{l} = 127.3$. The bulk Reynolds number, based on the bulk velocity of the flow without bubbles and the width of the channel, was $\mathrm{Re} = U_\mathrm{b} 2H/\nu_\mathrm{l} = 3786$. The number of bubbles was in the range 18–72 with $N = 7.29 \times 10^{-7}$ and $\mathrm{Eo} = 0.3125$. The computations were done using grids with $192 \times 160 \times 96$ points, uniformly spaced in the streamwise and the spanwise direction, but unevenly spaced and clustered near the wall in the wall-normal direction. Lu and Tryggvason (2006) found that the lift force drives nearly spherical bubbles away from the walls, as for the laminar flow case. The velocity in the bubble-free wall layer is well described by the standard law

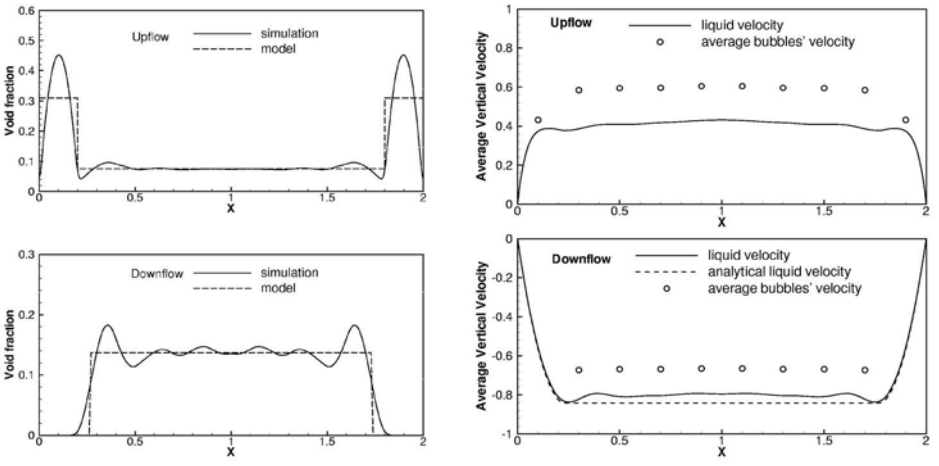


Fig. 8.4. Left: the average void fraction profiles across the channel for the simulations shown in Fig. 8.3 for upflow (top) and downflow (bottom). The dashed line is the results of a simple analytical model. Right: the average liquid velocity across the channel for upflow (top) and downflow (bottom). The open circles are the average bubble velocities. For downflow, the dashed line shows the predictions of a simple analytical model. Reprinted from Lu *et al.* (2006), with permission from Elsevier.

of the wall and the flow rate can, therefore, be found analytically, as for laminar flow. Even for a very thin wall layer (less than 50 wall units thick) the turbulence is sustained, but for thick wall layers the boundaries sometimes vary in time due to meandering of the bubbly core. Figure 8.5 shows the average liquid velocity. The velocity profile for turbulent liquid flow without bubbles is also shown. The pressure gradient for the flow without bubbles is adjusted so that the total driving force, the imposed pressure gradient and the weight of the mixture, is the same for both flows. For turbulent flows, the velocity in the middle of the channel is relatively uniform in the absence of bubbles, and since the main effect of adding the bubbles is to make the velocity there completely uniform, adding the bubbles causes surprisingly little change in the liquid velocity. The main increase in the liquid velocity takes place in the bubble-free wall layer, where the velocity profile remains nearly the same for the flows with and without bubbles. While the turbulent velocity profile without bubbles is not completely as flat as it is after adding the bubbles, the differences are small. Since the flow in the core of the channel is uniform, the turbulent Reynolds stresses there are zero and in the buffer layer these are reduced. The increase of the velocity in the buffer layer and the wall region is also cut short at the outer edge of the wall layer and replaced by the uniform velocity characterizing the bubbly core. Simulations with bubbles of different sizes in the same flow (Lu and Tryggvason, 2007) show that the bubble size has relatively little

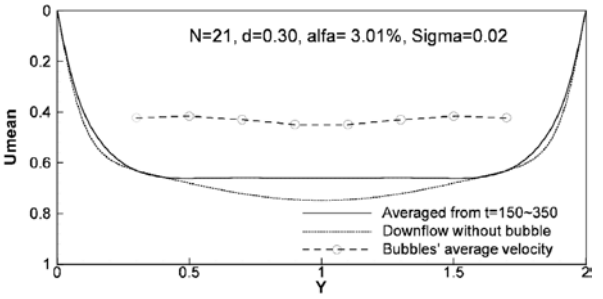


Fig. 8.5. The average liquid velocity from a simulation of turbulent bubbly downflow in a vertical channel. The velocity profile for flow without bubbles is also shown, along with the average velocity of the bubbles. Reprinted with permission from Lu and Tryggvason (2006). Copyright 2008, American Institute of Physics.

effect on the velocity profile, even for a polydisperse mixture. The rise velocity of smaller bubbles is, of course, smaller than for larger bubbles, and this conclusion obviously only holds as long as all the bubbles are nearly spherical. For details about these studies, see Lu and Tryggvason (2006, 2007).

The distribution of bubbles in a vertical channel is determined by the lateral migration of bubbles due to lift. For nearly spherical bubbles, as discussed above, the lift is toward the wall for upflow and away from the wall for downflow, thus creating exactly the correct motion to move the mixture toward hydrostatic equilibrium. For deformable bubbles, the situation is quite different. In some cases deformation can lead to a lift force in a direction opposite to that for a nearly spherical bubble and in some cases deformation, particularly when the bubble rise is unsteady, results in a lift force that is essentially zero. To examine the effect of bubble deformation in a turbulent upflow, Lu and Tryggvason (2008) simulated the motion of 21 bubbles in a rectangular vertical channel, driven by a constant pressure gradient. The initial turbulent flow was the same as for the investigation of bubbles in turbulent downflow, but the surface tension was varied to give $\mathrm{Eo} = 0.45$ for the nearly spherical bubbles and $\mathrm{Eo} = 4.5$ for the more deformable ones. The flow was simulated using a finer grid than for downflow, or $256 \times 192 \times 128$ points, with unevenly distributed grid points in the wall-normal direction. In Fig. 8.6 we show the bubble distribution and isocontours of the vertical velocity in the middle plane of the channel at one time for both cases after the flow has reached an approximate steady state. In the frame on the left the bubbles are nearly spherical, resulting in a prominent wall peak in the bubble distribution. On the right the bubbles are more deformable and, therefore, they stay in the middle of the channel. The void fraction for the spherical bubbles is well predicted by the hydrostatic model, but the deformable bubbles result in a void fraction that is highest in the center region

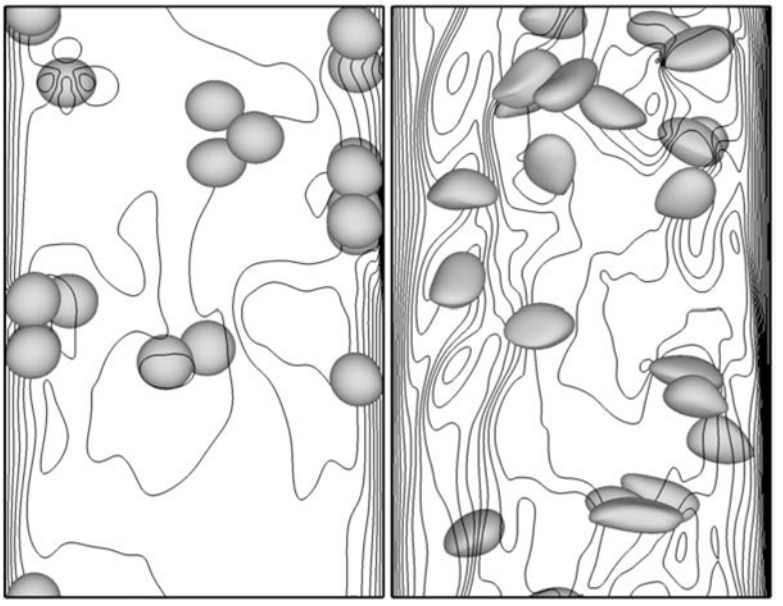


Fig. 8.6. Two frames from simulations of turbulent bubbly upflow in a vertical channel. The bubble distribution and isocontours of the vertical velocity in the middle plane of the channel are shown. In the frame on the left the bubbles are nearly spherical, resulting in a prominent wall peak in the bubble distribution. On the right the bubbles are much more deformable and stay in the middle of the channel. Reprinted with permission from Lu and Tryggvason (2008). Copyright 2008, American Institute of Physics.

of the channel and tapers off towards the walls. This difference in the void fraction distribution, shown in Fig. 8.7, has profound implications for the average velocity, as seen in Fig. 8.8, where the average liquid velocity and the average bubble velocity are plotted for both cases. For the nearly spherical bubbles there is a significant reduction in the average velocity (and thus the flow rate) after the bubbles are added, but the more deformable bubbles have a relatively minor impact on the liquid velocity, once the total pressure gradient has been adjusted to account for the reduction in average density of the mixture. The slip velocity of the bubbles is smaller for the deformable bubbles, since their drag is higher, but they rise significantly faster than the spherical ones since the liquid velocity is higher.

While the basic structure of bubbly flows in a vertical channel has been observed experimentally before, the computational studies discussed here allowed us to explore the dynamics in detail in a well-controlled and characterized situation. Results for both laminar and turbulent flows show that the structure of the flow is determined by the lift on the bubbles. At steady state the flow in the center of the channel is in hydrostatic equilibrium for both upflow and downflow (as also found

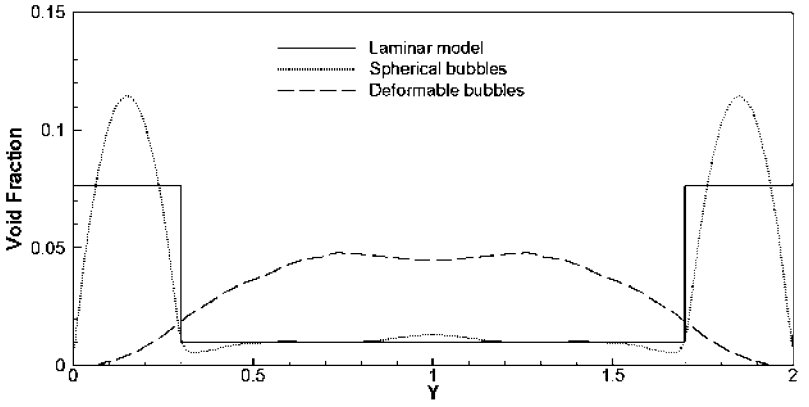


Fig. 8.7. The average void fraction versus the cross channel coordinate for the simulations shown in Fig. 8.6, along with predictions of an analytical model. Reprinted with permission from Lu and Tryggvason (2008). Copyright 2008, American Institute of Physics.

by Azpitarte and Buscaglia (2003)), and the dynamics is well described by results for homogeneous flows. The wall layer, however, is very different for upflow and downflow. For downflow, where there are no bubbles near the wall, the flow is particularly simple. In upflow, when the wall layer contains a large number of bubbles, the dynamics depends sensitively on both the number of bubbles and their deformability.

8.4 Discussion

The simulations of bubbly flows discussed in this chapter are good examples of the opportunities and challenges in applying DNS to understand complex multiphase flows. Natural problems tend to have a large range of scales, and material properties can vary greatly. On the computer, however, it is easiest to work with problems where the range of scales is small and the values of the material properties differ by a modest amount. Thus, a compromise is generally needed between what is desirable (real material properties and system size) and what is practical (or possible). For bubbly flows the limitations are many. First of all is system size. Real industrial systems often involve a large number (often hundreds or thousands) of bubbles interacting with complex flow structures. Second is viscosity. We are often interested in air bubbles in water, and even for small bubbles the rise Reynolds number is in the hundreds. System size and Reynolds number are closely related. As the Reynolds number increases, the number of grid points needed to resolve the flow around each bubble increases and fewer bubbles can therefore be included. Of considerably lesser practical importance – but often of a major significance when

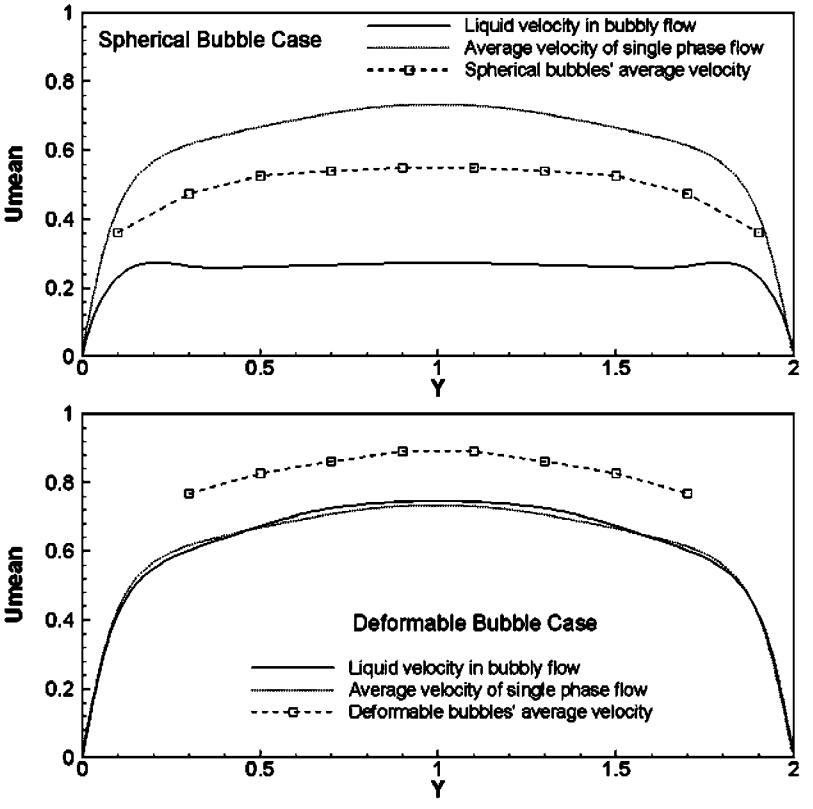


Fig. 8.8. The average liquid velocity versus the cross-channel coordinate for the simulations shown in Fig. 8.6. Reprinted with permission from Lu and Tryggvason (2008). Copyright 2008, American Institute of Physics.

talking to our experimental colleagues – is the density difference between the bubbles and the air and the surface tension for very small bubbles. For air bubbles in water the density ratio is about 800 at normal temperature and pressure. Using this ratio in a computation generally is difficult. Some methods simply will not work, and for others the solution of the pressure equation takes a long time, compared with cases where the density difference is smaller. The density inside the bubble affects the evolution of the flow in two ways. First, the buoyancy force is directly proportional to the density difference times the gravity acceleration and, second, the inertia of each bubble is determined by its density. Of those, the first one is more important. In many cases the buoyancy force is what drives the fluid (bubble and liquid) motion and getting it right affects the rise velocity. The bubble inertia is only important for unsteady bubble motion and usually it is the inertia of the surrounding fluid that is much more important than the inertia of the bubble itself.

Thus, what matters is the sum of the "added mass" and the mass of the bubble. For a spherical bubble the added mass is half the mass of water displaced by the bubble. Although at a first glance it might seem that the importance of the density ratio is larger for the inertia than for the buoyancy force, one has to remember that the total mass multiplies the acceleration, which itself depends on the density ratio. Thus, buoyancy is what drives the system and inertia is a second-order effect. These arguments are somewhat similar to those made in deriving the Boussinesq approximation for low density differences, where the full difference is included for the buoyancy term but the inertial terms are computed using the average density. The difficulties with surface tension arise from the fact that very high values can cause parasitic currents that dominate the flow (Chapter 7). Often, however, it is not necessary to set the surface tension as high as it should be. For very small bubbles, surface tension keeps the bubbles completely spherical. For high Reynolds number flows the bubbles are essentially spherical once the Eötvös number is about 0.1 and increasing the surface tension further (lowering Eo) causes no changes. Thus, once the surface tension is sufficiently high so that the bubbles stay spherical, the flow is independent of the exact value of the surface tension and there is no need to increase it further. It is important to stress that the validity of these approximations is specific to bubbly flows and that other systems will in general require different considerations.

In spite of these limitations – which get less severe every year as better methods are developed and computers get faster – DNS have already shown that they will transform the way we study bubbly flows. The two examples discussed here are only meant to demonstrate what can be done. For other simulations of bubbly flows see, for example, Tomiyama *et al.* (1993, 2002), Takada *et al.* (2001), Kanai and Miyata (2001), Kawamura and Kodama (2002), Sankaranarayanan *et al.* (2002), van Sint Annaland *et al.* (2006), and Bothe *et al.* (2006). Similar studies have been done for flows with suspended solid particles and the reader can consult Feng *et al.* (1994, 1995), Hu (1996), Johnson and Tezduyar (1997), Choi and Joseph (2001), and Pan *et al.* (2002) for an introduction to the literature. The immediate future will see increasingly more complex systems being examined by DNS and the results being used to help derive closure relations for computations of industrial systems relying on equations describing the average flow field.

9

Atomization and breakup

Various applications and natural processes involve large deformations and eventual breakup of liquid jets, layers, and droplets. When liquid masses fragment in a small number of pieces one speaks of breakup. More intense phenomena where, for instance, a liquid jet is broken into seemingly microscopic droplets are called atomization, although the term is somewhat incorrect, since the individual pieces are still far larger than atomic scales.

Nevertheless, atomization is a striking process in which finely divided sprays or droplet clouds are produced. This is often based on the ejection of a high-speed liquid jet from an *atomizer nozzle*. Many other configurations exist, such as sheets ejected at high speed from diversely shaped nozzles, or colliding with each other. As with many of the multiphase phenomena investigated in this book, atomization offers a rich physical phenomenology which is still poorly understood. Considerable progress has been made in the development of methods for atomization simulations during the last few years, and advances in hardware are making it possible to conduct simulations of unprecedented complexity.

9.1 Introduction

There are many important motivations for the study of spray formation, droplet breakup, and atomization. To take a first example from natural phenomena, spray formation atop ocean waves occurs when sufficiently strong winds strip droplets from the crests of the waves. Breaking waves also create bubbles that, when bursting at the surface, create a very fine mist that can rise high into the atmosphere. These various phenomena associated with ocean waves play an important rôle in ocean–atmosphere exchanges and interactions and thus in climate-change dynamics.

From the industrial point of view, engineers need to study and control spray properties in various ways. In some cases, relatively narrow distributions of droplet sizes are required to ensure uniform evaporation of the droplets. In other cases,

large droplets are useful as they create mixing. On the other hand, for fire hoses, atomization is to be avoided as much as possible in order to obtain longer coherent jets.

A particulary important application is combustion. The initial atomization is the most critical stage of burning liquid fuel. In the modeling of combustion of sprays by computer codes such as KIVA (Amsden, 1993; Torres and Trujillo, 2006), where the droplets are modeled as point particles, the initial atomization is the largest unknown. The importance of the initial droplet generation has stimulated the invention of a large number of atomizers and a large body of literature devoted to the study of such devices; see, for example, the books by Lefebvre (1989) and Bayvel and Orzechowski (1993).

The full simulation of atomizing high-speed jets is an extremely complex calculation involving a wide range of length scales, some examples of which are shown at the end of this chapter. Rather than aiming only at extremely massive simulations of this sort, another option is to advance our fundamental understanding of the process. For this purpose, researchers isolate some fundamental, simple configurations and attempt to extract features that yield physical insight into the mechanisms involved. One such configuration is a thin sheet of liquid suspended in air. The sheet ends with a rim that may be destabilized in various ways. The stability and other properties of this rim are analyzed in Section 9.2. Another basic flow configuration related to the aerodynamic instability of atomizing jet is the *two-phase mixing layer* depicted in Fig. 9.9.

The study of such layers has been intense, theoretically, experimentally, and more recently numerically as well. We sketch these various approaches in what follows. The theoretical approach is based in part on the theory of small perturbation of flows, or linear stability theory, and is itself rather complex. We devote some time to this theory in Section 9.3, then discuss experimental and theoretical findings. The ultimate objective is to fully simulate atomization from nozzles.

9.2 Thread, sheet, and rim breakup

9.2.1 The Plateau–Rayleigh jet instability

At relatively low velocity, a liquid stream exiting from a nozzle breaks into a sequence of droplets as everyday experience of water flowing from a faucet shows.

This observation was first made in the 19th century by Savart, using a stroboscopic technique (Eggers and Villermaux, 2008) and the instability was then studied in a rigorous mathematical manner by Plateau (1873), who showed that modulations of the jet radius of wavelength $\lambda > 2\pi r$, where r is the jet radius, were unstable. Later, Rayleigh (1879) computed the growth rate, showing that the maximum growth rate was obtained for $\lambda/r \simeq 9.01$.

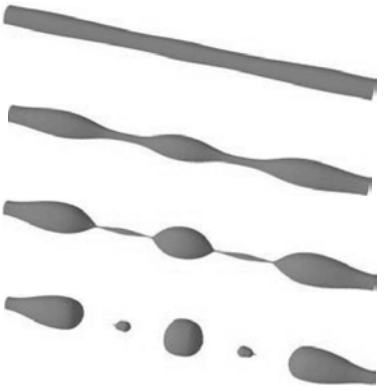


Fig. 9.1. A numerical simulation of the Plateau–Rayleigh jet instability in both the linear and nonlinear regimes (performed by the authors using the SURFER code, Zaleski *et al.*, 1992–2008).

9.2.2 Film and thread breakup

The Rayleigh analysis is linear – and is thus valid when the cylindrical shape is only slightly perturbed. As the perturbations grow, however, the surface area is steadily reduced and surface energy is transformed into kinetic energy and ultimately dissipated into heat. Narrower regions or "necks" form between bulges on the thread or jet. The thickness of the jet decreases continuously until it breaks, leaving sometimes a number of additional "satellite" droplets. The results of computations performed with the SURFER code (Zaleski *et al.*, 1992–2008) are displayed in Fig. 9.1.

The final instants of the thinning of the neck were investigated using lubrication-type equations by Eggers and Dupont (1994). These equations can be used for jets or for free thin films. They relate the film thickness or thread radius h and the average velocity u through coupled equations. The x-direction is the axis of the jet or the direction of flow in the film. The mass conservation equation can be easily derived. The amount of mass in an elementary piece of sheet or thread is proportional to h^n, where $n = 1$ for a film and $n = 2$ for a thread. The variation in time of the mass is thus proportional to $nh^{n-1}\partial_t h$. On the other hand, the mass flux is proportional to $h^n u$, where u is the velocity in the film or thread along the axial coordinate x. The mass variation from fluxing is thus $\partial(h^n u)/\partial x = nh^{n-1}u(\partial h/\partial x) + h^n(\partial u/\partial x)$. Balancing the two mass-variation terms then leads to

$$\frac{\partial h}{\partial t} + u\frac{\partial h}{\partial x} + \frac{h}{n}\frac{\partial u}{\partial x} = 0. \tag{9.1}$$

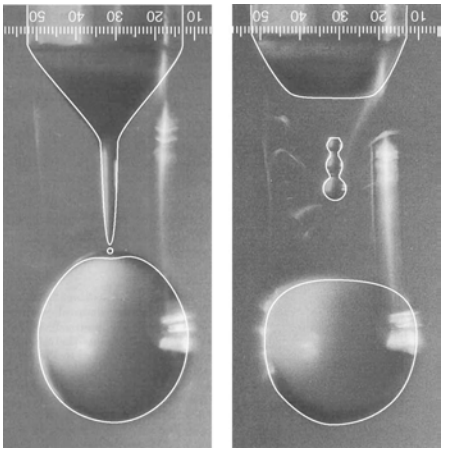


Fig. 9.2. Thread breakup. The photograph of a pendant droplet experiment in which water flows slowly from a nozzle is superposed with the result of a SURFER axisymmetric simulation performed by Denis Gueyffier. On the left the droplet is shown immediately after the first pinching. On the right the droplet is shown immediately after a second pinching. Reprinted and redrawn from Peregrine *et al.* (1990) and Gueyffier *et al.* (1999) with permission from Elsevier.

The momentum equations are more complex. For thin films see Oron *et al.* (1997), who derive

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}-\frac{4\mu}{\rho h}\frac{\partial}{\partial x}\Big(h\frac{\partial u}{\partial x}\Big)+\frac{\sigma}{\rho}\frac{\partial \kappa_1}{\partial x}=0, \tag{9.2}$$

where, as in (A.4), but with sign reversed for convenience of rotation,

$$\kappa_1=-\frac{\partial^2 h/\partial x^2}{\big[1+(\partial h/\partial x)^2\big]^{3/2}} \tag{9.3}$$

is the two-dimensional curvature for the thin film. For a thread the momentum equation is (Eggers and Dupont, 1994)

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}-\frac{3\mu}{\rho h^2}\frac{\partial}{\partial x}\Big(h^2\frac{\partial u}{\partial x}\Big)+\frac{\sigma}{\rho}\frac{\partial}{\partial x}(\kappa_1+\kappa_2)=0, \tag{9.4}$$

where the second curvature κ_2 is obtained from (A.24)

$$\kappa_2=\frac{1}{h\big[1+(\partial h/\partial x)^2\big]^{1/2}}. \tag{9.5}$$

Notice that the exact expression for the curvature κ_2 is used instead of the first-order approximation $\kappa_2\sim 1/h\equiv 1/r$. This has the advantage that the static solution of the Navier–Stokes equations for the thread is also a static solution of Equation (9.4). When the lubrication approximation is used to study the nonlinear regime,

the breakup cannot be succesfully studied because just before and after breakup the interface slope $\partial h/\partial x$ becomes very large. Using the exact expression for the curvature avoids the difficulty, although it is not clear what the exact asymptotic range of validity of the equations is. Moreover, the validity of continuum mechanics with sharp interfaces is lost at the breakup instant. We shall further discuss these issues below.

A relatively simple and interesting regime is found when fluid inertia and surface tension dominate. The third, viscous, term in Equation (9.2) or (9.4) is then neglected. Keller and Miksis (1983) found in that case self-similar solutions that become singular as time t approaches the breakup time t_c. Length is found to scale as $[\sigma(t_c-t)^2/\rho]^{1/3}$ and velocity scales as $[\sigma/\rho(t_c-t)]^{1/3}$. A different regime was found by Eggers (1993; Eggers and Dupont, 1994) when fluid inertia, viscosity, and surface tension all contribute to the dynamics of breakup. A full numerical simulation of the breakup using the VOF method was performed by Popinet (2009), who was able to recover both scaling regimes.

It is interesting to note the finite-time breakup we just discussed can be rather accurately described without much knowledge about molecular-scale phenomena. It is generally believed that breakup occurs so "quickly" in a sense that adding molecular-scale effects to the model would change the outcome very little. An example of a full numerical simulation of breakup is shown in Fig. 9.2.

Interface topology may also change through the breakup of thin films, as "holes" form in the thin liquid or gas layer (Fig. 9.3, see also Fig. 10.16). Without taking intermolecular forces into account, film breakup occurs in infinite time. A sheet parallel to the x-direction and caught in a straining flow of the type $u=\omega x, v=-\omega y$ would evolve with thickness $h\sim h_0\exp[-\omega(t-t_0)]$, where h_0 is the arbitrary initial thickness at some arbitrary initial time t_0 and it would thus reach molecular scales h_m at time $t_0+\omega^{-1}|\ln h_m/h_0|$. The breakup time may thus become arbitrarily large, albeit in a logarithmic manner, as h_m/h_0 becomes small. Similarly, in a numerical simulation where the interface evolution in time includes automatic breakup or coalescence, varying the grid resolution Δx leads to large variations in the breakup time. However, in flows where large fluctuations of the strain rate ω are observed, breakup may be relatively fast.

Moreover, in the case of thin films, attractive or repulsive forces between interfaces play an essential role in determining whether breakup will or will not occur. Small-scale forces and phenomena depend on the nature of intermolecular forces. In many circumstances these forces are affected by the presence of surface-active molecules. Perfectly clean water–air interfaces are difficult to manufacture in the laboratory and large organic molecules are often present in sufficiently large amounts to strongly affect interface dynamics. The surface properties affected are both surface tension and the plasticity of the interface, which in extreme cases

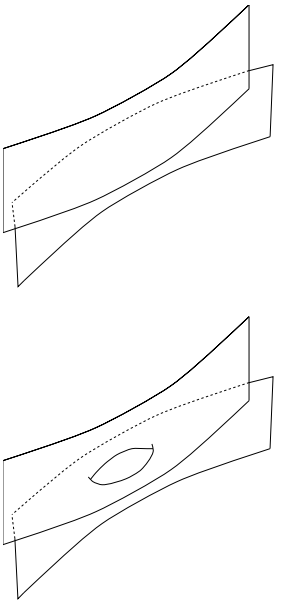


Fig. 9.3. Interface reconnection through sheet breakup. The computation of the final phase of the reconnection requires the estimation of microscopic forces.

may behave as a solid membrane, thus slowing down considerably the drainage of liquid from the film. The occurrence of such phenomena must be asserted on a case-by-case basis.

9.2.3 The Taylor–Culick rim

When liquid masses are violently shattered one observes the appearance of liquid sheets. These sheets end with a characteristic bulge or rim. The rim is created by surface tension, and it is relatively easy to predict its radius $R(t)$ and retraction speed $V(t)$ as a function of time for a sheet of constant thickness e. Conservation of mass implies that the mass flux from the sheet accounts for the increase in the mass $M(t)$ per unit length of the rim:

$$\frac{\mathrm{d}M}{\mathrm{d}t} = \rho V e. \tag{9.6}$$

Conservation of momentum with no air friction or gravity implies

$$\rho \frac{\mathrm{d}MV}{\mathrm{d}t} = 2\sigma. \tag{9.7}$$

In these balances, liquid viscosity is not taken into account. It is clear that the system has a characteristic length e, a characteristic time $t_\mathrm{c} = (\rho e^3/\sigma)^{1/2}$, and a

characteristic velocity $V_c = (\sigma/\rho e)^{1/2}$. It is easy to find a solution with steady velocity that verifies Equations (9.6) and (9.7). This solution is

$$V = \left(\frac{2\sigma}{\rho e}\right)^{1/2}, \quad M = \rho V e t. \tag{9.8}$$

The rim may retract extremely fast. For a typical splash corona, with a thickness about 100 μm and extending over $l = 10$ mm, the rim will retract in a time $l/V = 8.45 \times 10^{-3}$ s.

Viscosity does not affect the speed of the rim in our theory. This is because, if we take a sufficiently large control volume, the fluid is almost at rest in the sheet and no viscous stress is found on the boundary of the control volume. However, it can be shown that viscous effects will create a new length scale $l_\nu = (\nu t_c)^{1/2}$ (Brenner and Gueyffier, 1999) and that viscosity will affect the shape of the rim through the Ohnesorge number

$$\mathrm{Oh} = \mu/(\rho\sigma e)^{1/2}, \tag{9.9}$$

then $l_\nu = \mathrm{Oh}^{1/2} e$. When Oh is large, the rim is initially elongated with a variable thickness and a length l_ν; when Oh is small, the rim rapidly takes a circular shape. Brenner and Gueyffier (1999) have solved numerically the lubrication equations and observed various sheet shapes depending on Oh, shown on Fig. 9.4. At low Ohnesorge number a series of capillary waves appears ahead of the receding rim, leading to a "neck" region thinner than the undisturbed sheet thickness e. Song and Tryggvason (1999) have performed full simulations of the receding rim using the front-tracking method. An example of the results is shown in Fig. 9.5. The results are similar to those obtained using the lubrication equations, but the resolution of the Navier–Stokes equations allows the determination of the precise velocity field in the liquid and gas phases, yielding, for instance, vorticity distributions, as shown in Fig. 9.5.

The picture obtained by the simulations of Song and Tryggvason (1999) or Brenner and Gueyffier (1999) is, however, misleading. Continuing the simulations after dimensionless time $t' = t/t_c = 10$ at low Oh, we find that as liquid continues to accumulate in the rim, it becomes turbulent, as seen in Fig. 9.6. The turbulence is due to the interaction of several vorticity patches inside the rim, such as the one indicated by the dashed line in Fig. 9.5. The critical value for the onset of this turbulence is somewhere around $\mathrm{Oh} \simeq 0.07$. This critical number is attained in water for a film of thickness $e \simeq 3\,\mu$m, and thicker films thus have turbulent rim dynamics. In the turbulent regime, the neck thickness fluctuates strongly and may get close to the grid size, leading to breakup and detachment of the rim.

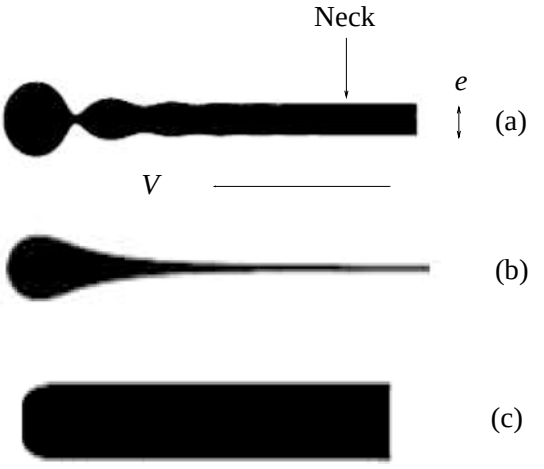


Fig. 9.4. Three rim shapes at various Ohnesorge numbers obtained using the model Equation (9.2): (a) $\mathrm{Oh} = 1.170 \times 10^{-3}$, (b) $\mathrm{Oh} = 11.70$, (c) $\mathrm{Oh} = 2.223 \times 10^{4}$. Reprinted with permission from Brenner and Gueyffier (1999). Copyright 1999, American Institute of Physics.

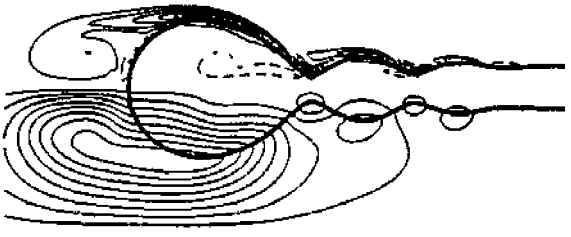


Fig. 9.5. Rim shapes obtained from front-tracking simulations at dimensionless time $t' = 6.53$ for $\mathrm{Oh} = 0.0098$ and $\rho_g/\rho_l = 0.1$. Upper half: vorticity contours; dashed lines show negative vorticity. Lower half: stream function contours. Reprinted with permission from Song and Tryggvason (1999). Copyright 1999, American Institute of Physics.

9.2.4 Rims leading to droplets and fingers

When thin sheets are realized experimentally, they exhibit rims that deform into finger-like structures perpendicular to the rim that detach droplets. One way to generate thin sheets is to pump fluid through a thin nozzle, as in the famous experiment by Crapper *et al.* (1973). A photograph of the experiment showing the sheet frontally and sideways is shown in Fig. 9.7. Sheets appear naturally in splash experiments, as seen in Chapter 10, or in some regimes of atomization experiments discussed later in this Chapter. Other ways to create thin sheets with rims is to pierce a preexisting thin film with an impacting solid object, as done by Rayleigh (1900), or to impact a jet on a small flat obstacle, as first done by Savart. A review

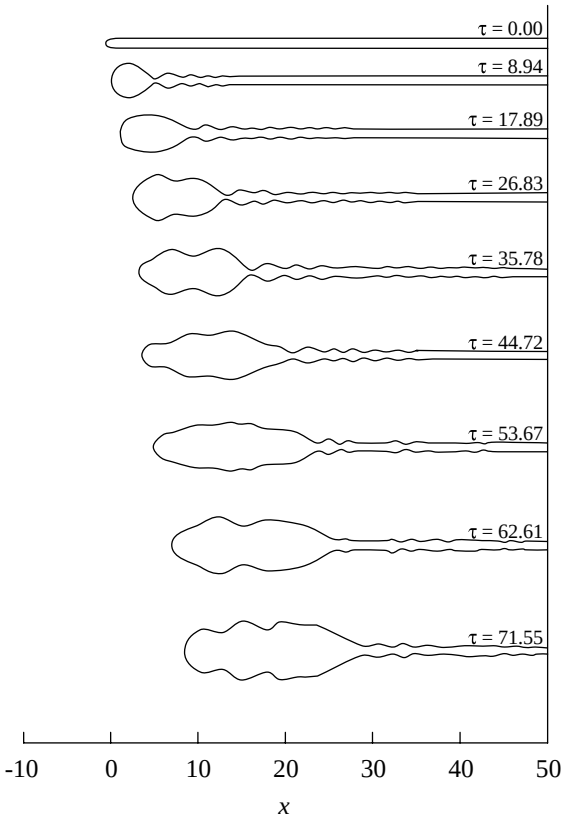


Fig. 9.6. The rim computed using Gerris for $\mathrm{Oh} = 0.0045$ and $\rho_g/\rho_l = 0.1$. Dimensionless times $\tau = t' = t/t_c$ are shown next to the simulation snapshots. The reference frame and the domain of the simulation are translated to the right roughly with the rim velocity so that the rim seems to move to the right by a lesser amount than it would in a reference frame anchored to the sheet. Simulation performed by Leonardo Gordillo.

of Savart's account of this experiment is given by Villermaux and Clanet (2002). Although the mechanisms by which the rim breaks into droplets may differ in each type of experiment, it is interesting to discuss simple ideas on how the rim deforms into fingers or droplets.

One theory for the origin of droplets in various breakup and atomization phenomena is that the Taylor–Culick rim detaches and then undergoes a Rayleigh jet instability. This mechanism was suggested by Dombrowski and Johns (1963). It requires, however, the neck in Fig. 9.4 to reach molecular lengths and does not seem to be realized in experiments.

A completely different mode in which a rim may lead to droplets is the focusing instability of wavefronts. A surface moving locally, normal to itself, at constant

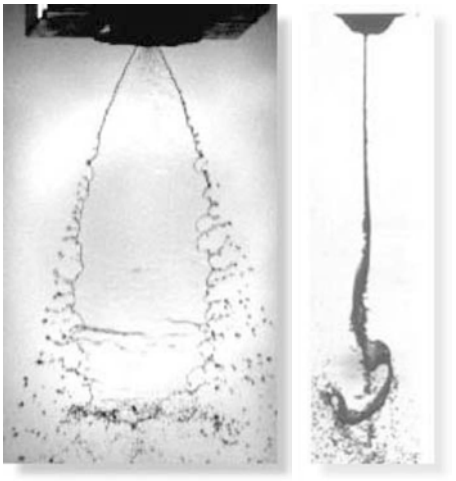


Fig. 9.7. A free sheet experiment. Reproduced with permission from Crapper *et al.* (1973). Copyright 1973, Cambridge University Press.

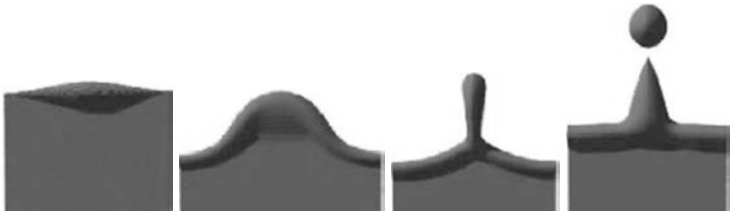


Fig. 9.8. A possible mechanism for droplet formation, involving an initially deformed bulging end rim.

velocity behaves as a wavefront. When the surface is moving in the direction of its center of curvature, it generally focuses at this point, then self-intersects, forming a so-called "kinematic singularity," put forward by Yarin and Weiss (1995) in the context of splash corollas. However, simulations show that the formation of a droplet by this mechanism requires very large initial deformations of the rim to yield significant amounts of mass in the fingers. Thus, it seems unlikely that this is the main mechanism responsible for the breakup of rims.

Another mechanism is the destabilization of a bulging rim, which is similar to that of an isolated cylinder in the Plateau–Rayleigh instability of Section 9.2.1, but here the rim is attached to the sheet. An example is shown in Fig. 9.8. The instability requires that the wavelength of the perturbation of the rim satisfies $\lambda \gg e$. Indeed, if the thickness of the sheet e is too large then the radius $R(t)$ of the rim grows too rapidly. It then becomes quickly of the order λ/π, and above the ratio

$R/\lambda = 1/\pi$ the Plateau–Rayleigh instability is suppressed. In the calculations of Fig. 9.8, $\lambda/e = 130$.

An analytical stability theory describing the various instability modes of a Taylor–Culick end rim attached to a sheet has been proposed by Roisman *et al.* (2006).

9.3 High-speed jets

9.3.1 Structure of the atomizing jet

As the velocity of the jet increases beyond the Plateau–Rayleigh regime of Section 9.2.1, a complex sequence of events occurs. The intact length of the jet increases, then decreases again, and the jet eventually both develops a large-scale, quasi-helical instability and also forms small ligaments, sheets, and droplets. The different regimes through which the jet passes are called the dripping regime, first wind-induced regime, second wind-induced regime, and finally atomization regime. The last regime is the one in which small-scale structures develop of size ℓ much smaller than the jet diameter ($\ell \ll D$). It is the most challenging from the point of view of technology and numerical simulations.

The global structure of the flow, as seen in experimental photographs, shows a diverse and complex mixture of liquid masses. The dense central region is called the liquid core. It is surrounded by various structures, either sheet-like or finger-like, that grow downstream and eventually produce detached droplets. This characterizes the fiber or membrane breakup. The nature of the instability and the characteristic scales depend on the parameter regime and on the flow inside the nozzle. The boundary layer size in the gas and the liquid upstream of the nozzle, as well as the turbulence level, are particularly important parameters.

In various regions of the flow the detached droplets can collide and possibly coalesce or be caught in a rapid shear or accelerating flow and suffer a further breakup, called *secondary atomization*.

The important dimensionless numbers are the Weber numbers based on liquid or gas properties and the diameter D of the liquid nozzle $\mathrm{We}_i = \rho_i U_i^2 D/\sigma$, where $i = \mathrm{l}$ or g, the Reynolds numbers $\mathrm{Re}_i = \rho_i U_i D/\mu_i$, also based on the liquid jet diameter, and for coaxial jets the momentum flux ratio $M = \rho_\mathrm{g} U_\mathrm{g}^2/(\rho_\mathrm{l} U_\mathrm{l}^2)$. Two additional numbers are the density and viscosity ratios, $r = \rho_\mathrm{g}/\rho_\mathrm{l}$ and $m = \mu_\mathrm{g}/\mu_\mathrm{l}$ respectively.

Experiments show that the core or "potential cone" length L of the liquid near the axis that is not disturbed by instability decreases when M increases. For large $U_\mathrm{g}/U_\mathrm{l}$ a good approximation is $L/D \simeq 6/\sqrt{M}$, where D is the jet diameter. For the single jet atomizer the liquid core length is close to $L/D \simeq 6(\rho_\mathrm{l}/\rho_\mathrm{g})^{1/2}$, a result already proposed by Taylor (1963). These results may be explained using the

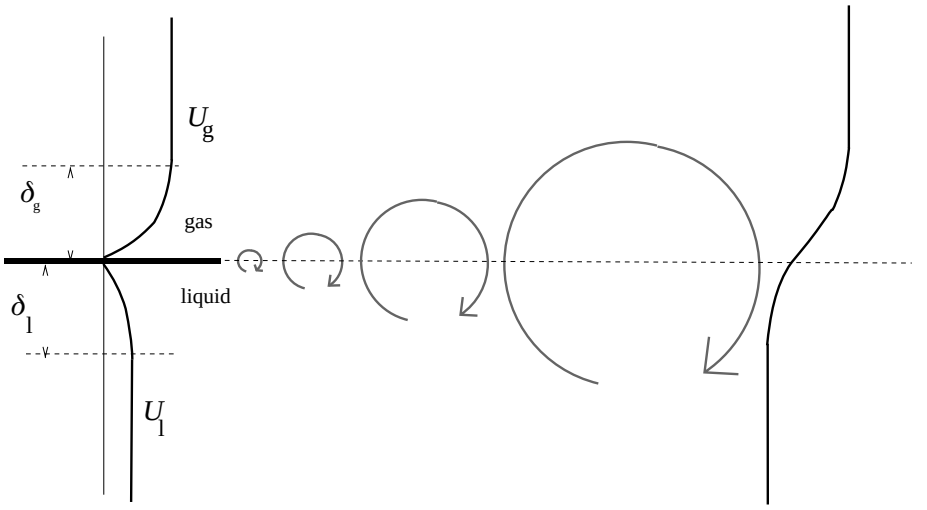


Fig. 9.9. A mixing layer behind a splitter plate is used to model the flow near the exit of the nozzle in a co-flowing atomizer. Upstream, the flow has two boundary layers of thickness δ_g and δ_l. Downstream, the boundary layers rearrange and the liquid boundary layer reverses. The size of the boundary layers upstream is exaggerated.

concept of a *two-phase mixing layer*; see Fig. 9.9. In a mixing layer, boundary layers of thickness δ_i ($i = \mathrm{l}$ or g) play an important rôle in the flow stability properties. Dimensionless numbers may alternately be defined based not on the liquid jet diameter as before, but on the boundary layer sizes. We note these new numbers with an asterix: $\mathrm{We}_i^* = \rho_i(U_i - U_{\mathrm{int}})^2\delta_i/\sigma$ and $\mathrm{Re}_i^* = \rho_i|U_i - U_{\mathrm{int}}|\delta_i/\mu_i$, where U_{int} is the fluid velocity on the interface.

9.3.2 Mechanisms of droplet formation

One of the most difficult conceptual issues in atomization is understanding the physical mechanisms that lead to atomization. Several such mechanisms have been proposed and confirmed experimentally. The three most important ones are the effect of upstream liquid turbulence, the stability of idealized liquid jets or sheets, and cavitation in the liquid.

The liquid-turbulence theory explains droplet formation as a consequence of the deformation of the interface by strong vortices in the liquid. Vortices of size ℓ, comparable to the nozzle diameter D, have a kinetic energy $\rho_\mathrm{l}\ell^3U_\mathrm{l}^2$. To deform the interface into a ligament or filament of size ℓ they must create a surface energy $\sigma\ell^2$, which leads to a ligament size $\ell \simeq D\mathrm{We}_\mathrm{l}^{-1}$. More complicated dependencies are obtained if $\ell \ll D$, in which case smaller vortices may have less energy. This mechanism requires Re_l to be sufficiently large for turbulence to be well developed in the liquid.

The stability theory instead assumes an idealized flow out of the nozzle. For example, the flow may be modeled as the mixing layer profile of Fig. 9.9. The growth of small perturbations leads to waves observed at the liquid surface. Further destabilization of two-dimensional waves, or direct growth of three-dimensional perturbations, then leads to the ligaments observed in Fig. 1.2. The earliest stability theories are related to the Kelvin–Helmholtz theory of a single-phase mixing layer.

Cavitation mechanisms arise when regions of very low pressure are present upstream of the nozzle, leading to the growth of vapor bubbles or sheets in the liquid. These bubbles may nucleate in low- or negative-pressure regions or preexist as small-size bubble seeds. The appearance of cavitation marks an important change in the jet shape and is expected at very high speed when the pressure variations ahead of the nozzle are large.

It is reasonable to assume that the various theories apply in different regions of parameter space. High liquid Reynolds number would lead to liquid-turbulence effects, while high gas Weber number leads to aerodynamic effects. This is one of the features of the Hopfinger diagram of Fig. 9.10.

9.3.3 Stability theory

Stability theory is rather complex and comes in many flavors. The simplest, but not necessarily the most applicable theory, retains the effects of surface tension and inertia, but not of viscosity.

9.3.3.1 Elementary Kelvin–Helmholtz analysis

Here, the dynamics is linearized assuming a perturbation of the interface of small amplitude h_1 leading to modes with exponential growth. The interface height is written as

$$h = h_0 + h_1 \exp(st + \mathrm{i}\omega t + \mathrm{i}kx)\,, \tag{9.10}$$

where the growth rate s and the frequency ω depend on the wavenumber k and the physical parameters of the problem. Expressions for the velocity and pressure exhibit exponential growth at the same growth rate and frequency. Calculations are easiest if one considers a flow with a sharp velocity profile, as shown in Fig. 9.11(a), where the sharp profile is marked "KH". In the single-phase case $\rho_\mathrm{g} = \rho_\mathrm{l}$ it is found (see Appendix D for detailed calculations) that the growth rate is

$$s = \frac{k}{2}|U_\mathrm{g} - U_\mathrm{l}|. \tag{9.11}$$

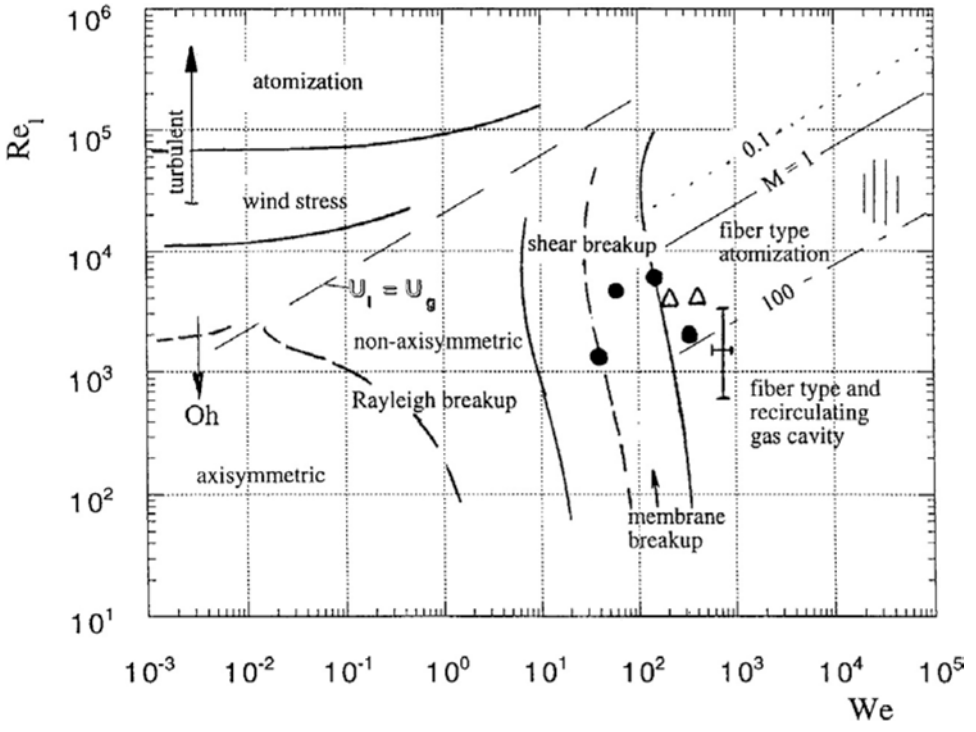


Fig. 9.10. A diagram of the various atomization regimes, as a function of the liquid Reynolds number $\mathrm{Re} = \mathrm{Re}_\mathrm{l}$ and the gas Weber number $\mathrm{We} = \mathrm{We}_\mathrm{g}$. The "atomization" regime corresponds to the liquid-turbulence-dominated regime, the "Oh" regime to the dominance of liquid viscosity, and thus large Ohnesorge number Oh. The aerodynamic regimes are to the right of the diagram. Reproduced from Lasheras and Hopfinger (2000), with permission from Annual Reviews, Inc.

For the two-phase case the more complex expression (D.21) holds. In the limit $\rho_\mathrm{g} \ll \rho_\mathrm{l}$ however, it reduces to

$$s \simeq \left(\frac{\rho_\mathrm{g}}{\rho_\mathrm{l}}\right)^{1/2} k|U_\mathrm{g} - U_\mathrm{l}|. \tag{9.12}$$

Both expressions predict an indefinitely large growth rate as the wavenumber increases. This situation is physically not meaningful, as perturbations of small height h_1 would lead to very large deformations of the flow at arbitrary small times. The time evolution of the problem would thus be unbounded and indeterministic, except for very smooth initial conditions. This situation is called an "ill-posed problem in the sense of Hadamard" in mathematics and, interestingly, it also arises in averaged models of two-phase flow (Prosperetti, 2007), albeit for reasons unrelated to the Kelvin–Helmholtz instability.

Taking into account viscosity or surface tension damps the growth of the Kelvin–Helmholtz instability at high wavenumbers. The effect of surface tension is the easiest to calculate. For $\rho_g \ll \rho_l$ the wavelength of maximum growth is given by $\lambda_m = 3\pi D \mathrm{We}_1^{-1}$ (see Appendix D). The effect of viscosity may be also estimated in a rather simple but inconsistent way by introducing piecewise linear boundary layers in the base flow of the stability analysis, and keeping the inviscid equations for the stability analysis (see Appendix D). The result is the wavelength of maximum growth $\lambda_m \sim (4\pi/3)(\delta_g \rho_g/\rho_l)$. This expression is not well verified experimentally. It is thus necessary to turn to a consistent inclusion of viscosity in the analysis.

9.3.3.2 Orr–Sommerfeld analysis

By keeping viscosity in order to improve physical realism, the relevant equations for the flow are the Navier–Stokes equations.

One obvious effect of viscosity is that it removes the sharp velocity jump. Solutions of the Navier–Stokes equations cannot sustain jumps in velocity, which are replaced by boundary layers of the type shown in Fig. 9.11(a), with thickness δ_g and δ_l. Recall that the velocity at the interface is denoted U_{int}.

The parameters are connected by the tangential stress condition (2.57). In the absence of any surface tension gradient these conditions read for a parallel flow $u(z,t)$

$$\mu_g \left.\frac{\partial u}{\partial z}\right|_g = \mu_l \left.\frac{\partial u}{\partial z}\right|_l \tag{9.13}$$

and hence

$$\mu_g \frac{|U_g - U_{int}|}{\delta_g} = \mu_l \frac{|U_l - U_{int}|}{\delta_l}, \tag{9.14}$$

from which follows the next relation between the dimensionless numbers defined in Section 9.3.1:

$$\frac{\mathrm{Re}_g^*}{\mathrm{Re}_l^*} \frac{m^2}{n^2 r} = 1, \tag{9.15}$$

where $n = \delta_g/\delta_l$ is the ratio of boundary-layer sizes. Moreover, the boundary-layer size is not arbitrary but results from viscous diffusion effects. For instance, a temporal instability set up $\delta_i \sim (\nu_i t)^{1/2}$ leads to

$$n \sim (m/r)^{1/2}. \tag{9.16}$$

One interesting result for typical values of fluid densities and viscosities is that the velocity jump in the liquid is much smaller than in the gas, $|U_l - U_{int}| \ll |U_g - U_{int}|$, which is equivalent to saying that the vorticity present at the trailing edge

of the separating plate has been mostly diffused into the gas. This is visible in Fig. 9.12(a).

The full linearized Navier–Stokes equations are the Orr–Sommerfeld equations, given in Appendix D. Here, we only discuss some relevant results. A complex structure of modes is found; however, some of them may be interpreted in a simple way. One branch of modes converges asymptotically to inviscid solutions; however, this convergence is rather slow for the relatively small Weber and Reynolds numbers relevant for atomization. Another mode is connected to the stability of very viscous shear layers. For this mode, the analysis of the behavior for large $\mathrm{Re_g}$ and $\mathrm{Re_l}$ shows that the wavenumber k_m of the maximum growth rate scales as $\sqrt{\mathrm{Re_g^*}}$, while the maximum growth rate scales as $k_\mathrm{m}|U_\mathrm{g}-U_\mathrm{l}|$ (Boeck *et al.*, 2007) .

This observed scaling with $\mathrm{Re_g^*}$ is caused by an unstable mode described by Hooper and Boyd (1983). The physical mechanism producing this mode was later explained by Hinch (1984). To recognize the contributions of these authors, this mode is referred to as the H-mode.

As shown on Fig. 9.13, agreement is obtained between the prediction of Orr–Sommerfeld stability theory and numerical computations using the SURFER code (Zaleski *et al.*, 1992–2008) in the "temporal setup" case, i.e. in the case where periodic boundary conditions, discussed in Section 3.7, are used.

In the particular case shown in Fig. 9.13, the quality of the agreement depends on which type of average is used for viscosity in mixed cells, whether arithmetic or harmonic. It may also depend on subtle details of the initialization, as the initial perturbation is small and may be corrupted by numerical errors.

9.4 Atomization simulations

9.4.1 Two-dimensional, temporal simulations

Using the method described in this book, relatively low-resolution (here, low resolution means a 128×128 grid) simulations of a two-layer configuration may be performed in a very short time. The flow configuration at the initial time follows the description in Fig. 9.11a with periodic boundary conditions in the horizontal direction. Periodic simulations correspond to the "temporal" version of stability theory. A typical result is shown in Fig. 9.11b.

With finer resolution the ligaments are stretched much longer (Fig. 9.14, on a 512×512 grid).

9.4.2 Two-dimensional spatially developing simulations

A more realistic result is obtained when entry conditions are imposed that mimic the flow at the exit of the nozzle. One way to represent entry conditions is to specify

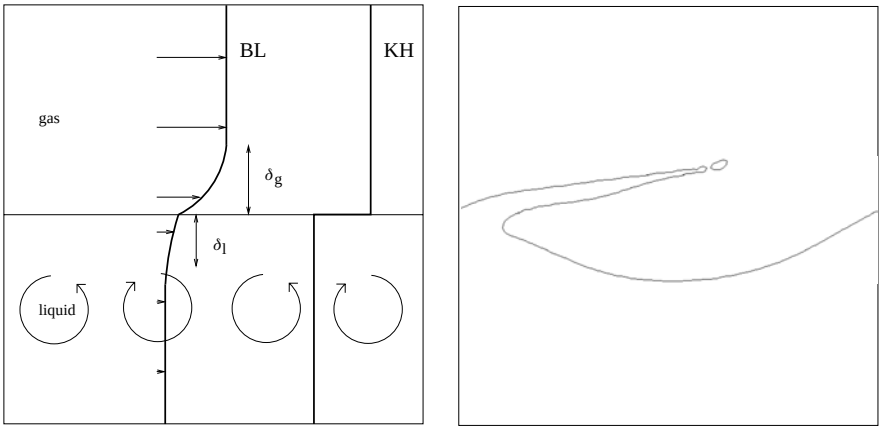


Fig. 9.11. (a) Setup of the "temporal" mixing layer problem. The profile marked "BL" is a realistic profile with boundary layers evolving in a mixing layer. Profile "KH" is a sharp profile used for ease of calculation and setup. The instability may be triggered by an array of vortices in the liquid layer in the bottom. (b) Simulation at low (128×128) resolution. The ligament is shown at $t = 8.4$ just after breakup. Reprinted with permission from Boeck *et al.* (2007).

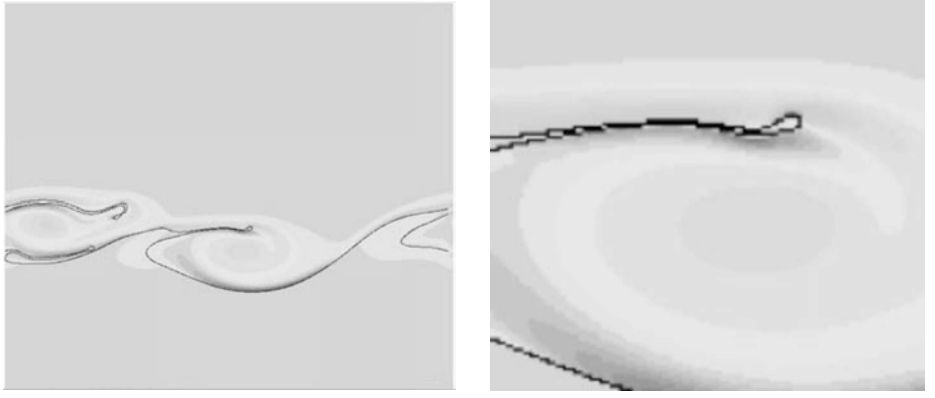


Fig. 9.12. (a) Typical high-resolution simulation. The colors indicate the vorticity. (b) Detail of simulation showing the end rim and a "Kelvin–Helmholtz roller" trapped below the sheet. See plate section for color version.

an entry profile for the velocity, setting $u(0,y,t) = u_{\text{entry}}(y)$ as a boundary condition. As an example, we show in Fig. 9.15 the results of a simulation in which the profile u_{entry} was flat with value U_l with the exception of two linear boundary layers each about 10% of the diameter. Parameters are $U_l = 300\,\text{m/s}$, $U_g = 0$, $\mu_l = 6 \times 10^{-3}\,\text{Pa s}$, $\mu_g = 1.7 \times 10^{-5}\,\text{Pa s}$, $D = 0.2\,\text{mm}$, $\rho_g = 20\,\text{kg/m}^3$, $\rho_l = 1000\,\text{kg/m}^3$,

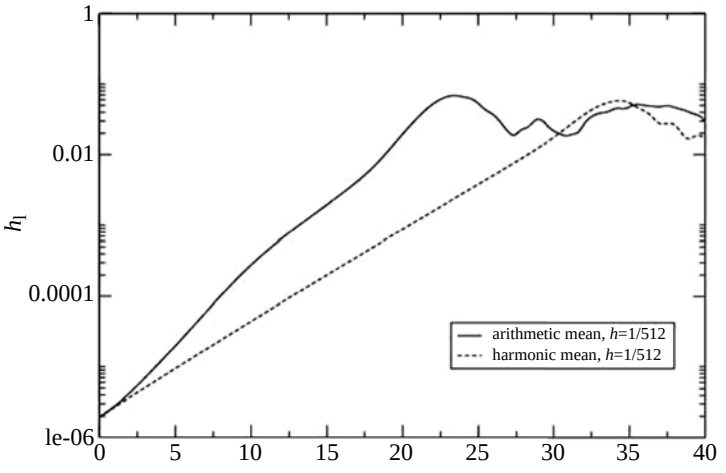


Fig. 9.13. Amplitude growth for the arithmetic and harmonic mean of the viscosity with small initial amplitude. The harmonic mean result is very close to linear theory even at small amplitudes, while the arithmetic mean is close to linear theory only when the interface height is larger than one grid size. Reprinted with permission from Boeck *et al.* (2007).

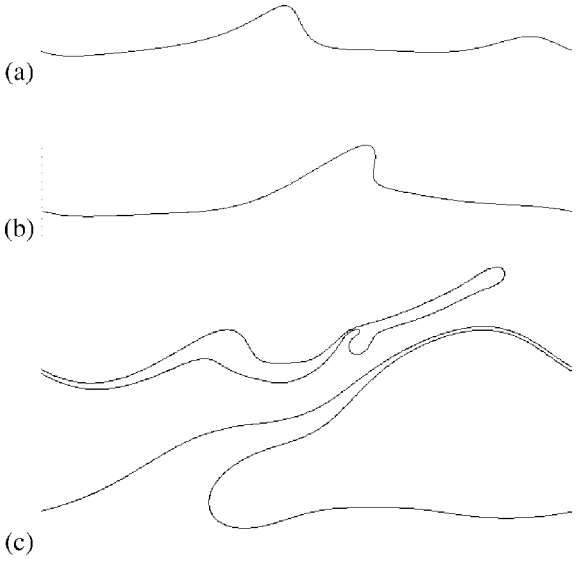


Fig. 9.14. Three stages of ligament formation and elongation on a 512×512 grid, at higher Reynolds number, $\mathrm{Re_g} = 4000$, and $\mathrm{We_g} = 500$, $r = 0.1$, at times (a) $t = 4$, (b) $t = 4.8$, (c) $t = 9.76$. Reprinted with permission from Boeck *et al.* (2007).

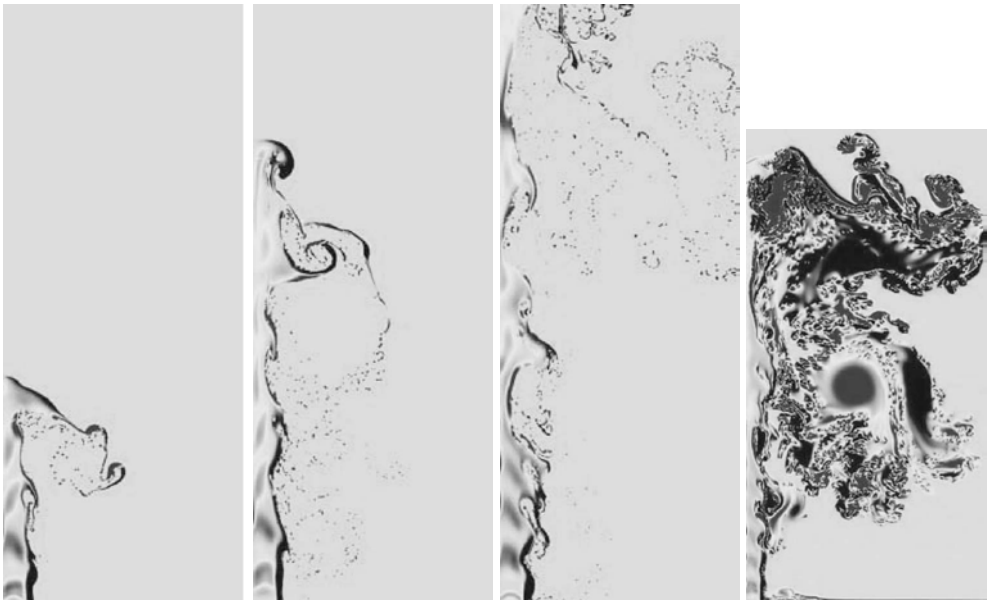


Fig. 9.15. The spatial development of an atomizing jet simulated in two dimensions using SURFER (obtained by Leboissetier (2002)). The color scheme represents the vorticity levels in the liquid. The vorticity in the gas is masked in the three figures on the left. A very strong coherent structure is seen in the gas region in the last figure on the right. Parameters are given in the text. See plate section for color version.

and domain size $2 \times 8\,\mathrm{mm}^2$, which leads to the following dimensionless parameters based on the boundary-layer size: $\mathrm{Re}_\mathrm{g}^* = 7060$, $\mathrm{Re}_\mathrm{l}^* = 850$, $r = 2.3 \times 10^{-2}$, $m = 2.83 \times 10^{-3}$, $\mathrm{We}_\mathrm{g}^* = 1500$, and $\mathrm{We}_\mathrm{l}^* = 6.38 \times 10^4$. Based on jet diameter, the dimensionless numbers are now $\mathrm{Re}_\mathrm{g} = 70\,600$, $\mathrm{Re}_\mathrm{l} = 8500$, $\mathrm{We}_\mathrm{g} = 15\,000$, and $\mathrm{We}_\mathrm{l} = 6.38 \times 10^5$. The smallest cell size in the simulation is $\Delta x = 2\,\mu\mathrm{m}$, so $\Delta x/D = 10^{-2}$.

The outcome of this simulation depends strongly on the amount of turbulence injected at the entrance. This turbulence was injected as small vortices of random orientation and of two sizes. The large vortices had a size matching the jet radius, the small ones the boundary layers. Leboissetier (2002) has shown that without the injection of these small vortices there is very little atomization. However, the simulation has an enormous Re_g and is not sufficiently well resolved to avoid a large numerical viscosity effect. Thus the relative jet stability observed in the absence of perturbation at the inlet is an artifact for these conditions.

A more realistic alternative to the specification of entry profiles is to simulate the flow inside the nozzle. This is what is done in Fig. 9.16. In this case the velocity

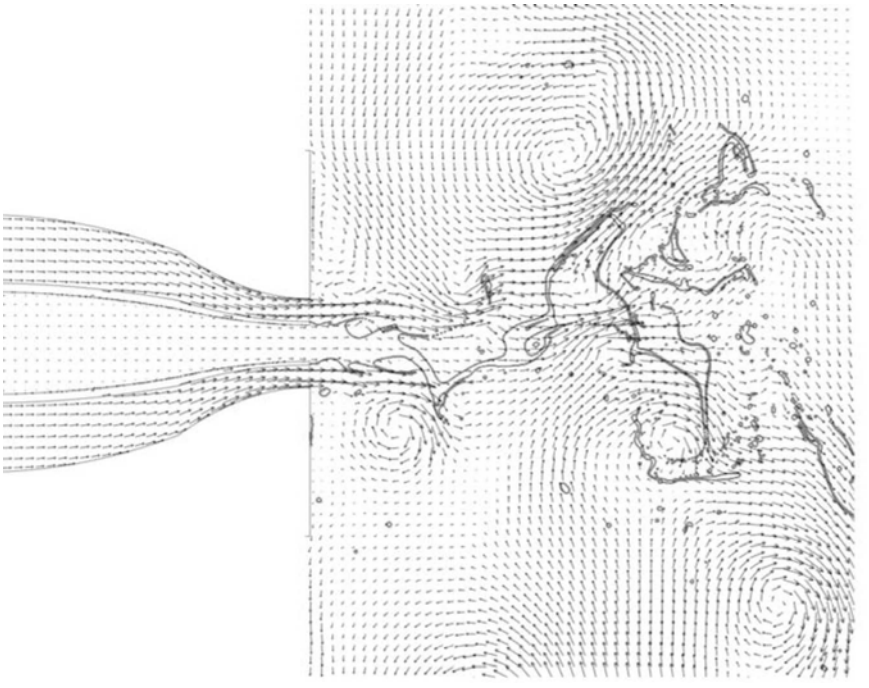


Fig. 9.16. Simulation in two dimensions of coflowing liquid and gas jets including the flow inside the nozzle. Governing parameters are given in the text. As in Fig. 9.15, large coherent structures are seen in the gas phase. Simulation carried out by Daniel Fuster. Reprinted from Fuster *et al.* (2009b) with permission from Elsevier.

profile near the injection was almost flat. Therefore, boundary layers are almost nonexistent. Moreover, the Reynolds and Weber numbers were much smaller. Parameters of the simulation are $U_\mathrm{l} = 20\,\mathrm{m/s}$, $\mathrm{U_g} = 100\,\mathrm{m/s}$, $\rho_\mathrm{g} = 2\,\mathrm{kg/m^3}$, $\rho_\mathrm{l} = 20\,\mathrm{kg/m^3}$, $\mu_\mathrm{g} = 10^{-4}\,\mathrm{Pa\,s}$, $\mu_\mathrm{l} = 10^{-3}\,\mathrm{Pa\,s}$, $\sigma = 0.03\,\mathrm{N/m}$, liquid injector radius $R = 4 \times 10^{-4}\,\mathrm{m}$, separator plate width $e = 80\,\mu\mathrm{m}$, which leads to these dimensionless numbers: $m = 0.005$, $r = 0.1$, $\mathrm{Re}_\mathrm{l}^* = 80$, $\mathrm{Re}_\mathrm{g}^* = 145$, $\mathrm{We}_\mathrm{l}^* = 48.5$, and $\mathrm{We}_\mathrm{g}^* = 19.4$. The smallest cell size in the simulation is $\Delta x = 7\,\mu\mathrm{m}$, so $\Delta x/D = 0.88 \times 10^{-2}$ and $\Delta x/e = 0.088$. Because there is no boundary layer, the dimensionless numbers are constructed using the separator plate width as length scale. The instability and the formation of waves are then obtained without the addition of noise. Well-developed waves are seen very close to the nozzle exit, in agreement with experimental observations (Raynal, 1997).

The observed instabilities were compared with linear theory and experiments in Fuster *et al.* (2009a,b). It was found that, for moderate momentum ratios M, linear theory and simulations agree, but that for large momentum ratios

($M \sim 16$) and air–water conditions (Fuster *et al.*, 2009a) a difference is found between linear theory and experiments, while simulations agree rather well with experiments. This is explained by large nonlinear effects immediately after the separator plate.

9.4.3 Three-dimensional calculations

There are two competing viewpoints on the three-dimensional instability. In the direct mechanism, transient-growth theory is used to predict the algebraic non-exponential growth of a three-dimensional perturbation (Yecko and Zaleski, 2005); this point of view has not yet been tested, either experimentally or numerically. In the second point of view, as one moves downstream, the three-dimensional instability develops as a secondary instability on top of the two-dimensional structure previously formed. There are several different mechanisms for the secondary instability in the literature. The simplest idea is that two-dimensional instabilities create thin sheets with end rims, as discussed in Section 9.2.3. The rims may then lead to droplets through the mechanisms discussed in Section 9.2.4.

Other mechanisms involve three-dimensional instabilities that arise before the sheets are fully formed; for instance, the two different types of "Rayleigh–Taylor" instabilites discussed by Marmottant and Villermaux (2002) and Ben Rayana *et al.* (2006).

Full three-dimensional numerical simulations are arguably the best hope for resolving these issues. In Fig. 9.17 we show a numerical simulation of the temporal development of the three-dimensional structure in a shear layer. The simulation is initialized with a horizontal velocity jump (without boundary layers). The initial condition contains a small two-dimensional perturbation of the interface of the form $h(x,y,t) = h_0 + \varepsilon \cos(kx + \phi(y))$, where ε is a small parameter. The phase ϕ has a small modulation in the third dimension $\phi(y) = a\cos(k_y y)$, where a is another small parameter. Interestingly, the simulation shows an exponential growth of the three-dimensional perturbation, before the rim is fully formed, showing that at least in this case the capillary effects in the rim are not relevant. The perturbation seems to grow because of the areodynamic pressure exerted by the air flow on the liquid sheet. Another striking effect is the eventual formation of several holes in the sheet. The holes cause the appearance of a typical "fish-bone" patterns, with lateral liquid cylinders branching from the side of the main ligament. In a sense the holes are not realistic, because they are expected to occur in reality when the sheet thickness reaches molecular length scales, while in the simulation they occur when the sheet thickness becomes as small as the grid spacing. However, experimental

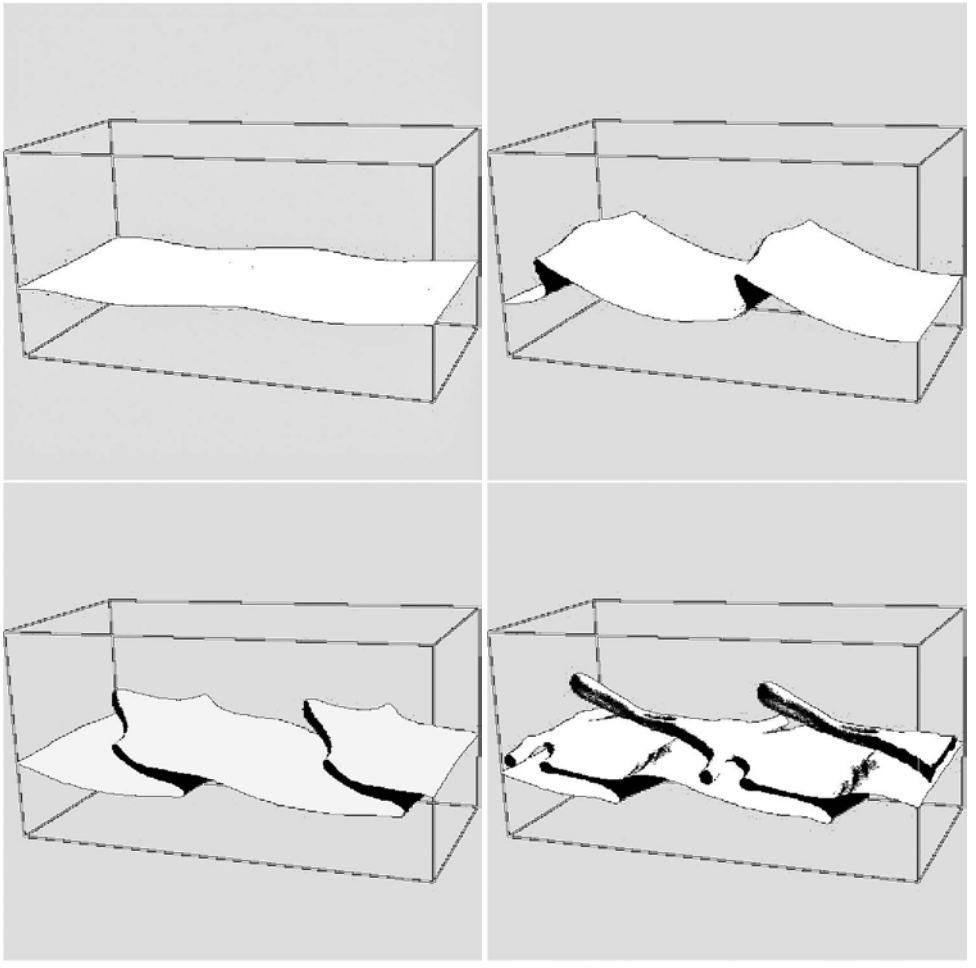


Fig. 9.17. The development of the Kelvin–Helmholtz instability simulated in three dimensions (obtained by Jie Li; see Li (1996)).

photographs show the fish-bone pattern in some cases, which seems to indicate that hole formation may also represent what happens in reality.

Simulations of three-dimensional spatially developing jets show similar ligament and droplet formation. A simulation of a coaxial jet at relatively low velocity is shown in Fig. 9.18 together with the adaptive grid used to generate it. Parameters are the same as for Fig. 9.16, except that the small separator plate near the injection has a smaller thickness $e = 50\,\mu\text{m}$ and the grid is coarser: the smallest cell size in the simulation is $\Delta x = 15.6\,\mu\text{m}$, so $\Delta x/D \simeq 2 \times 10^{-2}$ and $\Delta x/e \simeq 0.3$. Some turbulence is added near the inlet to generate the instability; however, the

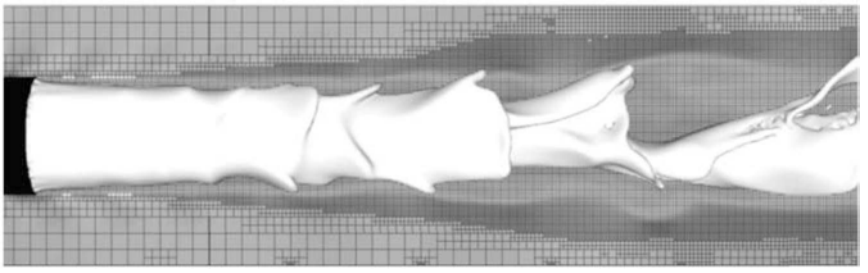
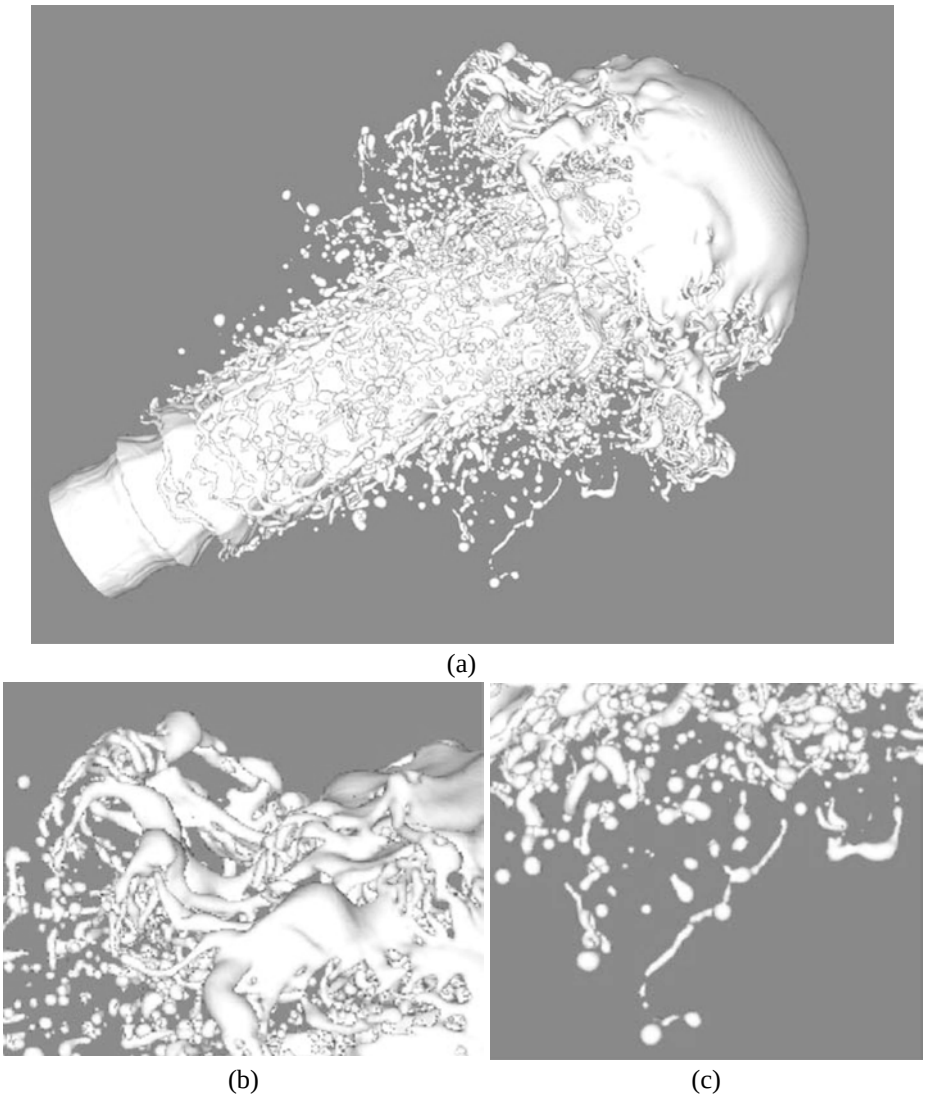


Fig. 9.18. The spatial development of coflowing atomizing jets simulated using Gerris in three dimensions. Gerris uses an adaptive oct-tree mesh which is also shown in the figure. Parameters are given in the text. Reprinted from Fuster *et al.* (2009b) with permission from Elsevier.

turbulence level is not as high as in the real experiments. It should be noted that the waves grow downstream much more slowly than in the case with the nozzle shown in Fig. 9.16. This constrast is currently being investigated, and may be due to the presence of the separator plate, of boundary layers of various size, and to the effect of the momentum ratio M. As the waves become nonlinear, ligaments reminiscent of Fig. 1.2 are seen.

A simulation of a diesel-fuel jet discharging in high-pressure still air at relatively higher velocity is shown in Fig. 9.19. In this case there is again no boundary layer and there is no separator plate either, so the Reynolds and Weber numbers are based on the injector diameter D. The parameters are $\mathrm{Re_g} = \mathrm{Re_l} = 1.2 \times 10^4$, $r = m = 3.59 \times 10^{-2}$, $\mathrm{We_g} = 823$, and $\mathrm{We_l} = 2.2 \times 10^4$. The smallest cell is such that $\Delta x/D = 2.3 \times 10^{-2}$. The Reynolds and Weber numbers are thus considerably higher than in the previous case, albeit not as high as in the two-dimensional diesel-jet case of Fig. 9.15. As in the latter case, considerable amounts of ligaments and droplets are produced. It is possible to distinguish sheets, especially near the tip. These sheets are broken by holes, producing numerous fish-bone-like patterns and ligaments. The ligaments then break into droplets. These types of simulation are both more realistic than the Leboissetier simulation of Fig. 9.15 because they are three-dimensional, and also better resolved because of the lower Reynolds and Weber numbers. The Gerris code used in the three-dimensional case also has lower numerical viscosity. These types of simulation are getting close to what is required for the ultimate goal of atomization simulations: predicting the droplet size. The results of simulations such as the one in Fig. 9.19 may be analyzed to produce probability distribution functions (PDFs) of droplet size. It is generally found that these distributions agree qualitatively with experimentally determined PDFs (Fuster *et al.*, 2009b). However, when a simulation is repeated with a finer grid, the PDF is shifted towards smaller droplet sizes: using a smaller Δx results in smaller droplets, as can be



(a)

(b) (c)

Fig. 9.19. The spatial development of a high-speed atomizing jet simulated using Gerris. Parameters are given in the text. (a) A view of the jet development near dimensionless time $Ut/D \sim 6$. (b) A zoom of the top-right region, showing how the sheet near the tip of the jet breaks into filaments and droplets. (c) A zoom of the bottom-right region, showing droplets and ligaments on the side of the jet. Simulation obtained by Anne Bagué and Stéphane Popinet.

expected when simulations are somewhat underresolved. However, this is a problem on the verge of being solved, with a combination of faster computers, better codes, and a clever use of adaptive mesh refinement.

10

Droplet collision, impact, and splashing

Droplet collisions and impacts are so spectacular that they have come to symbolize the beauty and fascination of fluid mechanics. Although simulations of two-dimensional and axisymmetric systems go back to the early times of two-phase flow simulation, those of fully three-dimensional configurations have become possible only recently. It remains difficult, however, to perform realistic simulations of laboratory experiments.

10.1 Introduction

Droplet impacts are of major industrial interest. In what is perhaps the most significant application, fuel droplets impact on the walls of pipes and combustion chambers. There they may spread and form thin films or shatter into a spray of smaller droplets. Impacts also have an obvious relevance to ink-jet printing and spray coating. In other industrial processes, droplet impacts are of interest in metallurgy (Liow *et al.*, 1996; Bierbrauer, 1995) and gas-injection processes. High-speed droplet impacts may damage turbines operating with multiphase flows. In hypothetical severe nuclear reactor accidents, molten-core debris may impact on containment walls, splashing at very large velocity. In agriculture, impacts are related to the effect of rain on soil erosion (Farmer, 1973), or the spread of pesticides as they are sprayed on plants. Rain also influences air–sea interactions, enhancing the gas exchange and perhaps damping sea waves (Sainsbury and Cheeseman, 1950; Tsimplis and Thorpe, 1989). Droplet breakup, atomization, impacts, and splashes also cause the accumulation of charge in droplets, as shown by the 1905 Nobel physics laureate Philip Lenard following the work of Hertz (Lenard, 1892). Meteorite and planetary impacts, as well as very high-speed impacts of metals, e.g. bullets, may be described as fluid-mechanical phenomena because the energy of the impact is such that inertia is much larger than the plastic and elastic stresses in the materials (see for instance O'Keefe and Ahrens (1986)). Droplet splashing is thus a model of impacts on planetary bodies, resulting for instance in moon

craters. The central peak in moon craters is probably a remnant of the so-called Worthington jet in droplet splashes that we describe below.

At another extreme in spatial scales, droplet collisions are a model for the collision of atom nuclei (Menchaca-Rocha *et al.*, 2000).

A remarkable early work on droplet impact, including spectacular photographs of impacting fluid droplets or solid bodies, was performed by Worthington (1908). It was Edgerton (1987), however, who took high-speed photographs that acquired extraordinary fame. Since then, a considerable literature on collisions and impacts has accumulated. Simultaneously with the engineering and natural science studies, a lot of work on splashes has been motivated by the need for graphics, since the early high-speed movies shot in the 1930s in preparation for the Disney animation *Fantasia* (Michel Ledoux, personal communication) to current work on computer graphics (Foster and Fedkiw, 2001; Bargteil *et al.*, 2006).

In the remainder of this chapter we focus on numerical work and cite theoretical and experimental work when relevant. Further information on experimental work may be found in the reviews by Rein (1993) and Yarin (2006).

10.2 Early simulations

The first simulations of droplet collision, impact, and splashing were performed by Harlow and Shannon (1967), who used the original MAC method, i.e. a staggered grid with bulk marker particles, and ignored viscosity and surface tension. Foote (1975) also used the MAC method, but included both surface tension and viscosity. His results for the evolution of rebounding droplets, at low Reynolds and Weber numbers, were compared with experimental observations by Bradley and Stow (1978), who found good agreement, but made the controversial observation that "this complicated treatment gives little insight into the physical processes involved." Obviously, just as in experimental measurements, the amount of insight gained should depend on the ability of the physicist to extract it from their simulation tools. Another approach has been pioneered by Fukai *et al.* (1995), who used a moving finite-element method and examined the effects of wetting. Foote's results were apparently overlooked by Fukai *et al.*, who concluded, after a review of previous investigations, that fixed-mesh techniques like the MAC method are not well suited for this problem! Foote's results and those presented here do not support that conclusion.

10.3 Low-velocity impacts and collisions

The outcomes of droplet–droplet impacts and droplets impacting on walls and solid films are often similar, so we shall treat them together in this section. At relatively low speed the outcomes of a head-on collision of two droplets can be: (a) a bounce

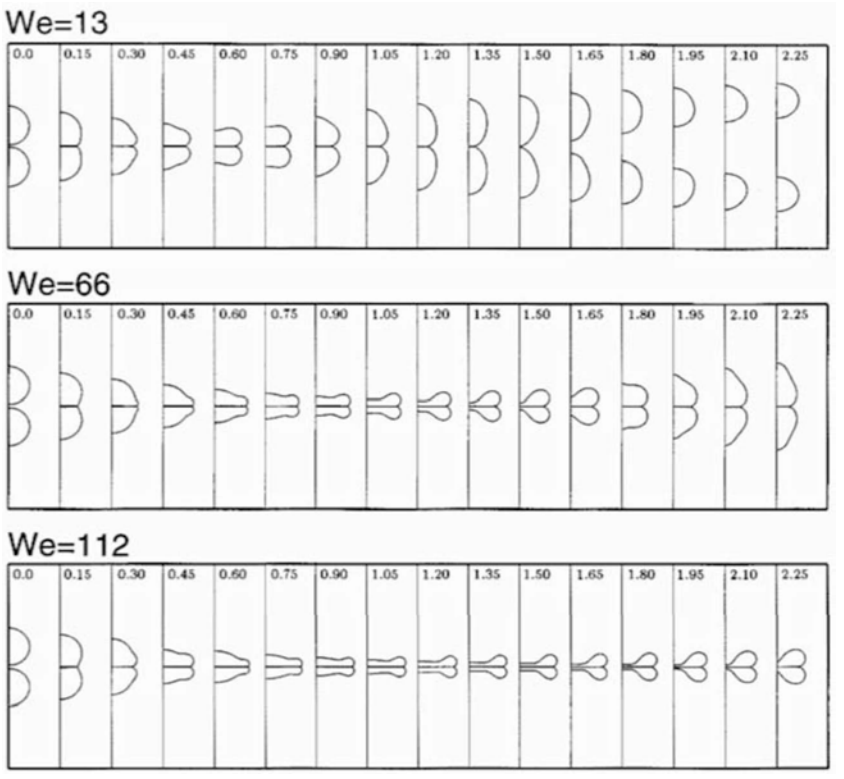


Fig. 10.1. Simulation of a head-on droplet–droplet collision. Parameters are given in the text. The nondimensional time based on initial velocity and droplet diameter is noted in each frame. Reprinted with permission from Nobari *et al.* (1996). Copyright 1996, American Institute of Physics.

without any coalescence, (b) merging of the droplets, (c) a partial merging followed by the separation of the droplets, and (d) shattering or breakup in more than two droplets, which occurs at the highest velocities. Head-on collisions of two droplets of the same size are not too different from the collision of one droplet with a flat wall if free-slip boundary conditions are assumed at the wall and wetting effects are ignored. As a matter of fact, in the impact of a droplet on dry walls, similar regimes are found: (a) bounce back, (b) deposition, i.e. spreading of the droplet on the wall, and (c) breakup or splash together with the production of a spray of smaller droplets. Furthermore, a number of these regimes are also observed in impacts on wet walls and deep fluid pools. However, when the thickness of the film is large compared with the radius of the droplet, a deep crater is formed, and in some cases a bubble is entrained as the crater closes. This bubble formation is called “regular entrainment,” following Pumphrey and Elmore (1990). Regular entrainment is an intermediate regime: if the droplet velocity is either too small or too large, the

bubble does not form, as discussed in Prosperetti and Oğuz (1993). The closing of the crater is often accompanied by thin, rapidly rising vertical jets (Morton *et al.*, 2000). When the velocity of impact is increased beyond the regular entrainment regime, a thick jet (the "Worthington jet") forms from which one or several droplets pinch off. The height of the jet and the number of detaching droplets increases with the velocity of impact.

For impact on deep pools, many authors have studied the formation of vortex rings. In their early work, Thomson and Newall (1885) found the paradoxical result that vortex rings form at relatively low impact velocities, but that at higher velocity the impact leads to a splash, as there is relatively little penetration of the vorticity.

The relevant parameters for these various regimes are the Reynolds, Weber, and Ohnesorge numbers:

$$\mathrm{We} = \frac{\rho U^2 D}{\sigma}, \quad \mathrm{Re} = \frac{\rho U D}{\mu}, \quad \mathrm{Oh} = \frac{\mathrm{We}^{1/2}}{\mathrm{Re}} = \frac{\mu}{(\sigma \rho D)^{1/2}}, \tag{10.1}$$

where D is the droplet diameter and U the impacting droplet velocity. In this chapter we use ρ and μ for the properties of the impacting liquid droplet and ρ_a and μ_a for the properties of the ambient gas or carrier fluid. For ordinary laboratory impacts, gravity may be neglected compared with inertia, surface tension, and viscosity. However, for the study of regular bubble entrainment and crater evolution, gravity is relevant and is measured by the Froude number:

$$\mathrm{Fr} = \frac{U^2}{gD}. \tag{10.2}$$

Other numbers are sometimes needed, such as the dimensionless roughness of the surface, wettability characteristics, diameter-to-film-height ratio.

Computations of head-on collisions of pairs of droplets were carried out by Nobari *et al.* (1996) using the front-tracking method to follow the interface and the method of Section 7.2 to compute surface tension, which ensures that the net capillary force on a drop is zero. Results are shown on Fig. 10.1, in which $\mathrm{Re} = 96$, $\rho/\rho_\mathrm{a} = 15$, $\mu/\mu_\mathrm{a} = 350$, and different Weber numbers are used. Computations were done on a uniform grid with 64×256 grid points in the radial and axial directions, respectively. The simulations show a characteristic jet, or "impact foot," spreading sidewise as the droplets are pressed against each other. It is seen that as the Weber number increases this jet becomes increasingly thinner. As the colliding or impacting droplets flatten, they eventually reach a maximum radius R_max, then recede and often bounce back. Many scalings have been proposed in the literature for the maximum radial extent. For example, if a simple balance is made between the initial kinetic energy, $K = \pi \rho U^2 D^3/12$, and the final surface energy of the "pancake," $E_S = 2\pi\sigma R_\mathrm{max}^2$, then the maximum radius scales as $R_\mathrm{max} \sim D\mathrm{We}^{1/2}$.

Other scalings have been proposed in the literature by Roisman *et al.* (2002) and Chandra and Avedisian (1991). They are often contradictory and only partially correlated to the experimental results, as the data are either too scattered or obtained over insufficient ranges of magnitude of the Reynolds and Weber numbers. This state of affairs demonstrates one of the uses of direct numerical simulations: analyzing the precise mechanism leading to a given scaling is easier when the output data of a large series of simulations can be represented and scaled to attempt to verify a given scaling theory.

Numerical simulations may thus help us to resolve the issue of the scaling of the maximum radius after impact. However, it must be noted that scaling theories are derived at asymptotically large values of Re and We and, thus, in regimes where droplet shattering, crown formation, and splashing are observed, which is the topic of Section 10.5.2.

10.4 More complex slow impacts

At ambient air pressure, droplets collision always leads at least to partial coalescence (higher air pressure may trigger a rebound). The thin air film separating the two droplets, or the droplet and the liquid pool, ruptures and the two droplets form one connected liquid mass. The expulsion of air from the thin air film could consistently be modeled only if the simulation is refined until the grid size is smaller than the length scale of the relevant microscopic physics, which is somewhere between 1 and 10 nm.

This represents a costly simulation. Present-day simulations on workstations using adaptive grid refinement reach the 100 μm scale for typical droplets in three-dimensional simulations (Nikolopoulos *et al.*, 2007) and the 10 μm scale in two-dimensional simulations (see Fig. 10.8 below). To reach 10 nm would require a 10^3 to 10^4 decrease in the grid spacing h. This is a tall order even if adaptive mesh refinement and massive parallelism are used.

Researchers have instead used one of two strategies. In front-tracking methods, the interface is reconnected at a prescribed time that seems physically reasonable. This was done for instance in the simulation of the three-dimensional impact shown in Fig. 10.2. In VOF and level-set methods instead, reconnection occurs automatically when interfaces are closer than one or two grid spacings. (Using a smoothed surface-tension method such as the one of Section 7.1.4 may increase this range.) Despite the crudeness of these procedures, realistic results are often obtained.

An example is off-axis droplet impacts. These are defined by a nonzero impact parameter I, defined as follows. The droplets move on parallel axes before collision. If the distance between the axes is χ, then $I = \chi/D$. For nonzero I the four collision outcomes described in Section 10.3 all occur, together with an additional

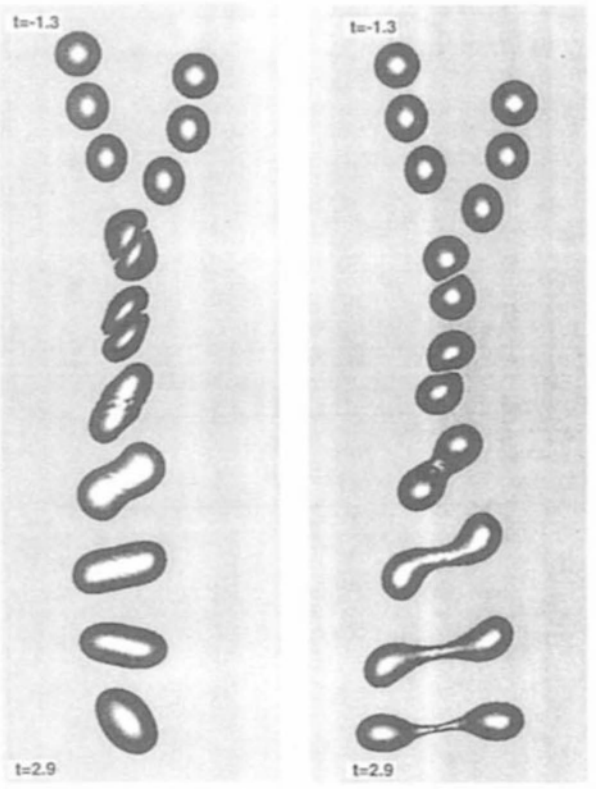


Fig. 10.2. Simulations of three-dimensional droplet collision using the front-tracking method. The initial conditions are shown on the top, and the results are then shown every 0.42 dimensionless time units. Time $t = 0$ is the time at which the initial spheres would touch if unperturbed, so the time of the first snapshot on top of the series of pictures is negative. The thin air film between the droplets is ruptured at $t = 0.46$ in both cases, but the impact parameter is different in the two runs. On the left $I = 0.5$, while on the right $I = 0.88$. The droplets stay coalesced in the first case, but they will separate again in the second case. Reprinted from Nobari and Tryggvason (1996) by permission of the American Institute of Aeronautics and Astronautics, Inc.

one: grazing collisions in which the droplets temporarily coalesce and then stretch to form an elongated object that may break again in two or more droplets. An example of such a collision is shown in Fig. 10.2.

Another type of more complex impact is obtained when droplets impact on deep pools. An example of a simulation of such an impact is shown in Fig. 10.3. As the droplet touches the liquid surface, it creates a crater visible on the first frame of the sequence, at time $t = 8.6$.

After the initial formation of the crater, capillary waves progress on the sides of the crater until they meet on the axis, around time $t = 10.1$. At that time, just before

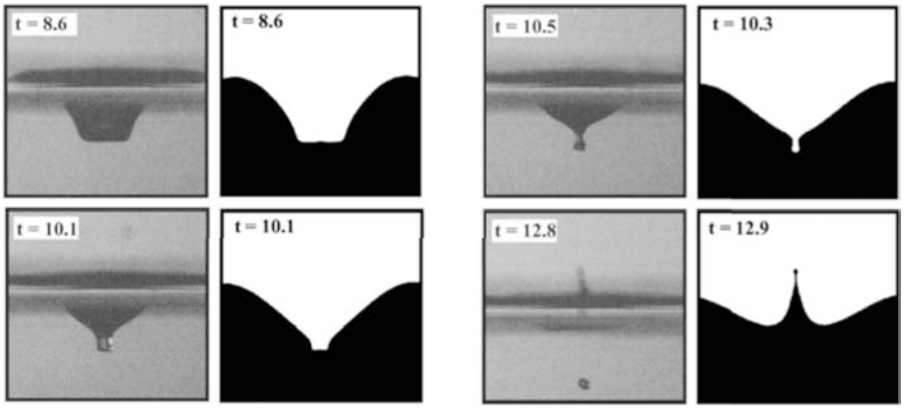


Fig. 10.3. Simulations of droplet impact on a deep pool compared with experiments. Reprinted with permission from Morton *et al.* (2000). Copyright 2000, American Institute of Physics.

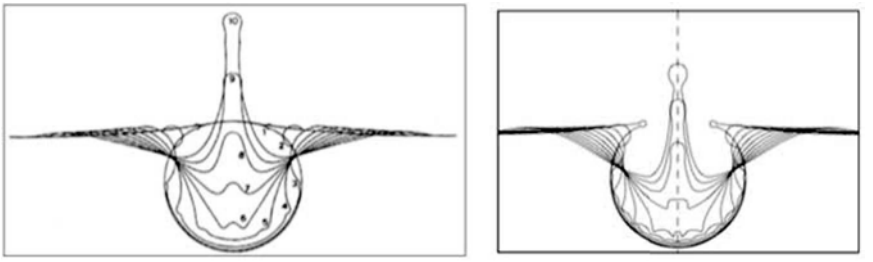


Fig. 10.4. Bursting of a bubble at a free surface. (a) Sketches by MacIntyre (1972) done after experimental photographs. Copyright 1972, American Geophysical Union. (b) Direct numerical simulation. Reprinted with permission from Duchemin *et al.* (2002). Copyright 2002, American Institute of Physics.

the trapping of the bubble, the crater assumes a remarkable conical shape. It is interesting to notice that this form is reminiscent of several other fluid phenomena were crater-like interfaces form. One of them is the crater formed after the break of a bubble near the water surface, shown in Fig. 10.4. The bottom of the crater exhibits the kind of capillary self-similarity in shape discussed in Section 9.2.2. The self-similar crater appears just at the transition between a regime where a bubble forms below the crater and one in which it does not form at all. This feature is similar to both bubble break and impact craters, as well as to a third form of crater, the crater formed when a layer of liquid is submitted to an oscillating vertical motion, generating the so-called Faraday waves. When the waves reach a critical amplitude

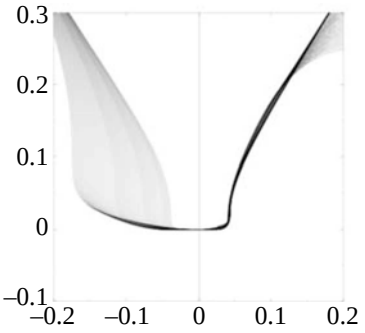


Fig. 10.5. The formation of a self-similar conical crater. For a well-defined bubble shape, a bubble bursting at a free surface will show a self-similar crater. Reprinted with permission from Duchemin *et al.* (2002). Copyright 2002, American Institute of Physics.

a self-similar crater forms again (Zeff *et al.*, 2000). The self-similarity of the crater is made evident when distances (both vertical and horizontal) are scaled by the length $(\sigma/\rho)^{1/3}(t-t_0)^{2/3}$, as shown in Fig. 10.5.

10.5 Corolla, crowns, and splashing impacts

When the impact velocity is high enough, droplet impact yields two characteristic phenomena. The liquid rises around the impact point through a circular, approximately vertical and expanding sheet, called a "corolla." In some cases the corolla sprouts fingers at its tip and becomes a beautiful crown or "corona," as in Fig. 1.3. Finally, droplets may be ejected from the crown, characterizing a shattering or splashing impact. Splashing impacts on deep pools, thin films, and dry surfaces are rather different, but share a few common characteristics. In this section we shall focus on splashing impacts on thin films, which have been the most studied impacts numerically.

10.5.1 Impacts on thin liquid layers

Figure 10.6 shows the initial conditions for the simulations described in this section using the VOF method and the CSF surface-tension scheme. A typical axisymmetric simulation is shown in Fig. 10.7. The dimensionless parameters are $\mathrm{Re} = 1000$, $\mathrm{We} = 8000$, $\rho/\rho_a = 500$, and $\mu/\mu_a = 20$. They correspond to a droplet of diameter $D = 6\,\mathrm{mm}$, with a realistic surface tension coefficient $\sigma = 0.065\,\mathrm{N/m}$, density $\rho = 10^3\,\mathrm{kg/m^3}$, and viscosity $\mu = 57\,\mathrm{cP}$ (obtained for instance by using a mixture of glycerine and water), impacting on a thin liquid layer, $H/D = 0.15$, with a velocity $U = 8.95\,\mathrm{m/s}$. The ambient fluid properties, however, are not realistic, as a larger gas density and viscosity have been taken to stabilize the calculation. The

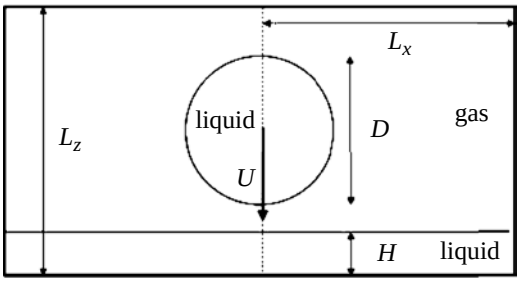


Fig. 10.6. The initial conditions for a splash on a thin liquid layer. This setup is used for simulations of axisymmetric and fully three-dimensional systems.

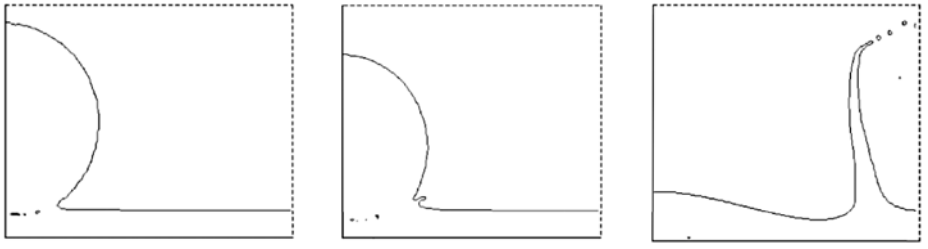


Fig. 10.7. Impact of a droplet on a thin liquid layer, at low resolution $D/\Delta x = 102$. Dimensionless times are: (a) $t = 0.05$, (b) $t = 0.1375$, (c) $t = 1.225$.

relatively large Weber number was selected by Josserand and Zaleski (2003) in order to investigate the effect of viscosity while keeping the effect of surface tension rather moderate.

The results for a resolution, $D/\Delta x$, of 102 grid points per diameter are shown in Fig. 10.7. The droplet is launched slightly above the liquid layer. Time is made dimensionless by $t = t'U/D$ and time $t = 0$ is the time at which the droplet would touch the layer if both were to stay undeformed. As is usual in VOF calculations, reconnection occurs quickly. A few air bubbles are trapped below the impacting droplet. Only some time after the impact does a jet form, as shown in Fig. 10.7(b). At an even later time a vertical corolla forms; see Fig. 10.7(c). A few droplets detach from the tip of the corolla. As in the case of the thin sheets discussed in Section 9.4.1, such topology changes occur because of the relatively low resolution.

Indeed, simulations at higher resolution show a completely different picture. Figs. 10.8(a) and 10.8(b) show the results at the same time as in the previously described calculation. It is seen that jet formation occurs much earlier than at moderate resolution. Moreover, a detailed analysis of the simulation shows that

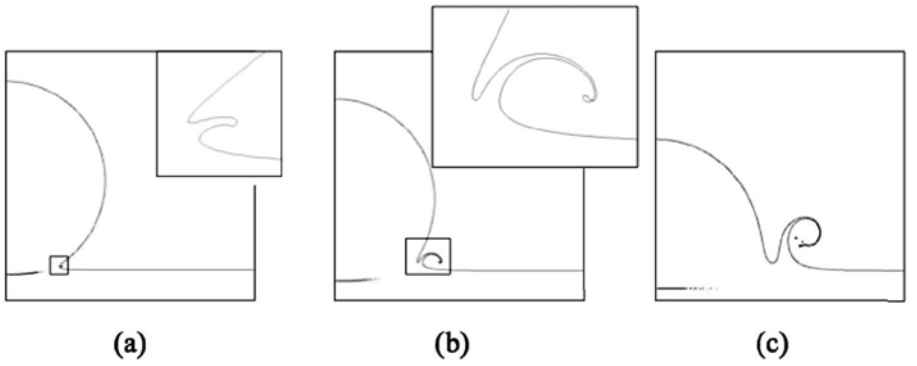


Fig. 10.8. Impact of a droplet on a thin liquid layer at high resolution $D/\Delta x = 804$. Dimensionless times are: (a) $t = 0.05$, (b) $t = 0.1375$, (c) $t = 0.3475$.

Fig. 10.9. Impact of a droplet on a thin layer at lower Re $= 100$. Dimensionless times are $t = 0.7$ and $t = 1.5$. Resolution is $D/\Delta x = 102$, as in Fig. 10.7. Reprinted with permission from Josserand and Zaleski (2003). Copyright 2003, American Institute of Physics.

jet formation is preceded by a phase in which a thin gas layer survives between the impact droplet and the liquid layer. Part of this gas layer can still be seen in Fig. 10.8(a). The shape of that gas layer is similar to that of the droplet boundary obtained analytically in the potential flow theory of Purvis and Smith (2005). As the jet forms, it has a thickness of about two grid points, which suggests that at these Reynolds and Weber numbers even thinner jets may form even earlier in even higher resolution computations.

However, as liquid viscosity is increased, jets are seen to form later and to have rounder tips, as in Fig. 10.9 for Re $= 100$. Eventually, for high-enough viscosity, no jet is seen, as in the Re $= 40$ case of Fig. 10.10. A scaling theory was proposed by Josserand and Zaleski (2003) that assumes a matching between an inner viscous region and an outer region of potential flow, as shown in Fig. 10.11. The theory locates the base of the jet in the vicinity of the point of intersection of a supposedly undeformed droplet shape with the undeformed-layer surface, at a distance $r_{\mathrm{J}}(t') \sim \sqrt{DUt'}$ from the axis of symmetry. This agrees well with experimental observations of the position of the base of the jet or corolla (Yarin, 2006). The

Fig. 10.10. Impact of a droplet on a thin layer at yet lower Re $= 40$. Dimensionless times are $t = 0.5$ and $t = 1.5$. Resolution as in Fig. 10.9. Reprinted with permission from Josserand and Zaleski (2003). Copyright 2003, American Institute of Physics.

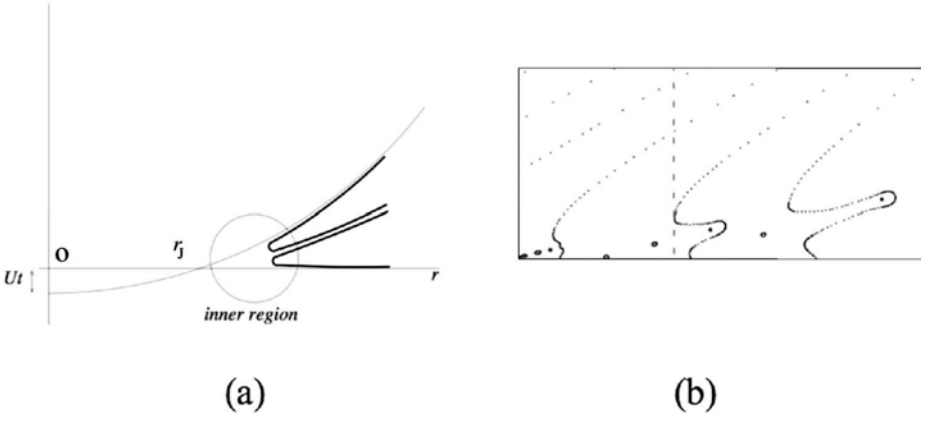


Fig. 10.11. Jet formation or prompt splash at early impact times. Vorticity is generated in the inner region. (a) Far from the jet base the drop surface and the layer surface are almost undeformed, as shown in this schematic drawing. The point at distance r_J is the imaginary intersection of the layer surface and the droplet surface at time t, assuming that both are underformed. The width of the inner region, the thickness of the jet, and the distance r_J all scale like $\sqrt{t}$ in the theory. (b) An actual simulation of jet formation. The picture is a small detail of the simulation shown in Fig. 10.13b. Notice that the picture has more dots in the highly curved regions of the interfaces. This is because one dot is generated in each cell, and cells are refined when curvature or vorticity is high.

inner viscous region is assumed to grow by diffusion and thus has size $\sqrt{\nu t'}$. This is also assumed to be the size of the jet when it forms. The theory of Josserand and Zaleski (2003) then predicts the velocity of the jet. The details of the theory are relatively lengthy, but the final result is that it predicts that a corolla will form when

$$\mathrm{We}^{1/2}\mathrm{Re}^{1/4} > K_c\,, \tag{10.3}$$

where K_c is a dimensionless number. The K_c number for the transition depends on the H/D ratio. We note that Rioboo *et al.* (2003) observe a transition from

Fig. 10.12. Detail of a spiralling corolla after impact of a glycerine droplet (Re ~ 1900, We ~ 2200, $\rho/\rho_a \sim 840$, $\mu/\mu_a = 940$) at dimensionless time $t = 0.343$. Reprinted with permission from Thoroddsen (2002).

deposition to corolla formation at $\mathrm{We}^{1/2}\mathrm{Re}^{1/4} \simeq 40$ for $H/D > 0.08$. The correlation (10.3) was previously stated, in the case of impacts on dry surfaces, by Stow and Hadfield (1981) in the form of the dimensional condition $UD^{3/5} > K_S$, which yields (10.3) when dimensionless numbers are used.

These results compare qualitatively with the experiments by Thoroddsen (2002), who also sees very thin corollas emerging from the droplet base just after impact. As time evolves, spiralling corollas may also develop. An example is shown in Fig. 10.12 for a droplet of diameter $D = 6.5\,\mathrm{mm}$, with surface tension coefficient $\sigma = 0.065\,\mathrm{N/m}$, viscosity $\mu = 16\,\mathrm{cP} = 16 \times 10^{-3}\,\mathrm{kg/(m\,s)}$ impacting on a thin liquid layer with a velocity $U = 4.65\,\mathrm{m/s}$. The photograph is at $t' = 0.480\,\mu\mathrm{s}$, which corresponds to a dimensionless time $t = 0.343$, which is close to the value $t = 0.347$ of Fig. 10.8(c). A simulation in conditions even more similar to those of Fig. 10.12 is shown in Fig. 10.13. As in our computations, the droplet is more viscous than water. In the case of water-droplet impacts, the corolla also forms very early, but it is immediately deformed into a complex spray (Thoroddsen, 2002). This raises the issue of destabilization of the axisymmetric corolla into three-dimensional structures, which we discuss in the next section.

10.5.2 Three-dimensional impacts

Three-dimensional impacts can be simulated using the same techniques and the same set up as for the two-dimensional case in the previous section. This poses no special difficulty except that the required computational power increases greatly.

As in the axisymmetric case, a jet first forms at the base of the droplet. The jet is immediately destabilized into a non-axisymmetric structure. To analyze its nature

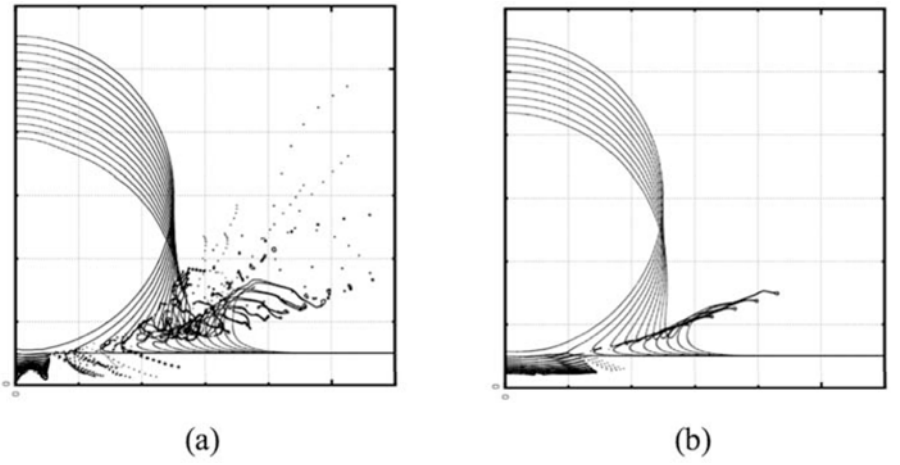


Fig. 10.13. Two simulations of jet formation in conditions similar to those of the Thoroddsen (2002) experiment. The oct-tree, VOF, Gerris code has been used. Parameters are $D = 6.3\,\mathrm{mm}$, $U = 4.65\,\mathrm{m/s}$, $\rho/\rho_a = 775$, $\sigma = 0.072\,\mathrm{N/m}$, which corresponds to $\mathrm{We} = 1892$. In the most refined cells, $D/\Delta x = 6144$. The thickness of the underlying liquid layer, $D/10$, is smaller than in the experiment. In (a) $\mu/\mu_a = 54$ and $\mathrm{Re} = 29\,295$ and in (b) $\mu/\mu_a = 1729$ and $\mathrm{Re} = 915$. Figure courtesy of Pascal Ray, CNRS.

we excite the instability by adding a small perturbation to the droplet surface. In spherical coordinates the surface of the perturbed spheroid at the initial time is

$$r(\theta,\phi) = D/2 + r_1 \sin(\theta)\cos(N\phi). \tag{10.4}$$

In the simulation of Fig. 10.14 the initial wavenumber of the perturbation is $N = 4$ and its initial amplitude is $r_1/D = 5\times10^{-3}$, which is barely visible in the image of the droplet at the given resolution.

At early times the perturbation reverses itself in the jetting corolla. This reversal is schematically explained in Fig. 10.15. As a result, the distance of the tip of the corolla to the vertical axis of the splash expands as

$$r_{\mathrm{tip}}(\phi) = r_0 - At\cos(N\phi), \tag{10.5}$$

where r_0 is the average distance to the axis.

As time evolves, nonlinear effects cause the fingers to crowd together and merge, so the wavelength between fingers increases and the number of fingers decreases. In addition, irregularities in the numerical approximation cause the separation of unphysical tiny droplets, as seen in Fig. 10.14 at the dimensionless time $t = 1.406$.

These fingers and the droplets appear at relatively early times compared with the dimensionless time at which the splash crown is seen in Fig. 1.3. This kind of splash is thus called a “prompt” splash. It is likely that the prompt splash arises from a perturbation amplified by the sudden rearrangement of the fluid velocity on

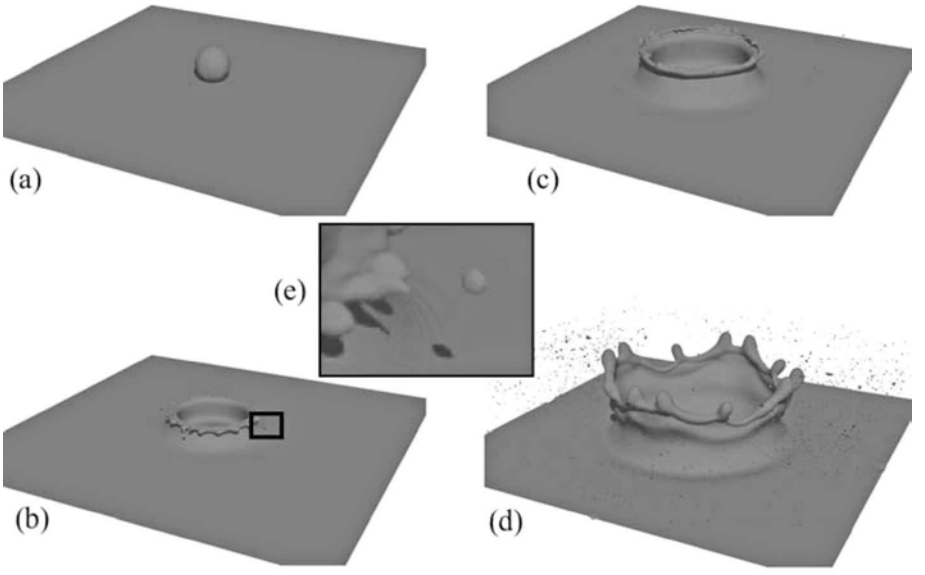


Fig. 10.14. Fall of a glycerine droplet on a thin water layer. Parameters are Re $= 600$, We $= 443$, $\rho/\rho_a = 1000$, $\mu/\mu_a = 94.11$, $D/\Delta x = 26$, and $H/D = 0.31$. Only one-quarter of the droplet is computed and the rest is reconstructed by symmetry. The whole computational domain has $n_z = 128$ points in the vertical direction ($n_z = L_z/h$, h is the constant grid spacing) and $n_x = n_y = 256$ points in the two horizontal directions ($n_x = L_x/h$). Results are shown at dimensionless times: (a) $t = 0$, (b) $t = 1.406$, (c) $t = 2.812$, (d) $t = 8.438$; (e) is a zoom of jet and droplet formation in frame (b).

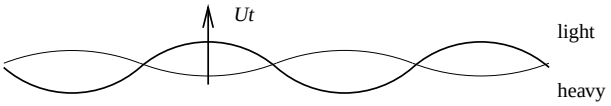


Fig. 10.15. When an interface undergoes a sudden acceleration that is oriented from the heavy to the light fluid, the so-called Richtmyer–Meshkov instability occurs and small wavy perturbations of the interface height reverse themselves and then grow. Another characteristic of the Richtmyer–Meshkov instability is that perturbations grow linearly in time, instead of the exponential growth observed, for instance, in the Rayleigh–Taylor instability.

impact, in a manner similar to the Richtmyer–Meshkov instability (Gueyffier and Zaleski, 1998). The Richtmyer–Meshkov instability is the phenomenon by which a perturbation of a flat interface is amplified when the interface is suddenly accelerated in a direction pointing from the heavy phase into the light phase (Richtmyer, 1960). The reversal of the waves on the interface and the linear dependence on time in (10.5) is characteristic of this instability.

The splashing events appearing at later times, when the corolla has risen much higher than the initial droplet diameter, are called “corona” or crown splashes. This

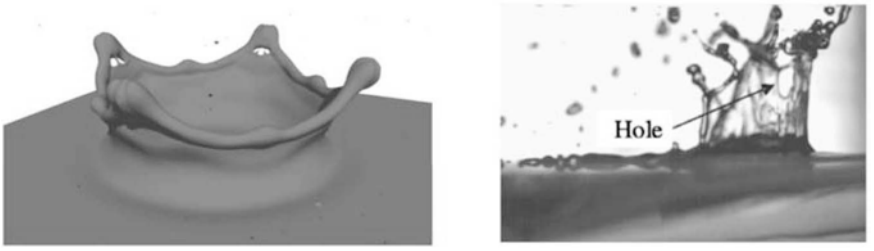


Fig. 10.16. Left: Impact of a droplet in conditions similar to Fig. 10.14 except $r_1/D = 5 \times 10^{-3}$, at $t = 9.842$ obtained by the authors using the SURFER code. Right: Experimental photograph from an oblique impact. Reprinted with permission from Roisman *et al.* (2006). Copyright 2006, American Institute of Physics.

is the situation obtained in the VOF simulation of Rieber and Frohn (1999) shown in Fig. 1.9. However, there is a catch in this type of crown splash simulation: the corolla may become too thin to maintain its integrity and breaks just before the stage shown in Fig. 1.9. The break often occurs right behind the rim; and as shown in Section 9.2.3, this is where low-viscosity rim necks are the thinnest. A cylinder detaches from the rim as in Fig. 10.7c. It is then quite naturally subject to the Plateau–Rayleigh jet instability. An example of a simulation with a hole behind the rim is shown on the left picture in Fig. 10.16. Can a real corolla break in this fashion? Recent experimental photographs have indeed shown corollas with holes, as for instance on the right image in Fig. 10.16.

In the absence of hole formation in the corolla, the mechanism for finger formation at the tip of the corolla at the late stage of crown formation may be related to the rim-to-finger mechanism discussed in Section 9.2.4. There are thus two stages in the destabilization of the corolla: at early stages the Richtmyer–Meshkov instability dominates, while at late stages various rim instabilities may take over. For impacts at high Weber and Reynolds numbers the two stages are separated by orders of magnitude of length and time scales. (Fig. 10.14d is at a nondimensional time of order 10, while we observed the jet in Fig. 10.8 to form at $t \simeq 0.03$.) It is thus not clear what influence the initial Richtmyer–Meshkov instability may have on the late-stage rim instability leading to crown formation. It is not even clear that the low-viscosity (e.g. water) splashes result from an *instability*, i.e. the growth of an initially small perturbation of a well-defined structure. Rather than an instability development, one may have a complex structure made of fingers and droplets. The objects in the structure would continuously change scale between the early and late stages. The ability to see this picture in simulations depends on the level of grid refinement, allowing us to see ever smaller structures, and the numerical dissipation of the code, filtering unresolved small-scale phenomena.

11

Extensions

Direct numerical simulations of multiphase flows are a rapidly growing field. In addition to continuous development of new and better numerical techniques and more extensive studies of the problems discussed so far in this Book, researchers are increasingly looking at new and more complex physical problems. In this chapter we examine briefly a few such extensions. We do not attempt to give an exhaustive list of all new applications of methods based on the "one-fluid" formulation of the fluid equations, but we hope that this introduction to the literature will be useful for our readers.

11.1 Additional fields and surface physics

Broadly speaking, new physics consists of new field equations, new surface effects, or both. Adding a new field is the simplest extension, but new fields often add new time- and length-scales. Mass transfer in liquids, for example, can lead to boundary layers that are much thinner than those resulting from either heat transfer or fluid motion. Resolving these boundary layers may introduce much more stringent resolution requirements than those necessary for the same problem in the absence of mass transfer.

The simplest new physics is probably heat transfer, where an advection/diffusion equation is solved for the temperature, and many authors have already studied multifluid problems involving heat transfer. The main complication is that large variations in the thermal conductivity across an interface can require fine grids and often it is better to use the harmonic mean of the conductivities at grid points where it is not defined, in the same way as for large differences in the viscosity (see Section 3.4). Given the relative simplicity of adding heat transfer, here we will focus on other effects.

11.1.1 Thermocapillary motion

The simplest extension of a "pure" heat transfer multifluid problem is to consider temperature-dependent surface tension. Surface tension usually decreases with increasing temperature and the nonuniform surface tension at the interface causes stresses that are transmitted to the fluids by viscous forces. Thus, bubbles and drops immersed in a liquid with a temperature gradient will move toward the hot region. Although often negligible under terrestrial conditions, thermocapillary forces can dominate in space, where buoyancy is absent (or greatly reduced). For material processing in microgravity, thermocapillary effects can be used to remove gas bubbles in melts before solidification, for example, and thermocapillary migration can also be important in the design of two-phase heat exchangers for space applications, since bubbles collecting on a heated surface can act as an insulator. For background information about thermocapillary motion of bubbles and drops, including experimental and analytical studies, see, for example, Subramanian and Balasubramaniam (2001).

The extension of methods based on the "one-fluid" formulation to handle variable surface tension is straightforward. The temperature is interpolated to the interface and surface tension is set as a given function of the temperature. (Nas and Tryggvason, 2003; Nas *et al.*, 2006) used a front-tracking method to simulate the motion of both a single drop and several interacting ones in both two- and three-dimensional domains. Similar computations have been done by Wang *et al.* (2004) and Gao *et al.* (2008). For computations of thermocapillary motion using the VOF method, see Bassano and Castagnolo (2003), for example.

11.1.2 Electrohydrodynamics

An electric field applied to a multifluid system, where an interface separates immiscible fluids, results in a force on the interface, due both to the migration of free charges to the interface and the dielectric mismatch between the fluids. For perfect dielectrics and conductors the force acts perpendicular to the surface, but in the general case the force will also have a tangential component, inducing fluid motion through viscous stresses. For general reviews of the electrohydrodynamics of multiphase flows, see Melcher and Taylor (1969), Arp *et al.* (1980), and Saville (1997).

The effect of an electric field is easily added to the methods described in this book. By using the leaky dielectric approximation of Taylor (1966) we must solve one Poisson equation for the electric potential and then find the force on the interface using the divergence of the Maxwell stress tensor. A front-tracking method was developed by Fernandez *et al.* (2005), who used it to examine the behavior of a suspension of many two-dimensional drops in a channel, in the presence of flow.

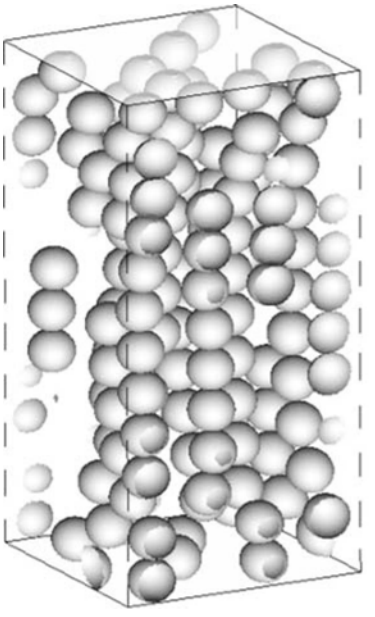


Fig. 11.1. One frame from a simulation of the effect of an electric field on a suspension of drops in a channel. The domain is a channel bounded by rigid top and bottom walls and periodic in the horizontal directions. Here, the drops are less conductive than the suspending fluid and the shear is relatively weak. Reprinted with permission from Fernandez (2008b). Copyright 2008, American Institute of Physics.

This study was subsequently extended in Fernandez (2008a,b) to cover a larger range of governing parameters and three-dimensional drops. Figure 11.1 shows one frame from a simulation of the effect of an electric field on a suspension of drops, taken from Fernandez (2008b). For the parameters used here the drops form long chains that span the height of the channel, as positive charges on one end of a drop attract negative charges on the other end of another drop. Hua *et al.* (2008) also used a front-tracking method to examine the effect of electric fields on the deformation of a drop. A combined VOF/level-set method was developed by Tomar *et al.* (2007), who emphasized the importance of using the harmonic mean when interpolating the electric properties for large differences in these values. These methods have been used to examine the effects of an electric field on film boiling by Tomar *et al.* (2005) and Welch and Biswas (2007).

We note that electric fields can also change the value of the surface tension at the interface between electrolytes (Davies and Rideal, 1966). Simulations of such flows would be nearly identical to those where surface tension depends on the temperature and will not be discussed here.

11.1.3 Mass transfer and chemical reactions

For many practical multiphase flow applications, the ultimate interest is in chemical reactions facilitated by the flow. The purpose of atomizing a liquid fuel is to burn it, and bubble columns are first and foremost chemical reactors. Thus, a full understanding of these processes requires us to solve not just for the fluid flow, but also for mass transfer and chemical reactions. Extensions of the methods described in this book to do so have just started. Simulations of mass transfer only can be found in Davidson and Rudman (2002), Yang and Maoa (2005), and Wanga *et al.* (2008), for example.

Reactions in gas–liquid and liquid–liquid multiphase systems can take place inside either phase or at the interface, may require suspended catalytic particles, and the products may form another phase. Several processes are usually involved, including mass transfer through the surface, adsorption and/or desorption to and from the surface, diffusion of species in the bulk phases, and penetration of reactants into catalytic particles (Ranade, 2001). Often, however, such as in the Fischer–Tropsch production of synthetic fuel and the catalytic hydrogenation of nitroarenes in bubble columns, the desorption and the penetration of the reactants into catalytic particles is fast and the diffusion of the gases in the liquid is the rate-limiting step. In those cases, particularly when the reactions take place close to the bubbles and, thus, depend sensitively on the mixing there, we would expect DNS to be particularly useful in providing insights into the overall process.

Khinast *et al.* (2003) showed that for stationary bubbles the results of chemical reactions are strongly dependent on the details of the flow in their wakes. For large and deformable bubbles with a stationary wake, for example, undesirable intermediate species can be trapped for a long time. This study was extended to many freely moving bubbles by Koynov *et al.* (2005), who showed that the sensitivity observed for a single stationary bubble carries over to flows with many freely moving bubbles but that the exact dependency on the governing parameters is even more complex. An examination of a specific system, the catalytic hydrogenation of nitroarenes described by Radl *et al.* (2008), further confirms the sensitivity of the overall process to the details of the flow. In this study the chemistry was taken to consist of two overall reactions with effective, measured, reaction rates. Other parameters were also taken to be representative of reasonable operating conditions. Since the mass transfer is slow and the reactions are fast, compared with the time scales of the flow, a finer grid was used for the mass transfer and the reactions (than for the fluid flow) and the study was limited to a two-dimensional system. The results showed that there was a significant difference between the selectivity computed numerically and simple film theories, when there was a prominent wake behind each bubble. The main challenge in the hydrogenation of nitroarenes is

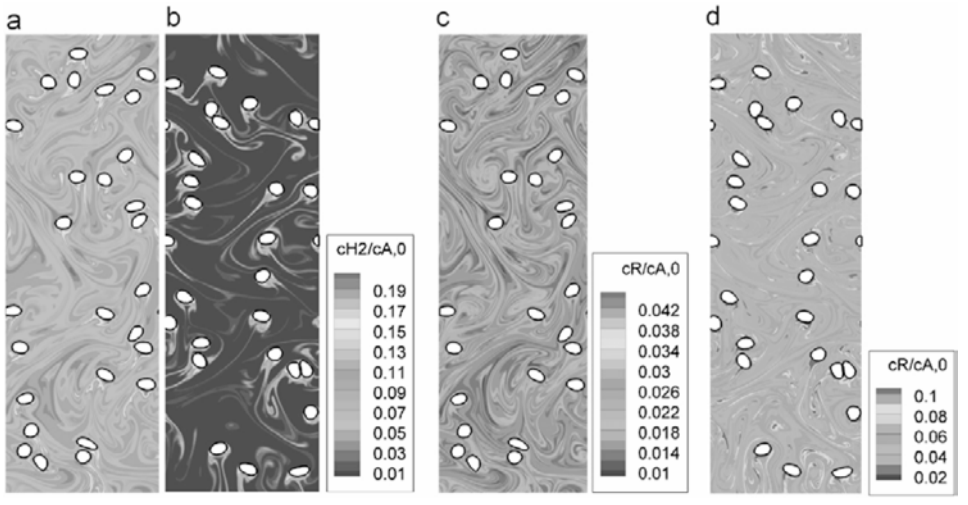


Fig. 11.2. Results from simulations of the catalytic hydrogenation of nitroarenes. The hydrogen (frames a and b) and hydroxylamine (frames c and d) concentration profiles are shown for one time for two simulations. In frames (a) and (c) the reaction rates are relatively slow, compared with the mass transfer, but in frames (b) and (d) the reaction is relatively fast. Reprinted from Radl *et al.* (2008), with permission from Elsevier. See plate section for color version.

that hydroxylamines form quickly but the final hydrogenation to arylamine is slow. Thus, there is a possibility of undesirable accumulation of hydroxylamines. The detailed process is very sensitive to the mixing caused by the bubbles and it is, in particular, important to mix the hydroxylamines with the liquid so they can react with dissolved hydrogen. A closed wake behind the bubbles is particularly undesirable, since hydroxylamines that form there have no hydrogen to react with. Figure 11.2, from Radl *et al.* (2008), shows snapshots of hydrogen and hydroxylamine concentration profiles around large bubble clusters for two different ratios of reactions rates to mass transfer. For relatively slow reactions (a), there is a significant amount of unreacted hydrogen present in the bulk phase, but for faster reactions (b), the dissolved hydrogen gas is consumed near the bubbles and the bulk concentration of hydrogen is nearly zero. Similarly, the bulk concentrations of the intermediate species (hydroxylamine) is much smaller in the case of faster reactions (compare frame c with a). Other investigations of reactive bubble systems can be found in Bothe *et al.* (2003), where a VOF method is used.

In some cases mass transfer can have a profound effect on the flow, even in the absence of chemical reactions. Early experiments on the rise of a bubble showed that the measured rise velocity was in better agreement with analytical solutions for the motion of a solid than the Hadamard (1911) and Rybczynski (1911) solution for a bubble. It is now well understood that the discrepancy is due to impurities in

the fluids, usually called "surface-active agents," "contaminants," or "surfactants" that make the bubble surface immobile. The observed behavior of many surfactants is well described by the surface-tension-gradient model proposed by Frumkin and Levich (1947), where surface tension depends on the concentration of surfactants and uneven concentration will lead to surface forces. Surface tension is generally reduced by an increase in the surfactant concentration, resulting in forces that move the fluid in such a way that the surfactant distribution becomes more uniform. Forces due to the fluid motion can, however, counter the effect of the surface tension, sometimes leading to a nonuniform steady-state surfactant distribution. Generally, we assume that the surface tension depends on the surfactant concentration and that the relationship is a given, experimentally measured, function. In the simplest case the surfactant is immiscible and stays on the surface. It is then advected by the surface material points, and if surface diffusion is neglected the total surfactant on a given surface element is a constant. Thus, changes in the surfactant concentration are inversely proportional to surface area; see Jan (1994), James and Lowengrub (2004), Drumright-Clarke and Renardy (2004), and Lee and Pozrikidis (2006). In the general case, however, surfactant is absorbed and desorbed at the interface and diffused into the liquid phase, posing a much more challenging mass transfer problem. Initial efforts to solve the fully coupled case can be found in Zhang *et al.* (2006), Muradoglu and Tryggvason (2008), and Tasoglu *et al.* (2008).

11.1.4 Boiling

So far, we have considered only incompressible flows of two or more immiscible fluids. For many important multiphase systems it is also necessary to consider phase changes. Boiling, in particular, is one of the most efficient ways of removing heat from solid surfaces and, therefore, is commonly used in energy generation and the cooling of hot surfaces. The large volume change and the high temperatures involved can make the consequences of design or operational errors catastrophic, and accurate predictions are highly desirable.

Boiling is usually initiated at a hot surface, although vapor bubbles sometimes also start at suspended nuclei away from surfaces (homogeneous boiling). Boiling at a hot surface is usually referred to as nucleate boiling when vapor bubbles are formed at well-defined nucleation sites (small crevices that are slightly hotter than the rest of the surface) and the vapor bubbles break away from the surface as they form. At higher heat-transfer rates the bubbles coalesce, covering the hot surface with a vapor film, and the process is referred to as film boiling. Vapor bubbles usually break off from the layer, but vaporization at the liquid–vapor interface replenishes the layer. Generally, film boiling is undesirable, since the vapor layer acts as an insulator, lowering the heat-transfer rate and increasing the temperature of the

hot surface. Fluid flow can have a significant influence on the vapor formation and the heat transfer rate. If the liquid is initially stagnant, we usually talk about pool boiling, but if the liquid is flowing, the process is called flow or forced-convection boiling.

Although boiling involves a liquid turning into vapor as heat is added, boiling can also take place when the system pressure is lowered. However, this situation is usually treated separately as cavitation, as discussed below. Condensation involves the same physical processes as boiling, except that vapor is turned into liquid as heat is removed. Although it is often assumed in modeling that the fluid consists of one pure material, both boiling and condensation can be modified when the liquid consists of a mixtures of fluids with different boiling points and by the presence of noncondensable gases in the vapor phase. When the pressure of the vapor in the gaseous stage is only a small fraction of the total pressure, so that the local expansion of the liquid as it changes phase is dynamically insignificant, we generally talk about evaporation rather then boiling. For introductions to boiling, see Dhir (1998) and Carey (2007).

Early computations of boiling relied on a number of simplifications, such as those of Lee and Nydahl (1989), who assumed that a vapor bubble remained hemispherical as it grew. More advanced computations started with Welch (1995), who simulated a fully deformable, two-dimensional bubble using moving triangular grids. He was, however, only able to follow the bubble for a relatively short time due to the distortion of the grids. Son and Dhir (1997) used a moving body-fitted coordinate system to simulate film boiling for both two-dimensional and axisymmetric flows, but were subject to similar limitations as Welch. The limitation to modest deformation of the phase boundary was overcome by Juric and Tryggvason (1998), who developed a front-tracking method for two-dimensional problems, and by Son and Dhir (1998) who used a level-set method to follow the phase boundary in axisymmetric geometries. Several authors have now developed methods suitable for film boiling and other situations that do not involve a nucleation site. Those include the front-tracking method of Esmaeeli and Tryggvason (2004a), the VOF methods of Welch and Wilson (2000) and Agarwal *et al.* (2004), the level-set method presented by Gibou *et al.* (2007), and the combined level-set/VOF method developed by Tomar *et al.* (2005). Several studies of film boiling have been done using these methods (e.g. Esmaeeli and Tryggvason, 2004b,c; Tomar *et al.*, 2008, 2009). For nucleate boiling, where the challenges are greater, progress has been slower, but a few authors have already reported such studies (Son *et al.*, 1999, 2002; Shin *et al.*, 2005b; Mukherjee and Dhir, 2004; Mukherjee and Kandlikar, 2007).

To simulate boiling flows we need to extend the approach presented in the earlier chapters of this book. We need to solve the energy equation, in addition to the momentum and mass conservation equation; we need to specify the temperature at

the phase boundary; we need to compute the rate at which liquid is converted into vapor and vice versa; and we need to account for the volume change accompanying the phase change.

In general, the energy equation must be solved to find the temperature distribution in the vapor and the liquid, although it is sometimes possible to ignore the vapor by assuming that it is at the saturation temperature (Son and Dhir, 1998). The energy equation can be written in the "one-field" form, in the same way as the momentum equation, by adding a source term $\dot{q}$ to account for the release of latent heat at the phase boundary:

$$\frac{\partial}{\partial t}(\rho cT) + \nabla \cdot \rho cT\mathbf{u} = \nabla \cdot k\nabla T - \dot{q}\delta(n). \tag{11.1}$$

The energy equation can be solved directly as written, smoothing the source term onto the fixed grid (Juric and Tryggvason, 1998; Esmaeeli and Tryggvason, 2004b), or it can be solved by imposing the interface temperature explicitly by modifying the numerical approximation near the interface. For large differences between the densities of the vapor and the liquid, the latter approach seems preferable.

The interface temperature must be specified. For a large number of situations of interest it is reasonable to assume thermodynamic equilibrium, where the temperature is continuous across the phase boundary. Since length scales resulting from the flow are considerably larger than those resulting from the thermodynamic conditions, it is usually also a good approximation to assume that the temperature at the phase boundary be equal to the saturation temperature at system pressure.

The heat source in Equation (11.1) is given by

$$\dot{q} = k\frac{\partial T}{\partial n}\Big|_{\mathrm{v}} - k\frac{\partial T}{\partial n}\Big|_{\mathrm{l}}, \tag{11.2}$$

where the subscript v stands for the vapor and l denotes the liquid, so the temperature gradients at the phase boundary must be estimated. That is easily done using the "normal probe" technique introduced by Udaykumar *et al.* (1999). In this approach we draw a line segment normal to the interface and interpolate the temperature at the end of the segment (usually about one and a half grid spacings or so away from the interface). This temperature, along with the given interface temperature, is then used to find the temperature gradient and the heat source. Once the heat source is found, the rate at which liquid is converted into vapor is given by $\dot{m} = \dot{q}/h_{\mathrm{lv}}$, where h_{lv} is the latent heat.

The conversion of liquid to vapor or vice versa is generally accompanied by a significant change in volume. We assume that the density in each phase is constant and the only volume expansion takes place at the interface. To find the divergence of the total velocity field at the phase boundary, we write $\mathbf{u} = \mathbf{u}_{\mathrm{v}}H + \mathbf{u}_{\mathrm{l}}(1-H)$,

where the velocity in each phase is assumed to have a smooth incompressible extension into the other phase. Taking the divergence, and using $\nabla \cdot \mathbf{u}_\mathrm{v} = \nabla \cdot \mathbf{u}_\mathrm{l} = 0$, yields

$$\nabla \cdot \mathbf{u} = (u_\mathrm{v} - u_\mathrm{l})\delta(n), \tag{11.3}$$

where $u_\mathrm{v} = \mathbf{u}_\mathrm{v} \cdot \mathbf{n}$ and $u_\mathrm{l} = \mathbf{u}_\mathrm{l} \cdot \mathbf{n}$ are the normal velocities on either side of the interface. To find the jump in the normal velocity across the interface, we use the Rankine–Hugoniot condition (2.46):

$$\rho_\mathrm{v}(u_\mathrm{v} - V) = \rho_\mathrm{l}(u_\mathrm{l} - V) = \dot{m}. \tag{11.4}$$

The volume expansion per unit interface area is found by eliminating V:

$$u_\mathrm{v} - u_\mathrm{l} = \dot{m}\Big(\frac{1}{\rho_\mathrm{v}} - \frac{1}{\rho_\mathrm{l}}\Big). \tag{11.5}$$

Inserting the expression for the velocity difference across the phase boundary and the heat source into Equation (11.3) results in

$$\nabla \cdot \mathbf{u} = \dot{m}\Big(\frac{1}{\rho_\mathrm{v}} - \frac{1}{\rho_\mathrm{l}}\Big)\delta(n). \tag{11.6}$$

The normal velocity V of the phase boundary is found to be

$$V = \frac{1}{2}(u_\mathrm{v} + u_\mathrm{l}) - \frac{\dot{m}}{2}\left(\frac{1}{\rho_\mathrm{v}} + \frac{1}{\rho_\mathrm{l}}\right). \tag{11.7}$$

Thus, this velocity is the sum of the average fluid velocity (the first term) and the relative motion of the interface as the fluid changes phase.

Most authors have neglected density variations in the vapor and the liquid, which could lead to buoyancy-driven currents. These are, however, easily taken into account by the Boussinesq approximation. Thermocapillary effects are usually also neglected, but as long as the interface temperature can be assumed to be constant, this is likely to be a good approximation. In any case, thermocapillary effects are easily included. In the energy equation, viscous dissipation and interface stretching terms are usually small compared with the latent heat and are neglected.

Direct numerical simulations of boiling flows are still very new and only a few situations have been examined in any details. We will show two examples here. Figure 11.3 shows three frames from a simulation of film boiling in a two-dimensional domain. An initially quiescent liquid pool rests on a hot, horizontal surface and the heat flux rate is sufficiently high so that the wall is blanketed by a thin vapor film. As the liquid evaporates, the liquid–vapor interface becomes unstable and bubbles are periodically released from the layer. The bubbles rise to the surface of the pool, break through the surface, and release the vapor. Initially, both the vapor and the liquid are at saturation temperature, and the bottom wall is kept

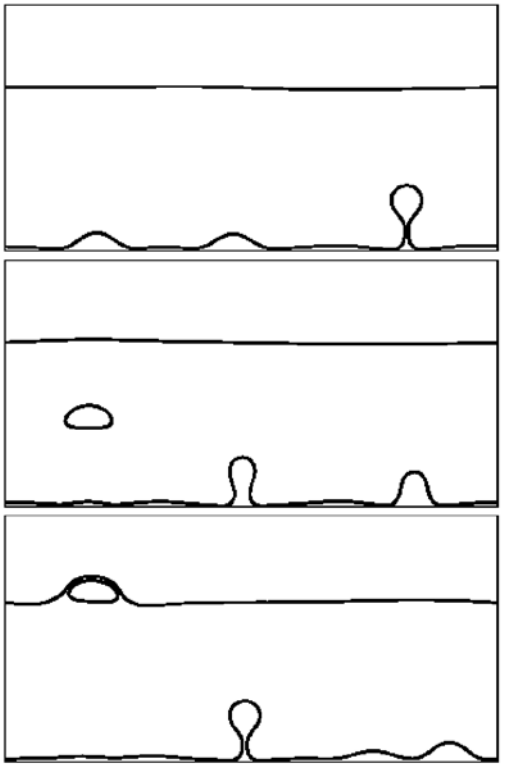


Fig. 11.3. Simulation of two-dimensional film boiling. A vapor layer blankets a hot wall and as the liquid evaporates, the liquid–vapor interface becomes unstable and bubbles are periodically released from the layer. The two phase boundaries are shown. Figure courtesy of Professor A. Esmaeeli.

at a constant temperature which is higher than the saturation temperature. The domain is periodic in the horizontal direction, it has a no-slip/no-through-flow wall at the bottom, and an open boundary at the top. The domain is resolved by a 64×160 grid, but the horizontal grid lines are unevenly spaced to resolve accurately the thin vapor film. For details, see Esmaeeli and Tryggvason (2004b,c).

The wall heat-transfer rate is usually given in terms of the Nusselt number, $\mathrm{Nu} = \lambda/(T_\mathrm{w} - T_\mathrm{sat})\partial T/\partial y|_{y=0}$, and in Fig. 11.4 the time-average Nusselt number is plotted versus the Jacob number (the nondimensional wall superheat), as computed by several simulations of film boiling. In addition, the predictions of two correlations of experimental results (Berenson, 1961; Klimenko, 1981) are included. Although the correlations are based on fully three-dimensional flows, the results overall are reasonably good. Welch and Wilson (2000) found a similar trend when they compared their two-dimensional results with Berenson's correlation.

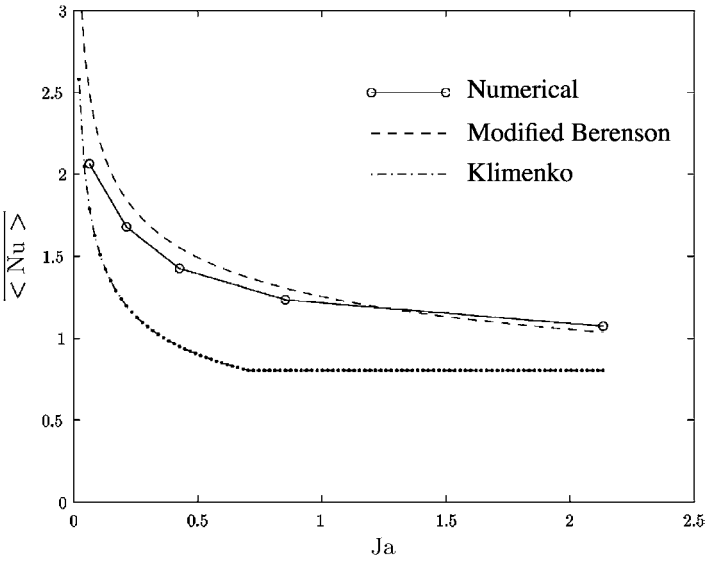


Fig. 11.4. The Nusselt number Nu versus nondimensional wall superheat Jacob number Ja from simulations of two-dimensional film boiling. Predictions of two experimental correlations (Berenson, 1961; Klimenko, 1981) are also shown. Reprinted from Esmaeeli and Tryggvason (2004c), with permission from Elsevier.

While film boiling seems to be reasonably well under control, nucleate boiling is of considerably more interest. A few authors have started to address nucleate boiling, as discussed above, but some development appears to be still needed before such simulations become routine. Several new issues must be addressed. First of all, the location of the nucleation site(s) must be specified. For simulations of the formation of a single bubble at a specific site (or a few bubbles) this is generally straightforward (experimentally this is achieved by generating an artificial site), but for real materials the nucleation site distribution is generally dependent on the surface finish, and sites become active and inactive depending on the superheat. Second, the formation of a vapor bubble usually affects the temperature distribution in the solid and, at least in principle, it is necessary to solve the energy equation there. For film boiling, on the other hand, specifying the surface temperature or the heat flux is generally a reasonable approximation. So far, most simulations of nucleate boiling have been done ignoring the variability of the solid temperature, although it should not be difficult to include it. The third major issue is that the initial bubble is very small and it is difficult, even with adaptive grid refinement, to capture the initial stage. Thus, placing a finite-size bubble at the nucleation site, particularly when simulating the repeated release of bubbles from a specific site, is generally approximate at best. The fourth major issue

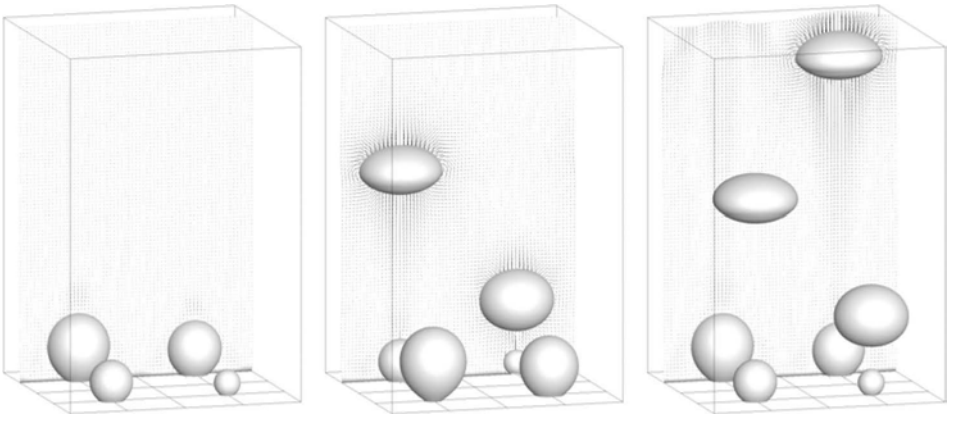


Fig. 11.5. Three frames from a simulation of nucleate boiling from four nucleation sites, computed on a $64 \times 64 \times 96$ uniform grid. The liquid is saturated water at atmospheric pressure. Figure courtesy of Dr. J. Lu.

is the inclusion of the so-called microlayer, left behind on the wall as the bubble expands and the apparent contact line moves away from the nucleation site. The evaporation of the microlayer is generally believed to contribute to the initial growth rate, but exactly how much as a function of the governing parameters is not completely understood. Since the microlayer is much thinner than any other length scale in the computations, Son *et al.* (1999) developed a thin-film model of the microlayer that they used as a subgrid model in their study of the nucleation bubble.

Figure 11.5 shows three frames from a simulation of the generation of vapor bubbles during the boiling of saturated water at atmospheric pressure. The bubbles are released from four nucleation sites and, as they leave, small bubbles are inserted at the nucleation sites. In this particular simulation no microlayer model was used and the resolution is relatively modest. Nevertheless, it suggests that direct numerical simulations of relatively complex boiling systems are poised to contribute in major ways to our understanding of boiling and our ability to predict the behavior of such systems.

11.1.5 Cavitation

Although cavitation can be treated as a special case of boiling, it is often sufficient to use a simpler model where the pressure in a cavitation bubble is taken to be either equal to the vapor pressure at system temperature or given by the ideal gas law. Early computations of the collapse of a vapor bubble were done by Chapman

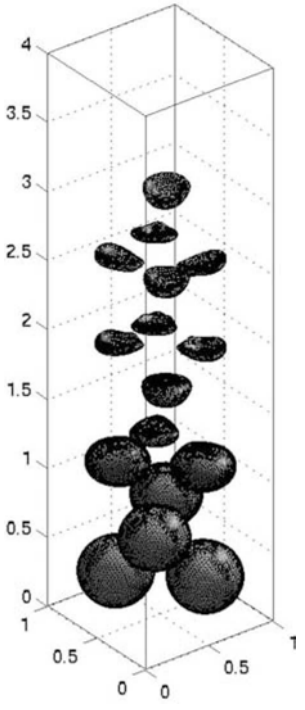


Fig. 11.6. One frame from a simulation of the propagation of a pressure wave into a domain initially containing several bubbles. The domain is open at the top, has a rigid bottom, and is periodic in the horizontal directions. Reprinted with permission from Delale *et al.* (2005). Copyright 2005, American Institute of Physics.

and Plesset (1972) and Mitchell and Hammitt (1973), using the MAC method. More recent studies include Yu *et al.* (1995), who examined the effect of fluid shear on the collapse of a vapor bubble near a wall, Popinet and Zaleski (2002), who examined the effect of viscosity on the collapse of a single bubble, and the simulation of shock propagation in a liquid containing many bubbles by Delale *et al.* (2005). Figure 11.6 shows one frame from a simulation of the propagation of a pressure wave into a fluid region initially containing several bubbles. The pressure inside the bubbles is equal to the vapor pressure and initially the pressure in the liquid is the same. At time zero the pressure at the top is increased, causing the bubbles there to collapse. As the bubbles near the top collapse, the pressure increases there, leading to a collapse of the bubbles deeper in the domain, and so on. Since the collapse takes a finite time and bubbles do not start to collapse until the bubbles above them have collapsed, the pressure wave travels downward with a finite speed. The speed and the overall shock behavior agree well with theoretical predictions. See Delale *et al.* (2005) for details.

11.2 Imbedded boundaries

It is perhaps a little bit of a stretch to classify flows over stationary solid objects as multiphase flows. However, in a development that has paralleled the progress made for fluid interfaces, the use of fixed regular meshes to handle complex solid boundaries has seen a significant comeback in recent years. In this section we will give a very brief introduction to the fundamental ideas, starting with the immersed boundary method of Peskin for elastic moving boundaries and then discussing the incorporation of complex solid boundaries.

11.2.1 The immersed boundary method of Peskin

The original immersed boundary method was designed specifically to model the motion of the heart. Peskin (1977) presented computations for a two-dimensional case and Peskin and McQueen (1989) extended the method to fully three-dimensional flows. The method is very similar to the front-tracking method described in Chapters 3 and 6, with the exception that the fluid properties (density and viscosity) are constant everywhere and the imbedded elastic solids (used to represent the heart walls) consist of bundles of fibers modeled as one-dimensional strings of marker points, in both two and three dimensions. Since they undergo only finite stretching, no redistribution of the marker points is necessary. The representation of the fibers, therefore, is considerably simpler than a front representing a fluid interface, which is discretized using unstructured meshes that must be updated as the interface deforms. Peskin and collaborators spent considerable effort on the accurate modeling of the fiber structure of the heart, but most of the early computations were done using a first-order time integration on collocated grids. A version using a higher order time integration method was presented by Lai and Peskin (2000). Peskin's work has stimulated a number of other simulations of biological systems using either the immersed boundary method directly or similar computational approaches. A review of some of those applications can be found in Peskin (2002), where the method is also discussed in detail. An example of the use of the immersed boundary method can be found in Fauci and Dillon (2006), where the biofluidmechanics of reproduction is reviewed. Other applications of the method are also listed there.

The original immersed boundary method predates much of the development of "one-fluid" methods for multiphase flows. While many of the difficulties encountered with multifluid and multiphase flows are absent when the fluid is homogeneous and the deformation of the immersed boundary is small, the basic "philosophy" adopted there has had a major impact on current work. Indeed, it is probably fair to rank Peskin's immersed-boundary method second only to the MAC and the VOF method in its impact on current "one-field" methods.

11.2.2 Solid boundaries

Approximating complex solid boundaries on rectangular structured grids by blocking out some cells was introduced at the beginning of numerical simulations of fluid flows. Rectangular boundaries aligned with the grid lines were obviously done first (Harlow and Welch, 1965), but techniques to include curved boundaries soon emerged. Viecelli (1969), for example, approximated curved boundaries by the closest cell edges and applied a pressure to the boundary so that the normal velocity component had the correct value. Since resolution was generally low, and since this approach offers only moderate opportunities to control the local resolution, using a rectangular grid with embedded boundaries quickly gave way to more involved approaches, such as body-fitted grids and unstructured grids, where it is possible to align a grid line with the boundary and to control the local resolution. However, just as regular structured grids eventually emerged as the method of choice for fluid interfaces, it has become clear that such grids offer many advantages for complex geometries. The primary reason is the simplicity by which very complex boundaries can be incorporated and the efficiency of numerical schemes based on regular structured grids.

Consider a fluid domain containing one or more solid bodies of arbitrary shape. If we use a regular grid to discretize the full domain, some of the grid points will fall in the fluid region and others will be inside the solid. To identify the solid and the fluid regions, we construct a marker function I such that

$$I(\mathbf{x}) = \begin{cases} 1 & \text{inside the fluid,} \\ 0 & \text{inside the solid.} \end{cases} \tag{11.8}$$

The velocity field, everywhere in the computational domain, can now be written as

$$\mathbf{u}(\mathbf{x}) = I(\mathbf{x})\mathbf{u}_{\mathrm{f}}(\mathbf{x}) + \big(1 - I(\mathbf{x})\big)\mathbf{u}_{\mathrm{S}}(\mathbf{x}), \tag{11.9}$$

where $\mathbf{u}_{\mathrm{f}}$ is the velocity in the fluid and $\mathbf{u}_{\mathrm{S}}$ is the velocity in the solid. The fluid velocity is governed by the Navier–Stokes equations for homogeneous fluids:

$$\frac{\partial \mathbf{u}_{\mathrm{f}}}{\partial t} + \nabla_h \cdot \mathbf{u}_{\mathrm{f}}\mathbf{u}_{\mathrm{f}} = -\frac{\nabla_h p}{\rho} + \nu \nabla_h^2 \mathbf{u}_{\mathrm{f}}, \tag{11.10}$$

and for a stationary body, or one moving with a prescribed velocity, $\mathbf{u}_{\mathrm{S}}$ is given.

Below, we first describe a very simple way to include a solid region in simulations based on the methods described in Chapter 3, and then we discuss the various challenges and approaches in more detail. If we solve for the flow field using a first-order, forward-in-time projection method, then the predicted fluid velocity is given by

$$\mathbf{u}_{\mathrm{f}}^{*} = \mathbf{u}^n + \Delta t \Big(-\nabla_h \cdot \mathbf{u}^n \mathbf{u}^n + \nu \nabla_h^2 \mathbf{u}^n \Big). \tag{11.11}$$

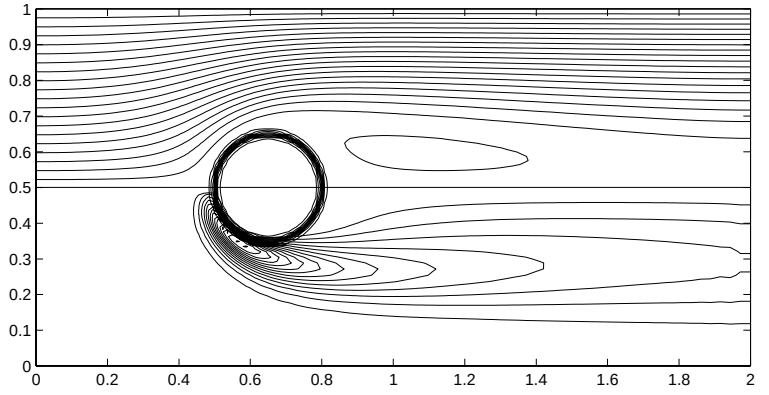


Fig. 11.7. The steady-state velocity around a circular cylinder at a Reynolds number of 60, computed on a 60×120 grid, stretched in such a way that the cylinder is resolved by about 25 grid points across its diameter. The stream function is shown in the top of the figure and the vorticity in the bottom half.

Here, we use $\mathbf{u}^n$ instead of $\mathbf{u}_{\mathrm{f}}^n$ on the right-hand side, since $\mathbf{u}^n = \mathbf{u}_{\mathrm{f}}^n$ in the fluid region. This velocity is then corrected by adding a pressure gradient:

$$\mathbf{u}_{\mathrm{f}}^{n+1} = \mathbf{u}_{\mathrm{f}}^* - \Delta t \nabla p. \tag{11.12}$$

The pressure gradient should make the final velocity,

$$\mathbf{u}^{n+1} = I\mathbf{u}_{\mathrm{f}}^{n+1} + (1-I)\mathbf{u}_{\mathrm{S}}^{n+1}, \tag{11.13}$$

divergence free, or $\nabla \cdot \mathbf{u}^{n+1} = 0$. However,

$$\begin{aligned} \nabla \cdot \mathbf{u}^{n+1} &= \nabla \cdot I\mathbf{u}_{\mathrm{f}}^{n+1} + \nabla \cdot (1-I)\mathbf{u}_{\mathrm{S}}^{n+1} \\ &= I\nabla \cdot \mathbf{u}_{\mathrm{f}}^{n+1} + I\nabla \cdot \mathbf{u}_{\mathrm{S}}^{n+1} + (\mathbf{u}_{\mathrm{f}}^{n+1} - \mathbf{u}_{\mathrm{S}}^{n+1}) \cdot \nabla I, \end{aligned} \tag{11.14}$$

where the last two terms are zero since the velocity in the solid is constant and the normal velocity is continuous at the solid surface. Thus, it is sufficient to find the pressure by considering only $\nabla \cdot \mathbf{u}_{\mathrm{f}}^{n+1} = 0$, which gives the same pressure equation as for a flow without a solid body:

$$\nabla^2 p = \frac{\Delta t}{\rho} \nabla_h \cdot \mathbf{u}^*, \tag{11.15}$$

where $\mathbf{u}^*$ is the predicted velocity in the whole domain, computed using (11.11).

This makes it particularly simple to extend a code written for fluid flow to include solids. We simply need to put the velocity inside the solids equal to $\mathbf{u}_{\mathrm{S}}^{n+1}$ at the end of the time step. Figure 11.7 shows an example of flow around a stationary cylinder computed using this approach.

Generally, I is taken to have a smooth transition from zero to one in a narrow zone around the surface of the solid. The exact shape of the transition zone can have some impact on the quality of the solution. A very thin zone leads to a jagged velocity field near the body and too thick a zone results in a loss of accuracy. Some authors have also found that using an asymmetric I yields better results (see Beckermann *et al.* (1999) in the context of a phase-field method and Al-Rawahi and Tryggvason (2002) who used front tracking).

A more formal way of including a stationary immersed solid was introduced by Goldstein *et al.* (1993), who added a force $\mathbf{f}_S$ to the Navier–Stokes equations that was adjusted in such a way that the velocity in the solid region remained zero. The force was computed by a feedback mechanism where the weighted difference between the computed velocity and the desired velocity in the solid, as well as the time integral of the difference (the displacement), was added to the force. The weights were adjusted to get as rapid convergence as possible, without making the system unduly stiff. The feedback mechanism was improved by Saiki and Biringen (1996), who used a more sophisticated control algorithm to compute the feedback. Another approach to compute the forces was suggested by Lai and Peskin (2000), who represented the surface by points that they connected to fixed points by a linear spring. The boundary points could thus move a little, but as their displacement becomes larger, so does the force pulling them back to their original position. In the language of Goldstein *et al.* (1993), the force was therefore proportional to the displacement only.

The feedback forcing is actually a roundabout way to impose the velocity in the solid. Suppose we use first-order forward differences for the time derivative. The semi-discrete form of Equation (11.10), with the force added, is then

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t\left(-\nabla_h \cdot \mathbf{u}\mathbf{u} + \nu\nabla_h^2\mathbf{u} - \frac{\nabla_h p}{\rho} - \mathbf{f}_S\right). \tag{11.16}$$

The challenge is to find $\mathbf{f}_S$ such that $\mathbf{u}^{n+1} = \mathbf{u}_S$. This is simpler than it seems. Since we already know $\mathbf{u}_S$, we are implicitly including $\mathbf{f}_S$ when we set the velocity in the solid to its desired value. There is, therefore, no need to compute the force explicitly and (after a somewhat convoluted detour) we are back to the simple method outlined at the beginning of the chapter!

The convergence rate of the straightforward immersed boundary method is at best first order. To get second-order accuracy we need to treat the velocities near the boundary more accurately. There are a few ways to do so. In the method introduced by Fadlun *et al.* (2000) the velocities at grid points in the fluid, immediately adjacent to the solid, are found by linear interpolation between the velocity at the solid surface (zero for stationary bodies) and the velocities further inside the fluid. Figure 11.8 shows this interpolation in more detail. The grid points marked by a

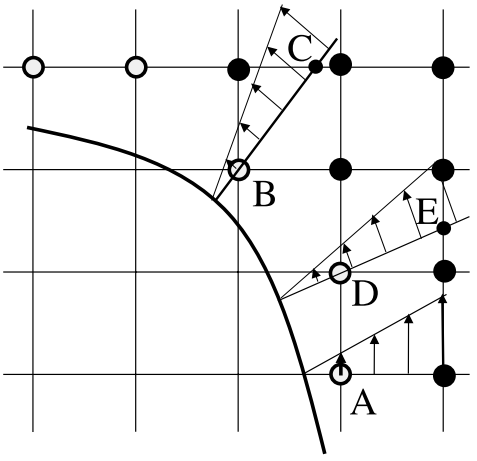


Fig. 11.8. Interpolation of the velocity for points near the solid boundary.

solid circle have all their neighbors inside the fluid, as do grid points further away from the solid. The points marked by open circles, on the other hand, have at least one neighbor inside the solid. Thus, after the velocity field has been updated by solving the Navier–Stokes equations for points marked by a solid circle, the velocities at the open circles are interpolated. The interpolation can be done in a number of ways. The simplest way, used by Fadlun *et al.* (2000), is to interpolate along each grid line (as done for point A). For boundaries that are approximately aligned with the grid this approach is straightforward, but for arbitrary boundaries the interpolation direction can be ambiguous. For point B, for example, we could just as well interpolate along the vertical grid line instead of the horizontal one. To remove this ambiguity, Gilmanov *et al.* (2003) suggested a modified procedure where we draw a line normal to the boundary, through point B, and find the velocity at the first point where this line crosses a grid line (point C). The velocity at C is found by linear interpolation between the nearest grid points and the velocity at point B is then found by linear interpolation between the velocity at C and the velocity at the boundary. Similarly, the velocity at D is found by interpolating between E and the solid surface. Minor variations of this approach are obviously possible, but as several authors have shown, determining the velocities at points near the boundary by interpolation improves the accuracy of the results.

By interpolating the velocity at the grid points next to the boundaries, we discard the Navier–Stokes equations for those points and, therefore, do not strictly conserve mass and momentum. Thus, Kim *et al.* (2001) suggested that instead of finding the velocity by interpolation between the surface velocity and the fluid velocity where it is known, we should set the velocity at points *inside* the body in such a way that

the interpolated velocity at the body surface had the correct value. We will not pursue the various improvements and variants that have been proposed to represent arbitrary solid regions on regular structured grids, but point the reader to the review by Mittal and Iaccarino (2005) for a more complete discussion.

The approach outlined above has been extended to freely moving solids by several authors. The simplest version is to use the Navier–Stokes equations with a variable density and assign the solids their correct density and a relatively high viscosity. The velocity field is updated as described in Chapter 3, except that at the end of each time step the velocity field inside each solid is updated by

$$\mathbf{u}_{\mathrm{S}}^{n+1} = \mathbf{u}_{\mathrm{Sc}}^{n+1} + \mathbf{r} \times \Omega_{\mathrm{S}}^{n+1}. \tag{11.17}$$

Here, $\mathbf{u}_{\mathrm{Sc}}$ is the centroid velocity of the solid body, Ω_{S} is the angular velocity, and $\mathbf{r} = \mathbf{x} - \mathbf{x}_{\mathrm{c}}^{n+1}$ is the distance to any point inside the solid from the centroid of the body $\mathbf{x}_{\mathrm{c}}^{n+1}$. To find $\mathbf{u}_{\mathrm{Sc}}^{n+1}$ and $\Omega_{\mathrm{S}}^{n+1}$ we can proceed in several ways. In the simplest approach we first advance the location and orientation of the solid body, using $\mathbf{u}_{\mathrm{Sc}}^{n}$ and Ω_{S}^{n}, update I, and then compute the average translational and rotational velocities inside the solid body at $n+1$ by averaging the velocity found in the first step:

$$M_{\mathrm{S}}\, \mathbf{u}_{\mathrm{Sc}}^{n+1} = \int (1-I)\rho \mathbf{u}\, \mathrm{d}v \quad \text{and} \quad \mathbf{I}_{\mathrm{S}}\, \Omega_{\mathrm{S}}^{n+1} = \int (1-I)\mathbf{r} \times \rho \mathbf{u}\, \mathrm{d}v, \tag{11.18}$$

where M_{S} is the mass of the solid and $\mathbf{I}_{\mathrm{S}}$ is the moment of inertia tensor. The accuracy of the flow near the surface can be increased by the interpolation technique described above for stationary bodies.

For freely moving solids that can collide with each other (or a solid wall), the motion is stopped as the solids come in contact and stresses build up in the body. The pressure in the fluid generally also increases near the contact point, but unlike for the collision of drops, where the pressure rise is sufficient to arrest the motion of the drops, for solids that are modeled as nondeformable bodies (and where the stress inside is usually not computed) it is necessary to add a short-range force to prevent the solids from interpenetrating each other (or a wall).

Several other possibilities exist to enforce a rigid body motion in a solid represented on a fixed grid. In their fictitious boundary method, Glowinski *et al.* (2001) enforced a zero deformation tensor by an iterative technique, and Kajishima and Takiguchi (2002) and Yabe *et al.* (2001) averaged the acceleration of the solid at time n to predict the velocity in the solid at $n+1$, for example. For elastic solids, the deformation must be computed by the appropriate strain–stress relationship; see Yabe *et al.* (2001) for a brief discussion. Methods that represent arbitrary solids on fixed Cartesian grids are often referred to as immersed boundary methods, thus extending Peskin's original terminology significantly.

11.2.3 Solidification

The solidification of liquids is important in natural processes, such as magma dynamics and lava flow, as well for the formation and dynamics of sea ice and glaciers. Similarly, solidification is of enormous importance in the processing of materials, since a large fraction of all materials used by mankind (metals, polymers, concrete, glasses, and so on) is usually in the liquid phase at some point.

When a material in its liquid state is cooled down to a sufficiently low temperature, parts of the material start to solidify. The solidification usually begins at the container boundary or around a seed of solid material present in the melt. As more heat is removed, the solid region grows, until all the melt has become solid. The process is determined by the transport of heat from the melt, which is needed to lower the temperature to the solidification temperature and to conduct latent heat released during the solidification away from the phase boundary. The phase boundary must remain at the solidification temperature and when heat is removed, latent heat is released by moving the boundary. The rate of release of latent heat, therefore, determines the velocity of the phase boundary. The challenge for numerical simulations is to accurately predict the temperature distribution in the material and the motion of the phase boundary.

In many situations the interface is unstable, and often the melt is a mixture of materials with different solidification temperatures, leading to both a complex phase boundary and a solid consisting of regions with a different composition. The prediction of how the microstructure of alloys depends on the details of how the solidification took place is one of the most challenging problems in metallurgy. It has been known for thousands of years that the various process parameters (such as heating and cooling rates and history) have a profound effect on the properties of the final product. While experience has allowed practitioners to tailor their processes for a desirable outcome, a detailed understanding is limited. Experimental investigations are usually difficult. The harsh thermal and chemical environments and limited optical access in directional solidification furnaces do, for example, make it nearly impossible to observe or measure the transient temperature and solute distribution. In addition to determining the composition of the microstructure, uneven solidification can lead to residual stresses, “freckles,” and voids.

For solidification it is necessary to solve the energy equation for the temperature and, for the solidification of a binary alloy, an equation for the solute concentration. For solidification the volume change is usually small and can often be neglected. The whole domain, containing both the melt and the solid, can therefore be taken to be incompressible. If the effect of the fluid flow can be neglected, the energy equation (and the species equation if appropriate) is sufficient to describe the dynamics. When fluid flow is included, the full Navier–Stokes equations must be

solved and the solid part included in the way described earlier in this chapter. For a pure material the solution strategy is very similar to boiling, except that we can usually ignore the volume expansion. We note that while the volume change at the interface can often by ignored, sometimes volume change in the melt can lead to convective currents. This can be included by the Boussinesq approximation, where a $\beta(T - T_o)$ term (here, β is the expansion coefficient) is added to the momentum equation.

In addition to the energy equation, we need interface conditions for the temperature. For a large number of situations of interest it is reasonable to assume thermodynamic equilibrium, where the temperature is continuous across the phase boundary. The interface temperature T_S is therefore equal to the temperature in the material on either side. The interface temperature can be found by thermodynamic considerations, leading to what is usually referred to as the Gibbs (or Gibbs–Thompson) condition. For a derivation see, for example, Alexiades and Solomon (1993). In its simplest form, the temperature is equal to the equilibrium solidification temperature T_m. Local conditions can, however, increase or decrease the interface temperature and the local Gibbs conditions are usually taken to be

$$T_S = T_m - T_m \frac{\sigma\kappa}{L} + mC_L - \frac{V}{\eta}. \tag{11.19}$$

Here, σ is the surface tension, κ the curvature, m the slope of the liquidus line, L is the latent heat, and C_L the solute concentration in the liquid next to the interface. Furthermore, V is the normal solidification velocity and η the kinetic mobility. The surface tension and the kinetic mobility are generally anisotropic, depending on the crystalline structure of the material. In principle, these need to be determined experimentally for each material, but in numerical work simple analytical relations are generally used. The microstructure resulting from an unstable interface usually depends strongly on the specific form of the anisotropy. The inclusion of other interfacial phenomena can lead to the addition of other terms.

The energy equation (along with the solute and fluid equations when needed) can be solved for both the solid and the melt using a single regular structured grid in the same way as described earlier in this book. Once the temperature distribution has been updated, the heat source at the interface and, therefore, the normal velocity can be determined. While determining the temperature next to the interface is subject to the same considerations as for boiling, it is generally found that, since the properties of the solid and the melt are more similar than for a liquid and its vapor, computing the temperature accurately is easier.

Solidification can be examined at many different scales and different questions are important at the different scales. At the largest scale, when the purpose is first and foremost to predict the cooling and solidification time of the various parts of

an object, and the details of the microstructures cannot be resolved, the enthalpy method is usually the method of choice. In its original form the enthalpy method included the transition from liquid to solid as a constant-temperature "mushy zone" where enthalpy changed gradually from the value in the solid to the value in the liquid. No attempt is made to account for the details of the microstructure, and when fluid flow is included the "mushy zone" is modeled as a porous material whose porosity changes from zero to unity as one goes from the liquid to the solid. For a background of the enthalpy method, see Alexiades and Solomon (1993); examples of computations using the enthalpy method can be found in Swaminathan and Voller (1993), Shyy and Rao (1994), and Shyy *et al.* (1996).

In some cases the solidification front is stable and remains essentially planar. In this case the simulation is relatively straightforward and no empirical input for the mushy zone is needed. Several authors have, for example, simulated in this way the rapid solidification of hot droplets impacting a cold solid surface (important for various spray and net-shape forming processes); see, for example, Pasandideh-Fard *et al.* (2002) and Che *et al.* (2004).

The most challenging problem in solidification, at least of those where it is reasonable to work with a continuum model, is the formation of microstructures. Microstructures generally arise through an instability of a planar or spherical interface where a protrusion of the interface reaches into melt that has properties that are more favorable for solidification than the melt next to the interface (it can be colder or its solute concentration can be lower). The protrusion, therefore, grows faster than the rest of the interface, leading to cellular or dendritic microstructures. For alloys, the composition of the solid that forms first is generally different from what is formed later, so the solidification history is preserved in the composition of the final part. The solidification of pure material can also lead to microstructures, where an initially regularly shaped interface undergoes an instability leading to a more complex structure. In the case of a pure material, the microstructures are, however, transient, and once the melt has solidified, all traces of the irregular interface shape are lost. Nevertheless, transient microstructures of pure material have often been studied as a proxy for the more complex alloy problem. For extensive reviews of the early literature on computations of solidification, see Wheeler *et al.* (1995), for example, as well as the comprehensive review of methods for alloy solidification and morphological stability theory by Coriell and McFadden (1993).

Numerical simulations are already helping us understand how alloys solidify and what conditions lead to specific microstructures. Early computations of the large-amplitude evolution include Ungar and Brown (1985), who used a boundary-conforming finite-element method to examine cellular solidification. Sethian and Strain (1992), using a level-set method, and Wheeler *et al.* (1993), using a phase-field method, simulated the growth of dendrites. Simulations of the growth of

dendrites in a pure material, in the absence of flow, have now become relatively routine; see, for example, Udaykumar *et al.* (1999), Kobayashi (1993), Schmidt (1996), and Karma and Rappel (1997, 1998). Methods for alloy solidification have also been developed, and the reader is referred to the literature for the current status. Recent references include the phase-field simulations by Cha *et al.* (2005), Nestler (2005), and Plapp (2007), as well as Tan and Zabaras (2007), who used a level-set method.

It is now well established experimentally that fluid flow can have a significant impact on the growth of microstructures (see Glicksman *et al.* (1986), for example), and although most simulations have focused on microstructure formation in the absence of flow, some progress has also been made for the coupled problem. Some of the first such simulations include the phase-field simulations of Tonhardt and Amberg (1998) and Beckermann *et al.* (1999) and the front-tracking computations of Shin and Juric (2000) and Al-Rawahi and Tryggvason (2002), for two-dimensional problems. Simulations of three-dimensional systems include Jeong *et al.* (2001), Boettinger *et al.* (2002), and Lu *et al.* (2005b), all of whom use phase–field methods, and the front-tracking simulations of Al-Rawahi and Tryggvason (2004).

Figure 11.9 shows one frame from a simulation of the growth of one three-dimensional dendrite from Al-Rawahi and Tryggvason (2004). In addition to the dendrite, the temperature and the velocity vectors in a plane cutting through the center of the domain are shown. The domain is a cube resolved by a 256^3 grid. A uniform inflow is specified on the left boundary, the top and bottom boundaries are periodic, and all gradients are set to zero at the outlet boundary. Initially a small spherical solid, perturbed slightly, is placed in the center of the domain. The incoming flow is slightly undercooled and, as latent heat is released at the phase boundary, the flow sweeps it from the front to the back. This results in a thin thermal boundary layer at the tip of the upstream growing arm and in a relatively uniform temperature in the wake. The growth rate of the upstream arm is therefore enhanced and the growth of the downstream arm is reduced. Growth of side branches is also suppressed on the downstream side. Although the same qualitative trend is seen in simulations of two-dimensional systems, the wake structure is significantly different in two and three dimensions. In two dimensions the fluid must flow around the tip of the dendrites growing perpendicular to the flow direction, whereas in three dimensions the fluid can flow around the side of the side branches. This results in a much larger wake for the two-dimensional dendrite. In addition to increasing the growth rate of the dendrite growing into the flow, the flow reduces the radius of the tip, in agreement with the experimental observations of Lee *et al.* (1993).

Computations of solidification are already important for the planning of casting, for example. However, such simulations generally rely on empirical

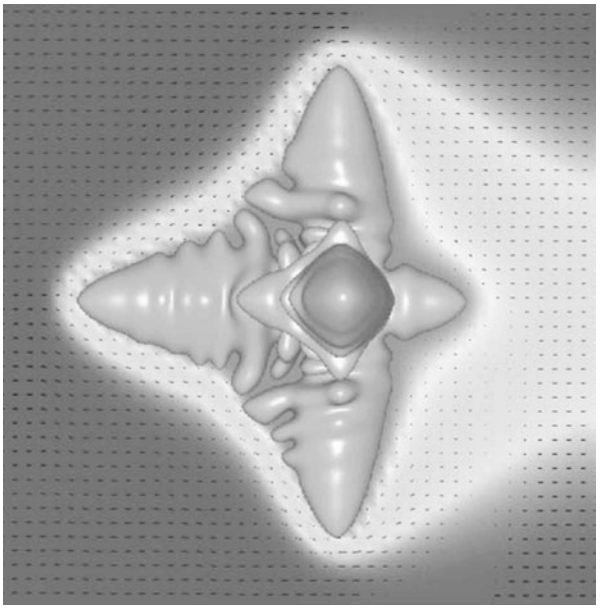


Fig. 11.9. The large-amplitude stage of a growing dendrite. In the liquid a dark area indicates a subcooled liquid. The light gray area is warmer. Reprinted from Al-Rawahi and Tryggvason (2004), with permission from Elsevier.

approximations to capture the behavior of unresolved processes and, thus, do not accurately predict the resulting microstructure. The use of numerical simulations to understand microstructure formation and how it depends on the process parameters holds the promise of revolutionizing material processing. The incorporation of such an understanding into computations of industrial systems, through appropriate subgrid models, could have significant economic implications as well.

11.3 Multiscale issues

In spite of the enormous information and understanding that DNS are providing for relatively complex flows, real systems provide challenges that still limit the range of situations that can be simulated, even when we limit our studies to systems well described by continuum theories. The problem is, as one might expect, one of scale. Generally, the smallest length scale of the flow has to be resolved by $\mathcal{O}(10)$ grid points, and as the number of available grid points increases, the range of scales that can be resolved increases. Current computer power makes it possible to simulate two-fluid systems in domains resolved by several hundred grid points in each spatial direction (for thousands of time steps) and simulations of systems resolved by over 1000^3 grid points are on the horizon. Such simulations will make

it possible to resolve length scales whose ratios span over two orders of magnitude. However, this is often not enough. Starting with simulations where what we might call the dominant small scales (see below) are fully resolved, it is frequently found that multiphase flows also can generate features much smaller than the dominant flow scales, consisting of very thin films, filaments, and drops.

The "natural" or "dominant" small scales in many multiphase systems are set by the balance of surface tension and viscosity and/or inertia. For the breakup of a jet, for example, this scale determines the average droplet size. In most cases, this scale corresponds to roughly where the appropriately chosen nondimensional numbers, such as Weber, Capillary, Ohnsorge, and Reynolds numbers, are $\mathcal{O}(1)$ (and the key word here is obviously "appropriately"). Smaller scales are, however, easily generated. Consider, for example, the relatively tame problem of the collision of two droplets that are a few hundred micrometers in diameter. As the drops collide, they deform and trap air in a thin film between them. The air drains out of the film and if the drops stay in contact long enough, the film ruptures. It is generally believed that the film thickness must get down to a few hundred angstroms before it ruptures. If the drops are 500 μm in diameter, say, the range of scales, from the thickness of the film to the diameter of the drop, is $\mathcal{O}(10^4)$. In principle, local, adaptive, grid refinement can be used to allocate the computational resources where they are needed and while doing so helps resolve small-scale features; this increases the complexity of the computations significantly and usually results in greatly increased computational time. Furthermore, adaptive grid refinement works best when the refinement is modest and often the range of scales is so large to make it impractical, or at least prohibitively expensive. For very simple situations, such as the head-on collision of two drops, it may be possible to capture this range by carefully exploiting the symmetry of the problem and allocating the resolution judiciously. However, for more complex systems, including several drops moving arbitrarily, such simulations are essentially impossible at the present time.

In many cases the structure of the small-scale features is relatively simple. Viscous effects usually dominate over inertia and surface tension effects are sufficiently strong to keep the geometry simple. Thus, the flow can be described analytically and by using the analytical solution it is possible to avoid resolving the small-scale features. This approach is perhaps best demonstrated by the point-particle approximations. For drops that are much smaller than the smallest flow scales (and whose Stokes number is much less than unity) several investigators have carried out simulations where the flow away from the particles is fully resolved, but the particles are approximated as point particles. Although the flow around the particle is not resolved by the computational grid, there are good reasons to believe that the accuracy of these models can be very high, and when the Reynolds number of the particle is low enough, analytical results for the forces between the

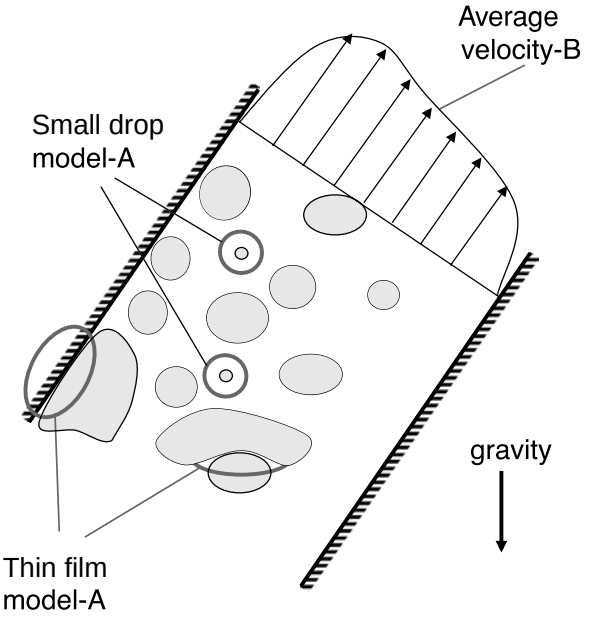


Fig. 11.10. Multiscale issues in multiphase flows and their classification according to E and Engquist (2003). While modeling "isolated defects" such as very thin films and drops is necessary to carry out the computations, the results are often intended to help develop a constitutive model of the flow.

particle and the fluid eliminate any empirical adjustments. Simulations of multiphase flows containing small bubbles, drops, or particles that are approximated as point particles go back many years (see Chapter 9 in Prosperetti and Tryggvason (2007) for a review), but it is only recently that researchers have started to use such models in simulations where "most" of the flow is captured fully using the techniques described in this book. Examples include computations of bubbles in slurry bubble reactors by Deen *et al.* (2004), where small catalytic particles are included as point particles, but the larger bubbles are fully resolved, and simulations of atomization by Tomar *et al.* (2010), where very small droplets are replaced by point particles. Thin films offer another obvious opportunity to include analytical (or quasi-anaytical) models of small-scale features. Studies of thin films have a long history, and model equations have been developed for a wide range of conditions. Little has been done so far, however, to incorporate these models into simulations of the larger scale motion. See, however, the study of Davis *et al.* (1989), where a thin-film model was coupled with a boundary integral computation to examine the collision of drops, the microlayer model of Son and Dhir (1998) for nucleate boiling, and Ge and Fan (2006), who examined the film boiling of drops colliding with hot particles.

As we go to smaller and smaller scales, we eventually reach a point where it is no longer fully justified to assume that the usual continuum hypothesis is accurate. It is then necessary either to change the modeling approach completely by, for example, using molecular simulations or possibly something like the dissipative particle dynamics approach, or to work with modified continuum formulations (such as phase-field models) designed to account for small-scale effects. See Nie *et al.* (2004) and Werder *et al.* (2005) for recent attempts to couple molecular dynamics with continuum simulations.

We end this section by noting that multiphase problems exhibit a wide range of multiscale issues, many of which have only recently been addressed in a more general setting. E and Engquist (2003) have, for example, classified multiscale problems into type A, dealing with isolated defects, and type B, constitutive modeling based on the microscopic models. The discussion above clearly involves type A problems, whereas the studies described in Chapter 8, where the goal is to understand the collective dynamics of large bubbly systems, is a type B problem. We show this schematically in the sketch in Fig. 11.10, which depicts bubbly flow in an inclined channel.

11.4 Summary

Multiphase flows play a major rôle in the workings of Nature and the enterprises of Man. They are important for energy production, manufacturing and chemical processes, wastewater treatment, agriculture, and many others processes critical for our current way of living. Considering the importance of such flows, it is not surprising that the origins of DNS of multiphase flows go back to the very beginning of computational fluid dynamics. However, progress was initially slow, and it is only recently that increased computer power and algorithmic development have allowed DNS to start to transform multiphase flow research. As the use of DNS for multiphase flow studies has increased and it has been applied to an increasing number of problems, it has become clear that in many situations it is necessary to examine problems that involve increasingly complex physics. In this chapter we have discussed a few extensions of the basic methodology presented earlier in this book to incorporate added physical complexity. The topics covered have not been discussed in any depth, nor is the list of topics complete. We hope, however, that this chapter has shown the reader that while much has been accomplished in DNS of multiphase flows, it remains an active field of research with major opportunities to make significant contributions towards the solution of important problems.

Appendix A

Interfaces: description and definitions

Following the motion of deformable interfaces separating different fluids or phases is at the center of accurate predictions of multiphase flows. In this section we introduce the basic mathematical notation used to describe their geometry and motion. The geometry of surfaces in three-dimensional space is a complex subject and the topic of several books. For the purpose of this book, however, we only need relatively elementary concepts and we can, in particular, leave out the introduction of the Christoffel symbols. While essential for more advanced treatment of curved spaces, their introduction involves considerable overhead that may not be of a central interest to many readers of this book. Our treatment will follow closely the presentation by Weatherburn (1927).

In three-dimensional space, the interface is a surface which separates the two fluid phases and in two dimensions it is a line. We start with lines and then move on to surfaces.

A.1 Two-dimensional geometry

The coordinates of a curve in two dimensions can be described by a vector function

$$\mathbf{x}(u) = (x(u), y(u)), \tag{A.1}$$

where u is a monotonically increasing scalar used to parameterize the curve. The vector $\mathrm{d}\mathbf{x}/\mathrm{d}u$ is tangent to the curve, and to obtain a unit tangent vector $\mathrm{d}\mathbf{x}/\mathrm{d}s$, where s is the arc length, it is necessary to normalize $\mathrm{d}\mathbf{x}/\mathrm{d}u$. The arc length is defined by $\mathrm{d}s^2 = \mathrm{d}x^2 + \mathrm{d}y^2$, and since $\mathrm{d}\mathbf{x}/\mathrm{d}u = (\mathrm{d}\mathbf{x}/\mathrm{d}s)(\mathrm{d}s/\mathrm{d}u) = \mathbf{t}(\mathrm{d}s/\mathrm{d}u)$, we can write $\mathbf{t} = (\mathrm{d}\mathbf{x}/\mathrm{d}u)/(\mathrm{d}s/\mathrm{d}u)$. From the definition of s, we have that $\mathrm{d}s/\mathrm{d}u = [(\mathrm{d}x/\mathrm{d}u)^2 + (\mathrm{d}y/\mathrm{d}u)^2]^{1/2}$. Introducing the shorthand notation

$$x_u = \mathrm{d}x/\mathrm{d}u, \quad x_{uu} = \mathrm{d}^2x/\mathrm{d}u^2, \quad y_u = \mathrm{d}y/\mathrm{d}u, \quad \text{and} \quad y_{uu} = \mathrm{d}^2y/\mathrm{d}u^2,$$

then

$$\mathbf{t} = \frac{(x_u, y_u)}{(x_u^2 + y_u^2)^{1/2}} \qquad \text{and} \qquad \mathbf{n} = \frac{(-y_u, x_u)}{(x_u^2 + y_u^2)^{1/2}}. \tag{A.2}$$

Here, we orient $\mathbf{n}$ so that $(\mathbf{t}, \mathbf{n})$ forms a right basis. The rate of change of the tangent vector with arc length is

$$\mathrm{d}\mathbf{t}/\mathrm{d}s = (\mathrm{d}^2\mathbf{x}/\mathrm{d}u^2)(\mathrm{d}u/\mathrm{d}s)^2,$$

so the curvature is given by

$$\kappa = \mathbf{n} \cdot \frac{\mathrm{d}\mathbf{t}}{\mathrm{d}s} = \frac{x_u y_{uu} - y_u x_{uu}}{(x_u^2 + y_u^2)^{3/2}}. \tag{A.3}$$

When the interface is described by a height function $y = h(x)$, then $x = u$, and $x_u = 1$, $x_{uu} = 0$, $y_u = \mathrm{d}h/\mathrm{d}x$, and $y_{uu} = \mathrm{d}^2h/\mathrm{d}x^2$. Equation (A.3) becomes

$$\kappa = \frac{\mathrm{d}^2 h}{\mathrm{d}x^2} \left[1 + \left(\frac{\mathrm{d}h}{\mathrm{d}x} \right)^2 \right]^{-3/2}, \tag{A.4}$$

and the tangent and the normal vectors are

$$\mathbf{t} = \frac{(1, \mathrm{d}h/\mathrm{d}x)}{\left[1 + (\mathrm{d}h/\mathrm{d}x)^2 \right]^{1/2}} \qquad \text{and} \qquad \mathbf{n} = \frac{(-\mathrm{d}h/\mathrm{d}x, 1)}{\left[1 + (\mathrm{d}h/\mathrm{d}x)^2 \right]^{1/2}}. \tag{A.5}$$

As an example of the use of the expressions derived above, consider an ellipse. We can describe it as a parametric curve by

$$x = a\cos u, \qquad y = b\sin u. \tag{A.6}$$

The tangent and the normal vectors are easily found to be

$$\mathbf{t} = \frac{(-a\sin u, b\cos u)}{(a^2\sin^2 u + b^2\cos^2 u)^{1/2}}, \qquad \mathbf{n} = \frac{(b\cos u, a\sin u)}{(a^2\sin^2 u + b^2\cos^2 u)^{1/2}}. \tag{A.7}$$

We reversed $\mathbf{n}$ with respect to (A.2) so that it is now oriented outwards, and the tangent is also reversed. (This still agrees with the formulae above if we change u into $-u$.) The curvature is

$$\kappa = -\frac{ab}{(a^2\sin^2 u + b^2\cos^2 u)^{3/2}}. \tag{A.8}$$

Notice that if $a = b$, then $\kappa = -1/b$, as in Chapter 2.

The ellipse can also be described in the *Monge* form by

$$y = \pm b(1 - x^2/a^2)^{1/2}. \tag{A.9}$$

The tangent and the normal vectors of the upper part are now

$$\mathbf{t} = \frac{(-a(a^2-x^2)^{1/2}, bx)}{\sqrt{a^4+(b^2-a^2)x^2}}, \qquad \mathbf{n} = \frac{(bx, a(a^2-x^2)^{1/2})}{\sqrt{a^4+(b^2-a^2)x^2}}. \tag{A.10}$$

Expression (A.1) results in

$$\kappa = \frac{-a^4 b}{[a^4+(b^2-a^2)x^2]^{3/2}}. \tag{A.11}$$

A.2 Three-dimensional geometry

While the concepts introduced for interfaces in two-dimensional space carry over to three dimensions, the technical details become more involved. As in two dimensions, the interface can by defined using a vector function

$$\mathbf{x}(u,v) = \Big(x(u,v), y(u,v), z(u,v)\Big), \tag{A.12}$$

where u and v are surface coordinates parameterizing the interface. See Fig. A.1. An arbitrary displacement in the surface can be written

$$\mathrm{d}\mathbf{x} = \frac{\partial \mathbf{x}}{\partial u}\mathrm{d}u + \frac{\partial \mathbf{x}}{\partial v}\mathrm{d}v = \mathbf{x}_u\,\mathrm{d}u + \mathbf{x}_v\,\mathrm{d}v. \tag{A.13}$$

The length is

$$\begin{aligned} \mathrm{d}s^2 &= \mathrm{d}\mathbf{x}\cdot\mathrm{d}\mathbf{x} \\ &= \mathbf{x}_u\cdot\mathbf{x}_u\,\mathrm{d}u^2 + 2\mathbf{x}_u\cdot\mathbf{x}_v\,\mathrm{d}u\,\mathrm{d}v + \mathbf{x}_v\cdot\mathbf{x}_v\,\mathrm{d}v^2 \\ &= E\,\mathrm{d}u^2 + 2F\,\mathrm{d}u\,\mathrm{d}v + G\,\mathrm{d}v^2 \end{aligned} \tag{A.14}$$

where we have followed Weatherburn (1927) and introduced the following notation

$$E = \mathbf{x}_u\cdot\mathbf{x}_u; \qquad F = \mathbf{x}_u\cdot\mathbf{x}_v; \qquad G = \mathbf{x}_v\cdot\mathbf{x}_v; \qquad H = |\mathbf{x}_u\times\mathbf{x}_v|. \tag{A.15}$$

It is easily shown that $H^2 = EG - F^2$. These quantities, as well as the second-order quantities

$$L = \mathbf{n}\cdot\mathbf{x}_{uu}, \qquad M = \mathbf{n}\cdot\mathbf{x}_{uv}, \quad \text{and} \quad N = \mathbf{n}\cdot\mathbf{x}_{vv}, \tag{A.16}$$

play a fundamental rôle in the theory of curved surfaces. The tangent vectors are orthogonal when $F = 0$ and orthonormal when in addition $E = G = 1$. It is not possible in general to have an orthonormal coordinate system adapted to an arbitrary interface. However, for some definitions we shall find it useful to think in terms of a locally orthonormal system; that is, a system for which $E = G = 1$ and $F = 0$ for a given point on the interface.

Quantifying the curvature of a surface in three dimensions is as important as for a two-dimensional curve. Since we have already introduced the curvature of a curve, it is natural to ask whether we can use the curvature of a curve, in the surface, to measure the curvature. Generally the answer is no: different curves have different curvatures. Furthermore, the normal to a curve lying in a surface

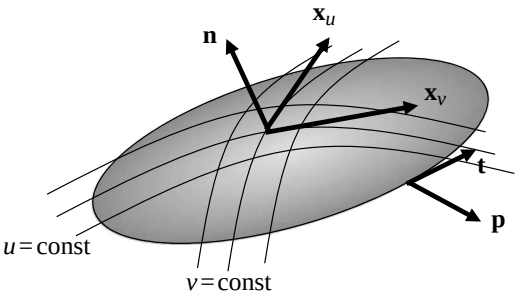


Fig. A.1. Surface coordinates.

is generally different than the normal to the surface (a curve in a plane usually will have a normal in the plane, thus perpendicular to a normal to the plane!). Some curves, however, may have a normal that coincides with the normal of the surface and it seems reasonable to attempt to use such curves to measure or quantify the curvature of a surface. For a curve, the rate of change of a normal vector with arc length (Equation (2.29)) can be written as $\mathrm{d}\mathbf{n} = -\kappa \mathrm{d}\mathbf{x}$ (since $\mathbf{t} = \mathrm{d}\mathbf{x}/\mathrm{d}s$), relating a small displacement along the curve to the change in the normal. A small displacement on the surface is $\mathrm{d}\mathbf{x} = \mathbf{x}_u\,\mathrm{d}u + \mathbf{x}_v\,\mathrm{d}v$ and the change in the normal is $\mathrm{d}\mathbf{n} = \mathbf{n}_u\,\mathrm{d}u + \mathbf{n}_v\,\mathrm{d}v$, where the subscript denotes differentiation. We now ask whether there are directions such that the change in the normal is related to the displacement along the surface by the same relationship that holds for a line. If that is true, we can write

$$\mathbf{n}_u\,\mathrm{d}u + \mathbf{n}_v\,\mathrm{d}v = -\kappa(\mathbf{x}_u\,\mathrm{d}u + \mathbf{x}_v\,\mathrm{d}v)$$

or

$$(\kappa\mathbf{x}_u + \mathbf{n}_u)\,\mathrm{d}u + (\kappa\mathbf{x}_v + \mathbf{n}_v)\,\mathrm{d}v = 0. \tag{A.17}$$

Taking the dot product of this equation with $\mathbf{x}_u$ and $\mathbf{x}_v$ respectively, yields

$$\begin{aligned}(\kappa E - L)\,\mathrm{d}u + (\kappa F - M)\,\mathrm{d}v = 0,\\ (\kappa F - M)\,\mathrm{d}u + (\kappa G - N)\,\mathrm{d}v = 0.\end{aligned} \tag{A.18}$$

Eliminating $\mathrm{d}u$ and $\mathrm{d}v$ gives a quadratic equation for κ:

$$H^2\kappa^2 - (EN - 2FM + GL)\kappa + (LN - M^2)^2 = 0, \tag{A.19}$$

and solving this equation gives the two principal curvatures. It can be shown that the principal curvatures are unique and are sufficient to describe the curvature of the surface. We are rarely concerned with the individual principal curvatures, but instead work with the *first* curvature or the *mean* curvature, defined as the sum of the principal curvatures:

$$\kappa = \kappa_1 + \kappa_2. \tag{A.20}$$

Adding up the solutions of (A.19) gives

$$\kappa = \frac{1}{H^2}(EN - 2FM + GL). \tag{A.21}$$

Sometimes the *mean* curvature is defined as the average of the principal curvatures. It obviously differs from the definition given here only by a factor of $1/2$.

The second (or *total*, or *Gauss*) curvature is the product of the principal curvatures and can be shown to be

$$\kappa_{\text{Gauss}} = \kappa_1 \kappa_2 = \frac{(LN - M^2)^2}{H^2}. \tag{A.22}$$

Once the curvatures have been found, the principal directions can be found by solving for $\mathrm{d}u/\mathrm{d}v$:

$$\frac{\mathrm{d}u}{\mathrm{d}v} = -\frac{\kappa_i F - M}{\kappa_i E - L}, \tag{A.23}$$

obtained from the first of the equations in (A.18), with $i = 1$ and $i = 2$ giving the two different directions. This is, however, rarely needed. For a surface given by the Monge parameterization

$$z = z(x, y),$$

we find that

$$\mathbf{x}_u = (1, 0, z_x) \qquad \text{and} \qquad \mathbf{x}_v = (0, 1, z_y)$$

and

$$E = 1 + z_x^2, \qquad F = z_x z_y, \qquad G = 1 + z_y^2, \qquad H^2 = 1 + z_x^2 + z_y^2.$$

The unit normal is

$$\mathbf{n} = \frac{\mathbf{x}_u \times \mathbf{x}_v}{H} = \frac{(-z_x, -z_y, 1)}{\sqrt{1 + z_x^2 + z_y^2}}$$

and the curvature is

$$\kappa = \frac{z_{xx}(1 + z_y^2) - 2 z_x z_y z_{xy} + z_{yy}(1 + z_y^2)}{(1 + z_x^2 + z_y^2)^{3/2}}.$$

A.3 Axisymmetric geometry

In axisymmetric geometry, the interface may be conveniently represented in cylindrical or spherical coordinates. Here, we consider the cylindrical coordinates r, z and ϕ and apply the previous definitions to an interface defined by $z = z(r)$. The surface coordinates are $u = r$, $v = \phi$, and $\mathrm{d}\mathbf{x} = r\mathrm{d}\phi\ \mathbf{e}_\phi + (\mathrm{d}r/\cos\theta)\mathbf{t}$, where θ is the angle of the normal $\mathbf{n}$ with the z axis. Thus, $\mathrm{d}s^2 = r^2\mathrm{d}\phi^2 + \mathrm{d}r^2/\cos^2\theta$.

Then $E = r^2, F = 0, G = 1/\cos^2\theta$, and $H = r/\cos\theta$. Differentiating, we find $L = -r\sin\theta$, $M = 0$, and

$$N = \frac{1}{\cos^2\theta}\frac{\sin\theta}{r}\frac{\partial \mathbf{t}}{\partial\theta}\cdot\mathbf{n}.$$

It is readily seen that $(\sin\theta/r)(\partial\mathbf{t}/\partial\theta) = \partial\mathbf{t}/\partial s$, then $N = \kappa_1/\cos^2\theta$, where κ_1 is the curvature of the curve $z = z(r)$ in the meridian half-plane (r,z). Finally, using (A.21), we get

$$\kappa = \kappa_1 + \frac{1}{R}, \tag{A.24}$$

where $R = r/\sin\theta$ is the distance to the axis along the normal. It is readily verified that this expression works for a sphere.

A.4 Differentiation and integration on surfaces

We shall frequently find it useful to define the surface gradient and surface divergence. To find the gradient of a scalar function ϕ, we shall use the fact that the gradient must necessarily be perpendicular to level curves of ϕ. For a displacement along a level curve $(\delta u, \delta v)$, the change in ϕ is zero, so $\mathrm{d}\phi = \phi_u\delta u + \phi_v\delta v = 0$, or

$$\delta u/\delta v = -\phi_v/\phi_u. \tag{A.25}$$

The displacement along the interface is

$$\delta\mathbf{x} = \mathbf{x}_u\delta u + \mathbf{x}_v\delta v.$$

Consider now another displacement $(\mathrm{d}u, \mathrm{d}v)$ for which

$$d\mathbf{x} = \mathbf{x}_u\,\mathrm{d}u + \mathbf{x}_v\,\mathrm{d}v.$$

If these displacements are orthogonal, then $\mathrm{d}\mathbf{x}\cdot\delta\mathbf{x} = 0$, or

$$\mathbf{x}_u\cdot\mathbf{x}_u\,\mathrm{d}u\,\delta u + \mathbf{x}_u\cdot\mathbf{x}_v(\mathrm{d}u\,\delta v + \delta u\mathrm{d}v) + \mathbf{x}_v\cdot\mathbf{x}_v\,\mathrm{d}v\delta v = 0\,.$$

Introducing the notation defined by Equation (A.15) and by using (A.25), this can be written as

$$(F\phi_u - E\phi_v)\,\mathrm{d}u + (G\phi_u - F\phi_v)\,\mathrm{d}v = 0.$$

Thus, the vector $(F\phi_u - E\phi_v, G\phi_u - F\phi_v)$ is perpendicular to $(\mathrm{d}u, \mathrm{d}v)$, or parallel to $(\delta u, \delta v)$, along the $\phi =$ constant level curve. The vector $(-b, a)$ is perpendicular to the vector (a, b) and, therefore, the gradient of ϕ is given by

$$\nabla_s\phi = \frac{1}{H^2}\left(G\frac{\partial\phi}{\partial u} - F\frac{\partial\phi}{\partial v}, E\frac{\partial\phi}{\partial v} - F\frac{\partial\phi}{\partial u}\right), \tag{A.26}$$

where we have divided by H^2 to ensure $\mathrm{d}\phi = \nabla_s\phi \cdot \mathrm{d}\mathbf{x}$. Similarly, the surface divergence of a vector is defined by

$$\nabla_s \cdot \mathbf{F} = \frac{1}{H^2}\mathbf{x}_u \cdot \left(G\frac{\partial \mathbf{F}}{\partial u} - F\frac{\partial \mathbf{F}}{\partial v} \right) + \frac{1}{H^2}\mathbf{x}_v \cdot \left(E\frac{\partial \mathbf{F}}{\partial v} - F\frac{\partial \mathbf{F}}{\partial u} \right). \tag{A.27}$$

Notice that if the coordinate system is orthonormal, such that $F = 0$ and $H = EG = 1$, then this expression reduces to the familiar form $\nabla_s = \mathbf{t}_1\partial/\partial\xi_1 + \mathbf{t}_2\partial/\partial\xi_2$, where the **t**s' are unit tangent vectors and the ξs' are the coordinates in the orthonormal system.

Applying the divergence to the normal vector results in

$$\nabla_s \cdot \mathbf{n} = \frac{1}{H^2}\mathbf{x}_u \cdot (G\mathbf{n}_u - F\mathbf{n}_v) + \frac{1}{H^2}\mathbf{x}_v \cdot (E\mathbf{n}_v - F\mathbf{n}_u). \tag{A.28}$$

Differentiating the relation $\mathbf{n} \cdot \mathbf{x}_u = 0$ yields $\mathbf{n}_u \cdot \mathbf{x}_u + \mathbf{n} \cdot \mathbf{x}_{uu} = 0$. Similar results are obtained differentiating $\mathbf{n} \cdot \mathbf{x}_v = 0$ with respect to first v and then u. Thus:

$$\begin{aligned} \mathbf{n}_u \cdot \mathbf{x}_u &= -\mathbf{n} \cdot \mathbf{x}_{uu} = -L, \\ \mathbf{n}_v \cdot \mathbf{x}_u = \mathbf{n}_u \cdot \mathbf{x}_v &= -\mathbf{n} \cdot \mathbf{x}_{uv} = -M, \\ \mathbf{n}_v \cdot \mathbf{x}_v &= -\mathbf{n} \cdot \mathbf{x}_{vv} = -N. \end{aligned} \tag{A.29}$$

Using these relations allows us to write

$$\nabla_s \cdot \mathbf{n} = -\frac{1}{H^2}(EN - 2FM + GL) = -\kappa. \tag{A.30}$$

An arbitrary vector **F** can be decomposed into a component normal to a surface and a component that is tangent to the surface. The normal component is given by $\mathbf{F}_n = (\mathbf{n} \cdot \mathbf{F})\mathbf{n}$ and the tangent component is found by subtracting the normal component from the original vector. The tangential component is therefore

$$\mathbf{F}_s = \mathbf{F} - (\mathbf{n} \cdot \mathbf{F})\mathbf{n} = (\mathbf{I} - \mathbf{nn}) \cdot \mathbf{F}, \tag{A.31}$$

where we have introduced the unit tensor **I** whose components are given by $I_{ij} = \delta_{ij}$. The relation $(\mathbf{n} \cdot \mathbf{F})\mathbf{n} = (\mathbf{nn}) \cdot \mathbf{F}$ is easily verified by writing it out in component form, $(n_i F_i)n_j = (n_i n_j)F_i$. When applied to the gradient operator, we have $\nabla_s() = (\mathbf{I} - \mathbf{nn}) \cdot \nabla()$. This can be used to show that the normal component of the gradient of the normal vector is zero. Thus, curvature can be found by either taking the surface divergence or the divergence of the normal vector. We leave it to the reader to show that

$$\nabla \cdot \mathbf{n} = -\kappa. \tag{A.32}$$

The use of this expression is easily demonstrated by applying it to a circle at the origin, identified as the $f = 0$ contour of the function $f(x,y) = x^2 + y^2 - R^2$.

The gradient is $\nabla f = (2x, 2y)$ and the normal vector is given by $\mathbf{n} = \nabla f/|\nabla f| = (x,y)/\sqrt{x^2+y^2}$. The divergence of the normal is then

$$\nabla \cdot \mathbf{n} = \frac{y^2}{(x^2+y^2)^{3/2}} + \frac{x^2}{(x^2+y^2)^{3/2}} = \frac{1}{R}, \tag{A.33}$$

where we have used that $R = \sqrt{x^2+y^2}$ when $f = 0$. We leave it to the reader to compute the mean curvature for an ellipse and other simple shapes.

We will, on occasion, need the following relationship:

$$(\mathbf{n} \times \nabla) \times \mathbf{n} = \kappa \mathbf{n}, \tag{A.34}$$

which is easily proven if we work in index notation. The ith component of the left-hand side is $(n_j \nabla_k \varepsilon_{ijk}) n_l \varepsilon_{mil} = -n_j \nabla_k n_l \varepsilon_{ijk} \varepsilon_{iml} = -n_j \nabla_k n_l (\delta_{jm}\delta_{kl} - \delta_{jl}\delta_{km}) = -n_m \nabla_l n_l + n_l \nabla_m n_l = -n_m (n_l)_l + n_l (n_l)_m$. The first term on the right-hand side is κn_m and the last term is zero, since $n_l (n_l)_m = (n_l n_l)_m - n_l (n_l)_m$, showing that $n_l (n_l)_m = (1/2)(n_l n_l)_m = 0$, since $n_l n_l = 1$.

Similarly, we can show that

$$\nabla_s \cdot \mathbf{I}_s = \kappa \mathbf{n}. \tag{A.35}$$

When a surface is discretized, or when deriving equations for physical processes taking place at an interface, we frequently have to work with integrals over a finite surface area. Several relations can be derived allowing us to transform between area integrals and contour integrals along the perimeter of the surface element. We shall not go through the detailed derivation here, but state that the divergence theorem for a curved surface takes the form

$$\int_A \nabla_s \cdot \mathbf{F} \, \mathrm{d}a = \oint_C \mathbf{F} \cdot \mathbf{p} \, \mathrm{d}l - \int_A \kappa \mathbf{F} \cdot \mathbf{n} \, \mathrm{d}a, \tag{A.36}$$

where $\mathbf{p} = \mathbf{t} \times \mathbf{n}$ is a vector in the surface, perpendicular to the boundary of the surface element (see Fig. A.1) and C is the curve bounding the area A.

Putting $\mathbf{F} = \mathbf{c}$, where $\mathbf{c}$ is a constant vector, the first term is identically zero and we have

$$\mathbf{c} \cdot \left(\oint_C \mathbf{p} \, \mathrm{d}l - \int_A \kappa \mathbf{n} \, \mathrm{d}a \right) = 0. \tag{A.37}$$

Since $\mathbf{c}$ is arbitrary and the above identity always holds, we must have

$$\oint_C \mathbf{p} \, \mathrm{d}l = \int_A \kappa \mathbf{n} \, \mathrm{d}a. \tag{A.38}$$

Applying this expression to an infinitesimal small area δA around a point on the surface and taking the limit, we obtain an alternative expression for the mean curvature:

$$\kappa \mathbf{n} = \lim_{\delta A \to 0} \frac{1}{\delta A} \oint \mathbf{p} \, \mathrm{d}l. \tag{A.39}$$

We also find that, for a closed surface, we must have

$$\int_A \kappa \mathbf{n}\, \mathrm{d}a = 0, \tag{A.40}$$

which has important consequences for the total surface force. Applying the divergence theorem (A.36) to the vector $\mathbf{F} \times \mathbf{c}$, where $\mathbf{c}$ is a again an arbitrary constant, yields

$$\int_A \nabla_s \times \mathbf{F}\, \mathrm{d}a = \oint_C \mathbf{p} \times \mathbf{F}\, \mathrm{d}l - \int_A \kappa \mathbf{n} \times \mathbf{F}\, \mathrm{d}a. \tag{A.41}$$

Other relations are easily derived by selecting $\mathbf{F}$ differently.

Appendix B

Distributions concentrated on the interface

The step function H, where $H = 1$ in one fluid and $H = 0$ everywhere else, introduced in Section 2.3 plays a fundamental rôle in our discussion of the "one fluid" approach. H can be constructed in different ways, but for our purpose it is convenient to express it in terms of an integral over the product of one-dimensional δ-functions:

$$H(x,y) = \int_V \delta(x-x')\delta(y-y')\,\mathrm{d}v'. \tag{B.1}$$

Here, the integration is over the region V bounded by the surface S and $\mathrm{d}v' = \mathrm{d}x'\mathrm{d}y'$; see Fig. 2.7. Obviously, $H = 1$ if the point (x,y) is inside the contour, where it will coincide with (x',y') during the integration, and $H = 0$ if it is outside and will never be equal to (x',y'). To simplify the discussion slightly we will consider a two-dimensional domain here; the extension to three dimensions is straightforward.

The interface is located where the gradient of the step function $\nabla H(x,y)$ is non-zero. Since the gradient of Equation (B.1) is taken with respect to the unprimed variables, the gradient operator can be put under the integral sign:

$$\nabla H = \int_V \nabla \left[\delta(x-x')\delta(y-y')\right] \mathrm{d}v'. \tag{B.2}$$

The gradient with respect to the unprimed variables can be replaced by the gradient with respect to the primed variables:

$$\nabla H = -\int_V \nabla' \left[\delta(x-x')\delta(y-y')\right] \mathrm{d}v', \tag{B.3}$$

where the prime on the gradient symbol denotes the gradient with respect to the primed variables. The resulting area (or volume in three dimensions) integral can then be transformed into a line (surface) integral by a variation of the divergence

theorem for gradients:

$$\nabla H = -\oint_S \delta(x-x')\delta(y-y')\mathbf{n}'ds'. \tag{B.4}$$

Although we have assumed that the area occupied by the marked fluid is finite so that S is a closed contour, the contribution of most of the integral is zero, so we can replace it by one over a part of the contour and drop the circle on the integral:

$$\nabla H = -\int_S \delta(x-x')\delta(y-y')\mathbf{n}'\,\mathrm{d}s'. \tag{B.5}$$

By introducing local coordinates, tangent s and normal n to the interface we can write

$$\delta(x-x')\delta(y-y') = \delta(s)\delta(n) \tag{B.6}$$

and evaluate the integral

$$-\int_S \delta(x-x')\delta(y-y')\mathbf{n}'\,\mathrm{d}s' = -\int_S \delta(s)\delta(n)\mathbf{n}'\,\mathrm{d}s' = -\delta(n)\mathbf{n}. \tag{B.7}$$

We shall frequently find it useful to define a function δ_S, which is concentrated on the interface just as the Dirac δ-function is concentrated on a point. By using a locally orthonormal coordinate system ξ_1, ξ_2, n, where n is as usual the coordinate normal to the interface, this is exactly the $\delta(n)$ introduced above. Thus, we set

$$\delta_S(\mathbf{x}) = \delta(n), \tag{B.8}$$

where δ is the ordinary Dirac distribution. The function δ_S has the property of changing volume integrals into surface integrals. For an arbitrary function f,

$$\int_V \delta_S(\mathbf{x}) f(\mathbf{x})\,\mathrm{d}v = \int_S f(\mathbf{x})\,\mathrm{d}s, \tag{B.9}$$

and the gradient of the characteristic function H is related to the surface distribution δ_S by

$$\nabla H = -\delta_S \mathbf{n}. \tag{B.10}$$

For computations with generalized functions we usually smooth the function onto a regular fixed grid. To do the smoothing it is useful to recall that a δ-function can be defined as the limit of a smooth peaked function when the "width" of the peak goes to zero, as shown in Fig. B.1. Thus, for example, we can write

$$\delta(x) = \lim_{\varepsilon\to 0} \frac{\varepsilon}{\pi(x^2+\varepsilon^2)} \qquad \text{or} \qquad \delta(x) = \lim_{\varepsilon\to 0} \frac{\mathrm{e}^{-x^2/4\varepsilon}}{2\sqrt{\pi\varepsilon}}. \tag{B.11}$$

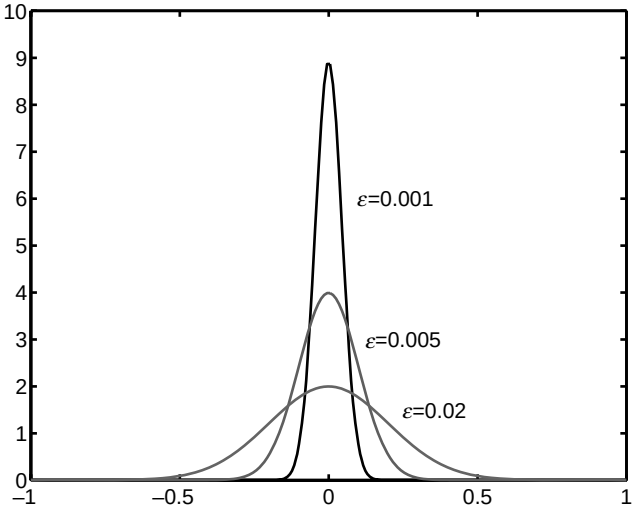


Fig. B.1. The delta function as a limit of a smooth function, as $\varepsilon \to 0$.

When the delta function is represented on a fixed grid, we stop the limiting process at a finite value of the width parameter, in the same way as finite-difference approximations can be obtained from the definition of the derivative. Since the δ-function is the derivative of the step function, the step function can be obtained by integration of the δ-function (or, if we pick a smoothed form for the step function, we can obtain the δ-function by differentiation). On a discrete grid, these operations can, of course, be done numerically.

Figure B.2 shows a circular cylinder represented by a two-dimensional step function, where we have smoothed the transition from one to zero over a few grid spacings. Differentiating this function results in a smooth approximation to the δ-function.

B.1 A simple example

We will use a simple one-dimensional example below to show how jump conditions can be incorporated into the governing equation as a singular source term. Consider the advection of a quantity f (a color or a species tracer) with uniform velocity U_0. The flow comes from the left and initially $f = 0$. At x_s, however, f is injected. The traditional way of solving this problem is to write the governing equation separately before and after x_s and couple them together with jump conditions at x_s. Thus, we have

$$\frac{\partial f}{\partial t} + U_0 \frac{\partial f}{\partial x} = 0 \quad \text{for} \quad x < x_s \quad \text{and} \quad x > x_s \,. \tag{B.12}$$

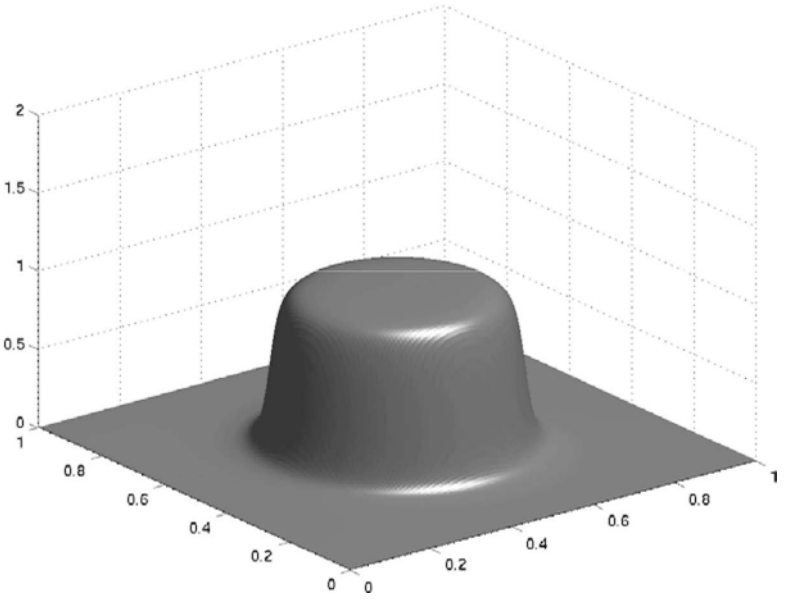


Fig. B.2. Smoothed circle.

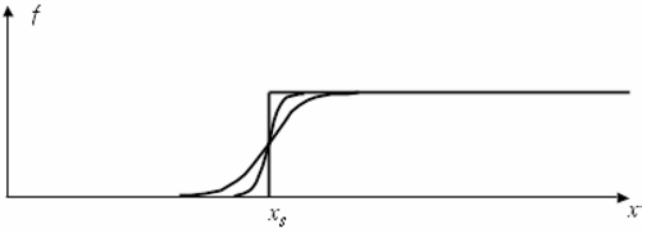


Fig. B.3. The approximation of a discontinuity by a smooth transition zone for a simple one-dimensional advection. The discontinuous solution and two smooth approximations are shown.

At x_s we have $[U_o f] = q$, where q is the injection rate of f. For this simple problem the solution is, of course, trivial: $f = 0$ to the left of x_s and $f = q/U_o$ to the right of x_s.

We can, however, also write this equation for the whole domain as

$$\frac{\partial f}{\partial t} + U_o \frac{\partial f}{\partial x} = q\delta(x - x_s). \tag{B.13}$$

To show that the two formulations are equivalent, we write

$$f(t,x) = f_1(t,x)H_1(x - x_s) + f_2(t,x)H_2(x - x_s), \tag{B.14}$$

where $H_1 = 1$ to the left of x_s and $H_1 = 0$ to the right of x_s, and further $H_2 = 1 - H_1$.

Substituting into (B.13) yields

$$H_1\Big(\frac{\partial f_1}{\partial t}+U_o\frac{\partial f_1}{\partial x}\Big)+H_2\Big(\frac{\partial f_2}{\partial t}+U_o\frac{\partial f_2}{\partial x}\Big)+U_o(f_2-f_1)\delta=q\delta, \qquad \text{(B.15)}$$

since $\partial H_2/\partial x=-\partial H_1/\partial x=\delta$. Gathering the terms, we recover the original equations, including the jump condition $q=U_o(f_2-f_1)=[U_o f]$.

When the δ-function is approximated by a smoother function, the discontinuity becomes a transition zone of finite width, as shown in Fig. B.3.

Appendix C

Cube-chopping algorithm

We consider the geometrical problem of a planar interface that intersects a right hexahedron of sides Δx_1, Δx_2, Δx_3 and volume $V_0 = \Delta x_1 \Delta x_2 \Delta x_3$. The plane is described by the equation

$$\mathbf{m} \cdot \mathbf{x} = m_1 x_1 + m_2 x_2 + m_3 x_3 = \alpha, \tag{C.1}$$

where the normal $\mathbf{m}$ is pointing outside the volume to be computed and the three coefficients m_i are known and positive. In the direct problem we compute the volume of the cell which is below a planar interface with the following expression:

$$V = h^3 C = \frac{1}{6 m_1 m_2 m_3} \left[\alpha^3 - \sum_{i=1}^{3} F_3(\alpha - m_i h) + \sum_{i=1}^{3} F_3(\alpha - \alpha_{\max} + m_i h) \right], \tag{C.2}$$

with $\alpha_{\max} = \sum_{i=1}^{3} m_i \Delta x_i$ and $F_n(z) = z^n$ when $z > 0$ and zero otherwise. Geometrically, and with reference to Fig. C.1, the first term $(\alpha^3 / 6 m_1 m_2 m_3)$ is the volume of the right tetrahedron under the triangle AEH. The first sum subtracts the volumes of the tetrahedra under the two triangles CEG and BFH, when the vertices E and H move beyond the cell faces. Finally, the second sum adds back the volume of the tetrahedron under the triangle DFG, when the line EH is completely outside the cell. Furthermore, we notice that all the right tetrahedra and triangles involved in the computation are similar. The function V varies from the value zero, when $\alpha = 0$, to V_0, when $\alpha = \alpha_{\max}$. In two dimensions the previous relation simplifies to

$$A = \frac{1}{2 m_1 m_2} \left[\alpha^2 - \sum_{i=1}^{2} F_2(\alpha - m_i \Delta x_i) \right]. \tag{C.3}$$

Let us assume for the moment that $\Delta x_i = 1$ and $\sum_{i=1}^{3} m_i = 1$, then $\alpha_{\max} = 1$ and the function $V = V(\alpha)$ has the following properties:

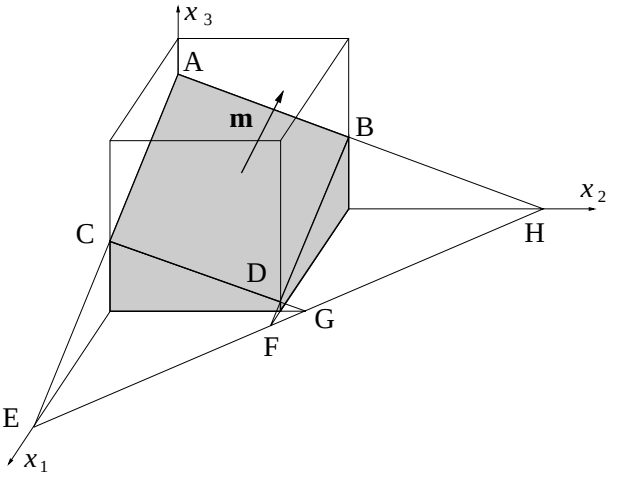


Fig. C.1. The "cut volume" is the gray region inside the right hexahedrical cell and below the planar interface ABCD.

(i) V is a continuous, one-to-one, monotonically increasing function of α with continuous first derivative;
(ii) both V and α vary in the range $[0,1]$ and there is an odd symmetry with respect to the point $(V,\alpha) = (1/2,1/2)$, and then the analysis can be restricted to the range $[0,1/2]$;
(iii) the relation $V = V(\alpha)$ is invariant with respect to a permutation of the indices, so we need to consider only the case $m_1 \leq m_2 \leq m_3$ in three dimensions and $m_1 \leq m_2$ in two dimensions;
(iv) in three dimensions the limit $m_1 \to 0$ is smooth and relation (C.2) becomes (C.3).

With some straightforward algebra the following expressions for the direct and inverse functions are readily computed (see Scardovelli and Zaleski, 2000, for more details).

C.1 Two-dimensional problem

Let $m_1 + m_2 = 1$ ($m_1 \leq m_2$) and $A_1 = m_1/(2m_2)$, then the analytical expressions for the direct and inverse problem are respectively

$$A = \frac{\alpha^2}{2m_1 m_2}, \qquad \text{for: } 0 \leq \alpha < m_1,$$

$$A = \frac{\alpha}{m_2} - A_1, \qquad \text{for: } m_1 \leq \alpha \leq 1/2,$$

$$\alpha = \sqrt{2m_1 m_2 A}\,, \qquad \text{for: } 0 \le A < A_1\,,$$
$$\alpha = m_2 A + \frac{m_1}{2}\,, \qquad \text{for: } A_1 \le A \le 1/2\,.$$

C.2 Three-dimensional problem

Let $m_1 + m_2 + m_3 = 1$ ($m_1 \le m_2 \le m_3$) and $\widetilde{m} = \min(m_{12}, m_3)$, where $m_{12} = m_1 + m_2$, then there are four consecutive ranges and the formulas for the direct problem are

$$V = \frac{\alpha^3}{6m_1 m_2 m_3}\,, \qquad \text{for: } 0 \le \alpha < m_1\,,$$
$$V = \frac{\alpha(\alpha - m_1)}{2m_2 m_3} + V_1\,, \qquad \text{for: } m_1 \le \alpha < m_2\,,$$
$$V = \frac{\alpha^2(3m_{12} - \alpha) + m_1^2(m_1 - 3\alpha) + m_2^2(m_2 - 3\alpha)}{6m_1 m_2 m_3}\,, \qquad \text{for: } m_2 \le \alpha < \widetilde{m}\,.$$

For the fourth interval there are two possible cases, one for $\widetilde{m} = m_3 < m_{12}$ and the other for $\widetilde{m} = m_{12} < m_3$:

$$V = \frac{\alpha^2(3 - 2\alpha) + \sum_{i=1}^{3} m_i^2(m_i - 3\alpha)}{6m_1 m_2 m_3}\,, \qquad \text{for: } m_3 \le \alpha \le 1/2\,,$$
$$V = \frac{2\alpha - m_{12}}{2m_3}\,, \qquad \text{for: } m_{12} \le \alpha \le 1/2\,.$$

The formulas for the inverse problem are

$$\alpha = \sqrt[3]{6m_1 m_2 m_3 V}\,, \qquad \text{for: } 0 \le V < V_1\,,$$
$$\alpha = \frac{1}{2}\left[m_1 + \sqrt{m_1^2 + 8m_2 m_3 (V - V_1)}\,\right], \qquad \text{for: } V_1 \le V < V_2\,,$$
$$P(\alpha) = a_3' \alpha^3 + a_2' \alpha^2 + a_1' \alpha + a_0' = 0\,, \qquad \text{for: } V_2 \le V < V_3\,.$$

Again, there are two cases in the fourth interval, one for $V_3 = V_{31} < V_{32}$ and the other for $V_3 = V_{32} < V_{31}$:

$$P(\alpha) = a_3'' \alpha^3 + a_2'' \alpha^2 + a_1'' \alpha + a_0'' = 0\,, \qquad \text{for: } V_{31} \le V \le 1/2\,,$$
$$\alpha = m_3 V + \frac{m_{12}}{2}\,, \qquad \text{for: } V_{32} \le V \le 1/2\,.$$

In the previous relations $V_1 = m_1^2/(\max(6m_2 m_3, \varepsilon))$ is an approximation of the actual value $m_1^2/6m_2 m_3$. This is because the limit for V_1 as $m_1, m_2 \to 0$ is zero; however, numerically, we need to avoid that the denominator of V_1 becomes zero, so ε is an arbitrary small number. The other limiting values of the range of validity of each relation are given by the following expressions: $V_2 = V_1 + (m_2 - m_1)/2m_3$,

$V_{31} = [m_3^2(3m_{12} - m_3) + m_1^2(m_1 - 3m_3) + m_2^2(m_2 - 3m_3)]/(6m_1m_2m_3)$, and $V_{32} = m_{12}/2m_3$. The coefficients of the two cubic polynomials are: $a_3' = -1$, $a_2' = 3m_{12}$, $a_1' = -3(m_1^2 + m_2^2)$, $a_0' = m_1^3 + m_2^3 - 6m_1m_2m_3V$, $a_3'' = -2$, $a_2'' = 3$, $a_1'' = -3(m_1^2 + m_2^2 + m_3^2)$, and $a_0'' = m_1^3 + m_2^3 + m_3^3 - 6m_1m_2m_3V$.

In the third and fourth regions, when $V_{31} \leq V \leq 1/2$, there is the need to find the roots of the cubic polynomial $P(\alpha)$. An analytical solution can be easily found following Abramowitz and Stegun (1964). With $a_3 = 1$, the discriminant $\Delta = p_0^3 + q_0^2$ is negative, and $P(\alpha)$ has three real roots $(\alpha_1, \alpha_2, \alpha_3)$. However, only α_2 satisfies the requirement that $P(\alpha)$ is an increasing function of α:

$$\alpha_2 = \sqrt{-p_0}(\sqrt{3}\sin\theta - \cos\theta) - \frac{a_2}{3},$$

where we have set $p_0 = a_1/3 - (a_2/3)^2$, $q_0 = (a_1a_2 - 3a_0)/6 - (a_2/3)^3$, and $\cos(3\theta) = q_0/\sqrt{-p_0^3}$.

We can now generalize the problem to rectangular grids and negative m_i. We divide expression (C.2) by the cell volume V_0 and obtain the following expression for the volume fraction $C = V/V_0$:

$$C = \frac{1}{6\Pi_{i=1}^3(m_i\Delta x_i)}\left[\alpha^3 - \sum_{i=1}^3 F_3(\alpha - m_i\Delta x_i) + \sum_{i=1}^3 F_3(\alpha - \alpha_{\max} + m_i\Delta x_i)\right]. \tag{C.4}$$

If we normalize the plane equation in order to have $\alpha_{\max} = \sum_{i=1}^3 m_i\Delta x_i = 1$, we are back to the equation examined in the previous discussion and the results obtained can be extended to a right hexahedron as well.

If one or more of the m_i are negative, the geometry can be mapped to the standard case with the linear transformation $x_i' = \Delta x_i - x_i$, which describes a mirror reflection of the figure with respect to the plane $x_i = \Delta x_i/2$. After the calculation of C or α the configuration is brought back to its actual position with similar reflections.

Finally, it is often required to calculate the volume cut by the same linear interface in a right hexahedron that is different from the grid cell; for example, to compute fluxes across the cell boundary. In this case with the linear transformations $x_i' = x_i - x_{0i}$ we translate the origin of the local coordinate system to the vertex $\mathbf{x}_0$ of the right hexahedron, where all sides $\Delta x_i'$ are positive.

Appendix D

The dynamics of liquid sheets: linearized theory

In this appendix we summarize some theoretical results for the motion of superposed liquid layers. The layers may be almost at rest, as a layer of gas above a layer of water, or be in shearing motion. In the first case we find many classical instabilities and oscillations, such as water waves, capillary waves, and the Rayleigh–Taylor instability. In the second case we find the Kelvin–Helmholtz instability.

D.1 Flow configuration

We consider two superposed horizontal fluid layers. Each phase n ($n = 1,2$) has constant density and viscosity, ρ_n and μ_n respectively, and a uniform horizontal velocity U_n. In two dimensions the interface height is given by $y = h(x,t)$ and in the unperturbed state $h(x,t) = h_0 = 0$. We assume that phase 2 is located above the interface, where $y > 0$, and phase 1 is below the interface, where $y < 0$. If $U_1 \neq U_2$, see Fig. D.1(a), there is a discontinuity in the velocity which may be maintained in the inviscid approximation, $\mu_n = 0$, that we consider first. Consider now a small perturbation of wavelength λ. Since the unperturbed state does not depend on the x coordinate and time t, we can write any physical quantity q as the sum of its unperturbed value q_0 and the small perturbation q' evolving as

$$q'(x,y,t) = q_1(y) \exp\left\{ \mathrm{i} \left[k(x - ct) + \varphi \right] \right\}, \tag{D.1}$$

where φ is an arbitrary phase, the wavenumber $k = 2\pi/\lambda$, and $c = c_\mathrm{r} + \mathrm{i}c_\mathrm{i}$ the complex wave speed, with $\gamma = kc_\mathrm{i}$ the growth rate and $\omega = kc_\mathrm{r}$ the frequency.

D.2 Inviscid results

D.2.1 The general dispersion relation

The *inviscid* Euler equation is obtained by letting $\mu = 0$ in the Navier–Stokes equation, (2.12):

$$\rho \frac{D\mathbf{u}}{Dt} = \rho \mathbf{g} - \nabla p. \tag{D.2}$$

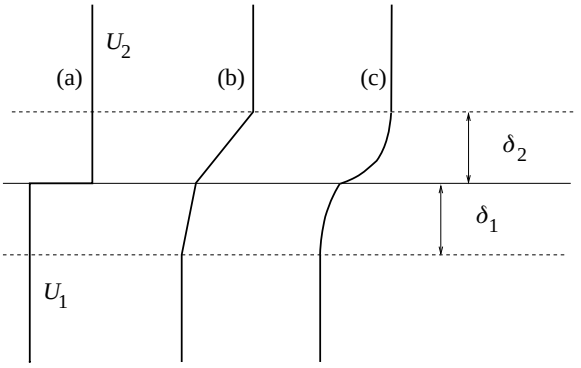


Fig. D.1. The initial or base flows discussed in this section. (a) The simplest profile: sharp velocity jump. (b) A piecewise linear flow with two boundary layers used in early studies. (c) A smooth boundary layers profile, used for comparisons with linear theory and in most full simulations.

Furthermore, we consider an incompressible flow, $\nabla \cdot \mathbf{u} = 0$, and the evolution equation for the density, $D\rho/Dt = 0$, as the density is different in the two fluids (see Equations (2.20) and (2.19)). We also need to consider the equation for the motion of the interface $y = h(x,t)$. We now consider a perturbed two-dimensional flow in the x,y plane and linearize the variables $\mathbf{u}$, ρ, p, and h around the steady-state profile. For example, the velocity field for phase n is

$$\mathbf{u} = (u,v) = \mathbf{u}_0 + \mathbf{u}' = (U_n, 0) + (u', v'), \tag{D.3}$$

where u' and v' are given by (D.1). The interface perturbation $h'(x,t)$ does not depend on y as in (D.1), since it represents the perturbation of a line, $y = h_0 = 0$, and not of a two-dimensional field. We linearize the motion equations around the base flow and drop the subscript n to get

$$\rho_0 \frac{\partial u'}{\partial t} + U \frac{\partial u'}{\partial x} = -\frac{\partial p'}{\partial x}, \tag{D.4}$$

$$\rho_0 \frac{\partial v'}{\partial t} + U \frac{\partial v'}{\partial x} = -\frac{\partial p'}{\partial y} - \rho' g, \tag{D.5}$$

$$\frac{\partial u'}{\partial x} + \frac{\partial v'}{\partial y} = 0, \tag{D.6}$$

$$\frac{\partial \rho'}{\partial t} + U \frac{\partial \rho'}{\partial x} = -v' \frac{\partial \rho_0}{\partial y}, \tag{D.7}$$

$$v'_S = \frac{Dh'}{Dt} = \frac{\partial h'}{\partial t} + U \frac{\partial h'}{\partial x}. \tag{D.8}$$

Because of (D.1), $\partial/\partial x \to (\mathrm{i}k)$, $\partial/\partial t \to (-\mathrm{i}kc)$ and let $\partial/\partial y \to D$, then we solve (D.4), (D.6), and (D.7), respectively, for p', u', and ρ' and substitute the results in (D.5) to get

$$-k^3\rho_0(U-c)v_1 + D\left[\rho_0 k(U-c)Dv_1 - \rho_0 k(DU)v_1\right] = gk\frac{D\rho_0}{U-c}v_1, \tag{D.9}$$

where we have simplified the term $\exp[\mathrm{i}k(x-ct)]$ on both sides. The interface evolution equation, (D.8), can be written as

$$h' = -\frac{\mathrm{i}v'_S}{k(U-c)}. \tag{D.10}$$

Away from the interface, the differential equation (D.9) becomes

$$(D^2 - k^2)v_1(y) = 0. \tag{D.11}$$

The solutions of this equation that satisfy the boundary conditions $v_1 \to 0$ as $|y| \to \infty$ and the uniqueness of the normal displacement h' of the interface at $y = 0$ are

$$v_1(y) = Ak(U_2 - c)\exp(-ky) \tag{D.12}$$

for $y > 0$ and

$$v_1(y) = Ak(U_1 - c)\exp(ky) \tag{D.13}$$

for $y < 0$, where A is a constant. From there, pressure amplitude p_1 and the interface h' may be found by some simple algebra. The dispersion relation $c = c(k)$ is then found by writing the second jump condition of Panel 2.5, without the viscous terms, where we also need the linearized expression of the interface curvature:

$$\kappa = \frac{\partial^2 h'}{\partial x^2}. \tag{D.14}$$

After some straightforward algebra we get

$$\rho_2 k^2(U_2 - c)^2 + \rho_1 k^2(U_1 - c)^2 = gk(\rho_1 - \rho_2) + \sigma k^3. \tag{D.15}$$

We let $\rho_\mathrm{p} = (\rho_1 + \rho_2)$ and $\rho_\mathrm{m} = (\rho_1 - \rho_2)$ and solve the quadratic equation (D.15) for the complex wave speed c as a function of the wavenumber k:

$$c(k) = \frac{1}{\rho_\mathrm{p}}(\rho_1 U_1 + \rho_2 U_2) \pm \left[\frac{\rho_\mathrm{m}}{\rho_\mathrm{p}}\frac{g}{k} + \frac{\sigma k}{\rho_\mathrm{p}} - \frac{\rho_1\rho_2}{\rho_\mathrm{p}^2}(U_1 - U_2)^2\right]^{1/2}. \tag{D.16}$$

D.2.2 Capillary–gravity waves

In the absence of horizontal motion, $U_1 = U_2 = 0$. Then the result is the *dispersion relation* for capillary–gravity waves:

$$c(k) = \left[\frac{\rho_\mathrm{m}}{\rho_\mathrm{p}}\frac{g}{k} + \frac{\sigma k}{\rho_\mathrm{p}}\right]^{1/2}. \tag{D.17}$$

For small k, i.e. large wavelengths, the gravity waves dominate the flow, while for large k or small wavelength the capillary waves dominate the flow. For air and water, the transition occurs around the famed *capillary length* $l_\mathrm{c} = \sqrt{\sigma/(\rho_\mathrm{water} g)}$ which is about 2.7 mm.

D.2.3 The Kelvin–Helmholtz instability

In the absence of gravity and capillarity we get the frequency

$$\omega(k) = \frac{k}{\rho_\mathrm{p}}(\rho_1 U_1 + \rho_2 U_2) \tag{D.18}$$

and the growth rate

$$\gamma(k) = \frac{k}{\rho_\mathrm{p}}(\rho_1\rho_2)^{1/2}|U_1 - U_2|. \tag{D.19}$$

(There is, of course, also a solution with negative γ which has no physical interest.) The solution grows exponentially as $\exp(\gamma t)$ and any initial perturbation of the interface is amplified. The instability so characterized is called the *Kelvin–Helmholtz* instability.

For large values of the ratio ρ_1/ρ_2, as for a liquid–gas mixing layer, the expression of the growth rate simplifies to

$$\gamma(k) = k\left(\frac{\rho_2}{\rho_1}\right)^{1/2}|U_1 - U_2|. \tag{D.20}$$

Gravity, viscosity, or capillarity introduces a radical change in the growth rate curve $\gamma(k)$. Instead of growing indefinitely with k, the growth rate reaches a maximum and then decreases. Expression (D.16) shows that when surface tension and gravity are included the growth rate is zero over some "cutoff" wavenumber k_c. For instance, with surface tension, but with no gravity or viscosity, there is a real growth rate given by

$$\gamma(k) = \left[\frac{\rho_1\rho_2 k^2}{\rho_\mathrm{p}^2}(U_1 - U_2)^2 - \frac{\sigma k^3}{\rho_\mathrm{p}}\right]^{1/2}, \tag{D.21}$$

when

$$k < k_c = \frac{\rho_1 \rho_2 (U_1 - U_2)^2}{\rho_p \sigma}. \tag{D.22}$$

When surface tension dominates and $\rho_1 \gg \rho_2$, one has $k_c \simeq \rho_2 (U_1 - U_2)^2/\sigma$. It is convenient to define the Weber number based on the wavelength λ as

$$\widetilde{\mathrm{We}}(\lambda) = \frac{\rho_2 \lambda (U_1 - U_2)^2}{\sigma}. \tag{D.23}$$

Instability occurs for all wavelengths such that $\lambda > 2\pi/k_c$ or $\widetilde{\mathrm{We}} > 2\pi$. The wavenumber of maximum growth rate is proportional to the cutoff wavenumber; indeed, from (D.21) one finds that maximum γ is obtained for $k_m = 2k_c/3$ or $\widetilde{\mathrm{We}} = 3\pi$. In order to further apply these results it is useful to define a Weber number based on some predefined length scale D, for instance the diameter of the jet or droplet, so that $\mathrm{We}_D = \rho_2 D (U_1 - U_2)^2/\sigma$. The predicted wavelength of maximum instability is then

$$\lambda_m = 3\pi D \mathrm{We}_D^{-1}. \tag{D.24}$$

D.2.4 Effect of thick boundary layers in the inviscid framework

Fluid viscosity causes the diffusion of velocity and changes the discontinuous base flow of Fig. D.1(a) into a profile connecting a gas and a liquid boundary layer of thickness δ_g and δ_l as in Fig. D.1(c). Viscous effects can be introduced in two ways: either by recomputing the stability problem in a simplified way using the same inviscid equations (D.2) but with a base flow $U(y)$ incorporating the boundary layers; or in a consistent way by also including the viscous effects in the flow equations. We describe the consistent approach in the next section and concentrate here on the inviscid framework.

When the boundary layers are modeled as piecewise linear profiles (see Fig. D.1(b)) the inviscid equations can still be solved in closed analytical form (Villermaux, 1998). The resulting growth rate $\gamma(k)$ still has a single maximum. In the case $U_g \gg U_l$, $\rho_g \ll \rho_l$ and $\delta_l = 0$, the maximum growth rate is attained for

$$k_m \sim 1.5 \left(\frac{\rho_g}{\rho_l}\right)^{1/2} \frac{1}{\delta_g}, \qquad \gamma(k_m) \sim \frac{\rho_g}{\rho_l} \frac{U_g}{\delta_g}, \tag{D.25}$$

and thus

$$\lambda_m \sim \frac{4\pi}{3} \left(\frac{\rho_l}{\rho_g}\right)^{1/2} \delta_g. \tag{D.26}$$

However, these results are not a good approximation of the full viscous solution even at large Reynolds numbers, for reasons that are discussed below.

D.3 Viscous theory for the Kelvin–Helmholtz instability

Although the theory described in Section D.2.4 is attractive by its simplicity, it is not realistic enough to allow quantitative predictions. By introducing viscosity we need consistently to consider the Navier–Stokes equations.

Just as for the Euler equation (D.2), the Navier–Stokes equations can be linearized to obtain the so-called Orr–Sommerfeld equations. The linear stability problem will be formulated here in two dimensions, i.e. we assume perturbations about the basic flow in the form of a streamfunction ψ_1 for the liquid and ψ_2 for the gas. Length, velocity, and time are made nondimensional using the gas velocity and gas boundary layer width. Then, $\mathrm{Re} = U_\mathrm{g}\delta_\mathrm{g}/\nu_\mathrm{g}$ and $\mathrm{We} = \rho_\mathrm{g}U_\mathrm{g}^2\delta_\mathrm{g}/\sigma$. The streamwise and cross-stream velocity components u and v are defined by

$$u = \frac{\partial\psi}{\partial y}, \quad v = -\frac{\partial\psi}{\partial x} \tag{D.27}$$

in both phases. Linearization about the basic velocity profile of Fig. D.1(c) implies that the solutions are exponentials in the time and streamwise coordinates, i.e.

$$\psi_1(x,y,t) = \exp(\mathrm{i}k(x-ct))\phi_1(y) \quad (y<0), \tag{D.28}$$

$$\psi_2(x,y,t) = \exp(\mathrm{i}k(x-ct))\phi_2(y) \quad (y>0). \tag{D.29}$$

With this *Ansatz*, we obtain an eigenvalue problem for complex ordinary differential equations in ϕ_1 and ϕ_2, which are coupled at the interface. The problem depends on the real wavenumber k as parameter, and the complex eigenvalues c determine the phase velocity and growth rate of the modes.

The differential equations are (Boeck and Zaleski, 2005)

$$(U_2-c)\left(D^2-k^2\right)\phi_2 - D^2U_2\phi_2 = \frac{1}{\mathrm{i}k\mathrm{Re}}\left(D^2-k^2\right)^2\phi_2, \tag{D.30}$$

$$(U_1-c)\left(D^2-k^2\right)\phi_1 - D^2U_1\phi_1 = \frac{r}{m}\frac{1}{\mathrm{i}k\mathrm{Re}}\left(D^2-k^2\right)^2\phi_1, \tag{D.31}$$

where D once again denotes the derivative with respect to y, $r = \rho_\mathrm{g}/\rho_\mathrm{l}$, and $m = \mu_\mathrm{g}/\mu_\mathrm{l}$. The boundary conditions at the interface are the continuity of the normal and tangential velocity, which yield

$$\phi_1 = \phi_2, \tag{D.32}$$

$$D\phi_1 + DU_1\frac{\phi_1}{c} = D\phi_2 + DU_2\frac{\phi_2}{c}, \tag{D.33}$$

as well as the continuity of the normal and tangential stress, which give

$$-\frac{k^2}{c\,\mathrm{We}}\,\phi_2 \;=\; -\frac{1}{r}\left(cD\phi_1+\phi_1 DU_1\right)-\frac{1}{m}\frac{1}{\mathrm{i}kRe}\left(D^3-3k^2D\right)\phi_1+cD\phi_2+\phi_2 DU_2 + \frac{1}{\mathrm{i}k\mathrm{Re}}\left(D^3-3k^2D\right)\phi_2, \tag{D.34}$$

$$m\left(D^2+k^2+\frac{1}{c}D^2U_2\right)\phi_2=\left(D^2+k^2+\frac{1}{c}D^2U_1\right)\phi_1. \tag{D.35}$$

These equations may be solved numerically, for instance using a Chebyshev polynomial method (Yecko *et al.*, 2002; Gordillo and Perez-Saborid, 2005). The results are discussed in Chapter 9.

References

Abramowitz, M. and Stegun, I. (eds.) (1964). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* National Bureau of Standards Applied Mathematics Series 55. US Government Printing Office, Washington, DC.

Afkhami, S., Zaleski, S., and Bussmann, M. (2009). A mesh-dependent model for applying dynamic contact angles to VOF simulations. *J. Comput. Phys.*, **228**: 5370–5389.

Agarwal, D., Welch, S.W.J., Biswas, G., and Durst, F. (2004). Planar simulation of bubble growth in film boiling in near-critical water using a variant of the VOF method. *ASME J. Heat Transfer*, **126**: 329–338.

Agresar, G., Linderman, J.J., Tryggvason, G., and Powell. K.G. (1998). An adaptive, Cartesian, front-tracking method for the motion, deformation and adhesion of circulating cells. *J. Comput. Phys.*, **43**: 346–380.

Al-Rawahi, N. and Tryggvason, G. (2002). Numerical simulation of dendritic solidification with convection: two-dimensional geometry. *J. Comput. Phys.*, **180**: 471–496.

Al-Rawahi, N. and Tryggvason, G. (2004). Numerical simulation of dendritic solidification with convection: Three-dimensional flow. *J. Comput. Phys.*, **194**: 677–696.

Alexiades, V. and Solomon, A.D. (1993). *Mathematical Modeling of Melting and Freezing Processes*. Hemisphere.

Amsden, A.A. (1993). KIVA-3: a KIVA program with block-structured mesh for complex geometries. Technical report, LA-12503-MS, Los Alamos National Lab., NM.

Antal, S.P., Lahey, R.T., and Flaherty, J.E. (1991). Analysis of phase distribution

in fully developed laminar bubbly two-phase flows. *Int. J. Multiphase Flow*, **15**: 635–652.

Aris, R. (1962). *Vectors, Tensors and Basic Equations of Fluid Mechanics*. Dover.

Arp, P.A., Foister, R.T., and Mason, S.G. (1980). Some electrohydrodynamic effects in fluid dispersions. *Adv. Colloid Interface Sci.*, **12**: 295–356.

Ashgriz, N. and Poo, J.Y. (1991). FLAIR: flux line-segment model for advection and interface reconstruction. *J. Comput. Phys.*, **93**: 449–468.

Ashgriz, N., Shirani, E., and Mostaghini, J. (2005). Interface pressure calculation based on conservation of momentum for front capturing methods. *J. Comput. Phys.*, **203**: 154–175.

Aubert, F., Aulisa, E., Manservisi, S., and Scardovelli, R. (2006). Interface tracking with dynamically-redistributed surface markers in unstructured quadrangular grids. *Comput. Fluids*, **35**: 1332–1343.

Aulisa, E., Manservisi, S., and Scardovelli, R. (2003a). A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows. *J. Comput. Phys.*, **188**: 611–639.

Aulisa, E., Manservisi, S., Scardovelli, R., and Zaleski, S. (2003b). A geometrical area-preserving volume of fluid advection method. *J. Comput. Phys.*, **192**: 355–364.

Aulisa, E., Manservisi, S., and Scardovelli, R. (2004). A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking. *J. Comput. Phys.*, **197**: 555–584.

Aulisa, E., Manservisi, S., Scardovelli, R., and Zaleski, S. (2007). Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry. *J. Comput. Phys.*, **225**: 2301–2319.

Azpitarte, O.E. and Buscaglia, G.C. (2003). Analytical and numerical evaluation of two-fluid model solutions for laminar fully developed bubbly two-phase flows. *Chem. Eng. Sci.*, **58**: 3765–3776.

Baker, G.R., Meiron, D.I., and Orszag, S.A. (1980). Vortex simulation of the Rayleigh–Taylor instability. *Phys. Fluids*, **23**: 1485–1490.

Baker, G.R., Meiron, D.I., and Orszag, S.A. (1982). Generalized vortex methods for free surface flows problems. *J. Fluid Mech.*, **123**: 477.

Bargteil, A.W., Goktekin, T.G., O'Brien, J.F., and Strain, J.A. (2006). A semi-Lagrangian contouring method for fluid simulation. *ACM Trans. Graphics (TOG)*, **25**(1): 19–38.

Bassano, E. and Castagnolo, D. (2003). Marangoni migration of a methanol drop in cyclohexane matrix in a closed cavity. *Microgravity Sci. Technol.*, **XIV**(1): 20–33.

Batchelor, G.K. (1970). *An Introduction to Fluid Dynamics*. Cambridge University Press.

Bayvel, L. and Orzechowski, Z. (1993). *Liquid Atomization*. Taylor and Francis.

Beckermann, C., Diepers, H.-J., Steinbach, I., Karma, A., and Tong, X. (1999). Modeling melt convection in phase-field simulations of solidification. *J. Comput. Phys.*, **154**: 468–496.

Bell, J.B., Colella, P., and Glaz, H.M. (1989). A second-order projection method for the incompressible Navier–Stokes equations. *J. Comput. Phys.*, **85**: 257–283.

Ben Rayana, F., Cartellier, A., and Hopfinger, E. (2006). Assisted atomization of a liquid layer: investigation of the parameters affecting the mean drop size prediction. In *Proc. ICLASS 2006, Aug. 27–Sept.1, Kyoto Japan*. Academic Publication and Printings. (ISBN 4-9902774-1-4).

Berenson, P.J. (1961). Film boiling heat transfer from a horizontal surface. *J. Heat Transfer*, **83**: 351–358.

Bierbrauer, F. (1995). Water drop impact on galvanised steel surfaces. In *Proc. The Splash and Free Surface Workshop* (ed. J.-L. Liow and P. Schwarz), The University of Melbourne, Melbourne, Australia.

Birdsall, C.K. and Langdon, A.B. (1985). *Plasma Physics via Computer Simulation*. McGraw-Hill.

Birkhoff, G. (1954). Taylor instability and laminar mixing. Rep. LA-1862; appendices in Rep. LA-1927, Los Alamos Scientific Laboratory (unpublished).

Blake, J.R. and Gibson, D.C. (1981). Growth and collapse of a vapour cavity near a free surface. *J. Fluid Mech.*, **111**: 123–140.

Blake, T.D. and Shikmurzaev, Y. (2002). Dynamic wetting by liquids of different viscosity. *J. Colloid Interface Sci.*, **253**: 196–202.

Boeck, T. and Zaleski, S. (2005). Viscous versus inviscid instability of two-phase mixing layers with continuous velocity profile. *Phys. Fluids*, **17**: 032106.

Boeck, T., Li, J., López-Pagés, E., Yecko, P., and Zaleski, S. (2007). Ligament formation in sheared liquid–gas layers. *Theor. Comput. Fluid Dyn.*, **21**: 59–76.

Boettinger, W.J., Warren, J.A., Beckermann, C., and Karma, A. (2002). Phase field simulations of solidification. *Annu. Rev. Mater. Res.*, **32**: 163–194.

Bonn, D., Eggers, J., Indekeu, J., Meunier, J., and Rolley, E. (2009). Wetting and spreading. *Rev. Mod. Phys.*, **81**: 739–805.

Boris, J.P. and Book, D.L. (1973). Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, **11**: 38–69.

Bothe, D., Koebe, M., Wielage, K., and Warnecke, H.J. (2003). FEDSM2003-45155: VOF-simulations of mass transfer from single bubbles and bubble chains rising in aqueous solutions. In *Proc. 2003 ASME Joint U.S.–European Fluids Eng. Conf.*, Honolulu, USA.

Bothe, D., Schmidtke, M., and Warnecke, H.-J. (2006). VOF-simulation of the lift force for single bubbles in a simple shear flow. *Chem. Eng. Technol.*, **29**: 1048–1053.

Boulton-Stone, J.M. and Blake, J.R. (1993). Gas-bubbles bursting at a free-surface. *J. Fluid Mech.*, **254**: 437–466.

Brackbill, J.U. and Ruppel, H.M. (1986). FLIP: a method for adaptively zoned, particle-in-cell calculations in two dimensions. *J. Comput. Phys.*, **65**: 314–343.

Brackbill, J.U., Kothe, D.B., and Zemach, C. (1992). A continuum method for modeling surface tension. *J. Comput. Phys.*, **100**: 335–354.

Bradley, S.G. and Stow, C.D. (1978). Collision between liquid drops. *Phil. Trans. R. Soc. Lond. A*, **287**: 635.

Brady, J.F. and Bossis, G. (1988). Stokesian dynamics. *Annu. Rev. Fluid Mech.*, **20**: 111–157.

Brenner, M. and Gueyffier, D. (1999). On the bursting of viscous films. *Phys. Fluids*, **11**: 737–739.

Briggs, W.L. (1987). *A Multigrid Tutorial*. SIAM Books, Philadelphia.

Bröder, D. and Sommerfeld, M. (2007). Planar shadow image velocimetry for the analysis of the hydrodynamics in bubbly flows. *Meas. Sci. Technol.*, **18**: 2513–2528.

Bunner, B. and Tryggvason, G. (2002a). Dynamics of homogeneous bubbly flows. Part 1. Rise velocity and microstructure of the bubbles. *J. Fluid Mech.*, **466**: 17–52.

Bunner, B. and Tryggvason, G. (2002b). Dynamics of homogeneous bubbly flows. Part 2. Velocity fluctuations. *J. Fluid Mech.*, **466**: 53–84.

Bunner, B. and Tryggvason, G. (2003). Effect of bubble deformation on the stability and properties of bubbly flows. *J. Fluid Mech.*, **495**: 77–118.

Carey, V.P. (2007). *Liquid Vapor Phase Change Phenomena: An Introduction to the Thermophysics of Vaporization and Condensation Processes in Heat Transfer Equipment*, 2nd edition, Taylor & Francis.

Cartellier, A. and Rivière, N. (2001). Bubble-induced agitation and microstructure in uniform bubbly flows at small to moderate particle. *Phys. Fluids*, **13**: 2165–2181.

Ceniceros, H.D. and Roma, A.M. (2005). A multi-phase flow method with a fast, geometry-based fluid indicator. *J. Comput. Phys.*, **205**: 391–400.

Cha, P.-H., Yeon, D.-H., and Yoon, J.-K. (2005). Phase-field model for multicomponent alloy solidification. *J. Cryst. Growth*, **274**: 281–293.

Chahine, G.L. and Duraiswami, R. (1992). Dynamic interactions in a multibubble cloud. *ASME J. Fluids Eng.*, **114**(4): 680–686.

Chan, R.K.-C. and Street, R.L. (1970). A computer study of finite-amplitude water waves. *J. Comput. Phys.*, **6**: 68–94.

Chandra, S. and Avedisian, C.T. (1991). On the collision of a droplet with a solid surface. *Proc. R. Soc. London, Ser. A: Math. Phys. Sci.*, **432**(1884): 13–41.

Chandrasekhar, S. (1961). *Hydrodynamic and Hydromagnetic Stability*. Oxford University Press.

Chang, Y.C., Hou, T.Y., Merriman, B., and Osher, S. (1996). Temporal evolution of periodic disturbances in two-layer Couette flow. *J. Comput. Phys.*, **124**: 449–464.

Chapman, R.B. and Plesset, M.S. (1972). Nonlinear effects in the collapse of a nearly spherical cavity in a liquid. *Trans. ASME, J. Basic Eng.*, **94**: 142.

Che, J., Ceccio, S.L., and Tryggvason, G. (2004). Computations of structures formed by the solidification of impinging molten metal drops. *J. Appl. Math. Model.*, **28**: 127–144.

Chen, S., Johnson, D.B., Raad, P.E., and Fadda, D. (1997a). The surface marker and micro cell method. *Int. J. Numer. Meth. Fluids*, **25**: 749–778.

Chen, S., Merriman, B., Osher, S., and Smereka, P. (1997b). A simple level set method for solving Stefan problems. *J. Comput. Phys.*, **135**: 8–29.

Chern, I.-L., Glimm, J., McBryan, O., Plohr, B., and Yaniv, S. (1986). Front tracking for gas dynamics. *J. Comput. Phys.*, **62**: 83–110.

Choi, H.G. and Joseph, D.D. (2001). Fluidization by lift of 300 circular particles in plane Poiseuille flow by direct numerical simulation. *J. Fluid Mech.*, **438**: 101–128.

Chorin, A.J. (1968). Numerical solutions of the Navier–Stokes equations. *Math. Comput.*, **22**: 745–762.

Chorin, A.J. (1967). A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys.*, **2**: 12.

Christiansen, J.P. (1973). Numerical simulation of hydrodynamics by the method of point vortices. *J. Comput. Phys.*, **13**: 363–379.

Clift, R., Grace, J.R., and Weber, M.E. (1978). *Bubbles, Drops, and Particles*. Academic Press.

Coriell, S.R. and McFadden, G.B. (1993). Morphological stability. In *Handbook of Crystal Growth*, vol. 1B (ed. D.T.J. Hurle), Elsevier, pp. 785–857.

Cortelezzi, L. and Prosperetti, A. (1981). Small amplitude waves on the surface of a layer of a viscous liquid. *Q. Appl. Math.*, **38**: 375–388.

Cortez, R. and Minion, M. (2000). The blob projection method for immersed boundary problems. *J. Comput. Phys.*, **161**: 428–453.

Coward, A.V., Renardy, Y.Y., Renardy, M., and Richards, J.R. (1997). Temporal evolution of periodic disturbances in two-layer Couette flow. *J. Comput. Phys.*, **132**: 346–361.

Crapper, G.D., Dombrowski, N., Jepson, W.P., and Pyott, G.A.D. (1973). A note on the growth of Kelvin–Helmholtz waves on thin liquid sheets. *J. Fluid. Mech.*, **57**: 671–672.

Cristini, V., Bławzdziewicz, J., and Loewenberg, M. (1998). Drop breakup in three-dimensional viscous flows. *Phys. Fluids.*, **10**: 1781–1784.

Cummins, S.J., Francois, M.M., and Kothe, D.B. (2005). Estimating curvature from volume fractions. *Comput. Struct.*, **83**: 425–434.

Daly, B.J. (1969a). A technique for including surface tension effects in hydrodynamic calculations. *J. Comput. Phys.*, **4**: 97–117.

Daly, B.J. (1969b). Numerical study of the effect of surface tension on interface instability. *Phys. Fluids*, **12**: 1340–1354.

Daly, B.J. and Pracht, W.E. (1968). Numerical study of density-current surges. *Phys. Fluids*, **11**: 15–30.

Dandy, D.S. and Leal, G.L. (1989). Buoyancy-driven motion of a deformable drop through a quiescent liquid at intermediate Reynolds numbers. *J. Fluid Mech.*, **208**: 161–192.

Davidson, M.R. and Rudman, M. (2002). Volume-of-fluid calculation of heat or mass transfer across deforming interfaces in two-fluid flow. *Numer. Heat Transfer Part B Fund.*, **41**: 291–308.

Davies, J.T. and Rideal, E.K. (1966). *Interfacial Phenomena*. Academic Press.

Davis, R.H., Schonberg, J.A., and Rallison, J.M. (1989). The lubrication force between two viscous drops. *Phys. Fluids A: Fluid Dyn.*, **1**: 77–81.

de Gennes, P.G. (1985). Wetting: statics and dynamics. *Rev. Mod. Phys.*, **57**: 827–863.

de Gennes, P.G., Quere, D., and Brochard, F. (2003). *Capillarity and Wetting Phenomena: Drops, Bubbles, Pearls, Waves*. Springer Verlag.

de Josselin de Jong, G. (1960). Singularity distribution for the analysis of multiple-fluid flow through porous media. *J. Geophys. Res.*, **65**: 3739–3758.

DeBar, R. (1974). Fundamentals of the KRAKEN code. Technical Report UCIR-760, LLNL.

Deckwer, W.-D. (1992). *Bubble Column Reactors*. Wiley.

Deen, N.G., van Sint Annaland, M., and Kuipers, J.A.M. (2004). Multi-scale modeling of dispersed gas–liquid two-phase flow. *Chem. Eng. Sci.*, **59**: 1853–1861.

DeGregoria, A.J. and Schwartz, L.W. (1985). Finger breakup in Hele–Shaw cells. *Phys. Fluids*, **28**: 2313.

Delale, C.F., Nas, S., and Tryggvason, G. (2005). Direct numerical simulations of shock propagation in bubbly liquids. *Phys. Fluids*, **17**: 121705.

Dhir, V.K. (1998). Boiling heat transfer. *Annu. Rev. Fluid Mech.*, **30**: 365–401.

Ding, H., Spelt, P.D.M., and Shu, C. (2007). Diffuse interface model for incompressible two-phase flows with large density ratios. *J. Comput. Phys.*, **226**: 2078–2095.

Dombrowski, N. and Johns, W.R. (1963). The aerodynamic instability and disintegration of viscous liquid sheets. *Chem. Eng. Sci.*, **18**: 203–214.

Drew, D.A. (1983). Mathematical modeling of two-phase flow. *Annu. Rev. Fluid Mech.*, **15**: 261–291.

Drew, D.A. and Passman, S.L. (1999). *Theory of Multicomponent Fluids*. Springer.

Drumright-Clarke, M.A. and Renardy, Y. (2004). The effect of insoluble surfactant at dilute concentration on drop breakup under shear with inertia. *Phys. Fluids*, **16**: 14–21.

Druzhinin, O.A. and Elghobashi, S.E. (2001). Direct numerical simulation of a three-dimensional spatially-developing bubble-laden mixing layer with two-way coupling. *J. Fluid Mech.*, **429**: 23–61.

Du, J., Fix, B., Glimm, J., Jia, X., Li, X., Li, Y., and Wu, L. (2006). A simple package for front tracking. *J. Comput. Phys.*, **213**: 613–628.

Duchemin, L., Popinet, S., Josserand, C., and Zaleski, S. (2002). Jet formation in bubbles bursting at a free surface. *Phys. Fluids*, **14**(9): 3000–3008.

Dussan, V.E.B. (1979). On the spreading of liquid on solid surfaces: static and dynamic contact lines. *Annu. Rev. Fluid Mech.*, **11**: 371–400.

E, W. and Engquist, B. (2003). The heterogeneous multiscale methods. *Commun. Math. Sci.*, **1**: 87–133.

Edgerton, H. (1987). *Stopping Time*. Harry N. Abrams, New York.

Edwards, D.A., Brenner, H., and Wasan, D.T. (1961). *Interfacial Transport Processes and Rheology*. Butterworth.

Eggers, J. (1993). Universal pinching of 3D axisymmetric free-surface flow. *Phys. Rev. Lett.*, **71**: 3458–3461.

Eggers, J. and Dupont, T.D. (1994). Drop formation in a one-dimensional approximation of the Navier–Stokes equation. *J. Fluid Mech.*, **262**: 205–221.

Eggers, J. and Villermaux, E. (2008). Physics of liquid jets. *Rep. Prog. Phys.*, **71**: 036601.

Ellingsen, K. and Risso, F. (2001). On the rise of an ellipsoidal bubble in water: oscillatory paths and liquid-induced velocity. *J. Fluid Mech.*, **440**: 235–268.

Ervin, E.A. and Tryggvason, G. (1997). The rise of bubbles in a vertical shear flow. *ASME J. Fluid Eng.*, **119**: 443–449.

Esmaeeli, A. and Tryggvason, G. (1996). An inverse energy cascade in two-dimensional, low Reynolds number bubbly flows. *J. Fluid Mech.*, **314**: 315–330.

Esmaeeli, A. and Tryggvason, G. (1998). Direct numerical simulations of bubbly flows. Part I. Low Reynolds number arrays. *J. Fluid Mech.*, **377**: 313–345.

Esmaeeli, A. and Tryggvason, G. (1999). Direct numerical simulations of bubbly flows. Part II. Moderate Reynolds number arrays. *J. Fluid Mech.*, **385**: 325–358.

Esmaeeli, A. and Tryggvason, G. (2004a). A front tracking method for computations of boiling in complex geometries. *Int. J. Multiphase Flow*, **30**: 1037–1050.

Esmaeeli, A. and Tryggvason, G. (2004b). Computations of film boiling. Part I: numerical method. *Int. J. Heat Mass Transfer*, **47**: 5451–5461.

Esmaeeli, A. and Tryggvason, G. (2004c). Computations of film boiling. Part II: multi-mode film boiling. *Int. J. Heat Mass Transfer*, **47**: 5463–5476.

Esmaeeli, A. and Tryggvason, G. (2005). A direct numerical simulation study of the buoyant rise of bubbles at $O(100)$ Reynolds number. *Phys. Fluids*, **17**: 093303.

Fadlun, E.A., Verzicco, R., Orlandi, P., and Mohd-Yusof, J. (2000). Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *J. Comput. Phys.*, **161**: 35–60.

Farmer, E.E. (1973). Relative detachability of soil particles by simulated rainfall. *Soil Sci. Soc. Am. J.*, **37**: 629–633.

Fauci, L.J. and Dillon, R. (2006). Biofluidmechanics of reproduction. *Annu. Rev. Fluid Mech.*, **38**: 371–394.

Fedkiw, R., Aslam, T., Merriman, B., and Osher, S. (1999). A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comput. Phys.*, **152**: 457–492.

Feng, J., Hu, H.H., and Joseph, D.D. (1994). Direct simulation of initial value problems for the motion of solid bodies in a Newtonian fluid. Part 1. Sedimation. *J. Fluid Mech.*, **261**: 95–134.

Feng, J., Hu, H.H., and Joseph, D.D. (1995). Direct simulation of initial value problems for the motion of solid bodies in a Newtonian fluid. Part 2. Couette and Poiseuille flows. *J. Fluid Mech.*, **277**: 271–301.

Fernandez, A. (2008a). Response of an emulsion of leaky dielectric drops immersed in a simple shear flow: drops more conductive than the suspending fluid. *Phys. Fluids*, **20**: 043303.

Fernandez, A. (2008b). Response of an emulsion of leaky dielectric drops immersed in a simple shear flow: drops less conductive than the suspending fluid. *Phys. Fluids*, **20**: 043304.

Fernandez, A., Che, J., Ceccio, S.L., and Tryggvason, G. (2005). The effects of electrostatic forces on the distribution of drops in a channel flow – two-dimensional oblate drops. *Phys. Fluids*, **17**: 093302.

Foote, G.B. (1973). A numerical method for studying liquid drop behavior: simple oscillations. *J. Comput. Phys.*, **11**: 507–530.

Foote, G.B. (1975). The water drop rebound problem: dynamics of collision. *J. Atmos. Sci.*, **32**: 390–402.

Fortes, A., Joseph, D.D., and Lundgren, T. (1987). Nonlinear mechanics of fluidization of beds of spherical particles. *J. Fluid Mech.*, **177**: 467–483.

Foster, N. and Fedkiw, R. (2001). Practical animation of liquids. In *Proc. 28th Annual Conf. on Computer Graphics and Interactive Techniques*. ACM Press, New York, NY, pp. 23–30.

Francois, M.M., Cummins, S.J., Dendy, E.D., Kothe, D.B., Sicilian, J.M., and Williams, M.W. (2006). A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework. *J. Comput. Phys.*, **213**(1): 141–173.

Frumkin, A. and Levich, V.G. (1947). O vliyanii poverkhnostno-aktivnikh veshestv na dvizhenie na granitse zhidkikh sred. *Zh. Fiz. Khim.*, **21**: 1183–1204.

Fukai, J., Shiiba, Y., Yamamoto, T., Miyatake, O., Poulikakos, D., Megaridis, C.M., and Zhao, Z. (1995). Wetting effects on the spreading of a liquid droplet colliding with a flat surface: experiment and modeling. *Phys. Fluids*, **7**: 236–247.

Furusaki, S., Fan, L.-S., and Garside, J. (2001). *The Expanding World of Chemical Engineering* (2nd edition). Taylor and Francis.

Fuster, D., Agbaglah, G., Josserand, C., Popinet, S., and Zaleski, S. (2009a). Numerical simulation of droplets, bubbles and waves: state of the art. *Fluid Dyn.* Res. **41**: 065001.

Fuster, D., Bagué, A., Boeck, T., Le Moyne, L., Leboissetier, A., Popinet, S., Ray, P., Scardovelli, R., and Zaleski, S. (2009b). Simulation of primary atomization with an octree adaptive mesh refinement and VOF method. *Int. J. Multiphase Flow*, **35**: 550–565.

Gao, P., Yin, Z., and Hu, W. (2008). Thermocapillary motion of droplets at large Marangoni numbers. *Adv. Space Res.*, **41**(12): 2101–2106.

Ge, Y. and Fan, L.-S. (2006). Three-dimensional direct numerical simulation for film-boiling contact of moving particle and liquid droplet. *Phys. Fluids*, **18**: 117104.

Gerlach, D., Tomar, G., Biswas, G., and Durst, F. (2006). Comparison of volume-of-fluid methods for surface tension-dominant two-phase flows. *Int. J. Heat Mass Transfer*, **49**: 740–754.

Gibou, F., Chen, L., Nguyen, D., and Banerjee, S. (2007). A level set based sharp interface method for the multiphase incompressible Navier–Stokes equations with phase change. *J. Comput. Phys.*, **222**: 536–555.

Gilmanov, A., Sotiropoulos, F., and Balaras, E. (2003). A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on Cartesian grids. *J. Comput. Phys.*, **191**: 660–669.

Glicksman, M.E., Coriell, S.R., and McFadden, G.B. (1986). Interaction of flows with the crystal–melt interface. *Annu. Rev. Fluid Mech.*, **18**: 307–335.

Glimm, J. and McBryan, O. (1985). A computational model for interfaces. *Adv. Appl. Math.*, **6**: 422–435.

Glimm, J., Marchesin, D., and McBryan, O. (1981). A numerical method for two phase flow with an unstable interface. *J. Comput. Phys*, **39**: 179–200.

Glimm, J., Grove, J.W., Li, X.L., Oh, W., and Sharp, D.H. (2001). A critical analysis of Rayleigh–Taylor growth rates. *J. Comput. Phys.*, **169**: 652–677.

Glowinski, R., Pan, T.W., Helsa, T.I., Joseph, D.D., and Periaux, J. (2001). A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: application to particulate flow. *J. Comput. Phys.*, **169**: 363–426.

Goldstein, D., Handler, R., and Sirovich, L. (1993). Modeling a no-slip flow boundary with an external force field. *J. Comput. Phys.*, **105**: 354–366.

Gordillo, J.M. and Perez-Saborid, M. (2005). On the first wind atomization regime. *J. Fluid. Mech*, **541**: 1–20.

Guet, S., Ooms, G., Oliemans, R.V.A., and Mudde, R.F. (2004). Bubble size effect on low liquid input drift-flux parameters. *Chem. Eng. Sci.*, **59**: 3315–3329.

Guet, S., Ooms, G., and Oliemans, R.V.A. (2005). Simplified two-fluid model for gas-lift efficiency predictions. *AIChe J.*, **51**: 1885–1896.

Gueyffier, D. and Zaleski, S. (1998). Formation de digitations lors de l'impact d'une goutte sur un film liquide. *C. R. Acad. Sci. Paris sér. II b*, **326**: 839–844.

Gueyffier, D., Nadim, A., Li, J., Scardovelli, R., and Zaleski, S. (1999). Volume of fluid interface tracking with smoothed surface stress methods for three-dimensional flows. *J. Comput. Phys.*, **152**: 423–456.

Hadamard, J. (1911). Mouvement permanent lent d'une sphere liquide et visqueuse dans un liquide visqueux. *C. R. Acad. Sci. Paris*, **152**: 1735–1738.

Hammerlin, G. and Hoffmann, K.-H. (1991). *Numerical Mathematics*. Springer.

Harlow, F.H. (1964). The particle-in-cell computing method for fluid dynamics. *Methods Comput. Phys.*, **3**: 319–343.

Harlow, F.H. and Fromm, J.E. (1965). Computer experiments in fluid dynamics. *Sci. Am.*, **212**: 104.

Harlow, F.H. and Shannon, J.P. (1967). The splash of a liquid drop. *J. Appl. Phys.*, **38**: 3855–3866.

Harlow, F.H. and Welch, J.E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. *Phys. Fluids*, **8**: 2182–2189.

Harlow, F.H. and Welch, J.E. (1966). Numerical study of large-amplitude free-surface motions. *Phys. Fluids*, **9**: 842–851.

Harvie, D.J.E. and Fletcher, D.F. (2000). A new volume of fluid advection algorithm: the Stream scheme. *J. Comput. Phys.*, **162**: 1.

Hervet, H. and de Gennes, P.G. (1984). Dynamique du mouillage: films précurseurs sur solide 'sec'. *C. R. Acad. Sci. Paris, Ser. II*, **299**: 499–503.

Hinch, E.J. (1984). A note on the mechanism of the instability at the interface between two shearing fluids. *J. Fluid Mech.*, **144**: 463–465.

Hirt, C.W. and Nichols, B.D. (1981). Volume of fluid (VOF) method for the dynamics of free boundaries. *J. Comput. Phys.*, **39**: 201–226.

Hirt, C.W., Cook, J.L., and Butler, T.D. (1970). A Lagrangian method for calculating the dynamics of an incompressible fluid with a free surface. *J. Comput. Phys.*, **5**: 103–124.

Hockney, R.W. and Eastwood, J.W. (1981). *Computer Simulation Using Particles*. McGraw-Hill.

Hooper, A.P. and Boyd, W.G.C. (1983). Shear-flow instability at the interface between two viscous fluids. *J. Fluid Mech.*, **128**: 507–528.

Hou, T.Y., Lowengrub, J.S., and Shelley, M.J. (2001). Boundary integral methods for multicomponent fluids and multiphase materials. *J. Comput. Phys.*, **169**: 302–362.

Hu, H.H. (1996). Direct simulation of flows of solid–liquid mixtures. *Int. J. Multiphase Flow*, **22**: 335–352.

Hu, H.H., Patankar, N.A., and Zhu, M.Y. (2001). Direct numerical simulations of fluid-solid systems using arbitrary-Lagrangian–Eularian techniques. *J. Comput. Phys.*, **169**: 427–462.

Hua, J. and Lou, J. (2007). Numerical simulation of bubble rising in viscous liquid. *J. Comput. Phys.*, **222**: 769–795.

Hua, J., Lim, L.K., and Wang, C.-H. (2008). Numerical simulation of deformation/motion of a drop suspended in viscous liquids under influence of steady electric fields. *Phys. Fluids*, **20**: 113302.

Huh, C. and Scriven, L. (1971). Hydrodynamic model of steady movement of a solid/liquid/fluid contact line. *J. Coll. Interf. Sci*, **35**: 85.

Ishii, M. (1987). *Interfacial Area Modeling*, Vol. 3, McGraw-Hill, Chapter 3, pp. 31–62.

Ishii, M. and Zuber, N. (1979). Drag coefficient and relative velocity in bubbly, droplet or particulate flows. *AIChE J.*, **25**: 843–855.

Issa, R.I. (1985). Solution of the implicitly discretised fluid flow equations by operator-splitting. *J. Comput. Phys.*, **62**: 40–65.

Jacqmin, D. (1999). Calculation of two-phase Navier–Stokes flows using phase-field modeling. *J. Comput. Phys.*, **155**: 96–127.

Jacqmin, D. (2000). Contact-line dynamics of a diffuse fluid interface. *J. Fluid Mech.*, **402**: 57–88.

James, A.J. and Lowengrub, J. (2004). A surfactant-conserving volume-of-fluid method for interfacial flows with insoluble surfactant. *J. Comput. Phys.*, **201**: 685–722.

Jamet, D., Lebaigue, O., Coutris, N., and Delhaye, J.M. (2001). Feasibility of using the second gradient theory for the direct numerical simulations of liquid–vapor flows with phase-change. *J. Comput. Phys.*, **169**: 624–651.

Jan, Y.-J. (1994). Computational studies of bubble dynamics. PhD thesis, The University of Michigan.

Jeong, J.-H., Goldenfield, N., and Dantzig, J.A. (2001). Phase field model for three-dimensional dendritic growth with fluid flow. *Phys. Rev. E*, **64**: 041602.

Johansson, B.C.V. (1993). Boundary conditions for open boundaries for the incompressible Navier–Stokes equation. *J. Comput. Phys.*, **105**: 233–251.

Johnson, A. A. and Tezduyar, T.E. (1997). 3D simulation of fluid–particle interactions with the number of particles reaching 100. *Comput. Methods Appl. Mech. Eng.*, **145**: 301–321.

Josserand, C. and Zaleski, S. (2003). Droplet splashing on a thin liquid film. *Phys. Fluids*, **15**: 1650–1657.

Juric, D. and Tryggvason, G. (1998). Computations of boiling flows. *Int. J. Multiphase Flow*, **24**: 387–410.

Kadioglu, S.Y. and Sussman, M. (2008). Adaptive solution techniques for simulating underwater explosions and implosions. *J. Comput. Phys.*, **227**: 2083–2104.

Kajishima, T. and Takiguchi, S. (2002). Interaction between particle clusters and fluid turbulence. *Int. J. Heat Fluid Flow*, **23**: 639–646.

Kanai, A. and Miyata, H. (2001). Direct numerical simulation of wall turbulent flows with microbubbles. *Int. J. Numer. Methods Fluids*, **35**: 593–615.

Kang, I.S. and Leal, L.G. (1987). Numerical solution of axisymmetric, unsteady free-boundary problems at finite Reynolds number. I. Finite-difference scheme and its applications to the deformation of a bubble in a uniaxial straining flow. *Phys. Fluids*, **30**: 1929–1940.

Karma, A. and Rappel, W.J. (1997). Phase-field simulation of three-dimensional dendrites: is microscopic solvability theory correct? *J. Cryst. Growth*, **174**: 54–64.

Karma, A. and Rappel, W.-J. (1998). Quantitative phase-field modeling of dendritic growth in two and three dimensions. *Phys. Rev. E*, **57**: 4323–4349.

Kashinsky, O.N. and Randin, V.V. (1999). Downward bubbly gas–liquid flow in a vertical pipe. *Int. J. Multiphase Flow*, **25**: 109–138.

Kawamura, T. and Kodama, Y. (2002). Numerical simulation method to resolve interactions between bubbles and turbulence. *Int. J. Heat Fluid Flow*, **23**: 627–638.

Keller, J.B. and Miksis, M.J. (1983). Surface tension driven flows. *SIAM J. Appl. Math.*, **43**(2): 268–277.

Kermeen, R.W. (1956). Water tunnel tests of NACA 4412 and Walchner profile 7 hydrofoils in non-cavitating and cavitating flows. Technical report, California Institute of Technology Hydro. Lab. Rep. 47-5.

Khinast, J.G., Koynov, A.A., and Leib, T.M. (2003). Reactive mass transfer at gas–liquid interfaces: impact of micro-scale fluid dynamics on yield and selectivity of liquid-phase cyclohexane oxidation. *Chem. Eng. Sci.*, **58**: 3961–3971.

Kim, J. and Moin, P. (1985). Application of a fractional step method to incompressible Navier–Stokes equations. *J. Comput. Phys.*, **59**: 308–323.

Kim, J., Kim, D., and Choi, H. (2001). An immersed-boundary finite-volume method for simulations of flow in complex geometries. *J. Comput. Phys.*, **171**: 132–150.

Kitagawa, A., Sugiyama, K., and Murai, Y. (2004). Experimental detection of bubble–bubble interactions in a wall-sliding bubble swarm. *Int. J. Multiphase Flow*, **30**: 1213–1234.

Klimenko, V.V. (1981). Film boiling on a horizontal plate – new correlation. *Int. J. Heat Mass Transfer*, **24**: 69–79.

Kobayashi, R. (1992). Simulations of three-dimensional dendrites. In *Pattern Formation in Complex Dissipative Systems* (ed. S. Kai). World Scientific, Singapore, pp. 121–128.

Kobayashi, R. (1993). Modeling and numerical simulations of dendritic crystal-growth. *Phys. D*, **63**: 410–423.

Koplik, J., Banavar, J.R., and Willemsen, J.F. (1988). Molecular dynamics of Poiseuille flow and moving contact lines. *Phys. Rev. Lett.*, **60**: 1282–1285.

Koynov, A., Khinast, J.G., and Tryggvason, G. (2005). Mass transfer and chemical reactions in bubble swarms with dynamic interfaces. *AIChE. J.*, **51**: 2786–2800.

Kuo, T.C., Pan, C., and Chieng, C.C. (1997). Eulerian–Lagrangian computations on phase distribution of two-phase bubbly flows. *Int. J. Numer. Methods Fluids*, **24**: 579–593.

Ladd, A.J.C. (1993). Dynamical simulations of sedimenting spheres. *Phys. Fluids A*, **5**: 299–310.

Lafaurie, B., Nardone, C., Scardovelli, R., Zaleski, S., and Zanetti, G. (1994). Modelling merging and fragmentation in multiphase flows with SURFER. *J. Comput. Phys.*, **113**: 134–147.

Lai, M.-C. and Peskin, C.S. (2000). An immersed boundary method with formal second order accuracy and reduced numerical viscosity. *J. Comput. Phys.*, **160**: 705–719.

Lance, M. and Bataille, J. (1991). Turbulence in the liquid phase of a uniform bubbly air–water flow. *J. Fluid Mech.*, **222**: 95–118.

Lasheras, J.C. and Hopfinger, E.J. (2000). Liquid jet instability and atomization in a coaxial gas stream. *Annu. Rev. Fluid Mech.*, **32**: 275–308.

Leboissetier, A. (2002). Simulation numérique directe de l'atomisation primaire d'un jet liquide à haute vitesse. Technical report, Université de Paris 6. Thèse de doctorat.

Lee, J. and Pozrikidis, C. (2006). Effect of surfactants on the deformation of drops and bubbles in Navier–Stokes flow. *Comput. Fluids*, **35**: 43–60.

Lee, L. and LeVeque, R.J. (2003). An immersed interface method for incompressible Navier–Stokes equations. *SIAM J. Sci. Comput.*, **25**: 832–856. .

Lee, R.C. and Nydahl, J.E. (1989). Numerical calculations of bubble growth in nucleate boiling from inception through departure. *J. Heat Transfer*, **111**: 474–479.

Lee, Y.-W., Ananth, R., and Gill, W.N. (1993). Selection of length scale in unconstrained dendritic growth with convection in the melt. *J. Cryst. Growth*, **132**: 226–230.

Lefebvre, A.H. (1989). *Atomization and Sprays*. Taylor and Francis.

Lenard, P. (1892). Über die Electricität der Wasserfälle. *Ann. Phys.*, **46**: 584–636.

Leonard, B.P. (1979). A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Comput. Methods Appl. Mech. Eng.*, **19**: 59–98.

Li, J. (1995). Calcul d'interface affine par morceaux (Piecewise linear interface calculation). *C. R. Acad. Sci. Paris, Sér. IIb (Paris)*, **320**: 391–396.

Li, J. (1996). Résolution numérique de l'équation de Navier–Stokes avec reconnection d'interfaces. Méthode de suivi de volume et application à l'atomisation. Technical report, Université de Paris 6. Thèse de doctorat.

Liow, J.L., Morton, D.E., Guerra, E., and Gray, N.B. (1996). Dynamics of splash formation in gas injected systems. In *The Howard Worner Int. Symp. on Injection in Pyrometallurgy, Melbourne*, pp. 175–190.

Lister, J. and Zhang, W.W. (1999). Similarity solutions for van der Waals rupture of a thin film on a solid substrate. *Phys. Fluids*, **11**: 2454–2462.

Liu, H., Krishnan, S., Marella, S., and Udaykumar, H.S. (2005). Sharp interface Cartesian grid method II: a technique for simulating droplet interactions with surfaces of arbitrary shape. *J. Comput. Phys.*, **210**: 32–54.

Liu, T.J. (1997). Investigation of the wall shear stress in vertical bubbly flow under different bubble size conditions. *Int. J. Multiphase Flow*, **23**: 1085–1109.

Liu, T.J. and Bankoff, S.G. (1993). Structure of air-water bubbly flow in a vertical pipe – I. Liquid mean velocity and turbulence measurements. *Int. J. Heat Mass Transfer*, **36**: 1049–1060.

Loewenberg, M. and Hinch, E.J. (1996). Numerical simulation of a concentrated emulsion in shear flow. *J. Fluid Mech.*, **321**: 395–419.

Longuet-Higgins, M.S. and Cokelet, E.D. (1976). The deformation of steep surface waves on water. *Proc. R. Soc. London Ser. A*, **358**: 1.

López, J., Hernandez, J., Gómez, P., and Faura, F. (2004). A volume of fluid method based on multidimensional advection and spline interface reconstruction. *J. Comput. Phys.*, **195**: 718–742.

Lopez De Bertodano, M., Lahey, R.T., Jr., and Jones, O.C. (1987). Development of a k–ε model for bubbly two-phase flow. *J. Fluids Eng.*, **13**: 327–343.

Lopez De Bertodano, M., Lahey, R.T., Jr., and Jones, O.C. (1994). Phase distribution in bubbly two-phase flow in vertical ducts. *Int. J. Multiphase Flow*, **20**: 805–818.

Louge, M.Y. (1994). Computer simulations of rapid granular flows of spheres interacting with a flat, frictional boundary. *Phys. Fluids*, **6**: 2253–2269.

Lu, J. and Tryggvason, G. (2006). Numerical study of turbulent bubbly downflows in a vertical channel. *Phys. Fluids*, **18**: 103302.

Lu, J. and Tryggvason, G. (2007). Effect of bubble size in turbulent bubbly downflow in a vertical channel. *Chem. Eng. Sci.*, **62**: 3008–3018.

Lu, J. and Tryggvason, G. (2008). Effect of bubble deformability in turbulent bubbly upflow in a vertical channel. *Phys. Fluids*, **20**: 040701.

Lu, J., Fernandez, A., and Tryggvason, G. (2005a). The effect of bubbles on the wall shear in a turbulent channel flow. *Phys. Fluids*, **17**: 095102.

Lu, J., Biswas, S., and Tryggvason, G. (2006). A DNS study of laminar bubbly flows in a vertical channel. *Int. J. Multiphase Flow*, **32**: 643–660.

Lu, Y., Beckermann, C., and Ramirez, J.C. (2005b). Three-dimensional phase field simulations of the effect of convection on free dendritic growth. *J. Cryst. Growth*, **280**: 320–334.

Luo, R., Pan, X.H., and Yang, X.Y. (2003). Laminar light particle and liquid two-phase flows in a vertical pipe. *Int. J. Multiphase Flow*, **29**: 603–620.

MacIntyre, F. (1972). Flow patterns in breaking bubbles. *J. Geophys. Res.*, **77**(27): 5211–5228.

MacNeice, P., Olson, K.M., Mobarry, C., de Fainchtein, R., and Packer, C. (2000). PARAMESH: a parallel adaptive mesh refinement community toolkit. *Comput. Phys. Commun.*, **126**: 330–354.

Marella, S., Krishnan, S., Liu, H., and Udaykumar, H.S. (2005). Sharp interface Cartesian grid method I: an easily implemented technique for 3D moving boundary computations. *J. Comput. Phys.*, **214**: 1–31.

Marmottant, P. and Villermaux, E. (2001). Ligament mediated drop formation. *Phys. Fluids*, **13**: S7.

Marmottant, P. and Villermaux, E. (2002). Atomisation primaire dans les jets coaxiaux. *Combustion (Rev. des Sci. Tech. Combustion)*, **2**: 89–126.

Martin, D.F. (1998). *An adaptive cell-centered projection method for the incompressible Euler equations*. PhD thesis, University of California, Berkeley.

Meiburg, E. and Homsy, G.M. (1988). Nonlinear unstable viscous fingers in Hele–Shaw flows. II. Numerical simulation. *Phys. Fluids*, **31**: 429.

Melcher, J.R. and Taylor, G.I. (1969). Electrohydrodynamics: a review of the role of interfacial shear stresses. *Annu. Rev. Fluid Mech.*, **1**: 111–146.

Menchaca-Rocha, A., Huidobro, F., Martinez-Davalos, A., Michaelian, K., Perez, A., Rodriguez, V., and Cârjan, N. (2000). Coalescence and fragmentation of colliding mercury drops. *J. Fluid Mech.*, **346**: 291–318.

Miller, G.H. and Colella, P. (2002). A conservative three-dimensional Eulerian method for coupled solid–fluid shock capturing. *J. Comput. Phys.*, **183**: 26–82.

Mitchell, T.M. and Hammitt, F.H. (1973). Asymmetric cavitation bubble collapse. *Trans. ASME, J. Fluids Eng.*, **95**: 29.

Mittal, R. and Iaccarino, G. (2005). Immersed boundary methods. *Annu. Rev. Fluid Mech.*, **37**: 239–261.

Morton, D., Rudman, M., and Jong-Leng, L. (2000). An investigation of the flow regimes resulting from splashing drops. *Phys. Fluids*, **12**(4): 747–763.

Mukherjee, A. and Dhir, V.K. (2004). Study of lateral merger of vapor bubbles during nucleate pool boiling. *J. Heat Transfer*, **126**: 1023–1039.

Mukherjee, A. and Kandlikar, S.G. (2007). Numerical study of single bubbles with dynamic contact angle during nucleate pool boiling. *Int. J. Heat Mass Transfer*, **50**: 127–138.

Muradoglu, M. and Kayaalp, A.D. (2006). An auxiliary grid method for computations of multiphase flows in complex geometries. *J. Comput. Phys.*, **214**: 858–877.

Muradoglu, M. and Tryggvason, G. (2008). A front-tracking method for computation of interfacial flows with soluble surfactants. *J. Comput. Phys.*, **227**: 2238–2262.

Nakamura, T. and Yabe, T. (1999). Cubic interpolated propagation scheme for solving the hyper-dimensional Vlasov–Poisson equation in phase space. *Comput. Phys. Commun.*, **120**: 122.

Nas, S. and Tryggvason, G. (2003). Thermocapillary interaction of two bubbles or drops. *Int. J. Multiphase Flow*, **29**: 1117–1135.

Nas, S., Muradoglu, M., and Tryggvason, G. (2006). Pattern formation of drops in thermocapillary migration. *Int. J. Heat Mass Transfer*, **49**: 2265–2276.

Navier, C.L. (1823). Sur les lois du mouvement des fluides. *Mém. Acado. Sci. France*, **6**: 389–440 (appeared in 1827).

Nestler, B. (2005). A 3D parallel simulator for crystal growth and solidification in complex alloy systems. *J. Cryst. Growth*, **275**: 273–278.

Nie, X.B., Chen, S.Y., E, W.N., and Robbins, M.O. (2004). A continuum and molecular dynamics hybrid method for micro-and nano-fluid flow. *J. Fluid Mech.*, **500**: 55–64.

Nikolopoulos, N., Theodorakakos, A., and Bergeles, G. (2007). Three-dimensional numerical investigation of a droplet impinging normally onto a wall film. *J. Comput. Phys.*, **227**: 1428–1469.

Nobari, M.R. and Tryggvason, G. (1996). Numerical simulations of three-dimensional drop collisions. *AIAA J.*, **34**: 750–755.

Nobari, M.R., Jan, Y.-J., and Tryggvason, G. (1996). Head-on collision of drops – a numerical investigation. *Phys. Fluids*, **8**: 29–42.

Noh, W.F. and Woodward, P. (1976). SLIC (simple line interface calculation). In *Proceedings, Fifth International Conference on Fluid Dynamics* (eds. A.I. van de Vooren and P.J. Zandbergen), Lecture Notes in Physics, volume 59. Berlin, Springer, Berlin, pp. 330–340.

Nordmark, H.O. (1991). High-order vortex methods with regridding. *J. Comput. Phys.*, **97**: 366.

Oguz, H. and Prosperetti, A. (1990). Bubble entrainment by the impact of drops on liquid surfaces. *J. Fluid Mech.*, **219**: 143–179.

Oguz, H. and Prosperetti, A. (1993). Dynamics of bubble growth and detachment from a needle. *J. Fluid Mech.*, **257**: 111–145.

O'Keefe, J.D. and Ahrens, T.J. (1986). Oblique impact: a process for obtaining meteorite samples from other planets. *Science*, **234**(4774): 346.

Oran, E.S. and Boris, J.P. (1987). *Numerical Simulation of Reactive Flow*. Elsevier.

Oron, A., Davis, S.H., and Bankoff, S.G. (1997). Long-scale evolution of thin films. *Rev. Mod. Phys.*, **69**: 931–980.

Osher, S. and Fedkiw, R. (2002). *Level Set Methods and Dynamic Implicit Surfaces*. Springer.

Osher, S. and Fedkiw, R.P. (2001). Level set methods: an overview and some recent results. *J. Comput. Phys.*, **169**: 463–502.

Osher, S. and Sethian, J. (1988). Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations. *J. Comput. Phys.*, **79**: 12–49.

Osher, S. and Shu, C.-W. (1991). High-order essentially nonsocillatory schemes for Hamilton–Jacobi equations. *SIAM J. Numer. Anal.*, **28**: 907–922.

Pan, T.W., Joseph, D.D., Bai, R., Glowinski, R., and Sarin, V. (2002). Fluidization of 1204 spheres: simulation and experiment. *J. Fluid Mech.*, **451**: 169–191.

Pasandideh-Fard, M., Chandra, S., and Mostaghimi, J. (2002). A three-dimensional model of droplet impact and solidification. *Int. J. Heat Mass Transfer*, **45**: 2229–2242.

Patankar, S.V. (1980). *Numerical Heat Transfer and Fluid Flow*. Taylor and Francis.

Peng, D., Merriman, B., Zhao, H., Osher, S., and Kang, M. (1999). A PDE based fast local level set method. *J. Comput. Phys.*, **155**: 410–438.

Peregrine, D.H., Shoker, G., and Symon, A. (1990). The bifurcation of liquid bridges. *J. Fluid. Mech.*, **212**: 25–39.

Peskin, C.S. (1977). Numerical analysis of blood flow in the heart. *J. Comput. Phys.*, **25**: 220.

Peskin, C.S. (2002). The immersed boundary method. *Acta Numer.*, **11**: 479–517.

Peskin, C.S. and McQueen, D.M. (1989). A three-dimensional computational method for blood flow in the heart I. Immersed elastic fibers in a viscous incompressible fluid. *J. Comput. Phys.*, **81**: 372–405.

Peskin, C.S. and Printz, B.F. (1993). Improved volume conservation in the computation of flows with immersed boundaries. *J. Comput. Phys.*, **105**: 33–46.

Pilliod, J.E., Jr. and Puckett, E.G. (2004). Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J. Comput. Phys.*, **199**: 465–502.

Plapp, M. (2007). Three-dimensional phase-field simulations of directional solidification. *J. Cryst. Growth*, **303**: 49–57.

Plateau, J. (1873). *Statique Expérimentale et Théorique des Liquide*. Gauthier-Villars.

Pomeau, Y. (2002). Recent progress in contact line dynamics: a review. *C. R. Acad. Sci. Paris, Sér. IIb, (Paris). Méc.*, **330**: 207–222.

Popinet, S. (2003). Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. *J. Comput. Phys.*, **190**: 572–600.

Popinet, S. (2008). The Gerris flow solver. `http://gfs.sf.net`.

Popinet, S. (2009). An accurate adaptive solver for surface-tension-driven interfacial flows. *J. Comput. Phys.*, **228**: 5838–5866.

Popinet, S. and Zaleski, S. (1999). A front-tracking algorithm for accurate representation of surface tension. *Int. J. Numer. Methods Fluids*, **30**: 775–793.

Popinet, S. and Zaleski, S. (2002). Bubble collapse near a solid boundary: a numerical study of the influence of viscosity. *J. Fluid Mech.*, **464**: 137–163.

Pöschel, T. and Schwage, T. (2005). *Computational Granular Dynamics: Models and Algorithms*. Springer.

Powell, K. (1998). Solution of the Euler equations on solution-adaptive Cartesian grids. In *Computational Fluid Dynamics Reviews*, (eds. M. Hafez, and K. Oshima), volume 1. World Scientific, pp. 65–92.

Pozrikidis, C. (1992). *Integral and Singularity Methods for Linearized Viscous Flow*. Cambridge University Press.

Pozrikidis, C. (2001). Interfacial dynamics for Stokes flow. *J. Comput. Phys.*, **169**: 250–301.

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. (1993). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

Prosperetti, A. (1981). Motion of two superposed viscous fluids. *Phys. Fluids*, **24**: 1217–1223.

Prosperetti, A. (2007). *Averaged Equations for Multiphase Flow.* Cambridge University Press, pp. 237–281.

Prosperetti, A. and Oğuz, H.N. (1993). The impact of drops on liquid surfaces and the underwater noise of rain. *Annu. Rev. Fluid Mech.*, **25**: 577–602.

Prosperetti, A. and Tryggvason, G. (2007). *Computational Methods for Multiphase Flow*. Cambridge University Press.

Puckett, E.G. (1991). A volume-of-fluid interface tracking algorithm with applications to computing shock wave refraction. In *Proceedings of the 4th Int. Symp. on Computational Fluid Dynamics*, Davis, CA (ed. H. Dwyer) pp. 933–938.

Puckett, E.G., Almgren, A.S., Bell, J.B., Marcus, D.L., and Rider, W.J. (1997). A high-order projection method for tracking fluid interfaces in variable density incompressible flows. *J. Comput. Phys.*, **130**: 269–282.

Pumphrey, H.C. and Elmore, P.A. (1990). The entrainment of bubbles by drop impacts. *J. Fluid Mech.*, **220**: 539–567.

Purvis, R. and Smith, F.T. (2005). Droplet impact on water layers: post-impact analysis and computations. *Phil. Trans. R. Soc. A*, **363**: 1209–1221.

Quirk, J.J. (1994). An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies. *Comput. Fluids*, **23**: 125–142.

Radl, S., Koynov, A., Tryggvason, G., and Khinast, J.G. (2008). DNS-based prediction of the selectivity of fast multiphase reactions: hydrogenation of nitroarenes. *Chem. Eng. Sci.*, **63**: 3279–3291.

Rallison, J.M. and Acrivos, A. (1978). A numerical study of the deformation and burst of a viscous drop in an extensional flow. *J. Fluid Mech.*, **89**: 191–200.

Ranade, V. (2001). *Computational Flow Modeling for Chemical Reactor Engineering*. Academic Press.

Rayleigh, J.W. Strutt, Baron. (1879). On the capillary phenomena of jets. *Proc. R. Soc. London*, **29**: 71–97.

Rayleigh, J.W. Strutt, Baron. (1900). *Scientific Papers*. Cambridge University Press.

Raynal, L. (1997). PhD thesis, Universite Joseph Fourier, Grenoble.

Rein, M. (1993). Phenomena of liquid drop impact on solid and liquid surfaces. *Fluid Dyn. Res.*, **12**(2): 61–93.

Renardy, Y. and Renardy, M. (2002). PROST: a parabolic reconstruction of surface tension for the volume-of-fluid method. *J. Comput. Phys.*, **183**: 400–421.

Rhie, C.M. and Chow, W.L. (1983). Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA J.*, **21**(11): 1525–1532.

Richtmyer, R.D. and Morton, K.W. (1967). *Difference Methods for Initial Value Problems*. Interscience, New York.

Richtmyer, R.D. (1960). Taylor instability in shock acceleration of compressible fluids. *Commun. Pure Appl. Math.*, **13**: 297–319.

Rider, W.J. and Kothe, D.B. (1998). Reconstructing volume tracking. *J. Comput. Phys.*, **141**: 112–152.

Rieber, M. and Frohn, A. (1999). A numerical study on the mechanism of splashing. *Int. J. Heat Fluid Flow*, **20**(5): 455–461.

Rioboo, R., Bauthier, C., Conti, J., Voué, M., and De Coninck, J. (2003). Experimental investigation of splash and crown formation during single drop impact on wetted surfaces. *Exp. Fluids*, **35**(6): 648–652.

Robinson, P.B., Blake, J.R., Kodama, T., Shima, A., and Tomita, Y. (2001). Interaction of cavitation bubbles with a free surface. *J. Appl. Phys.*, **89**(12): 8225–8237.

Roisman, I.V., Rioboo, R., and Tropea, C. (2002). Normal impact of a liquid drop on a dry surface: model for spreading and receding. *Proc. R. Soc. Lond. Ser. A*, **458**(2022): 1411–1430.

Roisman, I.V., Horvat, K., and Tropea, C. (2006). Spray impact: rim transverse instability initiating fingering and splash, and description of a secondary spray. *Phys. Fluids*, **18**(10): 102104–102104.

Rothman, D.H. and Zaleski, S. (1997). *Lattice-Gas Cellular Automata: Simple Models of Complex Hydrodynamics*. Cambridge University Press.

Rudman, M. (1997). Volume-tracking methods for interfacial flow calculations. *Int. J. Numer. Methods Fluids*, **24**: 671–691.

Rybczynski, W. (1911). Über die fortschreitende Bewegung einer flüssigen Kugel in einem zähen Medium. *Bull. Acad. Sci. Crac.*, **A**: 40–46.

Ryskin, G. and Leal, L.G. (1984). Numerical solution of free-boundary problems in fluid mechanics. Part 2. Buoyancy-driven motion of a gas bubble through a quiescent liquid. *J. Fluid Mech.*, **148**: 19–35.

Saiki, E.M. and Biringen, S. (1996). Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method. *J. Comput. Phys.*, **123**: 450–465.

Sainsbury, G.L. and Cheeseman, I.C. (1950). Effect of rain in calming the sea. *Nature*, **166**(4210): 79.

Sangani, A.S. and Didwania, A.K. (1993). Dynamic simulations of flows of bubbly liquids at large Reynolds numbers. *J. Fluid Mech.*, **250**: 307–337.

Sankaranarayanan, K., Shan, X., Kevrekidis, I.G., and Sundaresan, S. (2002). Analysis of drag and virtual mass forces in bubbly suspensions using an implicit formulation of the lattice Boltzmann method. *J. Fluid Mech.*, **452**: 61–96.

Sankaranarayanan, K., Kevrekidis, I.G., Sundaresan, S., Lu, J., and Tryggvason, G. (2003). A comparative study of lattice Boltzmann and front-tracking finite-difference methods for bubble simulations. *Int. J. Multiphase Flow*, **29**: 109–116.

Saville, D.A. (1997). Electrohydrodynamics: the Taylor–Melcher leaky dielectric model. *Annu. Rev. Fluid Mech.*, **29**: 27–64.

Scardovelli, R. and Zaleski, S. (2000). Analytical relations connection linear interfaces and volume fractions in rectangular grids. *J. Comput. Phys.*, **164**: 228–237.

Scardovelli, R. and Zaleski, S. (2003). Interface reconstruction with least-square fit and split Lagrangian–Eulerian advection. *Int J. Numer. Methods Fluids*, **41**: 251–274.

Schlick, T. (2002). *Molecular Modeling and Simulation*. Springer.

Schmidt, A. (1996). Computations of three dimensional dendrites with finite elements. *J. Comput. Phys.*, **126**: 293–312.

Schultz, W.W., Huh, J., and Griffin, O.M. (1994). Potential-energy in steep and breaking waves. *J. Fluid Mech*, **278**: 201–228.

Scriven, L.E. (1960). The Marangoni effects. *Chem. Eng. Sci.*, **12**: 98–108.

Serizawa, A., Kataoka, I., and Michiyoshi, I. (1975a). Turbulence structure of air–water bubbly flow – II. Local properties. *Int. J. Multiphase Flow*, **2**: 235–246.

Serizawa, A., Kataoka, I., and Michiyoshi, I. (1975b). Turbulence structure of air–water bubbly flow – III. Transport properties. *Int. J. Multiphase Flow*, **2**: 247–259.

Sethian, J.A. (2001). Evolution, implementation, and application of level set and fast marching methods for advancing fronts. *J. Comput. Phys.*, **169**: 503–555.

Sethian, J.A. and Strain, J. (1992). Crystal growth and dendritic solidification. *J. Comput. Phys.*, **98**: 231–253.

Shan, X.W. and Chen, H.D. (1993). Lattice Boltzmann model for simulationg flows with multiple phases and components. *Phys. Rev. E*, **47**: 1815–1819.

Shikmurzaev, Y.D. (1997). Moving contact lines in liquid/liquid/solid systems. *J. Fluid Mech.*, **334**: 211–249.

Shin, S. and Juric, D. (2002). Modeling three-dimensional multiphase flow using a level contour reconstruction method for front tracking without connectivity. *J. Comput. Phys*, **180**: 427–470.

Shin, S., Abdel-Khalik, S.I., Daru, V., and Juric, D. (2005a). Accurate representation of surface tension using the level contour reconstruction method. *J. Comput. Phys.*, **203**: 493–516.

Shin, S., Abdel-Khalik, S.I., and Juric, D. (2005b). Direct three-dimensional numerical simulation of nucleate boiling using the level contour reconstruction method. *Int. J. Multiphase Flow*, **31**: 1231–1242.

Shin, S.W. and Juric, D. (2000). Computation of microstructure in solidification with fluid convection. *J. Mech. Behav. Mater.*, **11**: 313.

Shopov, P.J., Minev, P.D., Bazhekov, I.B., and Zapryanov, Z.D. (1990). Interaction of a deformable bubble with a rigid wall at moderate Reynolds numbers. *J. Fluid Mech.*, **219**: 241–271.

Shu, C.W. and Osher, S. (1989). Efficient implementation of essentially non-oscillatory shock capturing schemes, II. *J. Comput. Phys.*, **83**: 32–78.

Shyy, W. and Rao, M.M. (1994). Enthalpy based formulations for phase-change problems with application to *g*-jitter. *Microgravity Sci. Technol.*, **7**: 41–49.

Shyy, W., Udaykumar, H.S., Rao, M., and Smith, R. (1996). *Computational Fluid Dynamics with Moving Boundaries*. Taylor and Francis.

Smereka, P. (1993). On the motion of bubbles in a periodic box. *J. Fluid Mech.*, **254**: 79–112.

So, S., Morikita, H., Takagi, S., and Matsumoto, Y. (2002). Laser doppler velocimetry measurement of turbulent bubbly channel flow. *Exp. Fluids*, **33**: 135–142.

Son, G. and Dhir, V.K. (1997). Numerical simulation of saturated film boiling on a horizontal surface. *J. Heat Transfer*, **119**: 525–533.

Son, G. and Dhir, V.K. (1998). Numerical simulation of film boiling near critical pressures with a level set method. *J. Heat Transfer*, **120**: 183–192.

Son, G., Dhir, V.K., and Ramanujapu, N. (1999). Dynamics and heat transfer associated with a single bubble during nucleate boiling on a horizontal surface. *J. Heat Transfer*, **121**: 623–631.

Son, G., Ramanujapu, N., and Dhir, V.K. (2002). Numerical simulation of bubble merger process on a single nucleation site during pool nucleate boiling. *ASME J. Heat Transfer*, **124**: 51–62.

Song, M. and Tryggvason, G. (1999). The formation of a thick border on an initially stationary fluid sheet. *Phys. Fluids*, **11**: 2487–2493.

Song, Q., Luo, R., Yang, X.Y., and Wang, Z. (2001). Phase distributions for upward laminar dilute bubbly flows with non-uniform bubble sizes in a vertical pipe. *Int. J. Multiphase Flow*, **27**: 379–390.

Stow, C.D. and Hadfield, M.G. (1981). An experimental investigation of fluid flow resulting from the impact of a water drop with an unyielding dry surface. *Proc. R. Soc. Lond. Ser. A*, **373**(1755): 419–441.

Subramanian, R.S. and Balasubramaniam, R. (2001). *The Motion of Bubbles and Drops in Reduced Gravity*. Cambridge University Press.

Sussman, M. (2003). A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comput. Phys.*, **187**: 110–136.

Sussman, M. and Fatemi, E. (1999). An efficient, interface preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM J. Sci. Comput.*, **20**(4): 1165–1191.

Sussman, M. and Puckett, E.G. (2000). A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows. *J. Comput. Phys.*, **162**: 301–337.

Sussman, M. and Smereka, P. (1997). Axisymmetric free boundary problems. *J. Fluid Mech.*, **341**: 269–294.

Sussman, M., Smereka, P., and Osher, S. (1994). A level set approach for computing solutions to incompressible two-phase flows. *J. Comput. Phys.*, **114**: 146–159.

Sussman, M., Fatemi, E., Smereka, P., and Osher, S. (1998). An improved level set method for incompressible two-phase flows. *Comput. Fluids*, **27**: 663–680.

Sussman, M., Almgren, A.S., Bell, J.B., Colella, P., Howell, L.H., and Welcome, M.L. (1999). An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys.*, **148**: 81–124.

Swaminathan, C.R. and Voller, V.R. (1993). On the enthalpy method. *Int. J. Numer. Methods Heat Fluid Flow*, **3**: 233–244.

Swarztrauber, P.N. and Sweet, R.A. (1979). Algorithm **541**: efficient Fortran subprograms for the solution of separable elliptic partial differential equations [D3]. *ACM Trans. Math. Software*, **5**(3): 352–364.

Sweby, P.K. (1984). High-resolution schemes using flux limiters for hyperbolic conservation-laws. *SIAM J. Numer. Anal.*, **21**: 995–1011.

Takada, N., Misawa, M., Tomiyama, A., and Hosokawai, S. (2001). Simulation of bubble motion under gravity by lattice Boltzmann method. *J. Nucl. Sci. Technol.*, **38**: 330–341.

Takagi, S., Matsumoto, Y., and Huang, H. (1997). Numerical analysis of a single rising bubble using boundary-fitted coordinate system. *JSME Int. J., Ser. B*, **40**: 42–50.

Takahira, H., Horiuchi, T., and Banerjee, S. (2004). An improved three-dimensional level set method for gas–liquid two-phase flows. *J. Fluids Eng.*, **126**: 578.

Takewaki, H. and Yabe, T. (1987). The cubic-interpolated pseudo particle (CIP) method: application to nonlinear and multi-dimensional hyperbolic equations. *J. Comput. Phys.*, **70**: 355–372.

Takewaki, H., Nishiguchi, A., and Yabe, T. (1985). Cubic interpolated pseudo-particle method (CIP) for solving hyperbolic-type equations. *J. Comput. Phys.*, **61**: 261–268.

Tan, L. and Zabaras, N. (2007). A level set simulation of dendritic solidification of multi-component alloys. *J. Comput. Phys.*, **221**: 9–40.

Tasoglu, S., Demirci, U., and Muradoglu, M. (2008). The effect of soluble surfactant on the transient motion of a buoyancy-driven bubble. *Phys. Fluids*, **20**: 040805.

Taylor, G.I. (1963). *Generation of Ripples by Wind Blowing over a Viscous Liquid*. Cambridge University Press, Cambridge, UK, pp. 244–254.

Taylor, G.I. (1966). Studies in electrohydrodynamics. I. The circulation produced in a drop by electrical field. *Proc. R. Soc. Lond. Ser. A*, **291**: 159–166.

Thompson, J.F., Soni, B.K., and Weatherill, N.P. (1998). *Handbook of Grid Generation*. CRC Press.

Thomson, J.J. and Newall, H.F. (1885). On the formation of vortex rings by drops falling into liquids, and some allied phenomena. *Proc. R. Soc. Lond.*, **39**: 417–436.

Thoroddsen, S.T. (2002). The ejecta sheet generated by the impact of a drop. *J. Fluid Mech.*, **451**: 373–381.

Tomar, G., Biswas, G., Sharma, A., and Agrawal, A. (2005). Numerical simulation of bubble growth in film boiling using a coupled level-set and volume-of-fluid method. *Phys. Fluids*, **17**: 112103.

Tomar, G., Gerlach, D., Biswas, G., Alleborn, N., Sharma, A., Durst, F., S.W.J. Welch, and Delgado, A. (2007). Two-phase electrohydrodynamic simulations using a volume-of-fluid approach. *J. Comput. Phys.*, **227**: 1267–1285.

Tomar, G., Biswas, G., Sharma, A., and Welch, S.W.J. (2008). Multi-mode analysis of bubble growth in saturated film boiling. *Phys. Fluids*, **20**: 092101.

Tomar, G., Biswas, G., Sharma, A., and Welch, S.W.J. (2009). Influence of electric field on saturated film boiling. *Phys. Fluids*, **21**: 032107.

Tomar, G., Fuster, D., Zaleski, S., and Popinet, S. (2010). Multiscale simulations of primary atomization using Gerris. *Comput. Fluids*, **39**(10): 1864–1874.

Tomiyama, A., Zun, I., Sou, A., and Sakaguchi, T. (1993). Numerical analysis of bubble motion with the VOF method. *Nucl. Eng. Des.*, **141**: 69–82.

Tomiyama, A., Tamai, H., Zun, I., and Hosokawa, S. (2002). Transverse migration of single bubbles in simple shear flows. *Chem. Eng. Sci.*, **57**: 1849–1858.

Tonhardt, R. and Amberg, G. (1998). Phase-field simulations of dendritic growth in a shear flow. *J. Cryst. Growth*, **194**: 406–425.

Torres, D.J and Trujillo, M.F. (2006). KIVA-4: an unstructured ALE code for compressible gas flow with sprays. *J. Comput. Phys.*, **219**(2): 943–975.

Tryggvason, G. and Aref, H. (1983). Numerical experiments on Hele–Shaw flow with a sharp interface. *J. Fluid Mech.*, **136**: 1–30.

Tryggvason, G. and Aref, H. (1985). Finger interaction mechanisms in stratified Hele–Shaw flow. *J. Fluid Mech.*, **154**: 284–301.

Tsimplis, M. and Thorpe, S.A. (1989). Wave damping by rain. *Nature*, **342**(6252): 893–895.

Udaykumar, H.S., Kan, H.C., Shyy, W., and Tran-Son-Tay, R. (1997). Multiphase dynamics in arbitrary geometries on fixed Cartesian grids. *J. Comput. Phys.*, **137**: 366–405.

Udaykumar, H.S., Mittal, R., and Shyy, W. (1999). Computation of solid–liquid phase fronts in the sharp interface limit on fixed grids. *J. Comput. Phys.*, **153**: 535–574.

Udaykumar, H.S., Mittal, R., Rampunggoon, P., and Khanna, A. (2001). A sharp interface Cartesian grid method for simulating flows with complex moving boundaries. *J. Comput. Phys.*, **174**: 345–380.

Ungar, L.H. and Brown, R.A. (1985). Cellular interface morphologies in directional solidification. IV. The formation of deep cells. *Phys. Rev. B*, **31**: 5931–5940.

Unverdi, S.O. and Tryggvason, G. (1992). A front-tracking method for viscous, incompressible, multi-fluid flows. *J. Comput. Phys.*, **100**: 25–37.

van der Vorst, H.A. (1992). Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, **13**: 631.

van der Vorst, H.A. (2003). *Iterative Krylov Methods for Large Linear Systems*. Cambridge University Press.

van Leer, B. (1979). Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *J. Comput. Phys.*, **32**: 101–136.

van Sint Annaland, M., Dijkhuizen, W., Deen, N.G., and Kuipers, J.A.M. (2006). Numerical simulation of gas bubbles behaviour using a 3D front tracking method. *AIChE J.*, **52**: 99–110.

Verschueren, M., van de Vosse, F.N., and Meijer, H.E.H. (2001). Diffuse-interface modelling of thermo-capillary flow instabilities in a Hele–Shaw cell. *J. Fluid Mech.*, **434**: 153–166.

Viecelli, J.A. (1969). A method for including arbitrary external boundaries in the MAC incompressible fluid computing technique. *J. Comput. Phys.*, **4**: 543–551.

Villermaux, E. (1998). On the role of viscosity in shear instabilities. *Phys. Fluids*, **10**(2): 368–373.

Villermaux, E. and Clanet, C. (2002). Life of a flapping liquid sheet. *J. Fluid Mech.*, **462**: 341–363.

Villermaux, E., Marmottant, Ph., and Duplat, J. (2004). Ligament-mediated spray formation. *Phys. Rev. Lett.*, **92**(7): 074501.

Vinje, T. and Brevig, P. (1981). Numerical simulations of breaking waves. *Adv. Water Resources*, **4**: 77–82.

Wang, S.K., Lee, S.J., Jones, O.C., Jr. and Lahey, R.T. Jr. (1987). 3-D turbulence structure and phase distribution in bubbly two-phase flows. *Int. J. Multiphase Flow*, **13**: 327–343.

Wang, Y., Lu, X., Zhuang, L., Tang, Z., and Hu, W. (2004). Numerical simulation of drop Marangoni migration under microgravity. *Acta Astronaut.*, **54**: 325–335.

Wanga, J., Lua, P., Wanga, Z., Chao, C., and Maoa, Z.-S. (2008). Numerical simulation of unsteady mass transfer by the level set method. *Chem. Eng. Sci.*, **63**: 3141–3151.

Weatherburn, C.E. (1927). *Differential Geometry of Three Dimensions*, Vol. I. Cambridge University Press, Cambridge.

Welch, S.W.J. (1995). Local simulation of two-phase flows including interface tracking with mass transfer. *J. Comput. Phys.*, **121**: 142–154.

Welch, S.W.J. and Biswas, G. (2007). Direct simulation of film boiling including electrohydrodynamic forces. *Phys. Fluids*, **19**: 012106.

Welch, S.W.J. and Wilson, J. (2000). A volume of fluid based method for fluid flows with phase change. *J. Comput. Phys.*, **160**: 662–682.

Werder, T., Walther, J.H., and Koumoutsakos, P. (2005). Hybrid atomistic-continuum method for the simulation of dense fluid flow. *J. Comput. Phys.*, **205**: 373–390.

Wesseling, P. (1992). *An Introduction to Multigrid Methods*. Wiley, New York.

Wesseling, P. (2001). *Principles of Computational Fluid Dynamics*. Springer-Verlag Berlin.

Wheeler, A.A., Murray, B.T., and Schaefer, R.J. (1993). Computations of dendrites using a phase-field model. *Phys. D*, **66**: 243–262.

Wheeler, A.A., Ahmad, N.A., Boettinger, W.J., Braun, R.J., McFadden, G.B., and Murray, B.T. (1995). Recent development in phase-field models of solidification. *Adv. Space Res.*, **16**: 163–172.

Worthington, A.M. (1908). *A Study of Splashes*. Longmans, Green and Co., London.

Xue, M., Xu, H.B., Liu, Y.M., and Yue, D.K.P. (2001). Computations of fully nonlinear three-dimensional wave–wave and wave–body interactions. Part 1. Dynamics of steep three-dimensional waves. *J. Fluid Mech.*, **438**: 11–39.

Yabe, T., Xiao, F., and Utsumi, T. (2001). The constrained interpolation profile (CIP) method for multi-phase analysis. *J. Comput. Phys.*, **169**: 556–593.

Yanenko, N.N. (1971). *The Method of Fractional Steps for Solving Multidimensional Problems of Mathematical Physics in Several Variables*. Springer-Verlag, Berlin.

Yang, C. and Maoa, Z.-S. (2005). Numerical simulation of interphase mass transfer with the level set approach. *Chem. Eng. Sci.*, **60**: 2643–2660.

Yang, X., Feng, J.J., Liu, C., and Shen, J. (2006a). Numerical simulations of jet pinching-off and drop formation using an energetic variational phase-field method. *J. Comput. Phys.*, **218**(1): 417–428.

Yang, X., James, A.J., Lowengrub, J., Zheng, X., and Cristini, V. (2006b). An adaptive coupled level-set/volume-of-fluid interface capturing method for unstructured triangular grids. *J. Comput. Phys.*, **217**: 364–394.

Yang, Y. and Udaykumar, H.S. (2005). Sharp interface Cartesian grid method III: solidification of pure materials and binary solutions. *J. Comput. Phys.*, **210**: 55–74.

Yarin, A.L. (2006). Drop impact dynamics: splashing, spreading, receding, bouncing. *Annu. Rev. Fluid Mech.*, **38**: 159–192.

Yarin, A.L. and Weiss, D.A. (1995). Impact of drops on solid surfaces: self-similar capillary waves, and splashing as a new type of kinematic discontinuity. *J. Fluid Mech.*, **283**: 141–173.

Yecko, P. and Zaleski, S. (2005). Transient growth in two-phase mixing layers. *J. Fluid Mech.*, **528**: 43–52.

Yecko, P., Zaleski, S., and Fullana, J.-M. (2002). Viscous modes in two-phase mixing layers. *Phys.Fluids*, **14**: 4115–4122.

Youngren, G.K. and Acrivos, A. (1976). On the shape of a gas bubble in a viscous extensional flow. *J. Fluid Mech.*, **76**: 433.

Youngs, D.L. (1982). Time dependent multimaterial flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics* (eds. K.M. Morton and M.J. Baines) Academic Press, New York, pp. 27–39.

Youngs, D.L. (1984). An interface tracking method for a 3D Eulerian hydrodynamics code. Technical report, AWRE. Technical Report 44/92/35.

Yu, D. and Tryggvason, G. (1990). The free surface signature of unsteady, two-dimensional vortex flows. *J. Fluid Mech.*, **218**: 547–572.

Yu, P.-W., Ceccio, S.L., and Tryggvason, G. (1995). The collapse of a cavitation bubble in shear flows – a numerical study. *Phys. Fluids*, **7**: 2608–2616.

Zaleski, S., Zanetti, G., Scardovelli, R., *et al.* (1992–2008). SURFER code. The code is covered by the GNU Public Licence and may be downloaded from http://www.lmm.jussieu.fr/∼zaleski.

Zeff, B.W., Kleber, B., Fineberg, J., and Lathrop, D.P. (2000). Singularity dynamics in curvature collapse and jet eruption on a fluid surface. *Nature*, **403**: 401–404.

Zenit, R., Koch, D.L., and Sangani, A.S. (2001). Measurements of the average properties of a suspension of bubbles rising in a vertical channel. *J. Fluid Mech.*, **429**: 307–342.

Zhang, D.Z. and Prosperetti, A. (1994). Ensemble phase-averaged equations for bubbly flows. *Phys. Fluids*, **6**: 2956–2970.

Zhang, J., Eckmann, D.M., and Ayyaswamy, P.S. (2006). A front tracking method for a deformable intravascular bubble in a tube with soluble surfactant transport. *J. Comput. Phys.*, **214**: 366–396.

Zhou, H. and Pozrikidis, C. (1993). The flow of ordered and random suspensions of two-dimensional drops in a channel. *J. Fluid Mech.*, **255**: 103–127.

Zinchenko, A.Z. and Davis, R.H. (2000). An efficient algorithm for hydrodynamical interaction of many deformable drops. *J. Comput. Phys*, **157**: 539–587.

Index

added mass, 203
advection term
  definition, 51
  discretization, 59, 61
AMR (adaptive mesh refinement), 19, 22, 71, 72, 225, 227, 232
analytical solutions, 1, 180, 181, 198, 201, 214, 285, 287, 292
atomization, 3, 4, 19, 72, 204, 205, 211, 212, 214, 215, 219, 222, 226, 228
  co-flowing, 4, 215
  effect of nozzle, 214, 219, 222, 223

bad elements
  in front tracking, 145
Berenson's correlation, 252
Bi-Conjugate Gradient STABilized method, 68
BiCGSTAB method, 68
boiling, 248
  pool, 251
  convection, 249
  film, 248, 249, 251, 252
  flow, 249
  nucleate, 254
  pool, 249
Bond number, 43
boundary condition, 29, 30, 37, 40, 50, 67, 69, 71, 141, 189
  for free surface, 40
  no slip, 37, 69
  outflow, 29, 30, 66, 70
  partial slip, 47
  periodic, 30, 71, 146, 189, 191, 195, 219
boundary integral method, 8–11
breakup of droplet
  in linear shear flow, 9, 10
  under impact or collision, 228, 230
breakup of rim, 205, 210, 211
breakup of thin threads, 158, 206–208
bubble columns, 19, 187
bubbles
  rising, 2, 4, 179, 189, 193, 196
  small, 203
bubbly flows
  homogeneous, 189–191
buoyancy, 203

capillary number, 43
CFL (Courant, Friedrichs, and Lewy) condition, 53, 54, 109
CFL number in VOF method, 109, 123–125
CIP (constrained interpolated propagation) method, 14, 75, 77, 91–93
closure relations, 187, 203
CLSVOF (coupled level-set volume-of-fluid) method, 128
coalescence, 8, 158, 159, 230, 232, 233
coherent structures, 223
collocated grid, 56, 58, 73
color function, 76–78, 80, 82, 95–98, 128, 129, 131, 161, 167, 184
combustion, 4, 205, 228
complex physics, 17, 19
craters, 5, 229–231, 233–235
crown splash, *see* droplet impact
CSF (continuous surface force) method, 13, 14, 161, 162
CSS (continuous surface stress) method, 161, 164–166, 169, 177–179, 181
curvature
  computed from height function, 184–186, 271
  computing in front tracking, 169–171, 175, 176
  computing in VOF, 163, 164, 167, 168, 184–186
  definition, 32
  expression for, 34, 35, 276
  Gauss, 274
  in axisymmetric geometry, 275
  mean
    definition, 33, 34, 169, 273, 274, 277
  total, 274

data structure
  in front tracking, 85, 86, 133, 134
dendrites, 264, 265
dendritic structure, 6, 264
density ratio, 202

disjoining pressure, 44, 45, 48
DNS (direct numerical simulation), 2, 3, 18, 19, 188, 195, 201, 203, 251
drafting, kissing, and tumbling, 18, 192
drop, *see* droplet
droplet formation, 213, 215, 225
droplet impact
  corolla, 213, 235, 236, 238–242
  crown splash, 232, 235, 240–242
  jet formation, 236
  prompt splash, 240

Eötvös number, 43, 188
  small, 203
electrohydrodynamics, 244
energy conservation, 25
ENO (essentially non-oscillating) scheme, 60, 61, 88

Fantasia (Disney animation), 229
Faraday waves, 234
Finite-volume method, 26, 50, 55, 86
fishbone patterns, 226
floatsam and jetsam, 12, 83, 116
flux-corrected transport method, 80
Fourier's law, 26
free surface, 1, 8, 11
  boundary condition at, 40, 45
  bubble bursting at, 204, 234, 235
front capturing method, 91
front-grid communications, 145–148
  weighting functions, 148
front-tracking volume-of-fluid method, 129
Froude number, 44

Galileo number, 43, 188
gas–liquid flows, 1, 6, 189, 291
Gerris code, 61
Gibbs–Thompson condition, 263
GMRES method, 68
Godunov method, 80
grid generation, 17

H-mode, 219
Hamaker's constant, 44, 45
heat transfer, 243
Heaviside or step function, 35, 41, 76, 77, 279
height-function method, 179, 181, 184
human aspect, 20
hybrid methods, 128
  CLSVOF, *see* CLSVOF (coupled level-set volume-of-fluid) method
  front-tracking VOF, *see* front-tracking volume-of-fluid method

immersed-boundary method, 14, 86, 149
indicator function, 68, 76, 77
industrial applications, 1, 3, 187
inertia, 203
interface
  geometrical description, 30, 31, 47, 270
intermolecular forces, 21, 22, 44, 45, 48, 208

jump conditions, 37, 39, 40
jump notation, 38

kinematic singularity, 213
kinematic viscosity, 29
Krylov method, 68

Lagrangian grids, 17
Laplace number, 43
Laplace's law, 45, 163, 177
large-scale structures
  in atomization, 214
  in bubbly flows, 192
LBM (lattice–Boltzmann method), 18
level-set method, 14, 15, 75, 86, 87, 89, 90, 96, 128, 175, 232, 264
lift force, 193, 196, 197, 199
linked list, 139–141, 143, 154
liquid fuel, 205
living systems, 1, 6
low Reynolds numbers, 53, 190
lubrication approximation, 206, 207, 210

MAC (marker-and-cell) method, 11–14, 51, 56, 59, 85, 109, 114, 121, 122, 125, 163, 229
marker function, 12–16, 34, 68, 74–77, 80–82, 84, 86–88, 92–94, 97, 115, 133, 139, 150, 152–158, 161, 162, 174, 175
marker particle (in front tracking), 14, 85, 86, 133, 151, 152
marker particle (in volume), 12, 229
mass conservation, 14, 22–24, 26, 28, 37, 41, 89, 90, 97, 112, 123, 136, 143, 206, 249
  in VOF, 97, 112, 123
mean curvature, *see* curvature
microstructure, 6, 20, 188, 262–265
mixing layer, 205, 215, 216, 220, 291
  stability theory, 216
    Kelvin–Helmholtz, 216, 288
    Orr–Sommerfeld, 218
    transient-growth theory, 224
momentum conservation, 24
  with interfaces, 39
moon craters, 228
  central peak, 229
Morton number, 43, 188
multigrid methods, 67
multigrid solvers, 67

Navier–Stokes equations, 11, 21
  definition, 25
Newtonian fluids, 22, 26, 28, 29, 40, 42
  definition, 24
non-Newtonian fluids, 20
nonuniform surface tension, 244
nucleate boiling, 253
Nusselt number, 252

oceans, 1, 3, 5, 204
Ohnesorge number, 43, 210, 231
one-fluid approach, 15–17, 21, 41, 42, 54, 161

PDF (probability distribution functions), 226
phase change, 5, 19, 37–39, 41, 91, 248
phase-field method, 15, 48, 90, 91, 264, 265
Plateau–Rayleigh jet instability (capillary instability), 4, 205, 206, 213, 214, 242
Poisson equation, 52, 64
potential flow
  in droplet impact, 237
potential flow simulations
  in bubbly flows, 188, 190, 192
pressure equation, 52, 64
pressure solver, 66
projection method, 12, 38, 51–53, 59
PROST (proper representation of surface tension) method, 165, 166, 179, 181, 183, 184

QUICK (quadratic upstream interpolation for convective kinematics) scheme, 60, 61

rain, 1, 4, 5, 228
Rayleigh–Taylor instability, 8, 11, 12, 85, 91
  in atomization, 224
redistribution
  in VOF, 120, 122, 130
  of markers in front tracking, 137
reinitialization
  in level-set method, 14, 88, 89
Reynolds number, 42
Richtmyer–Meshkov instability
  in droplet impact, 241, 242
rim, 205, 209, 210, 212–214

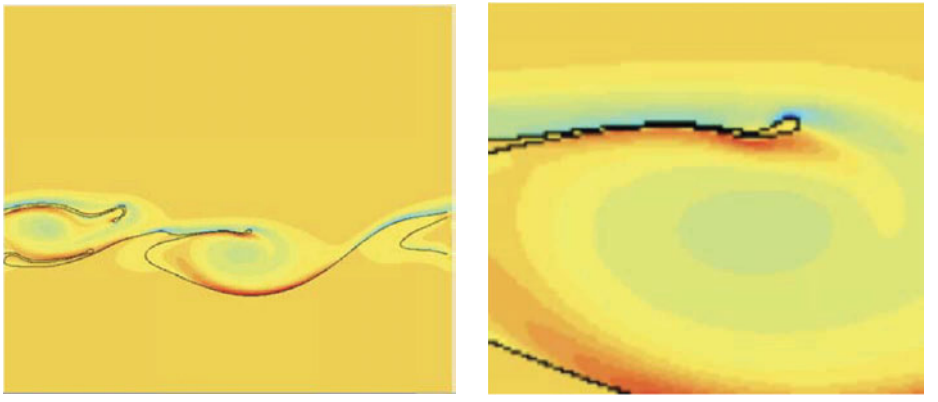
Saffman–Taylor instability, 10
self-similar solutions, 48, 208, 234, 235
sharp interface limit, 15, 16, 21, 22, 34, 37, 42, 44, 73, 80, 81, 91, 93
singular terms, 41
SLIC (simple line interface calculation) method, 82–84, 96, 116
SOR (successive over-relaxation) method, 66–68
spheres
  falling, 18
splash, 4, 5, 11, 13, 210, 211, 213, 228, 230, 231, 240
  prompt, 240
spray formation, 204
staggered grid (MAC grid), 12, 54, 56–58, 65, 69, 74, 114, 121, 125, 146, 153, 174
step function
  approximation in front tracking, 84, 166
surface geometry
  curvature, *see* curvature
  principal directions, 274
surface tension, 203
  axisymmetry, 168
  computed from marker functions, 161–167, 184, 185
  computed in front tracking, 168, 169, 171–177
  definition, 22, 38
  force (definition), 42
surface tension tensor, 38, 39, 164
surfactants
  in bubble dynamics, 189
suspensions, 6, 8, 10

Taylor–Culick rim, *see* rim
thermocapillary motion, 244
thin films, 19, 31, 44, 48, 206–208, 228
  splashing on, 235
time integration, 51, 53
topology change, 8, 44, 158, 159, 230, 232, 236
  in front tracking, 87, 134, 158, 159
  in VOF, 96

unit normal
  definition, 31–33
unstructured grids, 18, 85
  in front-tracking method, 135, 143
  in VOF, 131

viscous terms
  discretization, 61, 64
  harmonic mean, 63, 64, 221
VOF (volume-of-fluid) method, 12–15, 72, 81, 83, 84, 88–90, 94–98, 103, 124, 128, 129, 150, 161, 163, 165, 166, 169, 175, 182–184, 236, 242, 249
  advection, 108
    Eulerian implicit, 115, 116
    Lagrangian explicit, 115, 116
    unsplit, 116
  normal evaluation
    centered columns method, 100
    ELVIRA, 101–104, 107, 108, 123–125, 127, 128
    least-squares fit method, 103
    Youngs' method, 99, 103, 104, 107, 108
  reconstruction, 12, 83, 84, 95–99, 101, 103, 104, 106–108, 112, 116, 122–129, 183
  wisps, 116
volume change, 248, 262, 263
volume expansion, 250, 251
volume fraction
  in front tracking, 155
volume source at the interface, 41
vortex-in-a-box test, 123, 125–127, 131, 132, 151, 152

wall layer
  in bubbly flow, 196, 201
Weber number, 43
weighted ENO scheme, 128
wetting, 46, 48, 229, 230
wisps, 116
Worthington jet, 229



(a) (b)

Fig. 9.12. (a) Typical high-resolution simulation. The colors indicate the vorticity. (b) Detail of simulation showing the end rim and a “Kelvin–Helmholtz roller” trapped below the sheet.

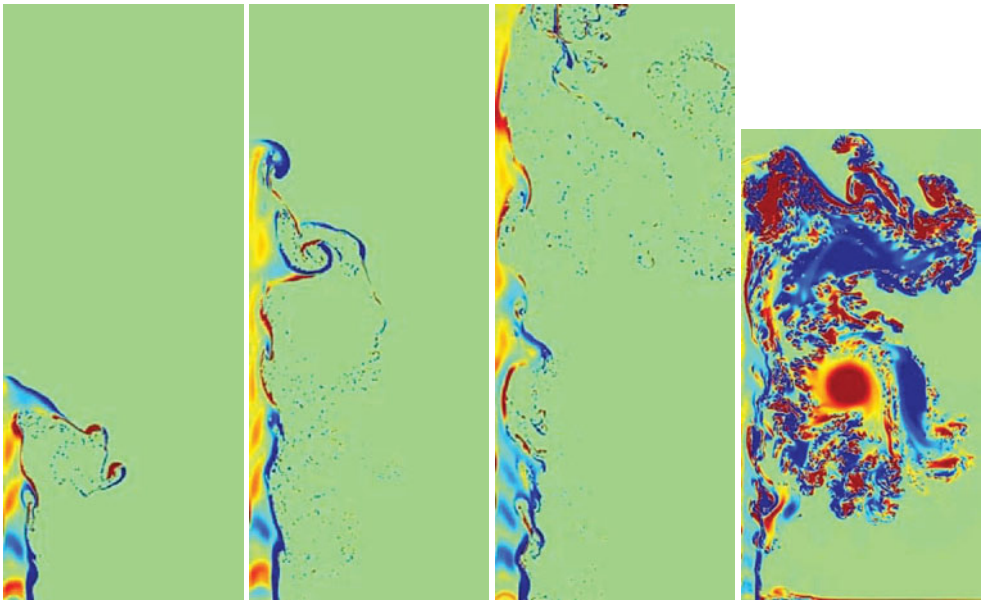


Fig. 9.15. The spatial development of an atomizing jet simulated in two dimensions using SURFER (obtained by Leboissetier (2002)). The color scheme represents the vorticity levels in the liquid. The vorticity in the gas is masked in the three figures on the left. A very strong coherent structure is seen in the gas region in the last figure on the right. Parameters are given in the text.

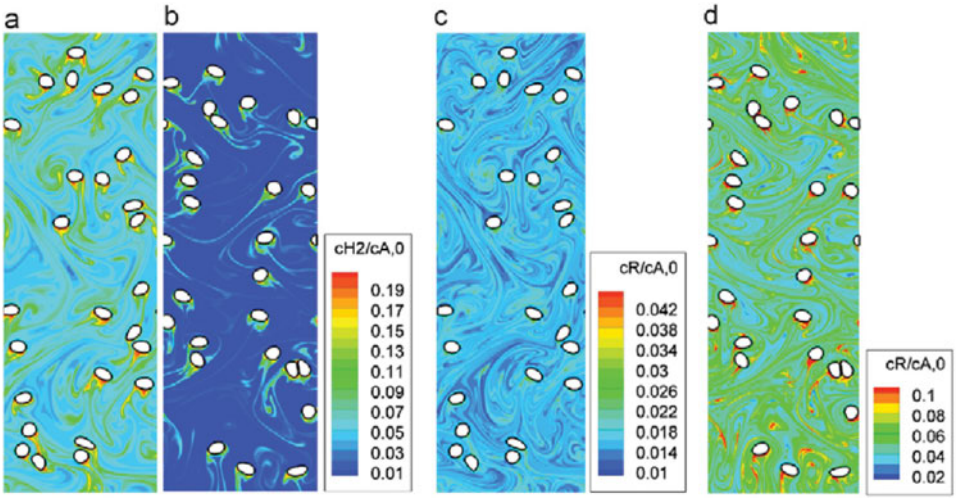


Fig. 11.2. Results from simulations of the catalytic hydrogenation of nitroarenes. The hydrogen (frames a and b) and hydroxylamine (frames c and d) concentration profiles are shown for one time for two simulations. In frames (a) and (c) the reaction rates are relatively slow, compared with the mass transfer, but in frames (b) and (d) the reaction is relatively fast. Reprinted from Radl *et al.* (2008), with permission from Elsevier.