Finite Element Computations for a Conservative Level Set Method Applied to Two-Phase Stokes Flow

DAG LINDBO

KTH Computer Science
and Communication

Master of Science Thesis
Stockholm, Sweden 2006

Finite Element Computations for a Conservative Level Set Method Applied to Two-Phase Stokes Flow

DAG LINDBO

Master's Thesis in Numerical Analysis (20 credits)
at the School of Engineering Physics
Royal Institute of Technology year 2006
Supervisor at CSC was Gunilla Kreiss
Examiner was Axel Ruhe

TRITA-CSC-E 2006:153
ISRN-KTH/CSC/E--06/153--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.csc.kth.se

Abstract

We mainly deal with the finite element formulation and implementation of a mass conserving level set method with application to two-phase flow. In particular, we have studied some fundamental issues in the level set computations, such as what are suitable finite elements for good accuracy, conservation of mass and efficiency. Here we find that a finite element method with linear interpolants outperforms more sophisticated quadratic and mixed order methods in many respects. We proceed to finite element calculations for the two-phase Stokes problem. Here we conclude that we get conservation of mass in flow calculations. Finally, we propose a method of having adaptive control of the level set function, and see that it helps maintain mass conservation in more demanding circumstances.

Referat

Vi gör finita elementberäkningar för en konservativ *level set*-metod tillämpad på Stokes-flöde med två faser. Specifikt så har vi studerat vilka finita element som ger bra noggrannhet, konservering av massa och efektivitet. Vi finner att element med linjär interpolation ger bättre resultat än mer sofistikerade, kvadratiska och blandade, element. Sedan gör vi földesberäkningar för Stokes tvåfasproblem med denna *level set*-metod. Här finner vi att vi kan få konservering av massa. Slutligen föreslår vi en metod för att ha adaptiv kontroll över representationen av gränsutan, och vi ser att detta hjälper masskonservaringen i mer krävande situationer.

Contents

Chapter 1

Introduction

The early 19th century saw the development of the mathematical description of fluid dynamics by some of the great intellectual heroes of the applied sciences: Euler, Stokes, Navier and others. The equations formulated in this era came to ever growing relevance in the second half of the 20th century, as computers started to provide the means of finding their approximate solutions by using numerical techniques.

Our ability to model different types of flows today is rather remarkable, but two topics remain debated and incomplete in both modeling and solution technique: turbulence and fluid-fluid interaction - two phenomena we see around ourselves all the time. This thesis shall shed a small amount of light on the second of these issues, the multi-phase flow problem.

Think , for example, of how droplets of oil behave in water. The primary challenge in multi-phase flow is of how to model the internal boundaries that separate the different fluids. We must define, in mathematical terms, these boundaries and how they move with, and affect, the flow. From an intuitive point of view this may seem simple, but we must have a model that is suitable for accurate and robust numerical computation. The representation and movement of the internal boundaries we call interface tracking. We must also take into account that different physical parameters govern the behavior of each fluid - giving rise to discontinuities of both physical and numerical consequence. Also, the shape of the internal boundaries are significant, as manifested by surface tension. There are several models that take all these concerns into account, but no winner is clear at this stage. And no one has proposed a model that deals with the contact behavior between a solid wall and a fluid-fluid interface in a convincing manner.

Alas, this thesis shall not deal with these challenges. Rather we shall focus on some of the interesting numerical questions that pertain to one class of methods for tracking the interface: level set methods (introduced in chapter 3). In particular we shall study the recent variant of the level set method proposed by Olson and Kreiss (restated in chapter 3), that promises to remedy the major drawback of level set methods applied to the multi-phase flow problem: that mass is not conserved in each fluid.

In their original paper Olsson and Kreiss provided an implementation based on finite difference schemes that provided excellent numerical results. For reasons of geometry and adaptivity, two common considerations in flow calculations, a finite element method was later proposed. Though extensive, this implementation left some questions to be addressed that are of interest in a finite element solution of this model. In this thesis we shall deal with what are suitable choices for finite elements and introduce some alternative discretizations of the level set equations, amongst other things.

We have made a new implementation for the this model in a finite element framework that offers great choice in the particular finite element used. Using this we can make interesting studies of which element is most suitable for these calculations. The implementation is object oriented (in C++), such that the level set solver can be coupled to an appropriate multi-phase flow solver. Here, we have implemented a solver for flow at low Reynolds number, i.e. Stokes flow. This allows us to test the conservative level set method in practical flow calculations - and the results are good.

The disposition of this thesis is as follows: Chapter 2 gives a brief summary of the finite element method. Then we introduce the level set method of Osher et. al. and the recent development by Olsson and Kreiss, in chapter 3. Chapter 4 introduces the two-phase Stokes problem, as an application of the level set method to multiphase flow. Chapters 5 and 6 develop finite element methods for these formulations, and the method is summed up in chapter 7. Then follows two chapters that present numerical results for the level set and Stokes calculations respectively, followed by a brief description of the implementation. Chapter 11 presents some novel level set results on the possibility of having adaptive control of the level set function, including a discussion about the calculation of interface curvature.

First and foremost I would like to thank my supervisor Gunilla Kreiss for never tiring of my questions and for her general enthusiasm and support of my work. For their valuable suggestions and our inspiring discussions I would like to thank Erik von Schwerin, Alexei Loubenets and Johan Hoffman of the NA-group at KTH. To a large extent, this project has been about implementation of the methods presented. This has taken a substantial amount of time and effort, so I would like to thank the FEniCS developers, in particular Anders Logg, for helping me with numerous technical issues that made the implementation a success.

Chapter 2

Elementary FEM

Here we shall give an overview of the finite element method (FEM), which will be used to solve all PDEs in the chapters that follow. The goal here is mainly to introduce notation and outline the fundamental concepts that make up FEM, such that the reader who has seen FEM before feels comfortable. We refer readers unfamiliar with FEM to [1] or [2], and the more advanced reader who seeks more details to [3]. We shall dwell briefly on some issues that will become relevant later, but omit many interesting aspects of the FEM, especially most aspects of its implementation.

2.1 Preliminaries

2.1.1 Boundary value problem

We wish to solve a PDE

$$\mathscr{L}u = f, \tag{2.1}$$

on a given domain Ω. Here, $u = u(x, y, ...) = u(\mathbf{x})$ and $\mathscr{L}$ is a linear differential operator. We must also impose boundary conditions, usually of type Dirichlet or Neumann. Dirichlet conditions will be written as

$$u(\mathbf{x}) = g(\mathbf{x}), \quad \forall \mathbf{x} \in \partial\Omega$$

and Neumann conditions as

$$\nabla u(\mathbf{x}) \cdot \mathbf{s} = g(\mathbf{x}) \quad \forall \mathbf{x} \in \partial\Omega,$$

where $\mathbf{s}$ is a unit normal to the boundary. The PDE and boundary conditions constitute our boundary value problem (BVP).

2.1.2 Inner product, weak solution and basis

It is necessary to equip ourselves with a space of functions, $\mathscr{W}$, that is rich enough to include all solutions of a BVP on Ω. We introduce the Euclidean inner product

$$(v, w) = \int_\Omega vw d\mathbf{x}$$

and norm

$$||v|| = \sqrt{(v, v)}$$

taking care that there are no elements in $\mathscr{W}$ such that this integral is divergent. Taking $u \in \mathscr{W}$ we say that u is a weak solution of (2.1) if

$$(\mathscr{L}u - f, w) = 0 \quad \forall w \in \mathscr{W} \tag{2.2}$$

We shall, in fact, seek an approximate solution, $\tilde{u}$, in a finite dimensional subspace $\mathscr{W}_h \subset \mathscr{W}$, with basis $\{w_k : k = 1...n\}$, i.e.

$$\tilde{u}(\mathbf{x}) = \sum_{k=1}^{n} c_k w_k(\mathbf{x}), \quad c_k \in \mathbb{R}.$$

We shall add requirements on the members of $\mathscr{W}_h$ as we construct the FEM, taking the first requirement to be that the inner product exists.

2.1.3 Variational formulation and forms

Take a basis $\{w_k\}$ of a finite dimensional function space $\mathscr{W}_h$. The mapping

$$L : \mathscr{W}_h \to \mathbb{R}$$

is called a linear form if it is linear in its argument. In the same manner, a bilinear form is a mapping

$$a : \mathscr{W}_h \times \mathscr{W}_k \to \mathbb{R},$$

where $\mathscr{W}_k$ is a finite dimensional function space too, possibly the same as $\mathscr{W}_h$. If we let

$$L(w) = \int_\Omega fw d\mathbf{x}$$

and

$$a(u, w) = \int_\Omega (\mathscr{L}u) w d\mathbf{x}$$

we have written the weak solution of (2.1) in terms of multi-linear forms

$$a(u, w) = L(w). \tag{2.3}$$

This we shall refer to as the variational formulation of the PDE.

2.2 The finite element method

Introducing the residual, $r(\mathbf{x}) = \mathscr{L}\tilde{u}(\mathbf{x}) - f(\mathbf{x})$, we see that the weak approximate solution of (2.1) becomes

$$(r, w) = 0 \quad \forall w \in \mathscr{W}_h. \tag{2.4}$$

This says that the residual is orthogonal to the entire subspace $\mathscr{W}_h$, i.e. (2.4) is a Galerkin condition, familiar from linear algebra!

If we take all v_k to satisfy zero Dirichlet boundary conditions we get, by construction, that $\tilde{u}$ satisfies the Dirichlet BVP above. The case of Neumann conditions is more complicated, and will be omitted here, but the choice that all basis functions satisfy zero boundary conditions is still essential. This is the second requirement on the members of $\mathscr{W}_h$.

Our goal here is to find a set of coefficients $\{c_k\}$ such that $\tilde{u}$ is an approximate weak solution according to (2.2). Taking w_l in the basis of $\mathscr{W}_h$ we have

$$\begin{aligned} (f, w_l) &= (\mathscr{L}\tilde{u}, w_l) \\ &= (\mathscr{L} \textstyle\sum_{k=1}^{n} c_k w_k, w_l) \\ &= \textstyle\sum_{k=1}^{n} c_k \left(\mathscr{L} w_k, w_l\right). \end{aligned}$$

Clearly, we have a linear system of equations $Ax = b$ for the unknown coefficients $x = [c_1, c_2, ..., c_n]$, where

$$A_{kl} = (\mathscr{L} w_k, w_l) = a(w_k, w_l)$$

and

$$b_k = (f, w_l) = L(w_l).$$

Thus, we see the convenience of writing the weak solution as a set of multi-linear forms. Note that the dimension of $\mathscr{W}_h$ is n, so A is square. The integrals can be evaluated by quadrature, but there are still a lot of matrix elements and the integrals are over all of Ω.

We have so far only derived a system of equations that approximately solve (2.1) under the Galerkin condition (2.4). Depending now on the specific choice of basis functions, one can obtain a wealth of numerical methods.

To get the FEM, we take a decomposition of Ω, e.g. a triangulation in $\mathbb{R}^2$, into M cells:

$$\Omega = \bigcup_{i=1}^{M} T_i$$

and let the inner product

$$(v, w) = \sum_{i=1}^{M} \int_{T_i} vw d\mathbf{x} = \sum_{i=1}^{M} (v, w)_{T_i}.$$

We have the matrix elements

$$A_{kl} = \sum_{i=1}^{M} (\mathscr{L} w_k, w_l)_{T_i}.$$

The crucial point here is that we may reduce the cost of computing the matrix elements a lot since we have considerable freedom in our choice of the basis functions. In particular, we may choose w_k to vanish almost everywhere. Then we get $(w_k, w_l) = 0$ for most combinations of k and l, making A sparse.

For basic and intuitive purposes, we may say that the finite element F_i is T_i and all w_k that have support on T_i. In some sense we may then understand the finite element method as an efficient way of calculating A, that uses localized basis functions. We see that A is assembled from contributions from each finite element.

In algorithmic form we write the assembly as

Algorithm 1 Assembly of A

```
$A = 0$
**for all** $T_k \subset \Omega$ **do**
  $C = \{m : \mathrm{supp}(w_m) \cap T_k \neq 0\}$
  **for all** $j \in C$ **do**
    **for all** $i \in C$ **do**
      $A_{ij} = A_{ij} + a(w_i, w_j)_{T_i}$
    **end for**
  **end for**
**end for**
```

See [19] for more details on the practical computations of general FEM.

The space of functions $\mathscr{W}_h$ must contain the basis functions w_k and a suitable number of their derivatives, so that $(\mathscr{L} w_k, w_l)$ exist. Clearly, if $\mathscr{L}$ is of order p (i.e. contains no higher derivatives than p:th), w_k should be p times non-zero differentiable, these derivative must lie in $\mathscr{W}$ and they must be bounded. However, it is often possible to do partial integration on the variational formulation, reducing $\mathscr{L}$ from order p to $p/2$. This reduces the requirements on $\mathscr{W}_h$ significantly.

So, the third and fourth restrictions on the members of $\mathscr{W}_h$ is that they have compact support and a suitable number of derivatives.

2.3 Finite elements

Thus far we have seen that the FEM splits the domain and solution space into finite elements. Constructing the basis functions has not been dealt with, and we shall only give a simple example here. Note that we may talk of e.g. "a finite element for ϕ", by which we mean a single reference finite element and not the whole set of triangulation and $\mathscr{W}_h$ that the expansion of ϕ requires. For future reference, we shall introduce some terminology for speaking of a reference element.

- The *shape* of a finite element is the geometry that defines the decomposition of Ω. In 2D, one usually has a triangle and in 3D a tetrahedron.

- The *type* of finite element is the class of functions that the basis function belongs to. The most common is Lagrange polynomials.

- The *order* of a finite element is roughly the order of the interpolation polynomial, e.g. a finite element of quadratic Lagrange polynomials has order two. If the basis functions are linear, the order is one etc.

When we construct a finite element method we choose a reference element and understand all other finite elements as appropriate transformations of the reference element.

An example is in order. In 1D, the simplest finite element takes the interval [0 1]. Based on the *chapeau* function

$$f(x) = \begin{cases} x+1, & x \in [-1\ 0) \\ -x+1, & x \in [0\ 1] \\ 0, & |x| > 1 \end{cases},$$

we get a finite element with two basis functions $f_1 = f(x)$ and $f_2 = f(x-1)$. Say we have $\Omega = [0\ 1]$, and we "triangulate" it by taking a partition with intervals of length h. A finite element method based on the element above will get

$$w_k(x) = f(x/h - k),$$

as plotted in Fig 2.1. The transformation from the reference element is merely a scaling and translation. We note that $\tilde{u}(x_k) = c_k$, since all other basis functions are zero on node k. This finite element will get A as a tridiagonal matrix - a very easy system to solve. Furthermore, as we see in Fig 2.1, on element i only w_i and w_{i-1} have support. Thus, the assembly of A will be computed as contributions of only four local element integrals per cell of the triangulation (see Alg. 1).

The finite element of type Lagrange, order 2 is somewhat analog to the linear element above. One has a piecewise quadratic polynomial that still has the property of being one on x_k and zero on all other nodes. The basis function is still continuous, but there is no second derivative on the nodes and edges of the triangulation.

More general elements can have continuous derivatives - such as spline interpolant elements. Such expressions, however, quickly become cumbersome and costly to evaluate. Additionally, in higher dimensions, the basis functions become substantially more complicated. The simplest one on a triangular element is akin to a pyramid.

2.4 Differentiation of functions in Lagrangean FE spaces

In this section the goal is to take care of the glaring problem of how to take derivatives of functions that are defined in terms of a particular piecewise polynomial

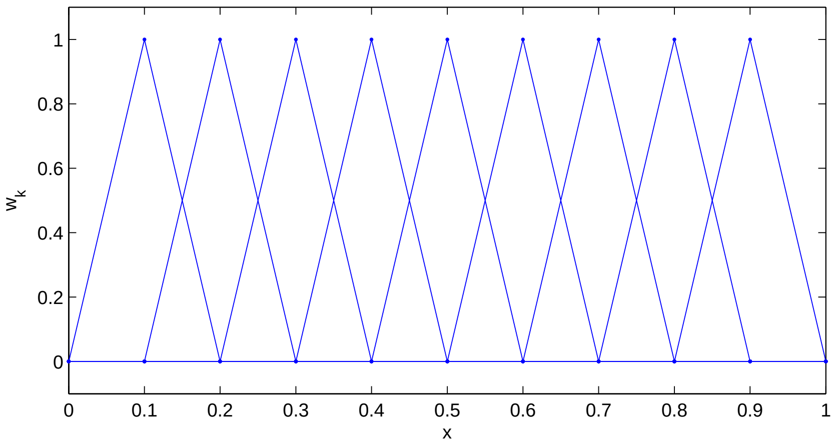


Figure 2.1. Chapeau functions w_k, $k = 1, ..., 9$ for $h = 0.1$

basis. Let

$$\phi = \sum_{k=1}^{n} c_k w_k$$

be piecewise polynomial of some order. On a Lagrange finite element, ϕ is continuous but not smooth. In particular, the derivative of ϕ is undefined in the nodes of the triangulation and across the edges between neighboring cells. Thus, there is no ordinary function, g, such that $g = D^{\alpha}\phi$. First we introduce the concept of a weak derivative and show some properties of such. Then we deal with differentiation from a local (i.e. element centered) point of view.

2.4.1 Global weak derivatives

For example, take ϕ to be piecewise linear and continuous over some $\Omega \subset \mathbb{R}$. Clearly, the derivative of such a ϕ is ill-defined in the nodes and constant between nodes, as in Fig (ref). It will be shown that this derivative exists in a weak sense - a matter that will require a bit of functional analysis.

Definitions

These definitions have been adopted from [3].

The space $L^1_{loc}(\Omega)$ is the space of all functions that are L^1-integrable in a compact interior of Ω. Next we define the weak derivative:

Take $f \in L^1_{loc}(\Omega)$. If there exists a $g \in L^1_{loc}(\Omega)$ that satisfies

$$\int_{\Omega} g w d\mathbf{x} = (-1)^{|\alpha|} \int_{\Omega} f w^{(\alpha)} d\mathbf{x}, \quad \forall w \in C_0^{\infty}(\Omega) \tag{2.5}$$

we say that g is a weak derivative of f: $g = D^{\alpha}_{w} f$.

First, we note that a continuous piecewise polynomial, order p, is locally L^1-integrable. The weak derivative to such a function is intuitively a piecewise polynomial of order $p-1$, discontinuous at a finite set of points. Let us investigate this:

Weak first derivative in a Lagrange FE space

Without loss of generality, take $\Omega = [0\ 1]$ and

$$f = \begin{cases} a_1x^2 + b_1x + c_1, & 0 \le x \le 0.5 \\ a_2x^2 + b_2x + c_2, & 0.5 < x \le 1 \end{cases}$$

Clearly, we would like

$$g = D_w f = \begin{cases} 2a_1x + b_1, & 0 \le x < 0.5 \\ 2a_2x + b_2, & 0.5 < x \le 1 \end{cases}$$

Now, attempting to verify this, we compute

$$\begin{aligned} \textstyle\int_0^1 fw'dx &= \textstyle\int_0^{0.5} fw'dx + \int_{0.5}^1 fw'dx \\ &= \langle fw\rangle_0^{0.5} - \textstyle\int_0^{0.5}(2a_1x+b_1)wdx + \langle fw\rangle_{0.5}^1 - \int_{0.5}^1(2a_2x+b_2)wdx \\ &= \langle fw\rangle_0^{0.5} + \langle fw\rangle_{0.5}^1 - \textstyle\int_0^1 gwdx \end{aligned}$$

We may put a constraint on the coefficients in f so that f is continuous. Then we have, in the indicated limit, that $(fw)(0.5-) = (fw)(0.5+)$ and

$$\int_0^1 fw'dx = -\int_0^1 gwdx + \langle fw\rangle_0^1.$$

That is, we get g as the weak derivative of f plus some boundary term. Note that if we have this in a FEM setting, then the test function w will satisfy zero Dirichlet conditions. Then we have $g = D_w f$ and our intuition holds.

Weak second derivative in Lagrange FE space

Lets take the same f as above and attempt to verify that there exists a weak second derivative given by

$$g = D_w^2 f = \begin{cases} 2a_1, & 0 \le x < 0.5 \\ 2a_2, & 0.5 < x \le 1 \end{cases}$$

The procedure is the same as in the calculation above, but we do integration by parts twice. The result is

$$\begin{aligned} \textstyle\int_0^1 fw''dx &= \textstyle\int_0^1 gwdx + \langle fw'\rangle_0^1 - \langle f'w\rangle_0^{0.5} - \langle f'w\rangle_{0.5}^1 \\ &= \textstyle\int_0^1 gwdx + \langle fw'\rangle_0^1 + (f'w)(0.5-) - (f'w)(0.5+) \end{aligned}$$

As before, we have used the continuity assumption on f. The result is clear: There is no weak second derivative, i.e. g does not satisfy (2.5). The additional terms are all non-zero - the boundary term is non-zero because there is no assumption that the derivative of the test function should vanish on the boundary, and the interior terms are unequal since there is no assumption on continuity of f'.

We shall need this result later, when we compute the curvature of the interface. It is a rather unsatisfactory result, since it would indicate that we cannot compute quantities that involve two or more derivatives of a known function. The solution to this is due to how the entries of the system matrix A is computed, i.e. as a superposition of contributions from the individual elements. In the FEM setting, it is thus possible to talk of derivatives defined locally.

2.4.2 Local derivatives

In the FEM setting we may cast the differentiation concept into another form. This is due to the assembly, Alg 1, where one adds up contributions from one element at a time. For practical computational purposes, we will thus only be concerned with the existence of element integrals. By construction, Lagrange polynomial basis functions are smooth in all such intervals - and derivatives may be taken locally, i.e. on each element.

Chapter 3

The level set method

The level set method (LSM) is a technique for interface tracking. It is one of several methods that have been successfully applied to the multiphase flow problem. In this chapter we shall briefly describe the LSM and some recent developments that are central to this thesis.

3.1 Level set foundations

Take a domain $\Omega \subset \mathbb{R}^n$. Let $\Omega_1 \subset \Omega$ and $\Omega_2 = \Omega \backslash \Omega_1$. We may say that the interface is the finite intersection of all closed sub-domains,

$$\Gamma = \left\{ \mathbf{x} : \mathbf{x} \in \cap \overline{\Omega_i} \right\},$$

meaning that Γ is the internal boundary between Ω_1 and Ω_2.

Now consider the *level set function* $\zeta(\mathbf{x}) \in \mathbb{R}$, $\mathbf{x} \in \Omega$ as a signed distance function, such that

$$\zeta(\mathbf{x}) = \left\{ \begin{array}{ll} \min_{\mathbf{x}_i \in \Gamma} ||\mathbf{x} - \mathbf{x}_i||, & \mathbf{x} \in \Omega_1 \\ -\min_{\mathbf{x}_i \in \Gamma} ||\mathbf{x} - \mathbf{x}_i||, & \mathbf{x} \in \Omega_2 \end{array} \right. . \tag{3.1}$$

We may thus define the interface between the two regions as being the implicit hypersurface

$$\Gamma = \{\mathbf{x} : \zeta(\mathbf{x}) = 0\}.$$

3.2 Level set advection and reinitialization

In some flow field $\mathbf{u} \in \mathbb{R}^n$, tracking the interface is a simple matter of solving a transport equation,

$$\zeta_t + \mathbf{u} \cdot \nabla \zeta = 0. \tag{3.2}$$

Thus far, the LSM was introduced by Osher et. al. in [4].

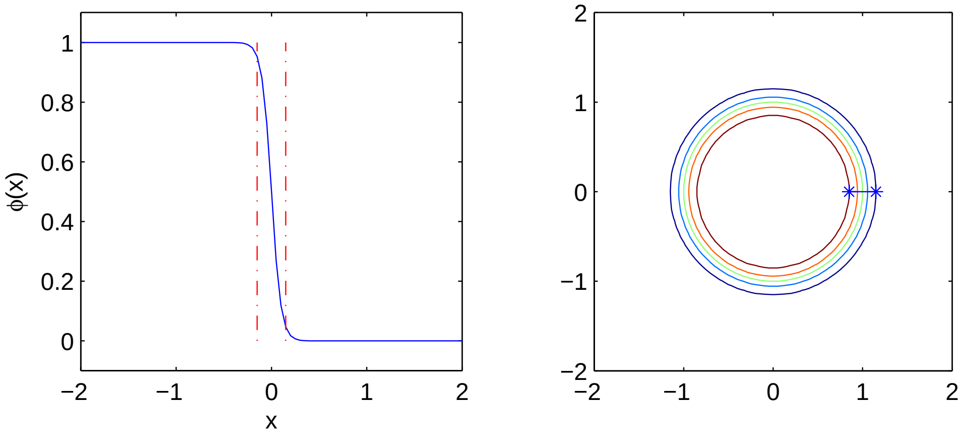


Figure 3.1. Left: 1D example of interface (3.5). Right: Contours of 2D bubble, (3.6) . Dashed line in left plot and line with stars in right plot indicate interface thickness, 6ε.

Unfortunately, ζ will not retain its property of being a signed distance function under the advection time-step. To address this serious drawback, it was suggested by Smereka et. al. in [5] that we consider

$$\zeta_\tau(\mathbf{x},\tau) + \operatorname{sgn}(\zeta_0)(|\nabla\zeta(\mathbf{x},\tau)| - 1) = 0. \tag{3.3}$$

Here, τ is time, but not the same time as t. Given a $\zeta_0 = \zeta(\mathbf{x}, t_n)$, we iterate the above equation until a steady state with respect to τ is reached. At that point, ζ will be restored to the desired distance function - a reinitialization of ζ.

Taking an advection time-step with (3.2) will move the interface to the correct location, but distorting the profile so that ζ is no longer a signed distance function (3.1). The reinitialization process in (3.3) will restore this property, but as it turns out, it will also move the interface.(REF???) This implies a loss of conservation of the volumes enclosed by Ω_1 and Ω_2.

3.3 Conservative level set method

The lack of conservation of mass makes the LSM significantly less appealing for the two-phase flow problem. Here we shall present a recent variant of the LSM that we expect shall not suffer from this lack of conservation. This formulation was first proposed by Olsson and Kreiss in [6].

We shall replace the expression for the level set function (3.1), with a regularized Heaviside function, ϕ, as in Fig 3.1. This function goes rapidly and smoothly from zero to one across a transition region of thickness 6ε centered on the interface.

With this level set function, the interface is instead defined by the 0.5-contour of ϕ,

$$\Gamma = \{\mathbf{x} : \phi(\mathbf{x}) = 0.5\}. \tag{3.4}$$

In 1D, a convenient function with this property is

$$\phi(x) = \frac{1}{1 + e^{x/\varepsilon}}, \tag{3.5}$$

giving an interface at $x = 0$. In 2D, we may want a circular bubble centered at $\mathbf{x}_c$ with radius r,

$$\phi(\mathbf{x}) = \frac{1}{1 + e^{(||\mathbf{x}-\mathbf{x}_c||-r)/\varepsilon}}, \tag{3.6}$$

Expression (3.6) will be a typical initial condition on ϕ later on.

Since we are interested in conservation properties of the method, we shall restrict ourselves to only considering divergence free velocity fields, $\nabla\cdot\mathbf{u} = 0$. In that setting, (3.2) becomes

$$\phi_t + \nabla \cdot (\phi\mathbf{u}) = 0 \tag{3.7}$$

We shall refer to this as the level set advection equation.

The original reinitialization, (3.3), is replaced by

$$\phi_\tau(\mathbf{x}, \tau) + \nabla \cdot [\phi(\mathbf{x}, \tau)(1 - \phi(\mathbf{x}, \tau))\hat{n}] = \varepsilon\nabla \cdot [\hat{n}(\nabla\phi(\mathbf{x}, \tau) \cdot \hat{n})], \tag{3.8}$$

where

$$\hat{n} = \hat{n}(\mathbf{x}) = \frac{\nabla\phi(\mathbf{x}, \tau = 0)}{|\nabla\phi(\mathbf{x}, \tau = 0)|} \tag{3.9}$$

is the unit normal to the interface. By solving (3.8) until steady state we will get ϕ as a regularized step function without changing the volume bounded by the 0.5-contour [6].

It is clear the the reinitialization equation should be solved under homogeneous Neumann boundary conditions, saying that there is no net transport of mass into or from the system during the reinitialization. Those are also suitable conditions for the advection equation.

Chapter 4

The two-phase Stokes problem

For incompressible flows at low Reynolds number Stokes equations are appropriate. While those equations have been established for over a century, it is far from clear how one should extend them to deal with the problem of two, or more, fluids present in the same system.

What we do in the level set framework is to incorporate the level set function into the Stokes equations in such a way that the one set of equations is valid in all of Ω. This is possible because the level set function is defined globally. The alternative could be considered more direct, but more substantially more complicated: solve Stokes equations for each domain, Ω_1 and Ω_2, separately and impose appropriate boundary conditions on the internal boundary - we shall not do that.

4.1 Stokes equations for two-phase flow

Stokes equations are given by

$$\left\{ \begin{array}{lcl} \nabla p(\mathbf{x}) & = & \frac{\Delta \mathbf{u}(\mathbf{x})}{Re} + \mathbf{F}(\mathbf{x}) \\ \\ \nabla \cdot \mathbf{u}(\mathbf{x}) & = & 0. \end{array} \right. \tag{4.1}$$

Here, $\mathbf{u}(\mathbf{x}) \in \mathbb{R}^n$ is the velocity components, $p \in \mathbb{R}$ is the pressure, Re is the Reynolds number, and $\mathbf{F} \in \mathbb{R}^n$ is the force due to surface tension. In the level set setting, (4.1) is solved over all of Ω, and the Reynolds number is thus not a constant:

$$Re = Re(\mathbf{x}) = \left\{ \begin{array}{ll} Re_1, & \mathbf{x} \in \Omega_1 \\ Re_2, & \mathbf{x} \in \Omega_2 \end{array} \right.$$

We wish to have a similar expression that uses the level set function, ϕ:

$$Re(\mathbf{x}) = Re_1 + (Re_2 - Re_1)\phi(\mathbf{x}). \tag{4.2}$$

This expression will go smoothly from Re_1 to Re_2 across the interface.

A multitude of boundary conditions for the Stokes problem could be interesting and reasonable to impose. We shall use inflow conditions on the velocity on one boundary, no slip conditions on some boundaries and zero pressure on the outflow boundary (with homogenous Neumann conditions on the velocity).

All that remains is to define $\mathbf{F}$ in terms of ϕ.

4.2 The surface tension

4.2.1 The ordinary way

The usual formulation from the level set literature, e.g. [4], is to let

$$\mathbf{F} = \sigma\kappa\hat{n}\delta. \tag{4.3}$$

Here, σ is the coefficient of surface tension, $\hat{n}$ is the unit normals to the interface given by (3.9), $\kappa = \kappa(\mathbf{x})$ is the curvature of the interface, and δ is a discrete variant of the Dirac distribution. If we let

$$\delta = |\nabla\phi(\mathbf{x})|, \tag{4.4}$$

then (4.3) becomes

$$\mathbf{F} = \sigma\kappa(\mathbf{x})\nabla\phi(\mathbf{x}). \tag{4.5}$$

A derivation of (4.3) can be found in [8].

4.2.2 Another way

A less common way to write down the surface tension, found in [9], is as follows

$$\begin{aligned} \mathbf{F} &= \nabla \cdot T \\ T &= \sigma(I - \hat{n}\hat{n}^T)\delta. \end{aligned} \tag{4.6}$$

Here we have the identity tensor, I, of rank two and the outer product, $\hat{n}\hat{n}^T$, yielding a tensor of rank two as well. As before, δ may be computed with (4.4), and σ is a constant. One apparent advantage with (4.6) over (4.5) is that the curvature need not be computed at all. Another advantage is the ease with which this expression can be integrated partially.

4.3 The curvature

Recall that we have the interface defined as the implicit hypersurface $\phi(\mathbf{x}) = 0.5$. As with the surface tension, there are two ways of computing the curvature of the interface. In later chapters we shall see the inherent difficulty of evaluating these expressions in a FEM setting.

4.3.1 The ordinary way

The level set literature, e.g. [7], present the following formula for the curvature:

$$\kappa = -\nabla \cdot \hat{n}, \tag{4.7}$$

with $\hat{n}$ as in (3.9). Expression (4.7) is in fact often taken as a definition of the curvature of an implicit surface.

4.3.2 Another way

In the case of a planar implicit curve there is, however, an interesting alternative that can be easily derived from Frenets' formulas for parametric curves. We shall outline the derivation here and refer to [10] for details.

Introduce the tangent vector, $\mathrm{Tang}(\phi) = [-\phi_y,\ \phi_x]$. Take an implicit curve, on standard form, $p(\mathbf{x}) = \phi(\mathbf{x}) - 0.5 = 0$ and introduce two parameterizations of this curve, $P(s)$ and $P(t)$. From differential geometry we have Frenets' formula for a parametric curve $P(s)$:

$$\kappa = -\frac{\mathrm{d}N}{\mathrm{d}s} \cdot T(P)^T,$$

where $N = \frac{P_s \times P_t}{|P_s \times P_t|}$ is the normal and $T = \frac{\mathrm{d}P}{\mathrm{d}s}$ is the tangent. By using the ordinary chain and quotient rules one easily gets an expression for the curvature of the implicit curve $p(\mathbf{x}) = 0$:

$$\begin{aligned} \kappa &= \frac{\mathrm{Tang}(\phi)\mathrm{Hess}(\phi)\mathrm{Tang}(\phi)^T}{||\nabla\phi||^3} \\ &= \frac{\begin{bmatrix} -\phi_y & \phi_x \end{bmatrix} \begin{bmatrix} \phi_{xx} & \phi_{xy} \\ \phi_{yx} & \phi_{yy} \end{bmatrix} \begin{bmatrix} -\phi_y \\ \phi_x \end{bmatrix}}{||\nabla\phi||^3} \end{aligned} \tag{4.8}$$

Of course, this result is invariant for $p(\mathbf{x}) = c$ and $cp(\mathbf{x}) = 0$, so (4.8) is valid for $\phi(\mathbf{x}) = 0.5$. It is possible that (4.8) is more suitable for certain numerical calculations than (4.7).

Chapter 5

FEM for the level set method

In this chapter we shall develop finite element methods for the components of the level set method. It should be understood that there are numerous alternatives concerning time discretization, stabilization and choice of finite elements. We will develop a number of methods for both advection and reinitialization, and attempt to draw some qualitative conclusions about their behavior from a theoretical perspective.

Each equation shall be written as a pair of bilinear and linear forms, as defined in chapter 2,

$$a(\phi, w) = L(w),$$

so that it is immediately clear how the linear system of equations will be assembled later on. Note that the unknown quantity, typically $\phi(\mathbf{x}, t_n + \Delta t)$, will only be present in the bilinear form and that the linear form will contain terms involving the known function $\phi(\mathbf{x}, t_n)$.

First, let w be understood as a test function belonging to some ordinary finite element space $\mathscr{W}_h$, such as piecewise polynomials. We shall discuss the alternatives for this space shortly. For brevity of notation, we shall simply write $(\cdot, w) = 0$ when we mean $(\cdot, w) = 0,\ \forall w \in \mathscr{W}_h$.

Also, we write $\phi^n \approx \phi(\mathbf{x}, t_n)$ from here on.

5.1 Advection

The natural choice for solving the advection equation (3.7) is either a pure upwind finite volume method or a one-sided finite difference scheme. The ordinary Galerkin finite element method is not stable. The reason for this is that it gives rise to a central difference-type approximation. We shall formulate such a method anyway, for reasons that will become apparent later.

Take the weak formulation of (3.7):

$$(\phi_t + \nabla \cdot (\phi \mathbf{u}), w) = 0$$

under homogeneous Neumann boundary conditions. Using the divergence theorem and natural boundary conditions we get

$$(\phi_t, w) = (\phi, \nabla w \cdot \mathbf{u}). \tag{5.1}$$

5.1.1 Euler in time

We may take the simplest possible temporal discretization of (5.1), forward Euler:

$$\frac{1}{\Delta t}(\phi^{n+1} - \phi^n, w) = (\phi^n, \nabla w \cdot \mathbf{u}),$$

so that we have a pair of forms $a(\phi, w) = L(w)$

$$(\phi^{n+1}, w) = (\phi^n, w) + \Delta t(\phi^n, \nabla w \cdot \mathbf{u}) \tag{5.2}$$

We note that this method will only be first order accurate in time.

5.1.2 Crank-Nicholson in time

To get a method which is second order in time we resort to the trusty Crank-Nicholson discretization of (5.1),

$$\frac{1}{\Delta t}\left(\phi^{n+1} - \phi^n, w\right) = \frac{1}{2}\left(\phi^{n+1} + \phi^n, \nabla w \cdot \mathbf{u}\right).$$

As a pair of forms, we have

$$(\phi^{n+1}, w) - \frac{\Delta t}{2}(\phi^{n+1}, \nabla w \cdot \mathbf{u}) = (\phi^n, w) + \frac{\Delta t}{2}(\phi^n, \nabla w \cdot \mathbf{u}) \tag{5.3}$$

5.1.3 Streamline upwind/Petrov-Galerkin stabilization

We expect that (5.3) behaves significantly better than the simple Euler formulation. But it is still not a satisfactory method, since the lack of upwind bias in the finite element will give rise to spurious oscillations in the solution. As of late, the most popular method to stabilize finite element methods for convection dominated PDEs has become the Streamline upwind/Petrov-Galerkin (SU/PG) method, introduced in [11].

There are at least two interpretations of the SU/PG stabilization, one being a transformation of the test function into something skewed to be more upwind - as in [11]. Here we shall see it as a clever residual weighting added to the original weak formulation. Let r be the residual of the advection equation (written in general form),

$$r = \phi_t + \mathbf{u} \cdot \nabla\phi.$$

The stabilized formulation can be written

$$(\phi_t, w) + (\nabla \cdot (\phi\mathbf{u}), w) + (s, r) = 0, \tag{5.4}$$

where s is the SUPG residual weighting. The expression for s, restated from [13] and [12], is

$$s = \frac{h}{2\|\mathbf{u}\|}(\mathbf{u} \cdot \nabla w).$$

Here, h is the local element size. The stabilization is thus dependent on the velocity field, so we can see that it adds some controlled amount of diffusion in the direction of characteristics. For a more thorough discussion, see e.g. [12].

It remains to take the Crank-Nicholson discretization in time of (5.4) to get a stable method for the level set advection. We have from (5.3) all but the stabilization term. The result is

$$a_{CN} + (\phi^{n+1}, s) + \frac{\Delta t}{2}(s, \nabla\phi^{n+1} \cdot \mathbf{u}) = L_{CN} + (\phi^n, s) - \frac{\Delta t}{2}(s, \nabla\phi^n \cdot \mathbf{u}), \tag{5.5}$$

where we introduced a_{CN} and L_{CN} as the left and right hand side of (5.3) respectively, for brevity.

5.2 Reinitialization

The reinitialization equation (3.8) is non-linear. However, we may perform a slight linearization and formulate the Crank-Nicholson Galerkin method without any worries about upwind bias or such. The weak solution to (3.8) fulfills

$$(\phi_\tau, w) + (\nabla \cdot [\phi(1-\phi)\hat{n}], w) = \varepsilon(\nabla \cdot [\hat{n}(\nabla\phi \cdot \hat{n})], w),$$

where the possibility of partial integration is irresistible, giving

$$(\phi_\tau, w) = (\phi(1-\phi), \nabla w \cdot \hat{n}) - \varepsilon(\nabla\phi \cdot \hat{n}, \nabla w \cdot \hat{n}). \tag{5.6}$$

Discretizing in time we get,

$$\begin{aligned} \tfrac{1}{\Delta\tau}(\phi^{n+1} - \phi^n, w) &= \left(\tfrac{\phi^{n+1}+\phi^n}{2}\left(1 - \tfrac{\phi^{n+1}+\phi^n}{2}\right), \nabla w \cdot \hat{n}\right) - \tfrac{\varepsilon}{2}(\nabla(\phi^{n+1}+\phi^n) \cdot \hat{n}, \nabla w \cdot \hat{n}) \\ &\approx \left(\tfrac{\phi^{n+1}+\phi^n}{2} - \phi^{n+1}\phi^n, \nabla w \cdot \hat{n}\right) - \tfrac{\varepsilon}{2}(\nabla(\phi^{n+1}+\phi^n) \cdot \hat{n}, \nabla w \cdot \hat{n}). \end{aligned}$$

By rearranging terms we get the variational formulation of the reinitialization,

$$\begin{aligned} (\phi^{n+1}, w) - \tfrac{\Delta\tau}{2}(\phi^{n+1}, \nabla w \cdot \hat{n}) + \tfrac{\varepsilon\Delta\tau}{2}(\nabla\phi^{n+1} \cdot \hat{n}, \nabla w \cdot \hat{n}) + \varepsilon\Delta\tau(\phi^{n+1}\phi^n, \nabla w \cdot \hat{n}) = \\ (\phi^n, w) + \tfrac{\Delta\tau}{2}(\phi^n, \nabla w \cdot \hat{n}) - \tfrac{\varepsilon\Delta\tau}{2}(\nabla\phi^n \cdot \hat{n}, \nabla w \cdot \hat{n}). \end{aligned} \tag{5.7}$$

This equation is not as complicated as it may appear. It remains to discuss how to compute the unit normals. This is, however, not as straight forward as one may think at first.

The first thing to realize is that the triangulation of Ω, the mesh, is unstructured. It is thus not possible to apply a difference approximation, based on a Taylor series, to the nodal values in a reasonable manner. Computing $\hat{n}$ according to (3.9), is done by normalizing the components of the gradient of ϕ. The general idea is simple:

If we wish to compute the gradient (in 2D, for clarity) of a known discrete function, $\mathbf{g} = [g_1\ g_2] = \nabla\phi$, we may instead write it as a pair of forms for each dimension,

$$\left\{ \begin{array}{lcl} (g_1, v) & = & (\frac{\partial\phi}{\partial x}, v) \\ (g_2, w) & = & (\frac{\partial\phi}{\partial y}, w). \end{array} \right. \tag{5.8}$$

Introduce a vector element $\mathbf{w} = [v\ w]$ and we have

$$\int_\Omega (\mathbf{g} \cdot \mathbf{w}) d\mathbf{x} = \int_\Omega (\nabla\phi \cdot \mathbf{w}) d\mathbf{x}, \tag{5.9}$$

which is equivalent to (5.8). Note that (5.9) is a valid variational form, since it is single valued, that determines a vector valued function. The difficulty with this expression is two-fold: it is not obvious that the integral will exist, and we don't know that the solution to the system of equations that (5.9) defines is well behaved.

To convince ourselves that the integral does exist, we recall the discussions in Sec 2.4 about global weak derivatives. Those results clearly show that the variational form (5.9) is well defined. Note, however, that this hinges on the continuity of ϕ. One major issue still remains.

5.3 Projections in FE spaces

Consider again the case of a continuous, piecewise linear, $\phi \in \mathbb{R}$. We have seen that a function $g^* = D_w\phi$ that satisfies (2.5) is piecewise constant. However, it is understood from (5.9) that $g = \sum c_k^* w_k$ is piecewise linear (like ϕ). So it is clear that solving the system of equations for $\{c_k^*\}$ will produce a projection from piecewise constant to piecewise linear and continuous. We shall illustrate that this solution may have some really nasty properties.

Take for example

$$g^* = \left\{ \begin{array}{ll} 0, & x < 0 \\ 1, & x \geq 0 \end{array} \right.$$

To get the projection onto a piecewise linear basis we solve the system of equations implied by (5.9), with w as chapeau functions (as in Fig 2.1), on an equidistant partition of $\Omega = [-1\ 1]$. Evaluating the variational form gives equations

$$\frac{1}{6}g_{k-1} + \frac{2}{3}g_k + \frac{1}{6}g_{k+1} = g_k^*, \quad k = 1, ..., N-1$$

and $g_0 = 0$, $g_N = 1$. The solution to this is in Fig. 5.1. Note that the computed solution deviates massively form the desired and intuitively correct solution.

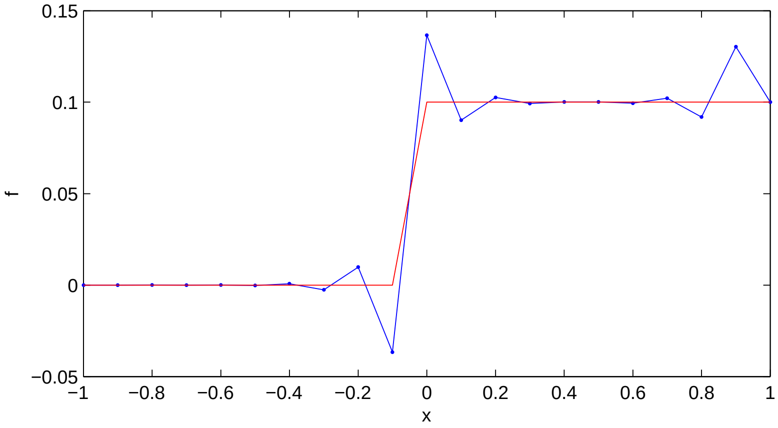


Figure 5.1. Solid line with dot: Projection of g^*, piecewise constant, onto g, piecewise linear and continuous (as computed with FEM). Solid line: The projection that one might consider most natural.

5.4 Choice of elements for the level set equations

Now the time is right to propose a suitable basis for the FEM that solves the level set advection and reinitialization equations. There are three concerns:

- Accuracy. Higher order finite elements will give higher order of accuracy in space. It could prove valuable to have higher order of accuracy in space than in time, since we will be calculating spatial derivatives of ϕ.

- Complexity. Higher order finite element methods become costly to solve, due in part to the complexity of assembling the systems.

- No projections. It is paramount to the computations that the effects we see in Fig. 5.1 do not appear.

Table 5.1 introduces notation for some finite element spaces that will be used. Note that the shape of the element is either triangular or tetrahedral, and that all elements are of type Lagrange.

5.4.1 Linear basis for ϕ

The natural choice is to let ϕ be continuous. One possibility is to let ϕ be expressed in $\mathscr{W}_c^1$, i.e. $\phi = \sum c_k w_k$, $w_k \in \mathscr{W}_c^1$. Then we would be wise to take $\mathbf{u} \in \mathscr{W}_{vc}^1$. Together with either variational form (5.2) or (5.3), the FEM for the level set advection is well defined. To get the SUPG stabilization defined we can choose $h \in \mathscr{W}_d^0$, since the cell size is constant across each cell.

$\mathscr{W}_c^1$	Linear	Continuous	
$\mathscr{W}_c^2$	Quadratic	Continuous	
$\mathscr{W}_d^0$	Constant	Discontinuous	
$\mathscr{W}_d^1$	Linear	Discontinuous	
$\mathscr{W}_{vc}^1$	Linear	Vector, continuous	$\mathscr{W}_c^1 \times \mathscr{W}_c^1$
$\mathscr{W}_{vc}^2$	Quadratic	Vector, continuous	$\mathscr{W}_c^2 \times \mathscr{W}_c^2$
$\mathscr{W}_{vd}^0$	Constant	Vector, discontinuous	$\mathscr{W}_d^0 \times \mathscr{W}_d^0$
$\mathscr{W}_{vd}^1$	Linear	Vector, discontinuous	$\mathscr{W}_d^1 \times \mathscr{W}_d^1$

Table 5.1. Finite element types.

To get the reinitialization we must first define the gradient computation. We may wish to use a mixed formulation, $g \in \mathscr{W}_{vd}^0$, due to the fact that the gradient of $\phi \in \mathscr{W}_c^1$ will be discontinuous and have polynomial degree 0. This will avoid the problems that we saw in the projection example, at the cost of reducing the accuracy of the method.

The alternative is to use a equal order method, i.e. using order 1 continuous Lagrange elements for $\hat{n}$ as well. This will be more accurate, while running the risk of bad oscillations due to the projection. In Tab 5.2, these two methods are defined.

c1/d0

Function	Element
ϕ	$\mathscr{W}_c^1$
$\mathbf{u}$	$\mathscr{W}_{vc}^1$
$\hat{n}$	$\mathscr{W}_{vd}^0$
h	$\mathscr{W}_d^0$

c1/c1

Function	Element
ϕ	$\mathscr{W}_c^1$
$\mathbf{u}$	$\mathscr{W}_{vc}^1$
$\hat{n}$	$\mathscr{W}_{vc}^1$
h	$\mathscr{W}_d^0$

Table 5.2. Set of elements for the functions for the LSM, based on a linear element for the level set function.

5.4.2 Quadratic basis for ϕ

To get an extra order of accuracy in space it can be reasonable suggest that ϕ be in $\mathscr{W}_c^2$ instead. The reasoning for the other elements is similar to the linear case. We still have a bit uncertainty whether it is more beneficial to lower the order of the element that $\hat{n}$ lives on, thus precluding a projection but losing accuracy, or to keep the same order but risk a bad projection. In Tab 5.3 two possible methods are proposed.

c2/d1

Function	Element
ϕ	$\mathscr{W}_c^2$
$\mathbf{u}$	$\mathscr{W}_{vc}^2$
$\hat{n}$	$\mathscr{W}_{vd}^1$
h	$\mathscr{W}_d^0$

c2/c2

Function	Element
ϕ	$\mathscr{W}_c^2$
$\mathbf{u}$	$\mathscr{W}_{vc}^2$
$\hat{n}$	$\mathscr{W}_{vc}^2$
h	$\mathscr{W}_d^0$

Table 5.3. Set of elements for the functions for the LSM, based on a quadratic element for the level set function.

This concludes the discussion about the finite element method for the interface tracking.

Chapter 6

FEM for the two-phase Stokes problem

In this chapter we shall formulate finite element methods for the two-phase Stokes problem, as defined in chapter 4. This shall pose additional difficulties, compared to the FEM for the LSM, in getting forms that are well defined mathematically and avoid projections.

We continue to use the notation for finite elements introduced in Tab 5.1

6.1 Variational forms for Stokes

There are several different ways to formulate a FEM for Stokes equations. We shall take (4.1) and write it in a matrix-vector form

$$\begin{bmatrix} -\frac{\Delta}{Re} & \nabla \\ \nabla & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \mathbf{F}(\mathbf{x}) \\ 0 \end{bmatrix},$$

or equivalently

$$\begin{bmatrix} -\frac{\Delta}{Re}\mathbf{u} + \nabla p \\ \nabla \cdot \mathbf{u} \end{bmatrix} = \begin{bmatrix} \mathbf{F}(\mathbf{x}) \\ 0 \end{bmatrix}. \tag{6.1}$$

We can take two test functions, $\mathbf{w} \in \{\mathscr{W}_k \times \mathscr{W}_k\}$ and $v \in \mathscr{W}_h$, and lump them into a vector $[\mathbf{w}\ v]$. Multiply (6.1) from the left with this test vector and integrate. This gives a weak form for Stokes: Find $\mathbf{u}$, p such that

$$\int_\Omega \left[-\frac{\Delta \mathbf{u}}{Re} + \nabla p \right] \cdot \mathbf{w} \mathrm{d}\mathbf{x} + (\nabla \cdot \mathbf{u}, v) = \int_\Omega \mathbf{F} \cdot \mathbf{w} d\mathbf{x}, \quad \forall \mathbf{w}, v$$

This can be integrated by parts, so that we have

$$\left(\frac{1}{Re}, \nabla \mathbf{w} \cdot \nabla \mathbf{u} \right) - (p, \nabla \cdot \mathbf{w}) + (\nabla \cdot \mathbf{u}, v) = \int_\Omega \mathbf{F} \cdot \mathbf{w} d\mathbf{x}. \tag{6.2}$$

Note that both Re and $\mathbf{F}$ vary in space, as functions of $\phi(\mathbf{x}, t_n)$, in the level set two-phase setting. There is no need to compute the surface tension force density explicitly. Expressions (4.5) and (4.6) are understood as candidates for substitution into the particular Stokes form. In practice, it may be beneficial to compute **F** separately (so as to guard against numerical breakdown related to the unit normals for example).

It is well known how to formulate a stable FEM for (6.2). There are two basic approaches: pressure stabilization or mixed element formulation. Which one is appropriate depends on the choices, with respect to finite elements, that were made for the level set formulation.

6.1.1 Mixed formulation - quadratic basis

A mixed element formulation is stable if we take basis functions for **u** being of one order higher than the functions for p. We refer to [14] for a proof of this. So, if we take $\mathbf{w}$ to be of order 2 and v to be of order 1, then the ordinary Galerkin FEM for (6.2) is stable.

By inserting the ordinary expression for **F**, (4.5), we get

$$\left(\frac{1}{Re}, \nabla\mathbf{w}\cdot\nabla\mathbf{u}\right) - (p, \nabla\cdot\mathbf{w}) + (\nabla\cdot\mathbf{u}, v) = -\sigma(\nabla\cdot\hat{n}, \nabla\phi\cdot\mathbf{w}). \tag{6.3}$$

This formulation involves the computation of the gradient of $\hat{n}$ - a concern noted prior. Consider instead the tensor-based formulation (4.6) of the force density. Now we can integrate the source term partially, getting

$$\left(\frac{1}{Re}, \nabla\mathbf{w}\cdot\nabla\mathbf{u}\right) - (p, \nabla\cdot\mathbf{w}) + (\nabla\cdot\mathbf{u}, v) = -\left(\sigma\delta, (I - \hat{n}\hat{n}^T)\cdot\nabla\mathbf{w}\right). \tag{6.4}$$

This expression is highly pleasing, since there are no derivatives taken on discontinuous functions, i.e. all weak derivatives exist globally.

Table 6.1 gives the set of finite elements that make this well defined and stable.

Function	Element
ϕ	$\mathscr{W}_c^2$
$\mathbf{u}$	$\mathscr{W}_{vc}^2$
p	$\mathscr{W}_c^1$
$\hat{n}$	$\mathscr{W}_{vd}^1$

Table 6.1. Set of elements for two-phase Stokes problem, based on a choice of quadratic element for ϕ.

6.1.2 Pressure stabilized formulation - linear basis

The case of a linear finite element method requires the pressure stabilization discussed previously, and is a bit more messy than the mixed formulation. This stabilization can be seen as a transformation of the velocity basis function, and can be found in e.g. [15], [18]. We restate the result here, since we will be implementing it:

$$\mathbf{w} \to \mathbf{w} + \eta \nabla v,$$

with $\eta = \frac{h^2}{5}$ (h is the local element size). The two-phase Stokes formulation is then

$$\left(\frac{1}{Re}, \nabla \mathbf{w} \cdot \nabla \mathbf{u}\right) - (p, \nabla \cdot \mathbf{w}) + (\nabla \cdot \mathbf{u}, v) + (\eta, \nabla p \cdot \nabla v) = \int_\Omega \mathbf{F} \cdot (\mathbf{w} + \eta \nabla v) d\mathbf{x}, \quad (6.5)$$

In the same way as we got (6.4) or (6.3), we may substitute expressions for $\mathbf{F}$ into (6.5). If $\phi \in \mathscr{W}_c^1$ then $\mathbf{u} \in \mathscr{W}_{vc}^1$, and we take the other functions to live on the elements specified in table 6.2.

Function	Element
ϕ	$\mathscr{W}_c^1$
$\mathbf{u}$	$\mathscr{W}_{vc}^1$
p	$\mathscr{W}_c^1$
$\hat{n}$	$\mathscr{W}_{vd}^0$
h	$\mathscr{W}_d^0$

Table 6.2. Set of elements for two-phase Stokes problem, based on a choice of linear element for ϕ.

6.2 Remarks

It is important to note that the order-one stabilized method may suffer accuracy problems due to the choice of a piecewise constant basis function for $\hat{n}$. Taking a element in $\mathscr{W}_{vc}^1$ is the alternative, but that will incur a projection. Numerical tests will decide which path is better.

This report does not attempt to address all numerical concerns related to the Stokes formulations presented here. In particular, we we shall not draw any conclusions with regard to which method of surface tension computation is more suitable - we have seen that the tensor based formulation for the surface tension gives a more solid mathematical expression, (6.4), and without further ado we shall stick to that. More fundamentally, we shall not make strong statements to the relative merit of the mixed and stabilized forms of Stokes. They are presented to relate to the various choices we have for the level set method, and we shall let a detailed study of the level set FEM determine which method performs best in that setting - and simply choose the corresponding Stokes formulation. These two restrictions are reasonable within the scope of the project.

Chapter 7

Summary of method, and general considerations

This chapter sums up the finite element solution to the coupled LSM-Stokes problem. We also make some remarks about linear algebra and practical computations.

7.1 Putting the method together

Setup Given a mesh and boundary conditions for Stokes, we

1. Choose spatial accuracy. That is, choose the order of the finite element that ϕ shall live on. $\phi \in \mathscr{W}_h$.
2. Initial conditions. In 2D, expression (3.6) is suitable.
3. Restrict the initial level set ϕ_{IC} onto the mesh and the particular finite element, by solving the identity form
   $(\phi, w) = (\phi_{IC}, w), \ \forall w \in \mathscr{W}_h$.
4. Choose level set advection form, either Euler (5.2), CN (5.3) or CN+SUPG (5.5).
5. Choose method of surface tension computation.
6. If we have quadratic or higher order basis of ϕ, the mixed Stokes formulations (6.4) or (6.3) is preferable. In a linear setting, use stabilized Stokes formulation.

Time-step To advance from time $t_n = n\Delta t$, knowing ϕ^n, to t_{n+1} do

1. Compute the interface normals. This is done solving (5.9),using ϕ^n and normalizing the components.
2. Compute the Reynolds numbers for the transition region, using (4.2).
3. Compute a flow field, $\mathbf{u}$, by solving the appropriate Stokes problem (items five and six in setup).

4. Move the interface, by using the computed $\mathbf{u}$ and the appropriate form for the advection equation (item four in setup). This gives some intermediate ϕ^*.
5. Recompute unit normals, based on ϕ^*
6. Iterate the reinitialization equation (5.7) until steady state. This gives the desired ϕ^{n+1}

7.2 Linear algebra and performance

Computing finite element solutions consists of two main tasks: Assembly and solution of linear systems, $Ax = b$. The assembly algorithm (Alg. 1) is really the heart of the finite element method, and is considered a given in this context (see [19] for a discussion of optimality in the evaluation of matrix elements). In contrast to this, the solution of the linear system is highly problem specific. A reasonable, yet small, computation will have $\sim 10^5$ unknowns. In dense linear algebra the solution of such a system is impossible both in terms of memory and CPU-time since it requires a LU-factorization. However, the finite element method will provide sparse systems. This means we will prefer Krylov subspace iterative methods.

This makes the situation more complicated, since the convergence of such methods crucially depend on the properties of A. This is in fact a major consideration, since we will not be able to do large computations if there are no efficient ways to solve the resulting linear systems. In particular, the most efficient methods, preconditioned Conjugate Gradient, are only applicable when A is symmetric and positive definite.

In essence the assembly of a simple bilinear form gives $A_{ij} = (\mathscr{L}w_i, w_j)$, as discussed in chapter 2. The real challenge is to determine if this system is positive definite. As seen in the time-stepping procedure above, we will have to solve numerous linear systems each time step. Take the simplest form, the Euler advection form,

$$(\phi^{n+1}, w) = (\phi^n, w) + \Delta t(\phi^n, \nabla w \cdot \mathbf{u}),$$

giving $A_{ij} = (w_i, w_j) = A_{ji}$. A matrix A is positive definite if $x^T(Ax) > 0 \quad \forall x \neq 0$. For the Euler for we get

$$x(Ax) = \sum_{i,j=0}^{N} x_i(w_i, w_j)x_j > 0 \quad \forall x \neq 0$$

Thus establishing that the advective PDE FEM can be computed with the Conjugate Gradient (CG) method. It follows that A is non-singular, which means that the system will admit a unique solution. Doing the same for the Crank-Nicholson method is more complicated, but the result is the same.

The situation is very different for the Stokes system. It is not positive definite (see [14]), meaning that we have to solve the system with pivoting Gaussian elimination. Modern implementations do, thankfully, provide high performance sparse Gaussian elimination that make the solution of the Stokes system reasonable.

Some concluding remarks about finite elements: When we speak of the number of unknowns in finite element methods we really mean the number of degrees of freedom (DOFS). A linear basis finite element has three degrees of freedom per triangle (the three corners of the triangle), i.e. the number of DOFS is the same as the number of vertices in the mesh. A quadratic finite element method adds one DOF on each edge, i.e. it will have four times as many DOFS as the linear FEM.

This is significant since neither the assembly nor the Krylov methods are of linear complexity. Computing the quadratic FEM costs significantly more. On the other hand, the order of accuracy is higher in theory - so if this higher order of accuracy can be obtained in practice then the quadratic FEM will be cheaper to compute for a given accuracy.

For the coupled advection/reinitialization procedure there are many options for the finite elements, and the final order of accuracy is far from clear.

Chapter 8

Numerical Results for LSM

8.1 Overview

In broad terms we are interested in two properties of the finite element methods presented in Ch. 5: conservation of mass, and order of accuracy. Getting conservation of mass is paramount since it fixes a deficiency in previous level set methods.

We demonstrated in Ch. 5 that it is not clear how one ought to construct a FEM for the advection and reinitialization steps. In particular, some open questions are:

- A classical Galerkin method for the advection step is not stable by itself. However, the subsequent reinitialization is expected to squash any oscillations that occur due to this instability. Is it necessary to use SUPG at all?
- There is a risk of numerical errors due to projections between different orders of basis functions. On the other hand, using mixed methods could lower the order of accuracy in the reinitialization step. Which of these conflicting concerns should one heed?
- Will the answer to the previous question depend on what order of element we choose for ϕ? That is, will a mixed "c1/d0" method have the same properties as a "c2/d1" method?
- What will the order of accuracy be for the methods presented in Ch. 5?
- Is it necessary to chose a higher order method to realize the goal of conservation of mass?

We shall perform numerical experiments in the following sections to address these questions to the best of our ability. For this we will need two model problems.

8.2 Model problems

The model problems are not really flow problems - they are advection problems. This entails two features:

- **u** is constant in time.
- The shape of the interface does not affect its behavior in any way.

The bulk of the numerical work is done in two spatial dimensions. Getting the same results in 3D as in 2D demands time and computational resources that are not available, and is unlikely to be more interesting. The implementation works fine 3D though, and without modifications. We use the same model problems as Kreiss and Olsson use in the paper that introduced the conservative level set method, [6].

8.2.1 Rotating bubble

We let

$$\mathbf{u} = \begin{bmatrix} -y \\ x \end{bmatrix}, \tag{8.1}$$

and the domain be $\Omega = [-1\ 1] \times [-1\ 1]$. This advection field can be seen in Fig 8.7. Note that this **u** will not alter the shape of the bubble. For that reason it is useful for investigating the accuracy of the computed interface positions at suitable times (where we have an exact solution to compare with).

8.2.2 Bubble in vortex

A more challenging situation will deform the interface. For this purpose we take

$$\mathbf{u} = \begin{bmatrix} \sin(\pi x)^2 \sin(2\pi y) \\ -\sin(\pi y)^2 \sin(2\pi x) \end{bmatrix}, \tag{8.2}$$

and the domain to be $\Omega = [0\ 1] \times [0\ 1]$. This field can be seen in Fig 8.7.

8.3 Conservation of mass

Let $\Omega_1 \subset \Omega$ be enclosed by an interface that is initially given by a function, such as 3.6. In 2D, conservation of mass means that the areas of Ω_1 is constant. So we define

$$I = \int_{\Omega_1} d\mathbf{x} = \int_{\Omega} \left(\frac{|\phi - 0.5|}{\phi - 0.5} + 1 \right) d\mathbf{x}, \tag{8.3}$$

and strive to see that I be constant as the interface moves. The computation of of this integral is done in a wasteful but robust manner (which should be fine for numerical tests): we take a linear interpolation from the computed nodal values and sample the integrand. With a large number of sample points the integral was seen to be well converged.

8.3.1 Conservation results for different methods

We have formulated a wealth of methods for the LSM advection phase. First we look at the possible benefits to be had when using a higher order finite element, and then we look at what effects the temporal discretization has on conservation.

The critical thing with conservation of mass is that there be no drift in the area value. Also of interest is the variance in this value. A good method will have a small variance around the correct area value.

Conservation and higher order finite elements

Here we shall run the rotation test until time $T = \pi/4$ for a set of methods to see if conservation of mass benefits from higher order finite element methods, or if SUPG stabilization improves the results. To see this we run the rotation test on a grid with $\Delta x = 1/40$, and a time step $\Delta t \sim \Delta x/10$.

Tab. 8.1 shows the conservation characteristics for two mixed and two equal order methods, with the Crank-Nicholson time step 5.3 but without the SUPG stabilization. The corresponding plot is in Fig. 8.1.

Method	Mean	Variance	Drift
c1/d0	0.28243100990099	0.00000000358301	0.00000059573371
c1/c1	0.28250669306931	0.00000000344123	0.00000049391243
c2/d1	0.28235643564356	0.00000000552880	0.00000108106786
c2/c2	0.28233427722772	0.00000000678285	0.00000123891007

Table 8.1. Conservation properties of four methods with Crank-Nicholson time stepping.

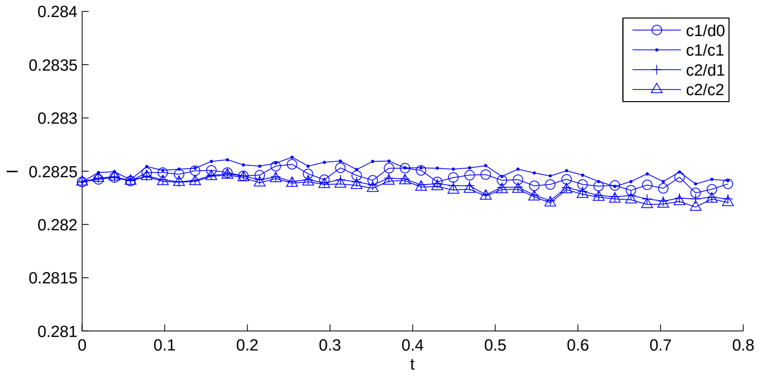


Figure 8.1. Rotation test: Mass, I, vs. time for four different methods that use Crank-Nicholson, $\Delta x = 1/40$, $\Delta t \sim \Delta x$.

Running the same test a second time, now with SUPG stabilized Crank-Nicholson methods gives the results in Tab. 8.2 and Fig. 8.2

Method	Mean	Variance	Drift
c1/d0	0.28241790099010	0.00000000364787	0.00000064738396
c1/c1	0.28249316831683	0.00000000371548	0.00000051739061
c2/d1	0.28233980198020	0.00000000613711	0.00000115610602
c2/c2	0.28232560396040	0.00000000752450	0.00000132440174

Table 8.2. Conservation properties of four methods with Crank-Nicholson time stepping and SUPG stabilization.

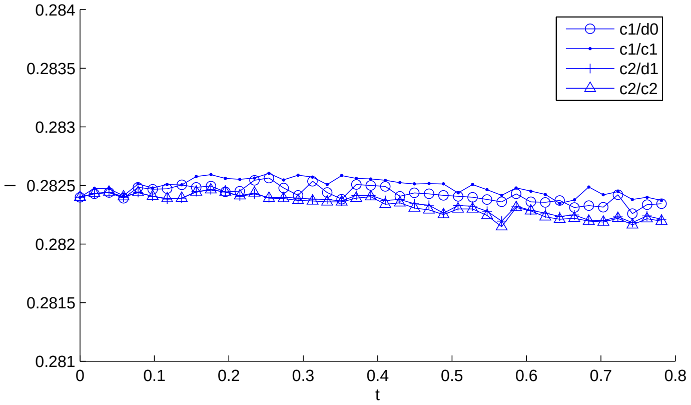


Figure 8.2. Rotation test: Mass, I, vs. time for four different methods that use Crank-Nicholson and SUPG, $\Delta x = 1/40$, $\Delta t \sim \Delta x$.

These result may seem a bit surprising. We take note of four things:

- The drift rate results for all methods are good, as compared with grid-size for example.
- The drift rate and variance is roughly double for the higher order methods. This implies that the conservation of mass is not improved.
- There is no discernible benefit, in terms of conservation of mass, from using a SUPG-stabilized method.
- There is no discernible difference in conservation of mass between an equal order and its corresponding mixed FEM.

Conservation and second order time-stepping

Having a second orders scheme for time stepping is desirable for many reasons. It is, however, not entirely obvious that a first order scheme will result in worse conservation of mass. For simplicity, this test will compare (5.2) and (5.3), without SUPG stabilization. Note that we are not changing the temporal discretization of the reinitialization, only the advection. We take a time step $\Delta t \sim \Delta x$. Conservation data is in Tab. 8.3, with the corresponding plot in Fig. 8.3.

Method	Mean	Variance	Drift
c1/c1 Euler	0.28281054901961	0.00000008682861	0.00000157667189
c1/c1 CN	0.28237862745098	0.00000000845766	0.00000165036104

Table 8.3. Conservation properties of two time-stepping methods for the same FEM.

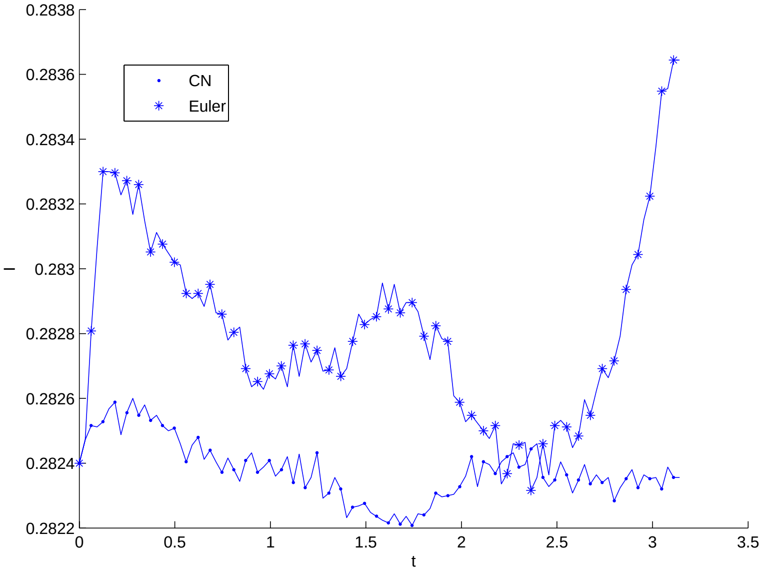


Figure 8.3. Rotation test: Mass, I, vs. time, for a first and a second order time-stepping method.

The numerical result clearly demonstrate that

- The second order scheme, CN, gives approximately the same drift value,
- but the variance is an order less. We conclude that the CN-scheme performs significantly better.

8.3.2 Convergence in conservation under grid refinement

Here we shall explore two things: that the drift rate and variance in I converges to zero under grid refinement, and the limits of conservation due to large curvature of the interface.

Rotation: convergence in drift and variance

To keep things simple, we choose to do this test for only one method - p1/p1. We let the interface thickness parameter, ε, be proportional to the grid size, so that the transition region is a fixed number of cells wide. Solving the rotation problem until a final time $T = \pi/4$ gives the conservation results seen in Fig. 8.4. The drift and variance values are in Tab. 8.4.

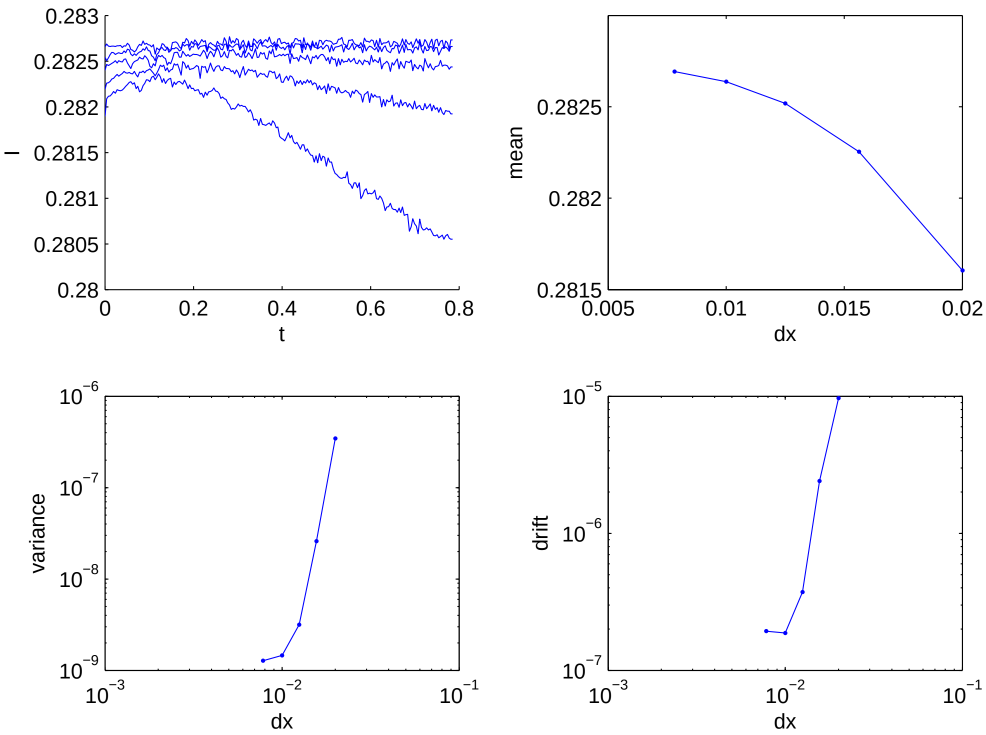


Figure 8.4. Top left: Mass, I, vs. time. Top right: mean of each I vs Δx. Bottom left: Variance of I vs. Δx. Bottom right: Drift rate of I vs. Δx.

dx	Mean	Variance	Drift
0.02	0.28160461386139	0.00000034498263	0.00000970167744
0.015625	0.28225473267327	0.00000002595914	0.00000241104578
0.0125	0.28251902970297	0.00000000317621	0.00000037262548
0.01	0.28263738613861	0.00000000145988	0.00000018728384
0.0078125	0.28269308910891	0.00000000127873	0.00000019294737

Table 8.4. Conservation properties for rotation test. $T = \pi/4$, $\Delta t \approx 0.00392$.

These result show that

- The conservation properties converge nicely under grid refinement.

Vortex: Loss of conservation due to excessive curvature

In our level set method, the interface has a thickness defined by a parameter ε. When the curvature of the interface becomes large, then it is clear that the interface will become poorly resolved. This could lead to a loss of mass conservation. We solved the vortex model until $T = 0.5$ and computed the mass, for a set of grids. A plot of how the interface deforms in this time interval is in Fig. 8.5. As before, the ratio between Δx and ε was kept fixed. The results for conservation are in Fig. 8.6.

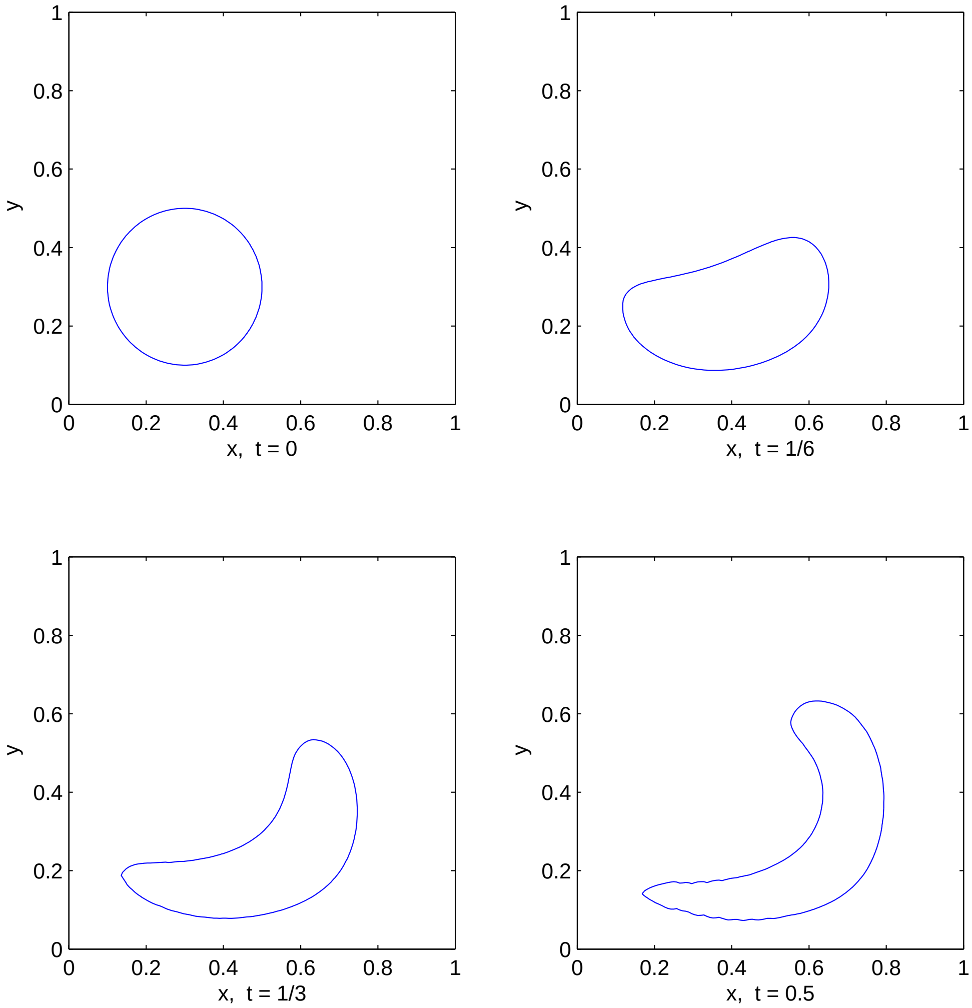


Figure 8.5. Vortex test: Interface at four times in [0 0.5]

We may conclude that

- The conservation properties of the method get better as the interface gets thinner.

8.4 Accuracy and convergence

In the best case, a finite elements with quadratic basis functions should be third order accurate in space, and the linear variant should be second order. The coupled

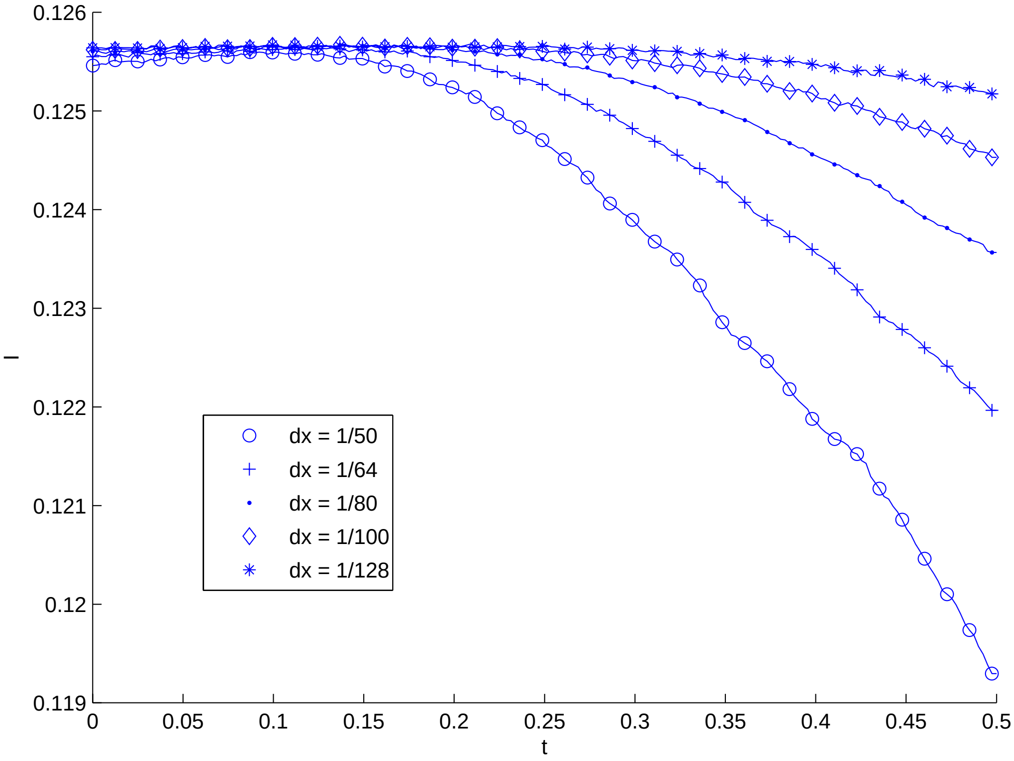


Figure 8.6. Vortex test: Mass, I, vs. time.

advection-reinitialization procedure is substantially more complicated than either calculation separately, due in part to the computation of $\hat{n}$ and the additional time-loop. From a mathematical point of view the mixed methods are more pleasing, since these will incur no projections.

For each method we run a rotation test, until $T = \pi/2$. By then the interface will have traveled a quarter of a revolution, e.g. if the bubble starts at $\mathbf{x}_0 = (0, 0.5)$ then $\mathbf{x}_T = (0.5, 0)$. Since the rotation field will not deform the bubble we have an exact solution to compare with. We choose three error metrics: L2-norm, ∞-norm and the location of the center of mass of the bubble (x_c, y_c) with,

$$x_c = \frac{\int_\Omega \phi x d\mathbf{x}}{\int_\Omega \phi d\mathbf{x}}, \qquad y_c = \frac{\int_\Omega \phi y d\mathbf{x}}{\int_\Omega \phi d\mathbf{x}}.$$

Define the order of accuracy of the method, p, as $err \sim (\Delta x)^p$ if we take a small enough time-step that the temporal error is small compared with the spatial error. We denote the exact solution as $\phi(\mathbf{x}, T) = \phi_*$.

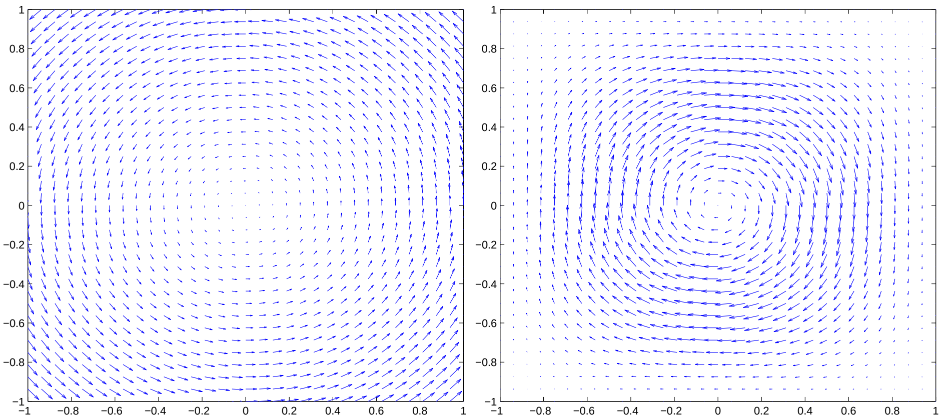


Figure 8.7. Left: Rotation field. Right: Vortex field.

8.4.1 Preliminary result

A FEM on linear elements is expected to be second order accurate in space, $p = 2$. Calculations with the "c1/c1" method, comparing with an exact solution gives a convergence in L2-norm of order $p = 0.83$. This is a very disappointing number, but it is not the whole story.

The immediate fear is that the time-stepping has missed the desired T. The error in the location of the center of mass for this calculation is $\sim 10^{-5}$ and converging fast. This indicates that the bubble is in the right place.

The crucial point that explains why the value for p is so low is that the discrete steady state solution of the reinitialization equation does not have exactly the profile that ϕ_* has. This is in fact neither a problem nor a surprise. The conservative level set method makes only the assertion that the area enclosed by the 0.5-contour be conserved - not that any initial profile be exactly restored.

With this in mind we make the following modification: Take an initial ϕ and compute the steady state solution of the reinitialization equation. Then solve until $T = \pi/2$. Take the exact solution, ϕ_* and reinitialize it - call this new reference solution ϕ_{**}. In the next section we shall see that the convergence results for $\|\phi_T - \phi_{**}\|$ are more convincing.

8.4.2 Accuracy results with respect to grid refinement

Take a sequence of grids that become finer and finer and compute the indicated errors on each. To get a value for p one can take the logarithm of the grid size versus the logarithm of the error and do a least squares fit of a line onto this - giving p as the slope of this line. These results are in Tab. 8.5. Looking at Fig. 8.8 we see that the quality of these accuracy numbers are good in most cases, but bad for some methods since they display erratic convergence.

We draw the following conclusions from these results:

Method	Conv order in $\|\phi_T - \phi_*\|_{L2}$	Conv order in $\|\phi_T - \phi_{**}\|_{L2}$	Conv order in $\|\phi_T - \phi_{**}\|_\infty$
c1/d0	0.381	0.303	-0.395
c1/c1	0.834	2.422	2.036
c1/c1 SUPG	0.884	2.432	2.095
c2/d1	0.584*	0.675*	-0.453*
c2/c2	2.913	2.886	2.090*

Table 8.5. Order of convergence in space, i.e. p in $err \sim (\Delta x)^p$, for a few methods for the LSM advection and reinitialization procedure

- The mixed linear/constant method, i.e. "c1/d0", is rubbish in terms of accuracy. The max norm even diverges! This loss of accuracy is probably due to the piecewise constant element for $\hat{n}$.

- The c1/c1 method performs very well. The convergence in L2-norm compared with the reinitialized exact solution is suspiciously good, but the convergence in max-norm is more sane.

- The SUPG stabilized method is only marginally better than its unstabilized counterpart. There is no clear significance in this result.

- The quadratic/linear mixed method, c2/d1, shows erratic convergence. The order of convergence reported in the table is of little significance.

- The quadratic equal order method, c2/c2, converges close to the $p = 3$ mark in L2-norm. The results for the max-norm calculations are rubbish, as seen in Fig. 8.8. Interestingly, there is no drop in convergence from comparing to the true translated solution, ϕ_*.

Which method is most accurate? A very tentative conclusion can be drawn by comparing L2-error for a particular grid ($\Delta x = 1/45$) - c1/c1: 0.00097, c2/d1: 0.00077, c2/c2: 0.00010. Despite the erratic convergence, the mixed quadratic method beats the equal-order linear one. The clear winner, in terms of accuracy is the equal order quadratic method.

We can also look at the accuracy results for the center of mass calculations. These can be seen in Fig. 8.9. It has not been possible to obtain a convergence order of any reasonable quality so we content ourselves with some tentative conclusions:

- All methods get *very* close to the desired point $(0.5, 0)$, even for coarse grids.

8.4.3 Accuracy of practical calculations

In a practical calculation we cannot take such small time-steps as in the previous section. Instead take a step that is close to the CFL-limit of the advection equation, $\Delta t \sim \Delta x$. For a sequence of grids we get an overall accuracy value of $p = 2.03$, in the max-norm, with the “c1/c1” method. This is only meant as an indication of overall accuracy, so we shall not do a detailed comparison with other methods. Note, however, that this result suggests a good second order temporal convergence.

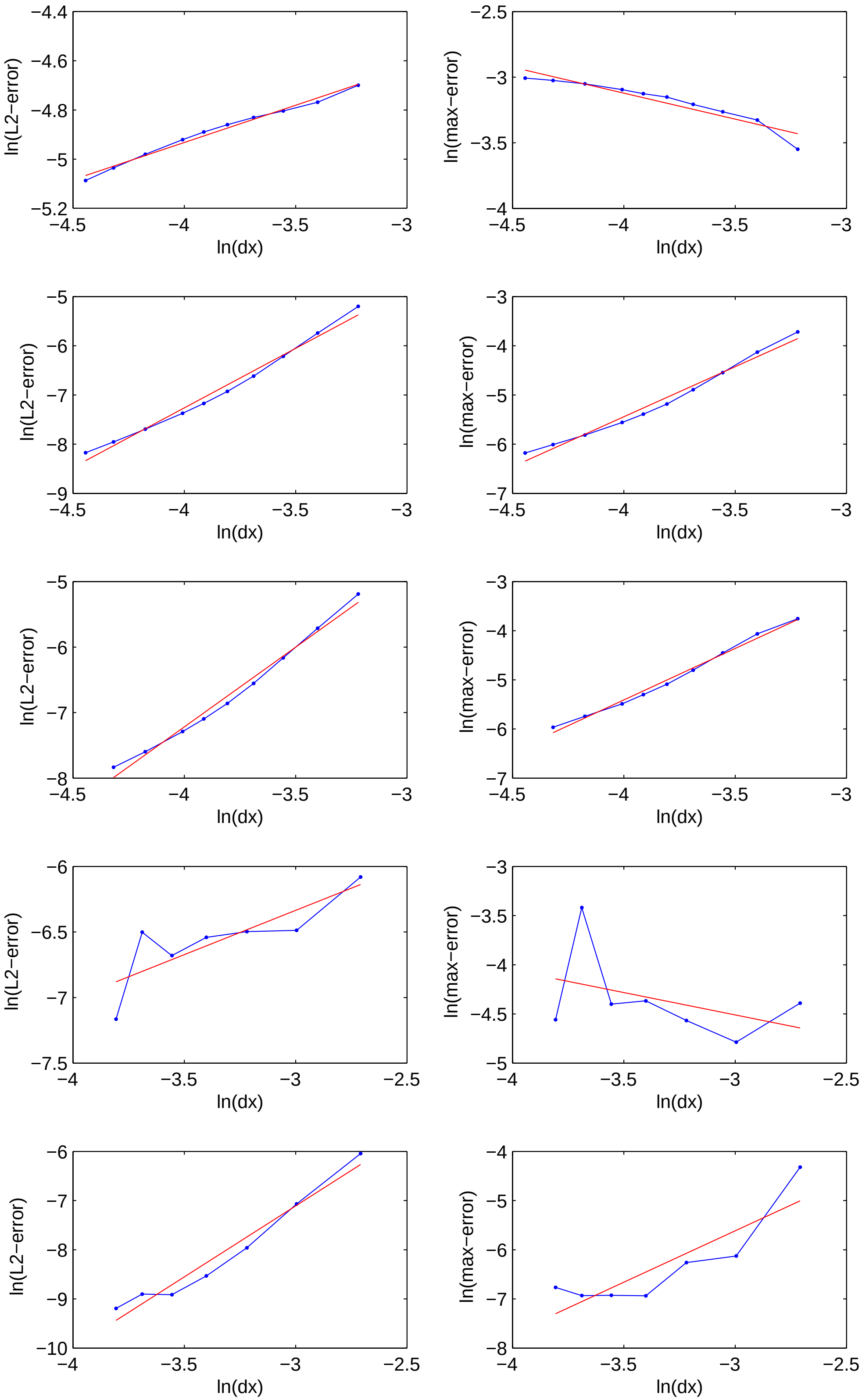


Figure 8.8. Left column: L2-norm. Right column: Max-norm. Top pair: c1/d0. Second pair: c1/c1. Third pair: c1/c1 SUPG. Fourth pair: c2/d1. Bottom pair: c2/c2.

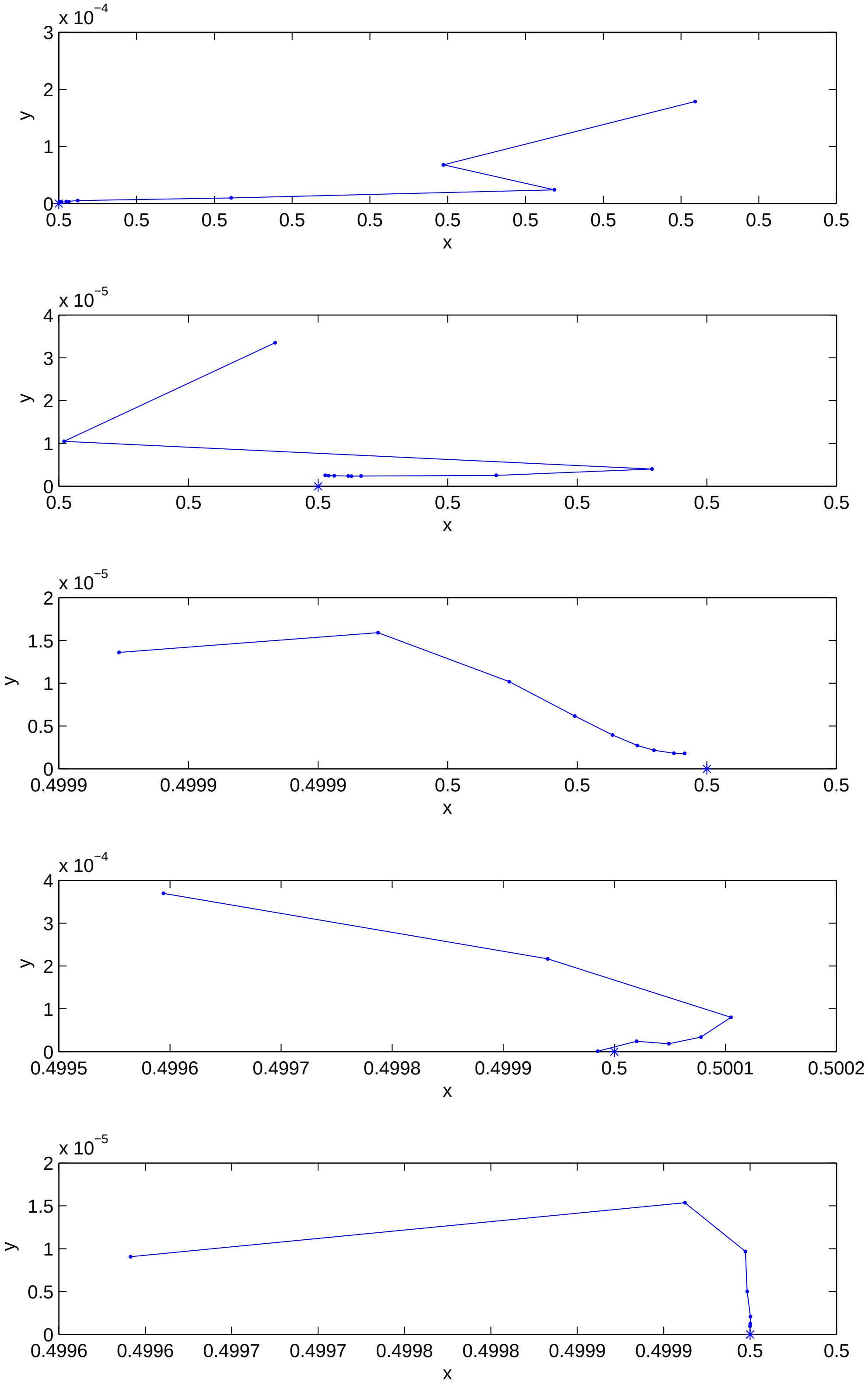


Figure 8.9. Center of mass points in xy-plane for a sequence of grids. Point marked with * is the desired center point (0.5,0). Top: c1/d0. Second: c1/c1. Third: c1/c1 SUPG. Fourth: c2/d1. Bottom: c2/c2.

Chapter 9

Numerical Results for two-phase flow

Having seen that the conservative level set method has nice numerical properties, we now present some flow calculations. In particular, we wish to see that conservation of mass can be obtained when the LSM solver is coupled to a flow solver. Also, we shall look at topology changes in the two regions, using a simple example.

Since we saw in the previous chapter that the equal order linear FEM, "c1/c1", performed very well in both accuracy and conservation, we shall use that method together with the Crank-Nicholson time-step.

With a linear finite element for ϕ, the only sane choice for the Stokes calculations is the pressure stabilized forms presented in chapter 6, i.e. (6.5) with the tensor based form of the surface tension, (4.6). This is good news, since the linear finite element method for the Stokes problem will be much cheaper to compute - involving roughly an eighth of the number of unknowns in a mixed quadratic/linear method.

9.1 Flow in narrow channel

The first test we run is described in Fig. 9.1, i.e. we have a bubble that must pass through a narrow channel. A finer mesh than the one pictured was used for calculations. The computed solution for different times can be seen in Fig. 9.2, where the flow field has been superimposed. These results look nice and convincing. Here the parameter values are $\sigma = 3$, $Re_1 = 1$, $Re_2 = 3$.

It is of great importance to see that mass is conserved. Fig. 9.3 has these results. We can see that there is no loss of conservation. The drift value is 0.000012, and the variance in mass is 9.44×10^{-10}.

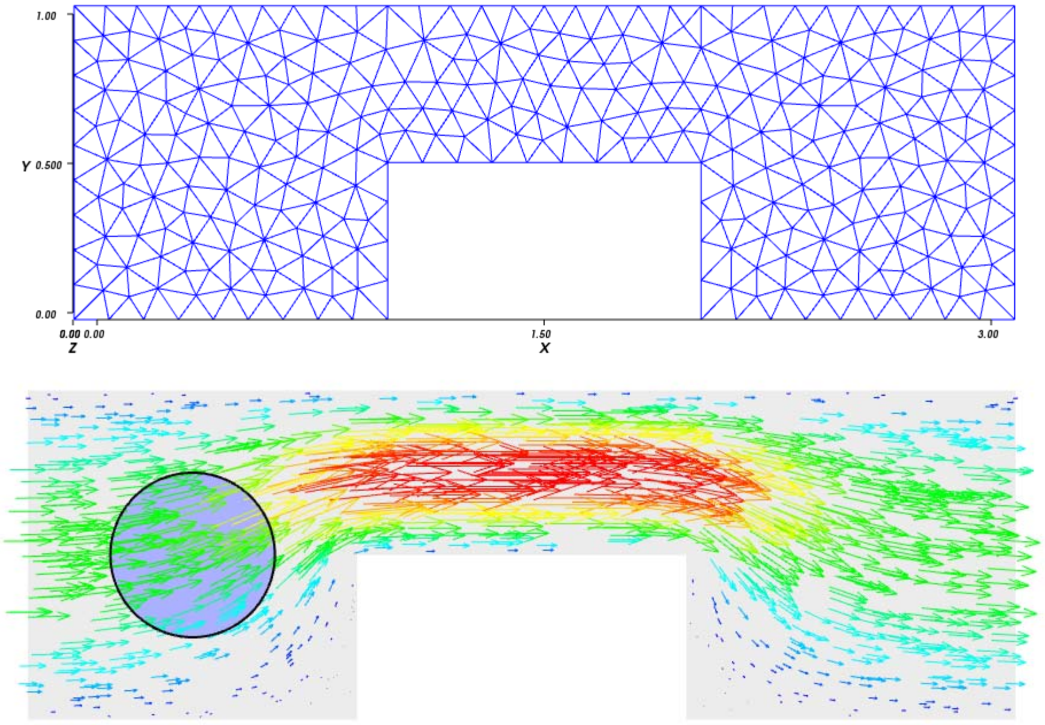


Figure 9.1. Top: Mesh for channel test. Bottom: Initial ϕ and first computed flow field, **u**.. As indicated in the second plot, the boundary conditions are as follows - left edge: parabolic inflow condition on **u**, right edge: outflow condition (i.e. zero pressure), all other: no slip.

9.2 Topology change

Some of the most complicated situations in two-phase flow involves merging and splitting of regions. One of the selling points of the level set method is that it handles these topology changes by construction. In practice, however, there are issues that arise.

First and foremost we are interested to see how topology changes affect conservation of mass for the conservative level set method. A simple, but representative, case is described in Fig. 9.4, i.e. a falling droplet hitting a surface. To get gravity we simply add it to the force term, **F**, in (6.4)

Fig. 9.6 shows a sequence of solutions to this problem, with the flow field superimposed. Note how strongly the the surface tension acts to straighten out the interface after it has begun merging. There is some amount of numerical error as the regions merge; this is something that we do not expect to be able to remove.

The results for conservation of mass can be seen in Fig. 9.5. It is interesting to note the following:

- That the mass increases sharply as the merging is about to take place, but then the mass decays.

- This effect is clearly less pronounced on finer grids, where the interface is thinner.
- After merging the mass becomes constant, but at a higher level than initially.

That the mass is not constant during the merging process is not really a major concern. It is, however, more alarming that the total mass has risen after the merging has completed. To address this we shall only do a small convergence observation.

Take the integral over the 0.5 level set, I, as calculated by 8.3. Consider

$$\begin{aligned} &\Delta I = I(T) - I(0) \\ &c\Delta I = \Delta x \end{aligned}$$

i.e. ΔI is the difference in mass incurred by the merging, as a multiple of the grid size. In Tab. 9.1 we see these quantities for a sequence of grids. This gives a convergence in ΔI with respect to grid refinement that is of order ~ 1.5, as seen in Fig. 9.7.

We can be reasonably content with these results. In the best of worlds one would have a method that gives area conservation exactly, even during interface merging, but the fact that we have been able to establish a convergence for this is at least some consolation.

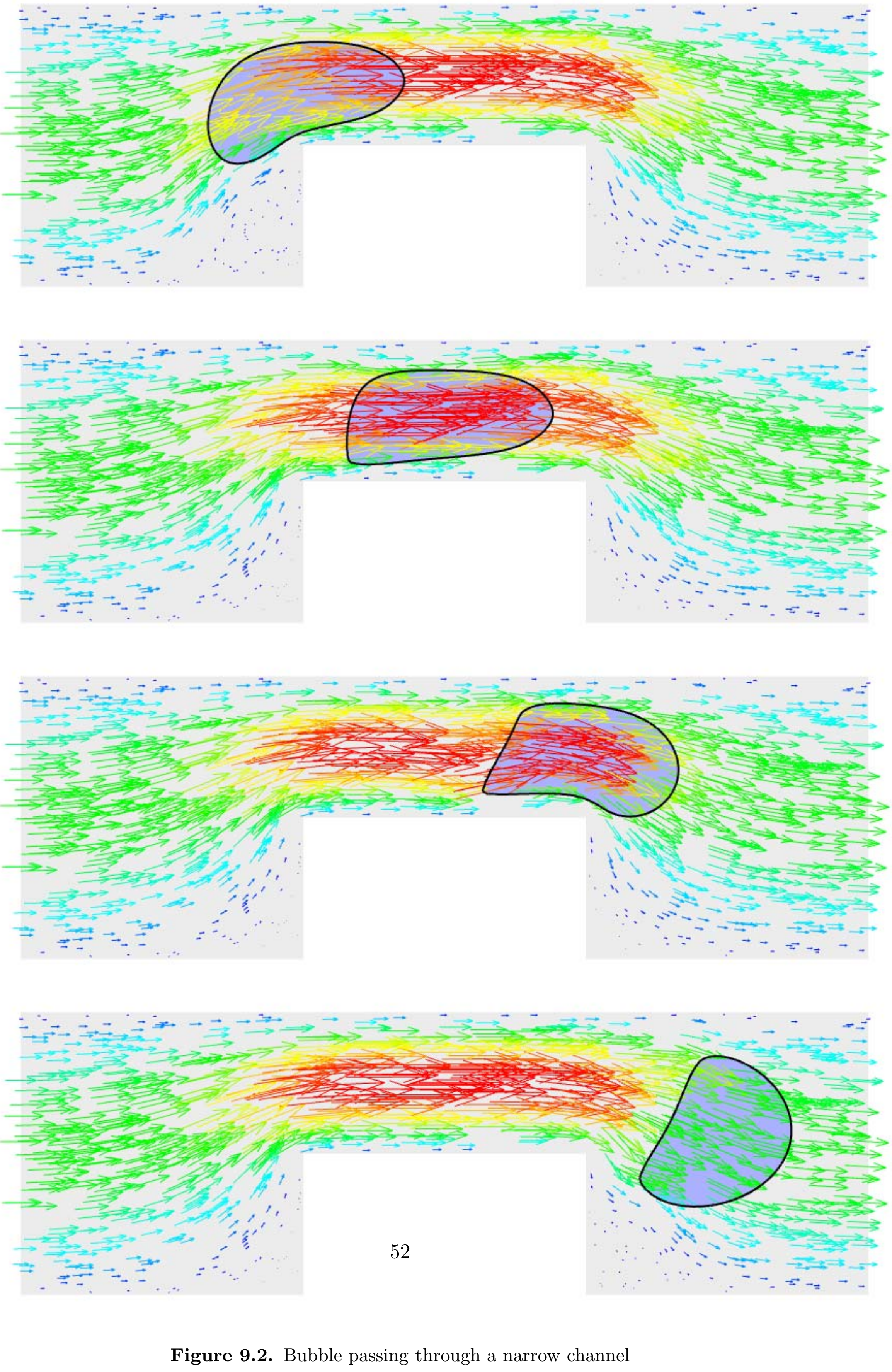


Figure 9.2. Bubble passing through a narrow channel

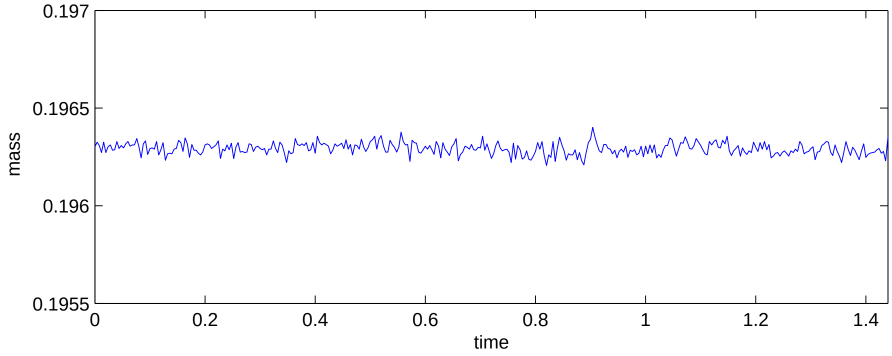


Figure 9.3. Flow through narrow channel: Mass vs. time

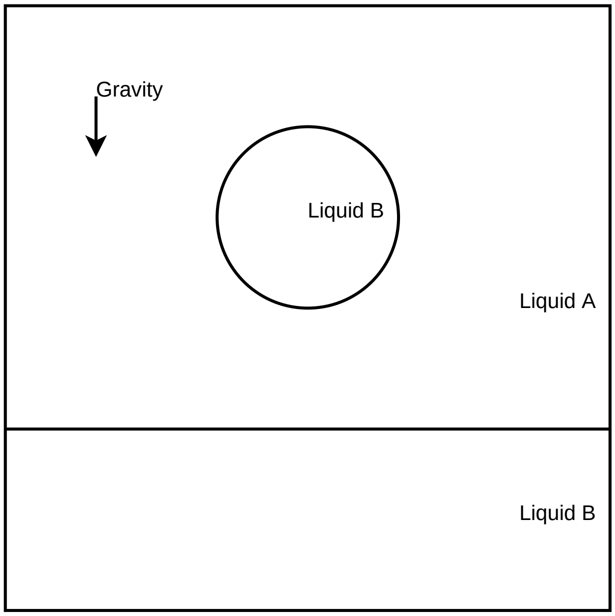


Figure 9.4. Falling droplet. Boundary conditions are no slip on all but the top edge, where the condition is zero pressure.

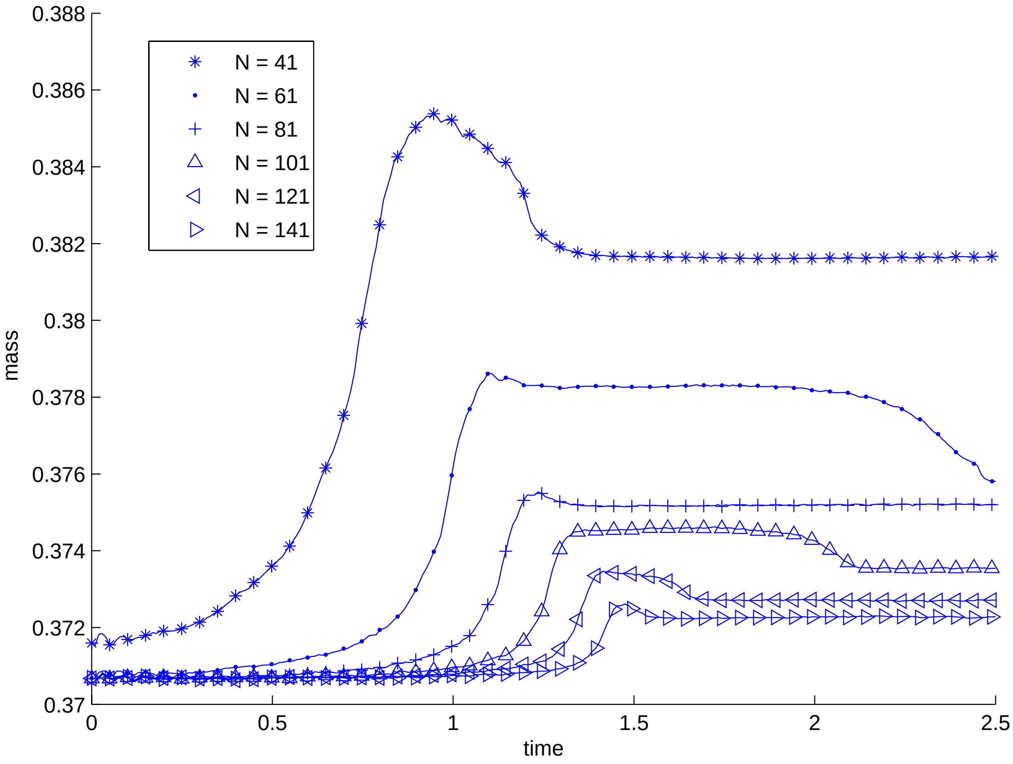
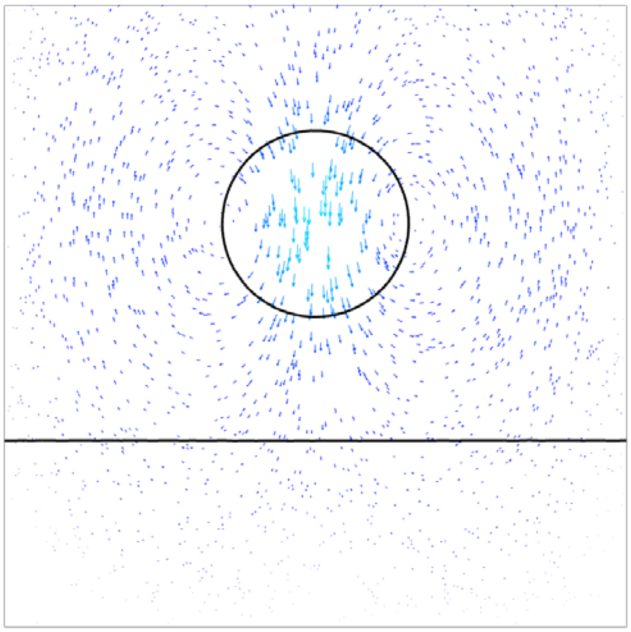
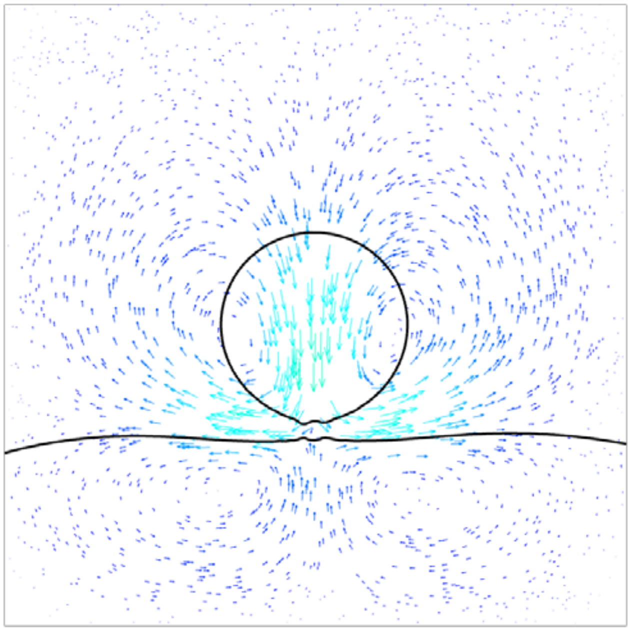
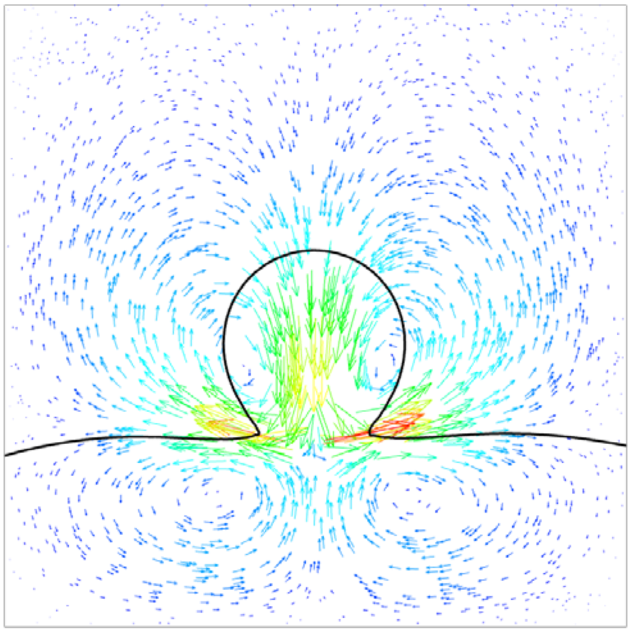


Figure 9.5. Mass, I, vs. time for different grids







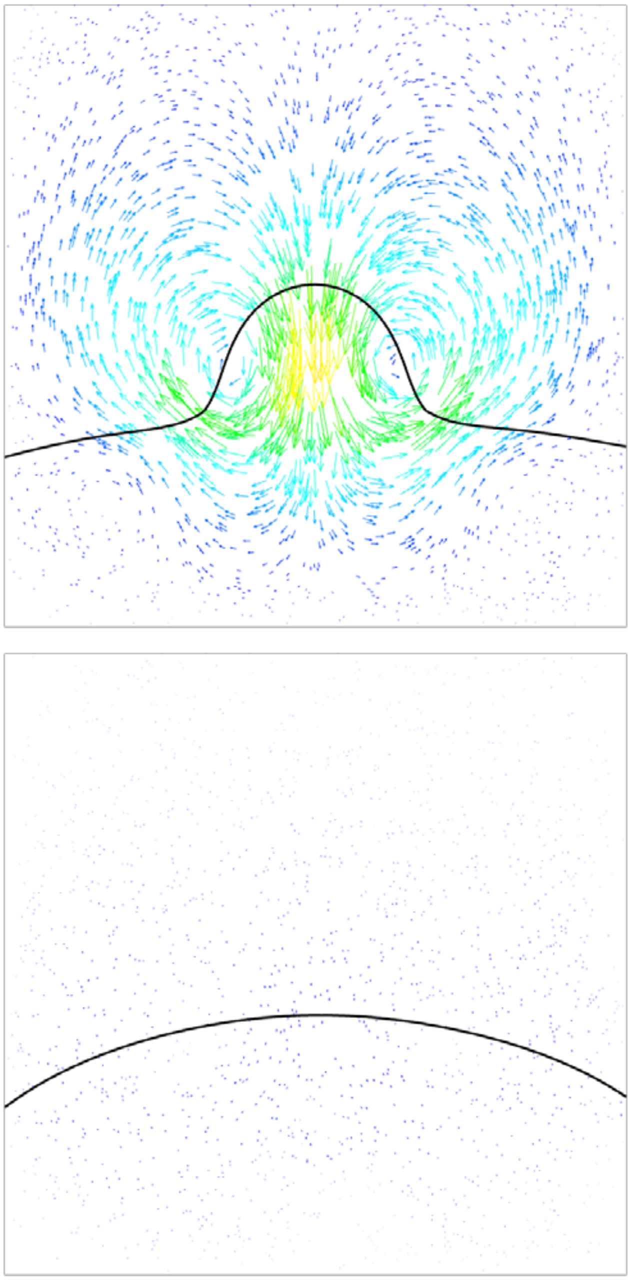


Figure 9.6. Falling droplet merging with surface

Δx	ΔI	c
1/41	0.0101	2.421
1/61	0.0051	3.197
1/81	0.0045	2.721
1/101	0.0029	3.454
1/121	0.0020	4.049
1/141	0.0016	4.454

Table 9.1. Convergence in conservation of mass

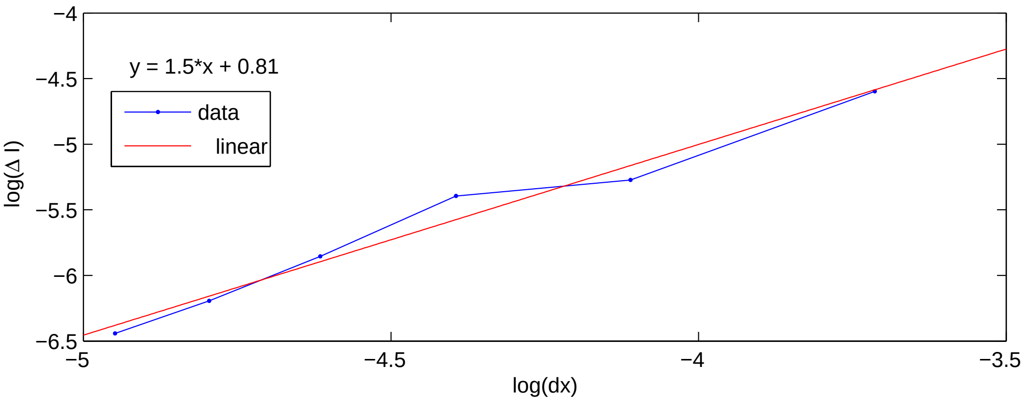


Figure 9.7. $\log(\Delta I)$ vs. $(\log(\Delta x)$, and convergence

Chapter 10

Implementation

A substantial amount of time has been spent on the implementation of the methods presented prior. This chapter shall give an overview of the features and structure of the code.

The selling points of the code is that it:

- is written in C++, making it fast and reliable.
- works in both 2D and 3D, without substantial changes.
- is object oriented, making it easy to add components (such as a Navier-Stokes solver or adaptive mesh refinement).
- is independent of finite element.
- can use parallel libraries for linear algebra, and advanced multigrid preconditioners.

The two last points are no small feats, but most credit must go to the excellent set of libraries and utilities that have been used - namely DOLFIN and FFC, parts of the FEniCS project, [20].

10.1 A set of object oriented solvers and tools

Fig. 10.1 show an overview of the implementation. The components are: a interface tracking solver, a flow solver, and a few classes with general purpose functions that are potentially used by more than one module.

- The `LSM_Solver` class contains the level set advection and reinitialization solver, as describes in Ch. 5. It can easily be configured at compile-time to run with or without SUPG stabilization, and with either of the time stepping schemes.

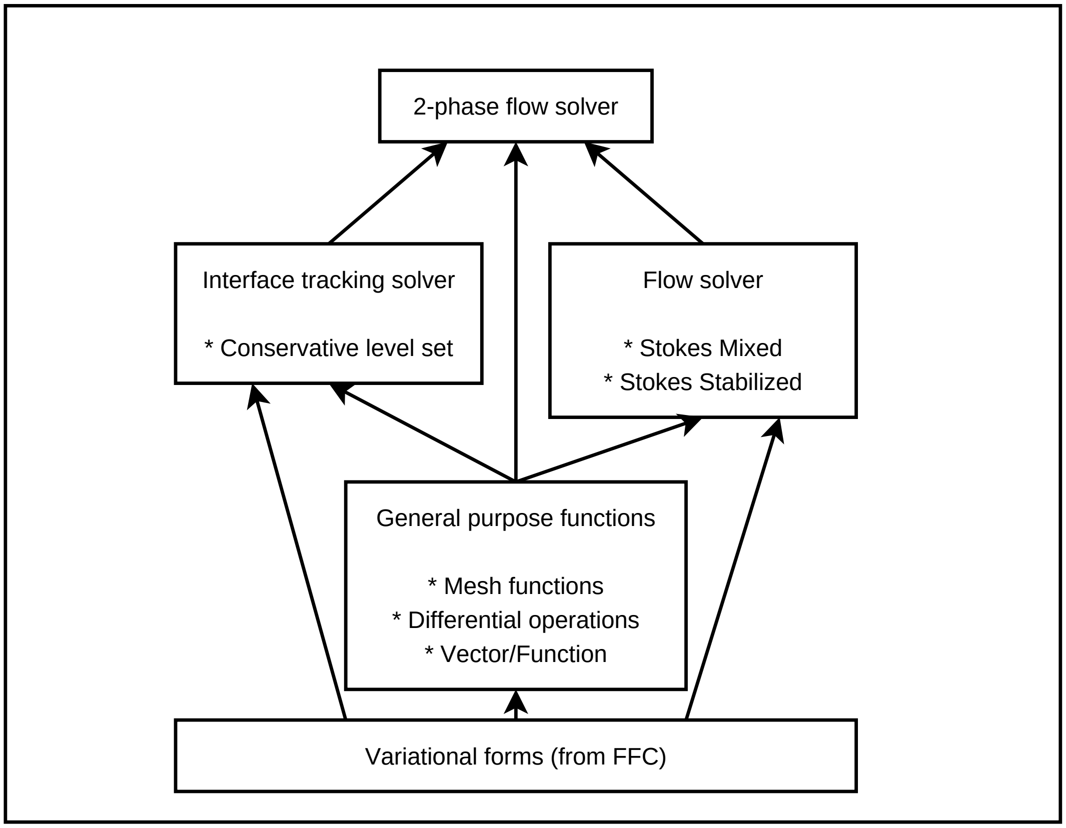


Figure 10.1. Overview of implementation

- We then have a set of flow solver classes, which currently includes a stabilized Stokes two-phase solver and a mixed element solver, contained in `Stokes_2ph_EOStab` and `Stokes_2ph_Mix` respectively (implemented according to the discussions in Ch. 6). Adding appropriate Navier-Stokes solvers poses no practical problem in this framework, but had to be left out due to time constraints.

- `Diff_fcns` contains routines for differential operations, such as gradient, unit normal, tangent, curvature, smoothing, and other calculations. Some of these functions are used very frequently in a simulation run.

- The `Mesh_fcns` class contains routines that do simple mesh operations, such as stretching the mesh, and computing the triangle diameters and areas on the mesh. There are also functionality for point-wise mesh refinement here, as well as a largely untested set of routines for adaptive mesh refinement and interface dependent mesh deformation. Due to the rapid development of DOLFIN, mesh adaptivity could not be fully realized.

- The `Vec_Func_fcns` contains functions that are useful when dealing with vectors and functions in the implementation. Examples are: restriction of a symbolic expression onto an arbitrary FE space, and different functionals over Ω (such as error and level set mass estimations).

10.2 A word about FEniCS

The FEniCS project aims to create an efficient package for the Automation of Computational Mathematical Modeling (ACMM), by using new insight into how a general finite element solver can be constructed. This includes the automation of the discretization of differential equations, which is handled by the Fenics Form Compiler (FFC), see [19].

The second component used in this implementation is the Dynamic Object-oriented Library for FINite element computation, or DOLFIN. The centerpiece of DOLFIN is the automatic assembly of the variational forms that FFC produces. DOLFIN also contains a mesh library, interfaces to linear algebra, and other useful functionality.

Chapter 11

Adaptive interface thickness

One clear drawback of the conservative level set method is the loss of conservation due to excessive curvature of the interface. Fig. 8.6 shows that we must have ε small compared with the maximum value of the curvature, κ.

In this chapter we shall propose a slight modification of the conservative level set method that opens up the possibility of having the thickness of the transition region as a variable. By doing this change we hope that conservation properties can be improved.

We saw earlier that the transition region should be resolved by at least eight triangles. As the interface get thinner we need to refine the mesh. In this implementation, adaptive mesh refinement was not possible within the time-frame. The results in this chapter have their own significance regardless, but keep in mind the possibility of mesh refinement.

11.1 Linking interface thickness to curvature

We need somehow to let ε depend on the curvature of the interface. To do this we opt for a qualitative approach: Introduce the interface thickness, $\xi = 6\varepsilon$. Initially, the interface is a bubble with a radius r, i.e. the 0.5-contour of ϕ has an initial curvature $\kappa = 1/r$. A simple approach links these by letting the radius be a multiple of the thickness, $r = \eta\xi$. This gives a nice expression for ε:

$$\varepsilon = \frac{1}{6\eta\kappa_{\max}},$$

where $\kappa_{\max}$ is the largest curvature in a vicinity of the interface. Since this is an implicit interface tracking method we can only roughly get this value - but that's enough for these purposes. Using this kind of expression is reasonable for the initial data, and if we keep η fixed then the interface will depend on the curvature at later times.

11.1.1 Locally adaptive interface thickness

Stage two is to let the interface thickness vary depending on the local curvature. The derivation of the reinitialization equation, in [6], takes ε to be a constant. If we take instead that $\varepsilon = \varepsilon(x, y)$, then the reinitialization equation becomes

$$\phi_\tau + \nabla \cdot [\phi(1 - \phi)\hat{n}] = \nabla \cdot [\varepsilon \hat{n}(\nabla \phi \cdot \hat{n})] \,,$$

In theory, we could compute

$$\varepsilon(x, y) = \frac{1}{6\eta\kappa(x, y)}.$$

This is overly simplistic, since κ will vary across the transition region. However, an appropriate amount, c, of Laplacian smoothing,

$$\kappa^* - \kappa = c\Delta\kappa^* \tag{11.1}$$

ssolved for κ^*, makes it feasible. Obviously, any chance of getting this to work demands that the interface curvature can be successfully evaluated. This will expand on the concerns of the computation of the unit normals - the problems due to projections and differentiating piecewise polynomial functions.

11.1.2 Conservation properties of κ-adaptive LSM

We would, of course, like to be able to make some theoretical statements of the conservation properties of this variant of the LSM. For these purposes, let's just look at the first method, i.e. ε constant in space and varying slowly in time.

Both advection and reinitialization PDEs are conservation laws. Thus, $M = \int_\Omega \phi dx$ is constant. This holds in the this setting as well. Take the area bounded by the 0.5-contour, I, as computed by (8.3). In $\mathbb{R}^2$,

$$I - M \sim L_\gamma \kappa_{\max} \varepsilon^2,$$

where L_γ is the length of the interface. The proof of this is in [16]. From this we see that the area, I, will converge to M as ε goes to zero. To address the issue at hand, take two circular interfaces, with $I = \pi r^2$ for some r,

$$I - M = L_\gamma \kappa \varepsilon^2 = 2\pi\varepsilon^2$$

$$I_* - M_* = L_\gamma \kappa \varepsilon_*^2. = 2\pi\varepsilon_*^2$$

In the setting where we have some ϕ and solve the reinitialization problem for both ε and ε_* we have $M = M_*$. This gives

$$I_* = I + 2\pi(\varepsilon^2 - \varepsilon_*^2)$$

So in the case of the interface getting thinner the area bounded by the 0.5-contour of ϕ will grow slightly. This is a simple geometric fact, and in the limit of either κ or $\varepsilon \to 0$ the difference will vanish.

11.2 Forms for computing curvature

In the formulation of the Stokes equations we saw that it was not necessary to compute κ explicitly. That is not possible here.

It has been observed in other FEM implementations for the level set two-phase flow problem that computing the interface curvature is a substantial difficulty. Olsson and Kreiss suggest in [16] that one should add diffusion to (11.3). Tornberg instead suggests in [17] that some diffusion be added to ϕ before $\hat{n}$ is computed, to eliminate small oscillations that could grow as derivatives are taken.

From chapter 4 we know

$$\kappa = -\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right) = -\nabla \cdot \hat{n},$$

so that

$$(\kappa, w) = -(\nabla \cdot \hat{n}, w) \tag{11.2}$$

$$(\kappa, w) = \int_\Omega (\hat{n} \cdot \nabla w) d\mathbf{x} \tag{11.3}$$

We saw in Sec 5.3 that a definitive source of numerical error is when there is a projection from discontinuous to continuous basis functions. This mismatch in order is inherent from the differentiation of a piecewise polynomial function. The form (11.2) is perfectly fine, i.e. there is no projection and the inner product exists, if for example $\hat{n} \in \mathscr{W}_{vc}^1$ and $w \in \mathscr{W}_d^0$. Recall the results from Sec 2.4 to see the existence of a global weak derivative of a continuous function. The problem is, however, that $\hat{n}$ is itself a derivative of ϕ, and thus it is not a continuous function unless one projects it onto a set of continuous basis functions.

Note that if we instead consider (11.3), we avoid taking the derivative on $\hat{n} \in \mathscr{W}_{vd}^*$. We are not permitted to take $w \in \mathscr{W}_d^*$ (since a weak derivative of such does not exist globally), and if we take $w \in \mathscr{W}_c^*$ there is projection.

It seems the only way to get a globally defined curvature form is to do a projection so that $\hat{n}$ is continuous. However, if we could differentiate the discontinuous function $\hat{n}$ then expression (11.2) would be fine, since we could choose the test functions to live on a finite element of one order less than $\hat{n}$ lives on (thus incurring no projection).

Within the FEM framework we may abandon the global concept of derivatives and settle for local derivatives on each finite element, as discussed in Sec 2.4. This means that we understand (11.2) as sum of contributions from finite elements:

$$(\kappa, w) = -(\nabla \cdot \hat{n}, w) = -\sum_i \int_{T_i} (\nabla \cdot \hat{n}) w dx.$$

It is not possible to compute κ explicitly without incurring a projection (which we may want to avoid) unless ϕ is at least piecewise quadratic - not even in a local sense.

Computations using either form is going to run into trouble at the boundary. For practical purposes, we multiply $\hat{n}$ by a boundary mask so that the $\hat{n}$ components go smoothly to zero near the boundary. This helps a lot.

11.3 Numerical results for curvature computation

We run the vortex test and compute the curvature at every time step. The set of elements for the curvature calculation are, of course, the same as the elements for the level set FEM. That is, we will have to choose a set of elements for that are sitable for curvature calculations (according to the concerns in the previous section) and that perform well in the numerical tests for the level set method (see chapter 8). We have used the common curvature expression, i.e. (4.7).

First, we take the equal order quadratic FEM, c2/c2 (as in Tab. 5.3), and get the results in Fig. 11.1. These are really bad. Even at time zero, i.e. computing the curvature of the initial data, there are large deviation from the exact solution. Furthermore, this problem grows as the interface gets thinner.

The results for the mixed order quadratic method, c2/d1, as seen to the right in Fig. 11.2, are more convincing. The only obvious conclusion is that the projection onto piecewise polynomial space has had a really adverse effect.

Interestingly, the results for the equal order linear FEM, c1/c1 (as in Tab. 5.2), are also good. In fact, there is no significant difference between the curvature results obtained by c1/c1 and c2/d1.

An important note is that none of these results in the time series were possible to get without first smoothing ϕ, as suggested by Tornberg. This is due to the fact that small oscillations are sure to enter the computed solutions. To get rid of these we apply a small amount of dissipation to ϕ, just as in (11.1).

Note also that the second curvature expression presented in chapter 4, i.e. (4.8), was tested and discarded due in part to the complexity of the expression. With the nice curvature results for the c1/c1 filtered method seen in this section, there is simply no need do a direct curvature calculation using (4.8).

11.4 Numerical results for variable thickness LSM

The first case uses the familiar reinitialization, but first computes a ε so that the interface thickness is small with respect to curvature. These results are in Fig. 11.3 - looking very nice.

The vital questions is whether we can improve conservation of mass or not. In Fig. 11.4, we see mass versus time, as usual. This result indicates that the area bounded by the 0.5-contour converges to M as the interface gets narrower. Also, there is no drop-off in mass for the adaptive case as opposed to the ordinary one. The behavior we see is in fact very pleasing, since it verifies the theory.

The numerical work proved that this approach is robust. Together with the feeling that the method is sound and useful if combined with adaptive mesh refinement,

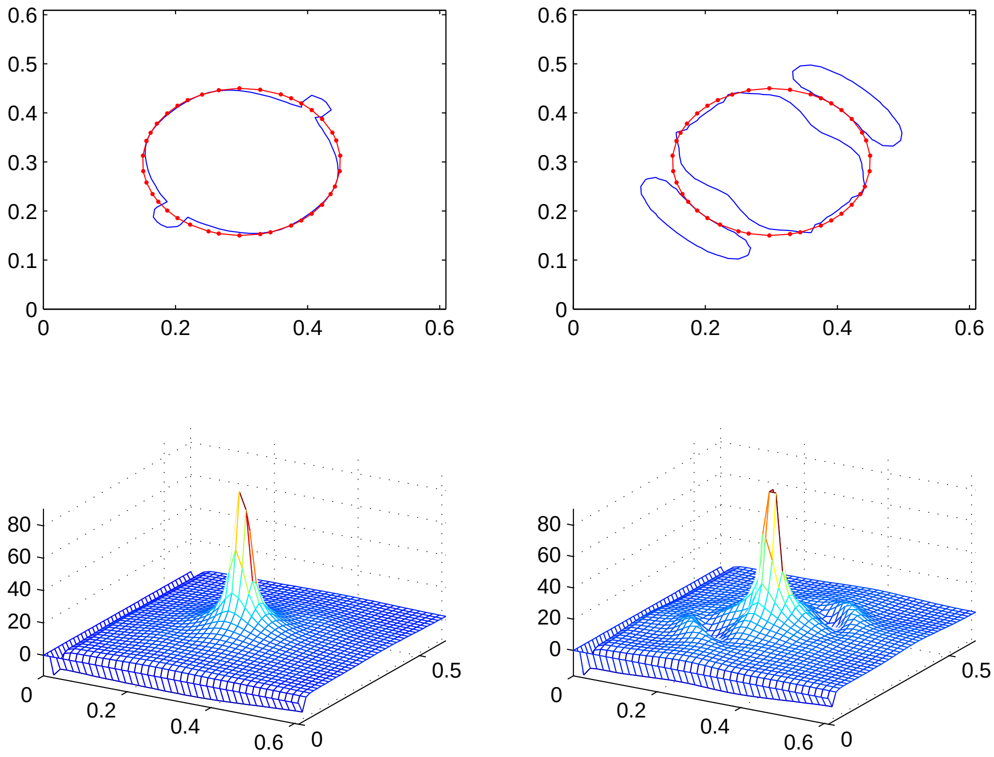


Figure 11.1. Curvature results for seconder order finite element method, c2/c2, at $t = 0$. Top pair: One contour of curvature. Line with dot marks exact solution. Bottom pair: Surface plot of curvature. Left column: $\varepsilon = 0.01$. Right column: $\varepsilon = 0.005$ (and twice as fine mesh).

this gives cause for optimism.

The next calculation uses the locally adaptive adaptive approach. The amount of smoothing to apply to the curvature is not determined exactly. We use $c = 0.1\Delta x$ and $c = 0.01\Delta x$, with the results given in Fig. 11.5. These results betray a problem with this method: robustness. In effect, there is a feedback loop from curvature to interface and back to curvature. If there are oscillations in the curvature calculation near the interface, then the interface will become jagged - and that will induce more oscillations in the curvature. It is possible to control this to some extent using the curvature smoothing approach, but it is somewhat fragile.

The conservation of mass is dubious at best, as the results are in Fig. 11.6. It seems reasonable to discard this method, due to the inherent stability problems and lack of consistent conservation behavior.

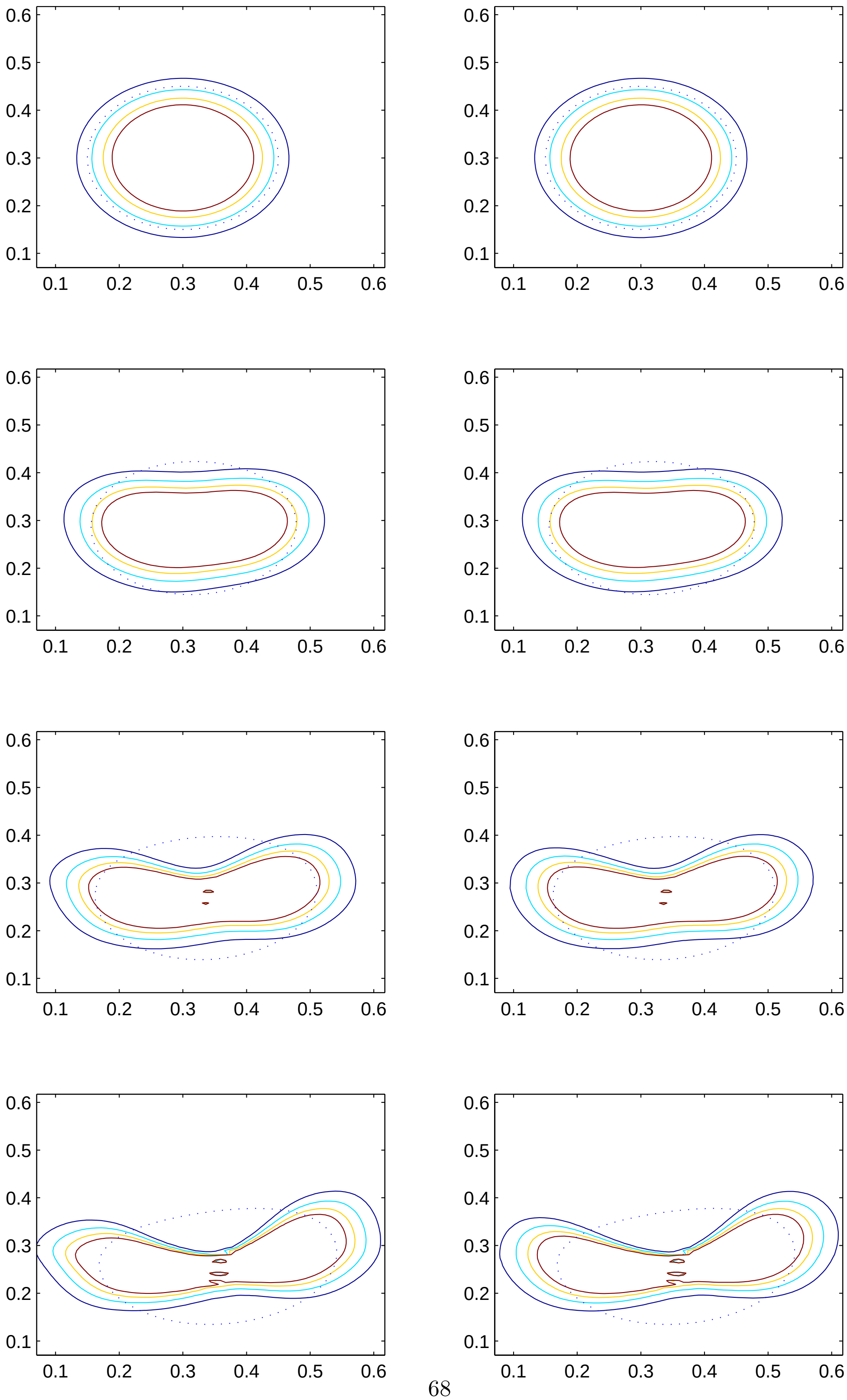


Figure 11.2. Curvature contours at four times. Left column: c1/c1 method. Right column: c2/d1 method. Dotted line shows the location of the interface.

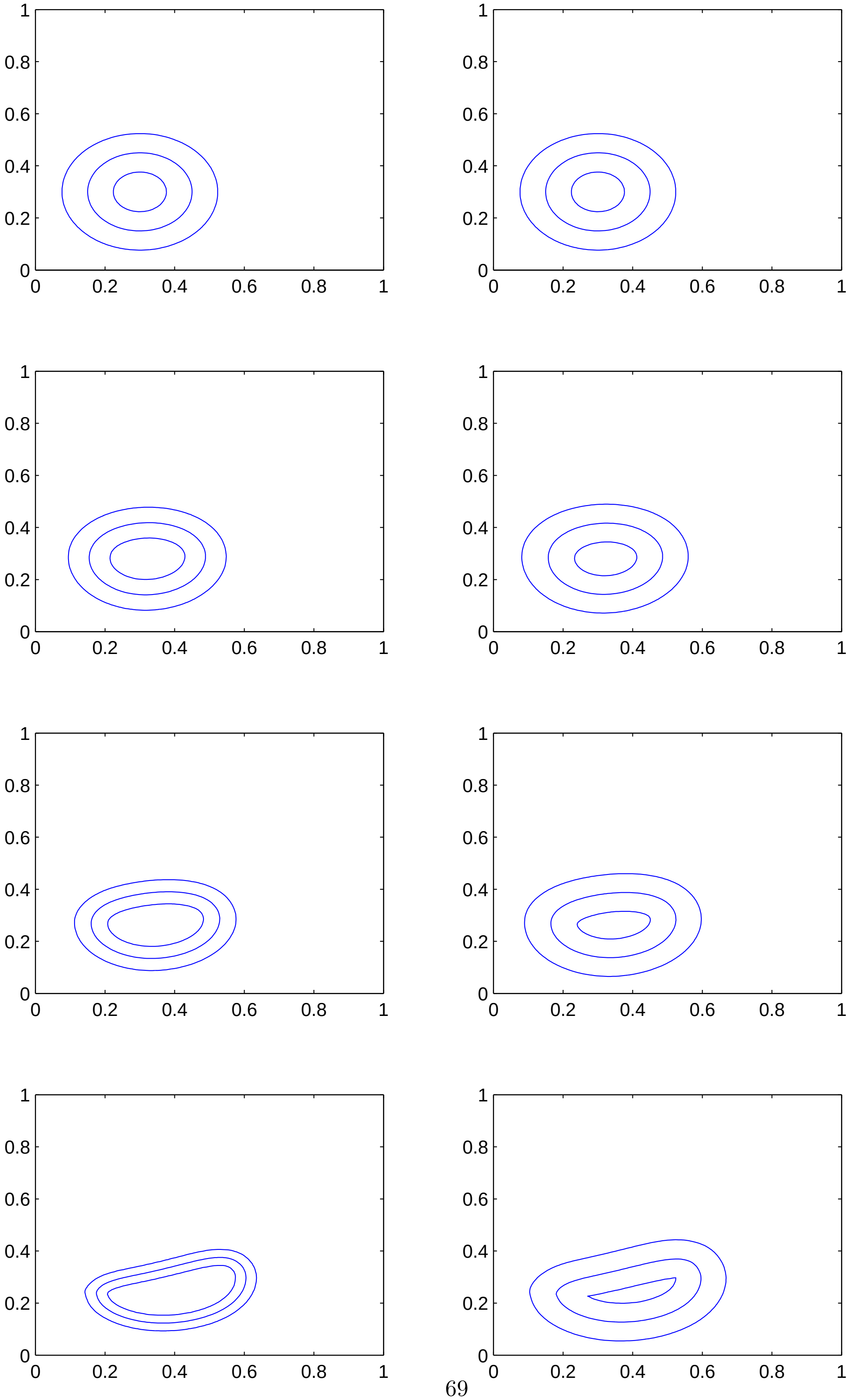


Figure 11.3. Contours of level set function, ϕ, at four times. Left column: Adaptive interface thickness. Right column: Ordinary conservative LSM.

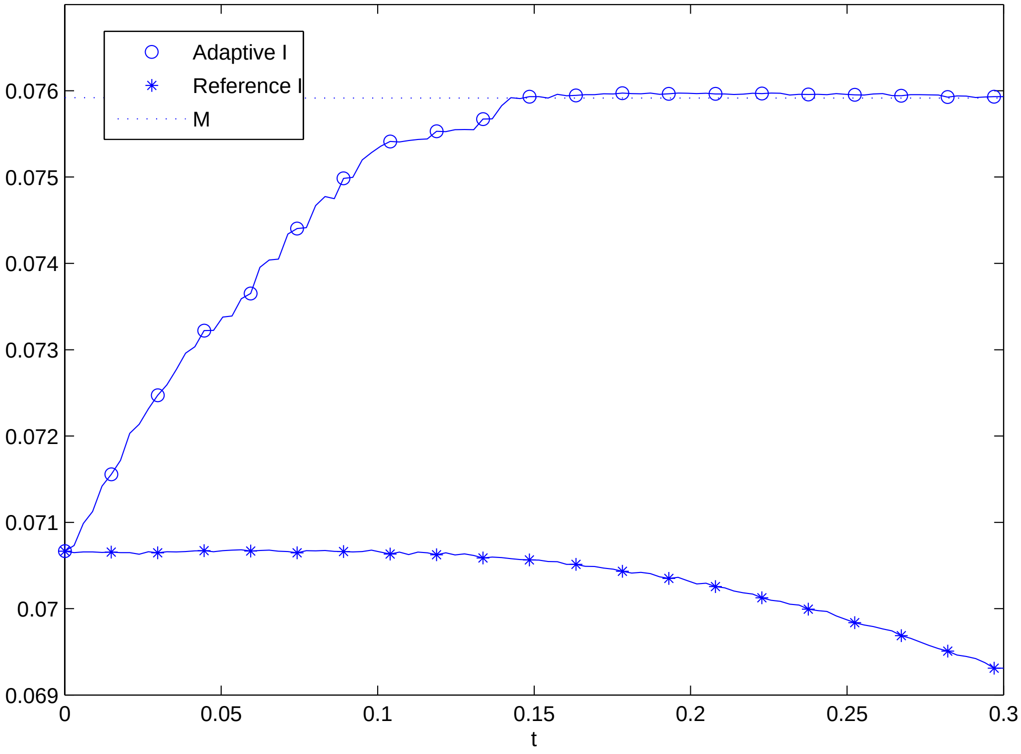


Figure 11.4. Conservation of mass, corresponding to Fig. 11.3

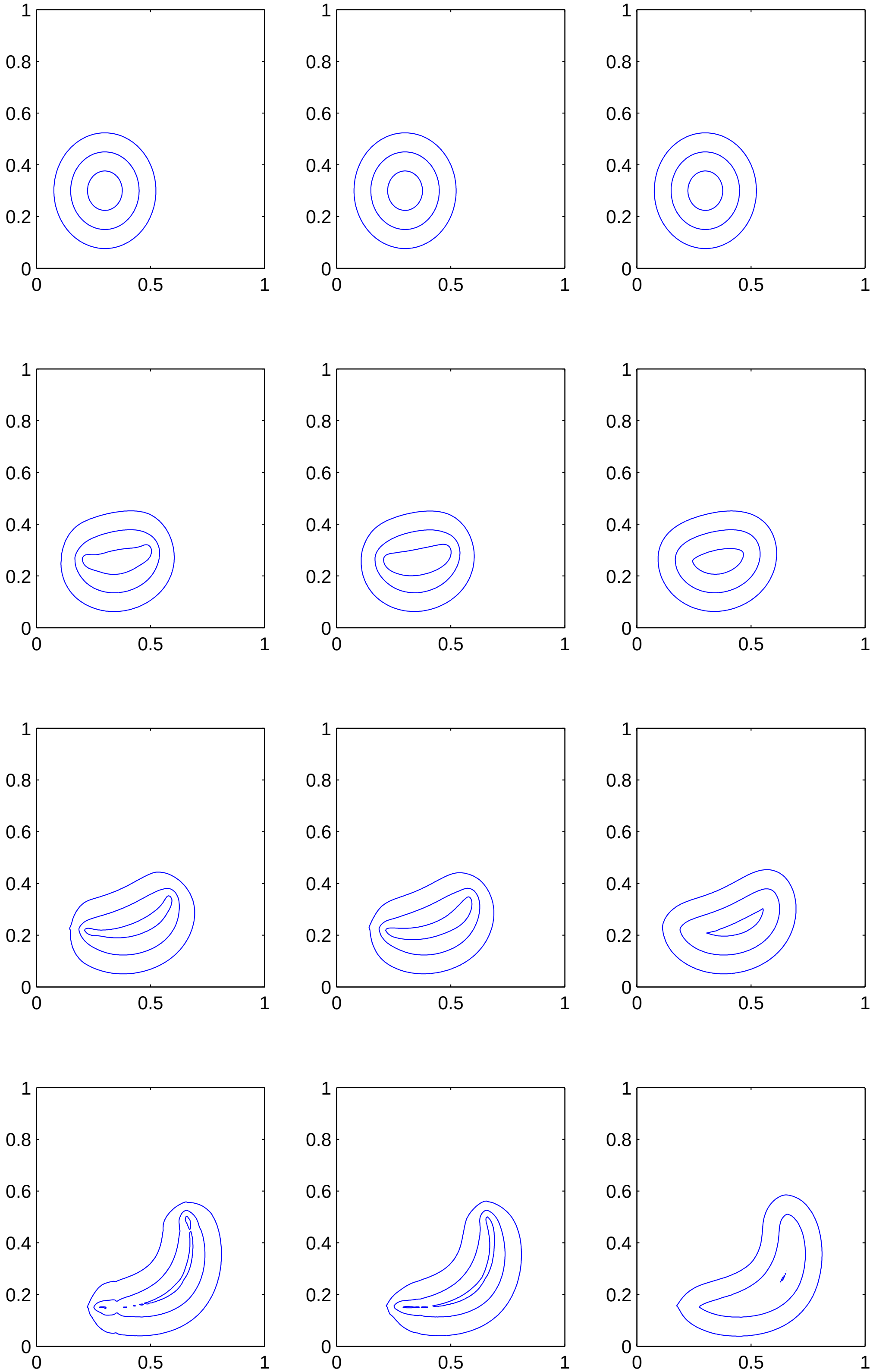


Figure 11.5. Contours of level set function, ϕ, at four times. Left column: Locally adaptive interface thickness, $c = 0.01\Delta x$. Center column: Locally adaptive interface thickness, $c = 0.1\Delta x$. Right column: Ordinary conservative LSM.

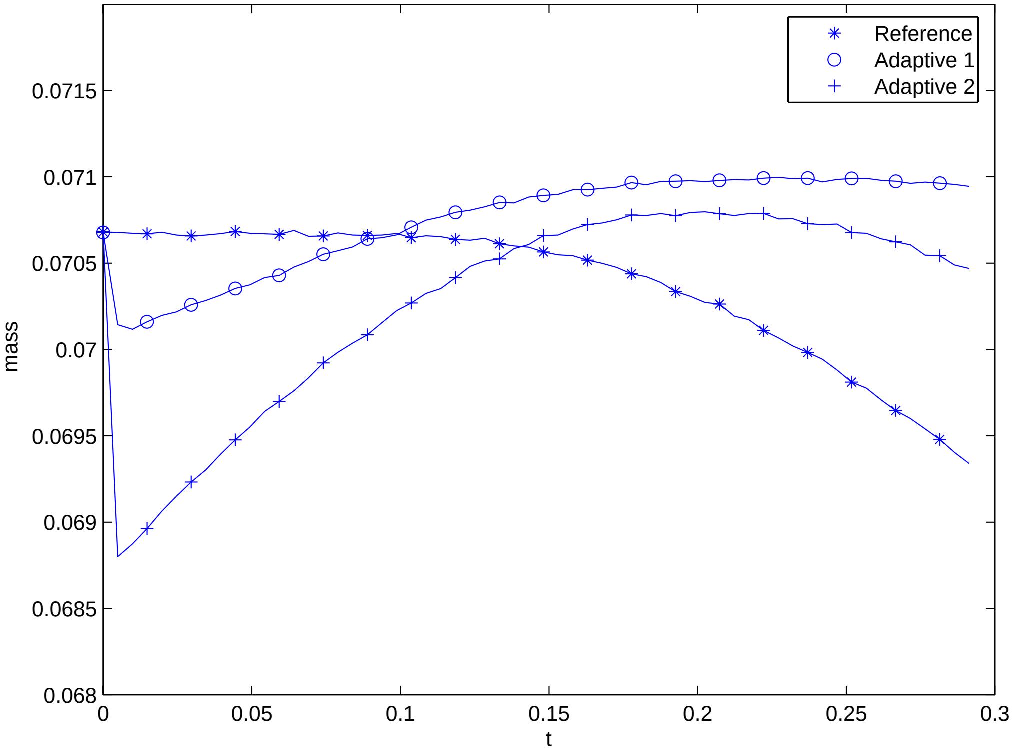


Figure 11.6. Conservation of mass, corresponding to Fig. 11.5

Chapter 12

Summary and conclusions

12.1 Summary

This report has attempted to provide a broad investigation into the finite element formulation and solution of a two-phase flow problem that uses a conservative level set method. In chapters 3 and 4 the level set method was presented both in its original formulation and the recent formulation by Olsson and Kreiss, as well as its application to the two-phase Stokes problem. Two ways of computing the surface tension force density were discussed, where one which is not often seen in the literature appeared to be more appealing.

We then derived or stated (in chapters 5 and 6) a variety of variational forms for the level set and Stokes PDEs that were thought to be of interest. Aside from the obvious ones, these included different time-stepping for the advection equation, the application of SU/PG stabilization to the advection equation, mixed and stabilized forms of the Stokes equations. This chapter elaborated on the possible finite elements for the problem at hand, emphasizing the possibility of choosing a mixed formulation where derivatives of ϕ were expressed on discontinuous finite elements of lower order. That stemmed from the observation that derivatives of a piecewise polynomial function will be discontinuous and piecewise polynomial of lower degree, and that the projection of a discontinuous function onto a continuous finite element may introduce numerical errors that are significant.

In the next couple of chapters we put these ideas through different numerical tests. These included two advection problems to determine the conservation properties of the advection/reinitialization methods. We also ran a set of test for to determine the convergence order of different methods with respect to grid refinement. These tests put the spotlight on the possible increases in accuracy to be had from choosing quadratic basis functions instead of linear, the possible loss of accuracy due to lowering the order of finite element for derivatives of computed functions and the effects of introducing a stable discretization of the advection PDE. Having seen which methods performed well we coupled the level set method to the Stokes solver to do some flow calculations. Here we wanted to confirm that the conservative characteristics of the method could be confirmed in flow calculations of some realism. The first test let a bubble pass through a narrow channel, deforming it in the process. Then we let a circular bubble fall and merge with a horizontal surface, looking at the conservation properties of the method when topology changes occur.

These tests provide some new insight into this method which, while far from revolutionary, are valuable in some sense. A large part of the contribution to the research in the NA-group and to TDB in Uppsala has been implementational. We have developed an extensive implementation using components form FEniCS, a current FEM research code, that provides more modularity, performance and choice than the previous code. However, not all the desired functionality could be delivered in the time-frame of this project (see next section), since we did not want to sacrifice the quality of the numerical work in chapters 8 and 9.

To conclude this project we developed some more experimental extensions to the conservative level set method. We proposed some simple expressions for linking interface thickness to its curvature, hoping to improve the conservation properties when the interface gets significantly deformed. This was done in two ways: letting the interface thickness be determined by the maximum of the interface curvature, and letting the interface thickness vary in space so that it depended on the local curvature of the interface.

12.2 Ambitions yet to be fulfilled

To take the method and implementation presented in this report to the level of actual practical usability one would need adaptive grid refinement. The reason for this is simple: The level set function, ϕ, is constant almost everywhere, but has a sharp transition across the interface. Form experience we know that this transition region must be approximately eight triangles wide to avoid disastrous numerical defects. This means that one has to use a really fine mesh, since taking a wide transition region is unphysical - unless one can refine the mesh only around the interface. We really mean a simple kind of adaptivity: ensuring that a gradient of known magnitude is resolved with a fixed number of triangles.

The 2D-calculations presented took very reasonable time to complete, but full 3D calculations using a uniform mesh were not possible, due to the enormous waste of resolution. As pointed out earlier, the concept of adaptive mesh refinement sits very well with the adaptive interface thickness approach. One could easily use a indicator function for the mesh refinement that is dependent on the interface curvature maximum. It would also be suitable to refine the mesh around large velocity gradients.

Another ambition in this project that failed to materialize as hoped for was to develop a Navier-Stokes module to go with the level set solver, as an alternative to the Stokes solvers. It was understood to be unrealistic to achieve a high performance Navier-Stokes solver within the scope of the project, so it was dropped - but the structure of the code is there so that it can be inserted into the existing framework in a modular fashion.

12.3 Conclusions

12.3.1 Conservation of mass

In Ch. 8 we were able to confirm that a finite element method for the conservative level set method does provide conservation of mass. All combinations of finite elements produced good conservation behavior. In particular we conclude the following:

1. Values for both drift and variance are roughly double for the quadratic finite element methods, while still being very small.
2. There are no discernible differences in conservation between the equal order methods and their mixed order counterpart.
3. Conservation is not improved by using SU/PG stabilization of the advection step.
4. Using second order, Crank-Nicholson, time-stepping does provide much better conservation properties than the simple Euler step.
5. Values for drift and variance converge rapidly as the grid gets finer.
6. Conservation of mass is lost when the interface thickness is not small compared to its curvature.

Take together, the most important conclusion is that, in terms of conservation of mass, neither a quadratic finite element method or the SU/PG stabilized method has been observed to be better than the linear finite element method with Crank-Nicholson time-stepping.

12.3.2 Accuracy

In contrast to the conservation calculations, the accuracy calculations provided great differences between methods, as seen in Ch. 8. The following table summarizes the convergence orders in space:

	Conv of	Conv of
Method	$\|\phi_T - \phi_{**}\|_{L2}$	$\|\phi_T - \phi_{**}\|_{\infty}$
c1/d0	0.303	-0.395
c1/c1	2.422	2.036
c1/c1 SUPG	2.432	2.095
c2/d1	0.675	-0.453
c2/c2	2.886	2.090

From these numbers and the plots in Fig. 8.8 we draw some clear conclusions:

1. The mixed linear/constant method, i.e. "c1/d0", is rubbish in terms of accuracy. The max norm of the error even diverges! This loss of accuracy is probably due to the piecewise constant element for $\hat{n}$.
2. The c1/c1 method performs very well. The convergence in L2-norm compared with the reinitialized exact solution is suspiciously good, but the convergence in max-norm is more sane.
3. The SUPG stabilized method is only better by the slimmest margin.
4. The quadratic/linear mixed method, c2/d1, shows erratic convergence, but is still the second most accurate method in the set.
5. The quadratic equal order method, c2/c2, converges close to the $p = 3$ mark in L2-norm. The convergence results in the max-norm are rubbish.
6. The center of mass calculation show that all methods produce results that are close, and converging, to the correct point - despite some bad convergence numbers in L2-norm.
7. A practical calculation, i.e. one with time step close to the CFL stability limit, shows an overall second order convergence using the equal order linear FEM.

We may conclude that the champion in terms of accuracy is the equal order quadratic method. The extra accuracy, however, is at the cost of a much more time-consuming calculation compared with the linear methods. The concern was that the linear finite element schemes were not living up to their promise of second order convergence in space, due to various errors in the calculation of the unit normals and the extra time-stepping of reinitialization. That fear was confirmed for the mixed order method but strongly rejected by the equal order method - making the quadratic finite element methods redundant in this setting.

12.3.3 Flow calculations

Conservation of mass was observed in the calculation where a bubble passes through a narrow channel. The case with two interfaces merging provided some interesting phenomena, where the mass peaked as the bubbles merged and then decayed to a level which is slightly higher than the pre-merge mass. This is not ideal, but we were able to establish a convergence towards zero of order ~ 1.5 for this error.

12.3.4 Curvature calculations

The investigations into the problematic calculation of the interface curvature showed that the equal order quadratic method suffered terribly while the mixed quadratic method was fine. We submit that this must be due to the projection onto the continuous finite element. Again, the equal order linear method preduced top notch results.

12.3.5 Adaptive interface thickness

The proposed approach for adaptive interface thickness proved very promising, for the case where we compute a ε based on the maximum curvature of the interface. Mass was not lost due to excessive curvature. The idea that used a locally adaptive interface thickness had to be discarded, due to lack or robustness, lack of conservation of mass and other failures of a more practical kind.

Bibliography

[1] Iserles, Arieh - A First Course in the Numerical Analysis of Differential Equations, 1996

[2] Eriksson, Estep, Hansbo, Johnson - Computational Differential Equations, 1996

[3] Brenner, S, Scott, LR - The Mathematical Theory of Finite Element Methods, 2002

[4] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146-159.

[5] M. Sussman, E. Fatemi, P. Smereka, S. Osher, An improved level set method for incompressible two-phase fows, Comp. Fluid 27 (1998) 663-680.

[6] E. Olsson, G. Kreiss, A conservative level set method for two phase Flow, J. Comp. Phys 210 (2005) 225-246.

[7] A.-K. Tornberg, B. Enhquist, A Fnite element based level set method for multiphase fow applications, Comput. Visual. Sci. 3 (2000) 93-101.

[8] J.U. Brackbill, D. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335-353.

[9] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski and G. Zanetti, Modeling Merging and Fragmentation in Multiphase Flows with SURFER, J. Comput. Phys. 113 (1994) 134-147

[10] R. Goldman ,Curvature formulas for implicit curves and surfaces, Comp. Aided Geom. Design 22 (2005) 632-658

[11] A. Brooks, T. Hughes, Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations, Computer Methods in Applied Mechanics and Engineering 32 (1982) 199-259

[12] T. Hughes, LP. Franca, GM. Hulbert, A new finite element formulation for computational fluid dynamics: VIII. The galerkin/least-squares method for

advective-diffusive equations, Computer Methods in Applied Mechanics and Engineering 73 (1989) 173-189

[13] T. Hughes, GN. Wells, Conservation properties for the Galerkin and stabilised forms of the advection?diffusion and incompressible Navier?Stokes equations, Computer Methods in Applied Mechanics and Engineering 194 (2005) 1141-1159

[14] C. Johnson, Numerical solutions of partial differential equations by the finite element method, Lund: Studentlitteratur, 1987.

[15] T. J. R. Hughes, L. P. Franca, and M. Balestra, A new finite element formulation for computational fluid dynamics. V. Circumventing the Babuska-Brezzi condition: a stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations, Comput. Methods Appl. Mech. Eng., 59 (1986), pp. 85-99.

[16] E. Olsson, G. Kreiss, A conservative level set method for two phase Flow II, Submitted

[17] A.-K. Tornberg, Interface Tracking Methods with Applications to Multiphase Flows, Dissertation KTH, Stockholm (2000). ISBN 91-7170-558-9

[18] A. Logg, Automating the finite element method - lecture series, http://www.fenics.org/pub/documents/fenics/presentations/logg-geilo-2006-03/logg-geilo-2006.pdf, (downloaded 2006-11-30).

[19] R.C. Kirby, A. Logg, A Compiler for Variational Forms, ACM Transactions on Matheatical Software 32 (2006) 417 - 444

[20] www.fenics.org

TRITA-CSC-E 2006:153
ISRN-KTH/CSC/E--06/153--SE
ISSN-1653-5715

www.kth.se