Interface Tracking Methods with Application
to Multiphase Flows

Anna-Karin Tornberg

Doctoral Dissertation
Stockholm 2000

Royal Institute of Technology
Department of Numerical Analysis and Computing Science
Akademisk avhandling som med tillstånd av Kungl Tekniska Högskolan framläggas till offentlig granskning för avläggande av teknisk doktorsexamen fredagen den 12 maj 2000 kl 10.00 i Kollegiesalen, Administrationsbyggnaden, Kungl Tekniska Högskolan, Valhallavägen 79, Stockholm.

ISBN 91-7170-558-9
TRITA-NA-0010
ISSN 0348-2952
ISRN KTH/NA/R-00/10-SE

© Anna-Karin Tornberg, April 2000
Abstract

The numerical tracking of interfaces is an important part in simulations of many physical processes. Three different techniques for interface tracking are presented: the segment projection, the level-set and the front-tracking methods. The segment projection method is an altogether new method in which the representation of an interface is comprised of a union of overlapping patches. When the interfaces are curves in $\mathbb{R}^2$, the patches are curve segments, chosen such that they can be represented as functions of one coordinate variable. These segment functions are discretized on one-dimensional Eulerian grids. To continuously track moving and deforming interfaces, dynamic construction and destruction of segments are employed.

Applications to multiphase flows involving immiscible fluids are studied in detail. The three tracking methods have been integrated into a finite element approximation of the two-dimensional incompressible Navier-Stokes equations. The interfaces separating the different fluids introduce singularities to the problem in the form of discontinuous density and viscosity across interfaces and singular surface tension forces. These quantities are more naturally represented numerically in the finite element setting, which uses a weak form of the equations.

Integrals in the weak formulation include the discontinuous density and viscosity functions. A rigorous analysis of the errors associated with integration of functions that are discontinuous across curves in $\mathbb{R}^2$ is presented. Regularized approximations of the discontinuous functions, designed to minimize these errors, are introduced. A similar analysis is made for the integration of singular functions with support on curves in $\mathbb{R}^2$.

Topology changes of interfaces, such as in the merging of bubbles, are naturally included in the level-set method. This method is however not so well suited for simulations of physical problems in which merging should not occur. In the segment projection method, the Eulerian grid facilitates the handling of merging, although it requires a specific algorithm. Different physical conditions may be used to define if and when merging should occur.

Numerical results are presented and include tests of convergence rates and comparison of the three methods in the case of one rising buoyant bubble. Runs with topology changes are presented for the level-set method, and runs with and without the possibility of merging are presented for the segment projection method.
Contents

1 Introduction 1
  1.1 Interface Tracking .................................................. 1
  1.2 The Multiphase Flow Problem ........................................ 4
  1.3 New Results and Outline ............................................. 7

2 Interface Tracking 11
  2.1 The Segment Projection Method ....................................... 11
  2.2 The Level-Set Method ................................................ 15
  2.3 The Front-Tracking Method .......................................... 17

3 The Multiphase Flow Problem 19
  3.1 The Navier-Stokes Equations ......................................... 19
  3.2 Interface Tracking Methods Applied to the Multiphase Flow
      Problem ............................................................... 21
    3.2.1 The Segment Projection Method ................................. 22
    3.2.2 The Level-Set Method ......................................... 24
    3.2.3 The Front-Tracking Method .................................... 24
  3.3 Discontinuities and Singularities .................................. 26

4 Numerical Integration of Discontinuous and Singular Functions 27
  4.1 Introduction .......................................................... 27
    4.1.1 Preliminaries .................................................... 29
    4.1.2 Numerical Test Case ............................................. 30
  4.2 Integration with Modified Quadrature ................................ 31
    4.2.1 Integration of the Characteristic Function $H(d(x))$ ........ 31
    4.2.2 Integration of the Product $H(d(x)) \cdot G(x)$ ............... 33
  4.3 Integration with Regularized Integrands: Error Analysis ............ 34
    4.3.1 Introduction of Moments ....................................... 35
    4.3.2 Analytical Error for Discontinuous Functions ............... 38
    4.3.3 Analytical Error for Singular Functions ..................... 45
    4.3.4 Relation Between Heaviside and Delta Function
        Approximations ..................................................... 53
    4.3.5 Numerical Error ................................................ 55

v
4.4 Integration with Regularized Integrands: Numerical Results . . . 59
4.4.1 Integration of the Approximation $H_w(d(x))$ . . . . . . . . 59
4.4.2 Integration of the Product $H_w(d(x)|G(x)$ . . . . . . . . . 61
4.4.3 Integration of the Delta Function Approximation . . . . . . . 65
4.5 Summary of the Quadrature Analysis . . . . . . . . . . . . . . . 67

5 Discretization
5.1 Discretization of the Navier-Stokes Equations . . . . . . . . . . 71
5.1.1 Time Discretization . . . . . . . . . . . . . . . . . . . . . . . . 71
5.1.2 Incompressibility . . . . . . . . . . . . . . . . . . . . . . . . . . 72
5.1.3 Spatial Discretization . . . . . . . . . . . . . . . . . . . . . . . 73
5.1.4 Solution of the Linear Algebraic System . . . . . . . . . . . 75
5.2 The Discrete Segment Projection Method . . . . . . . . . . . . . 76
5.2.1 Discretization of Segments . . . . . . . . . . . . . . . . . . . . 76
5.2.2 Advection and Reinitialization . . . . . . . . . . . . . . . . . . 77
5.2.3 Evaluating the Interfacial Force Term . . . . . . . . . . . . . 80
5.2.4 Density and Viscosity Fields . . . . . . . . . . . . . . . . . . . 81
5.3 The Discrete Level-Set Method . . . . . . . . . . . . . . . . . . . 83
5.3.1 Advection and Reinitialization . . . . . . . . . . . . . . . . . . 83
5.3.2 Evaluating the Interfacial Force Term . . . . . . . . . . . . . 85
5.3.3 Density and Viscosity Fields . . . . . . . . . . . . . . . . . . . 86
5.3.4 Advection and Reinitialization on a Refined Grid . . . . . . . 87
5.4 The Discrete Front-Tracking Method . . . . . . . . . . . . . . . . 88
5.4.1 Discretization and Advection . . . . . . . . . . . . . . . . . . . 88
5.4.2 Evaluating the Interfacial Force Term . . . . . . . . . . . . . 89
5.4.3 Density and Viscosity Fields . . . . . . . . . . . . . . . . . . . 90
5.5 Interaction Between Interfaces . . . . . . . . . . . . . . . . . . . 91
5.5.1 Merging Interfaces for the Segment Projection Method . . . 92
5.5.2 Merging Interfaces for the Level Set Method . . . . . . . . . 96
5.6 The Algorithm in Summary . . . . . . . . . . . . . . . . . . . . . . 96

6 Numerical Results
6.1 Convergence Results for a Buoyant Bubble . . . . . . . . . . . . . 99
6.2 Comparison for Single Buoyant Bubble . . . . . . . . . . . . . . 105
6.3 Remarks on Force Calculations in the Level-Set Method . . . . . 110
6.3.1 Curvature Calculations . . . . . . . . . . . . . . . . . . . . . . . 111
6.3.2 Force Evaluations . . . . . . . . . . . . . . . . . . . . . . . . . . 116
6.4 Bubble Dynamics with Topology Changes . . . . . . . . . . . . . 119
6.5 Sedimentation of Viscous Drops . . . . . . . . . . . . . . . . . . . 125

A Quadrature Rules

Contents
Acknowledgements

I would like to express my deep and sincere gratitude to my advisor, professor Björn Engquist, for his encouragement and support throughout my PhD studies. His great knowledge and experience make it truly inspiring to work with him, and I very much appreciate all enlightening and fruitful discussions. He has also at several occasions arranged most productive visits for me at University of California at Los Angeles (UCLA).

I am grateful to Dr. Olof Runborg for his valuable suggestions concerning the quadrature study presented herein, and for his many constructive editorial comments.

I would also like to thank Katarina Gustavsson for many interesting discussions, scientific and other, and for her nice company at different scientific meetings throughout Europe.

I am also grateful to Dr. Gunilla Kreiss, Prof. Jesper Oppelstrup, Dr. Björn Sjögreen and Prof. Anders Szepessy for always finding time for questions and scientific discussions.

I also want to extend many thanks to all my colleagues and friends at NADA, for enriching my everyday life at the department. Many thanks also to all the people at UCLA for welcoming me with great hospitality.

Finally, I would like to thank my family and friends for all their encouragement and interest. A big and special thanks to Fredrik for his neverending support and strong confidence in me.

Financial support has been provided by the Royal Institute of Technology (excellencejämför, internationaliseringsmedel), TFR (CMF), NUTEK (PSCI), FRN (PDC) and the NSF grant DMS97-06827, and is gratefully acknowledged.
Chapter 1

Introduction

Interfaces or internal boundaries are present in many different applications, such as high frequency wave propagation, solidification and multiphase flows. In each of these applications, the interfaces move according to some physical law. In the numerical simulations of these problems, it is important to keep an updated representation of the interfaces but there are often also other numerical challenges. For multiphase flows, each interface is separating two immiscible fluids. The two fluids will in general have different densities and viscosities and these physical quantities will therefore be discontinuous across each interface. In addition, surface tension forces act at each interface, with a strength that depends on the interface shape. Furthermore, the interfaces may change topology as separated volumes of the same fluid merge. All these effects require careful attention in the construction of numerical methods for multiphase flow problems.

1.1 Interface Tracking

Interface tracking methods are developed to describe and track moving and deforming surfaces or curves that represent various kinds of internal boundaries, interfaces and fronts. The law of motion of the interfaces might be directly connected to the shape of the interface as in geometrical applications such as motions by mean curvature. It might also be a passive tracking, where the interface is moved according to a given vector field. Very often the shape of the interface affects the vector field of advection, as it is determined according to some physical law. An example of this is the physics of multiphase flow.

In interface tracking methods, the interfaces are represented by continuously updated discretizations. These discretizations can be Lagrangian or Eulerian in nature, depending on the interface tracking technique that is used.

In the Lagrangian approach, marker particles are used to define the interfaces. This approach was used by Peskin [32], as he applied his immersed boundary method to calculations of blood flow in the heart. The moving walls of the heart
were treated as internal boundaries. More recent immersed boundary methods as well as front-tracking methods are also based on Lagrangian markers. Some parameterization connecting these markers is needed to make the description of the interface complete, [24, 33, 36, 44].

A different idea is employed in the level-set method, which is based on a fixed Eulerian grid. This method was introduced by Osher and Sethian [29]. In this method, the interfaces are defined as the zero level set of a continuous function, and this function is updated in order to follow the position of the interfaces. The level-set function is discretized on a Eulerian grid that is defined on the computational domain. If the law of motion is such that it only describes the movement of the interfaces, a vector field for updating the level-set function can be obtained by computing extension velocities everywhere in the computational domain, starting with the velocities prescribed at the interfaces, see for example [1, 31, 38]. By embedding the interfaces as a level set of a higher dimensional function, the dimension of the advection problem is increased by one. Hence, the computational expense is a drawback of the level-set technique. More localized level-set methods with lower computational cost have been developed for finite difference implementations, see [28, 38]. The natural ability of the level-set method to handle topological changes in the interfaces is for many applications an important advantage of the method. The level-set method has been applied to many different problems, such as kinetic crystal growth, Stefan problems, epitaxial growth of thin films and compressible and incompressible flow, [27, 28, 38, 43]. Methods like the level-set method, where the interfaces are not explicitly discretized, are often referred to as interface capturing techniques. For simplicity, we refer also to this method as an interface tracking method.

In this thesis, we introduce a new method which can be viewed as a compromise between the front-tracking and level-set methods. The new segment projection method relies on the partitioning of an interface into several parts. In the case of two-dimensional calculations, the interfaces are curves in the plane. Each curve is described by a union of curve segments, where the segments are chosen such that they can be given as functions of one spatial variable. The segments are overlapping, and the coordinates of points on the curve are either given by \((x, y) = (x, f_i(x))\) or \((x, y) = (g_j(y), y)\). The domain of the independent variable of each segment is the projection of the segment onto the coordinate axis for this variable. Information about the connectivity of segments is needed to complete the description. The segment functions \(f_i(x)\) and \(g_j(y)\) are discretized on Eulerian uniform one-dimensional grids in the \(x\) and \(y\) directions. Hence, the curve is explicitly discretized with points on the curve as in the front-tracking method, thereby keeping the lower dimensionality, but the discretization is Eulerian as in the level-set method. One example of a curve and its segment representation is given in Figure 1.1. The uniform discretizations of two of the segments are indicated in Figure 1.2.

Interfaces interact quite differently for different applications. In some cases, there might be superposition. For example, if the interfaces represent wave fronts, one interface might pass over another without interaction. In other cases,
two interacting interfaces will reconnect and create one single interface. This is exemplified by multiphase flow, where two separated volumes of the same fluid might merge into one connected volume, as in the case of merging bubbles. Finally, for some physical parameters in multiphase flow, there will be no merging, even if separated volumes of the same fluid get close. Two viscous drops might continue to move close together, being subjected to the same velocity field, without ever merging.

If there is a merger, there will be a change of topology of the interfaces. In the level-set method, the topology is determined by the zero level set of the level-set function \( \phi(x, t) \). If a change in topology occurs, this is only seen as a continuous evolution of \( \phi(x, t) \), which changes the pattern of the zero contours, and hence the topology of the problem. Merging is therefore naturally incorporated in the level-set method.

In both the front-tracking and segment projection method, each interface has a separate representation. If merging of two interfaces should occur, the two separate discretizations of the interfaces must be merged into one, which requires appropriate modifications. This is rather complicated for the front-tracking method, since there is no restriction in the positions of the discrete points of the two merging interfaces. For the segment projection method, merging is easier to perform, since the segments are defined as functions of one variable, and can more readily be compared to each other.

In the case of superposition, or when interfaces should not merge, nothing special needs to be done in the front-tracking and segment projection methods. In the level-set method, topological merging will always occur when interfaces defined by the same level-set function get close relative to the resolution of the grid. Therefore, explicit action must be taken to prevent merging.
1.2 The Multiphase Flow Problem

In multiphase flow simulations for immiscible flows, the exact positions and shapes of the interfaces separating the immiscible fluids contribute strongly to the physics of the problem. Surface tension forces act at each such interface, and the strengths of these localized forces are proportional to the local curvature of the interface. The fluids will in general have different densities and viscosities, and hence discontinuities in density and viscosity occur across interfaces. One example of a configuration, involving two different fluids is shown in Figure 1.3.

For these kinds of problems, one might consider a method based on interface fitted meshes. The mesh on which the fluid equations are discretized and solved is updated continuously to fit the deforming fluid interfaces. The interfaces are treated as internal boundaries on which boundary conditions are prescribed. Drawbacks with this method include the cost of the computation and the interpolation errors introduced when remeshing. One example of an algorithm based on interface fitted meshes is the work of Hu and Joseph, [20]. This approach is however common for for fluid-particle flows, see for example [25].

It is often more efficient to keep the grid of the flow variables fixed and to represent the interfaces by some interface tracking technique. Any such technique applied to this problem must include a method to compute surface tension forces and density and viscosity fields.

Many different interface tracking and interface capturing methods have been
applied to multiphase flow problems. One early method was the Marker in Cell (MAC) method [18]. In this method, a number of discrete Lagrangian particles are advected by the local flow. The distribution of these particles identifies the regions occupied by a certain fluid. In volume-tracking or Volume of Fluid (VOF) methods, a fractional volume function is defined to indicate the volume fraction of a certain fluid in each grid cell, see for example [19]. A review of such methods, together with a comparison between some of them is presented in [34]. In these methods, no explicit representations of the interfaces are defined, instead they are reconstructed locally. In order to model surface tension effects, which is difficult in the VOF methods, the continuum surface force (CSF) model was introduced by Brackbill et al., [7]. More recent implementations of these methods were made by Wu et al., [45].

![Figure 1.3](image)

**Figure 1.3.** Example of a configuration involving two fluids $A$ and $B$, with different density and viscosity ($\rho_A, \mu_A$) and ($\rho_B, \mu_B$).

A front-tracking method designed to simulate two-phase flow, or more specifically the motion of bubbles in a surrounding fluid, was introduced by Unverdi and Tryggvason, [44]. A finite difference technique was used for the discretization of the Navier-Stokes equations. In the front-tracking method, the separate data structures that represent the interfaces are based on a connected set of discrete points, where these points are advected individually by the local fluid velocity. Parts of the interface might thereby get depleted of points, and new points must be inserted into the data structure. At other parts of the interface, points must be removed to avoid clustering. This is a straight-forward procedure in a two-dimensional simulation, but much less so for three-dimensional calculations. It is quite hard to implement a merging algorithm in the front-tracking method due to the unstructured positioning of Lagrangian markers. Software including interface topology changes and three dimensional tracking has been developed by Glimm and coworkers, see [13] and the references therein for detailed description and applications.

The application of the level-set method to multiphase flows has been de-
scribed by Sussman et al. in [41, 42, 43]. In this formulation, the equations for updating the level-set function as well as the Navier-Stokes equations were discretized by finite difference techniques. The interfaces separating two fluids \(A\) and \(B\) are represented as the zero level set of a continuous function, designed to be of one sign in fluid \(A\), and of opposite sign in fluid \(B\). This level-set function is initialized as a signed distance function, carrying information about the closest distance to any interface. As the level-set function is advected by the flow, this property will not be retained. Sussman et al. [43] however showed that keeping the level-set function as a distance function is critical in order to obtain an acceptable conservation of mass. They introduced a reinitialization procedure designed to recover this property. The level-set method easily extends to three dimensional calculations.

The new segment projection method can also be applied to the multiphase flow problem. Each curve representing an interface is described as a union of overlapping curve segments. All the segments are advected by the flow, and the segment functions are updated in each time step. The geometry of the interface determines how many segments that are needed for the representation. After each advection step, the segment representation needs to be reinitialized. Dynamic creation and destruction of segments are necessary for tracking the deformable interfaces. The domain of definition of each segment and the connectivity of segments also need to be kept updated. When two regions of the same fluid merge, the segments must be reconnected to represent the new topology, which requires a specific algorithm. This step gives full control over the merging process and the surface physics may influence if or when merging should take place. The basic principles of the segment projections method extends to three dimensions, but the implementation is however more complex.

In all three methods, reinitialization between advection steps needs to be applied. In the front-tracking method, the concern is the distribution of the Lagrangian markers. In the level-set method, the distance function property for the level-set function needs to be restored, and in the segment projection method, the segment structure must be reviewed. The computational costs of these reinitialization procedures are different, ranging from the inexpensive insertion of points in the front-tracking method to the reinitialization of the level-set function.

The immiscible and incompressible flow is described by the incompressible Navier-Stokes equations. The interfaces separate the fluids and thus regions of different density and viscosity. The surface tension forces are located at the interfaces. The localization of the surface tension forces is defined by Dirac delta functions supported at the interfaces. Both for the front-tracking method in [44] and the level-set method in [43], the Navier-Stokes equations were discretized using finite difference techniques. In both cases, surface tension forces are regularized by the use of a mollified delta function, as described by Peskin [32].

In the weak form of the Navier-Stokes equations, the surface tension forces are included as line integrals along the interfaces. Furthermore, the differentiation of the discontinuous viscosity functions is replaced by the differentiation of test
functions, via Green's formula. This reduces the singularity of the problem that we need to treat numerically.

These advantages motivate a choice of a method based on a weak formulation of the equations, such as the finite element method. In addition, the finite element method provides the capability to use variable spatial resolution and to perform computations on domains of various geometrical shapes.

1.3 New Results and Outline

The new contribution with respect to the front-tracking and level-set methods are their integration into the finite element framework applied to the Navier-Stokes equations for incompressible and immiscible multiphase flow. The segment projection method is an altogether new method. It is also integrated into the same finite element framework.

The tracking properties are defined for all three methods, including the explicit discretizations of the interface representations and algorithms for updating these representations according to the law of motion. For the multiphase flow applications, the interfaces are advected by the flow field. In addition to the tracking properties, algorithms for the approximation of the surface tension forces and discontinuous density and viscosity are given for all three methods.

As was discussed earlier, merging and associated topology changes are naturally included in the level-set method. Due to the Eulerian discretization of the segments, it is feasible to simulate mergers also in the segment projection method. We have not developed any merging algorithm for the front-tracking method, since its unstructured distribution of points at the interfaces makes it less suitable for such operations.

The minimum distance for which two separate zero level-sets of a function can be resolved, depends on the resolution of the grid on which the function is discretized. Therefore, in the level-set method, two separate volumes of the same fluid merge when they get close enough compared to the resolution of this grid. In the segment projection method, when mergers should be allowed, a distance \( \delta \) can be set as the given distance at which two separate interfaces should merge. Since we can control when merging takes place, it is also possible to define other criteria to determine when it should occur.

Since the methods are integrated into a finite element framework, the weak form of the Navier-Stokes equations is discretized. This formulation includes line integrals along the interfaces, accounting for the surface tension forces, and integrals over the domain containing discontinuous coefficients, in terms of the density and viscosity fields. Motivated by the presence of the discontinuous density and viscosity in these integrals, the errors associated with the evaluation of integrals of functions discontinuous across curves in \( \mathbb{R}^2 \) have been analyzed. We study two approaches for this integral evaluation. In the first approach, a special quadrature method is introduced for the quadrature over triangles affected by the discontinuity. In the second approach, the discontinuity is smoothed out.
over a transition zone of a fixed width. The error for this second approach will consist of two parts: the analytical error made when replacing a discontinuous function by an approximation, and the numerical error from the integration of this approximation. These errors are analyzed, and we show that vanishing moments of a certain error function are needed to obtain a small analytical error. The regularity of the approximation is shown to be critical for the numerical error.

In the same context, we study the evaluation of integrals with integrands containing a Dirac delta function with support on curves in $\mathbb{R}^2$. As an alternative to evaluate the equivalent line integrals along these curves, the delta function is replaced by an approximation with support in small neighborhoods of each curve, and the integration is performed over the computational domain $\Omega \subset \mathbb{R}^2$. Specific conditions for creating approximations that will yield a certain order of accuracy are given, and a number of such approximations are computed.

These approximations could be convenient to use to evaluate the integral over the surface tension forces in the same framework as the other integrals in the variational formulation. To define the surface tension force in one point, the delta function is multiplied by the curvature and the normal vector of the interface, yielding the strength and direction of this force. To use this approach, the curvature and normal vectors must be computed in the region of support of the approximate delta function. This is natural for the level-set method, since the level-set function is discretized on the whole computational domain, but not for the other two methods, where all the points in the discretization lie on the interface.

In our study, the function multiplying the delta function is assumed to be smooth. If the computed curvature and normal vectors are not guaranteed to be smooth in the region of support of the approximate delta function, there are however still some concerns about using these approximations for the integration of the surface tension forces. This is in general the case in the level-set method, see section 6.3. We have made the choice to evaluate the equivalent line integral instead of using an approximate delta function in all three methods. This line integral is evaluated locally in each element intersected by an interface. The details of this evaluation are different for the three tracking methods.

Since we have smooth functions multiplying the discontinuous density and viscosity in the integrals in which they appear, the above mentioned concern does not arise in this case. In all three methods, the discontinuous density and viscosity fields are approximated by regularized functions chosen according to the results of the analysis made concerning integration of discontinuous functions.

The discretization of the Navier-Stokes equations is based on quadratic polynomials on a triangulated domain. The time-stepping scheme for the Navier-Stokes equations is a combined implicit-explicit second order method, with iterations for the non-linear terms. The resulting problem is solved using an iterated penalty method that enforces the incompressibility constraint. The system of equations is solved by a preconditioned conjugate gradient method.

We present computational results for a single bubble rising due to buoyancy.
The convergence rates of the three methods are computed, indicating essentially second order spatial accuracy for all three methods. The results from the three methods show good agreement, in both position and shape of the rising bubble. It is more difficult to obtain a good mass conservation with the level-set method compared to the other methods.

Furthermore, runs with topology changes including bubbles and a free surface are presented for the level-set and the segment projection methods. Bubbles merge while rising due to buoyancy, finally breaking a free surface.

Two runs of ten viscous drops settling in a box are presented for the segment projection method. In these runs, no mergers are allowed. The drops are initially distributed throughout the computational domain, and fall towards the bottom of the domain, where they settle. In one of the runs, the viscosity and surface tension of the drops are high enough for the drops to resist deformation, thereby resembling solid particles.

![Figure 1.4](image_url)  
Figure 1.4. Results from runs with the segment projection method. The arrows represent the velocity field. The results are presented in sections 6.4 and 6.5, respectively.
Chapter 1. Introduction

The outline of this paper is as follows. In chapter 2, the three different interface tracking methods are described. In chapter 3, the equations for multiphase flow are presented, together with the extensions needed to make the interface tracking methods applicable to this problem. The errors associated with the evaluation of integrals of discontinuous and singular functions are analyzed in chapter 4. The description and outline of that study is given in section 4.1. Results from the analysis are used in the evaluation of integrals containing the discontinuous density and viscosity. The discretizations of the full methods are given in chapter 5. The discretization of the Navier-Stokes equations is discussed in section 5.1, followed by descriptions of the discrete segment projection, level-set and front-tracking methods in sections 5.2-5.4. In chapter 6, the numerical results are presented. Sections 6.1-6.2 are devoted to a single buoyant bubble: in section 6.1 the convergence test is presented, and the comparison of the methods is made in section 6.2. Results from numerical tests of the force calculations in the level-set method are given in section 6.3, and some remarks regarding these calculations are given. In section 6.4, the runs with the level-set method and the segment projection method including topology changes are presented. Finally, the results obtained with the segment projection method for non-merging viscous drops are presented in section 6.5.

This thesis is partially based on the following papers.


Chapter 2

Interface Tracking

Introduce the interface $\gamma$ bounding a region $\Omega_A$ that may be multiply connected. The region $\Omega_A$ is contained in the computational domain $\Omega \subset \mathbb{R}^2$. In general, $\gamma$ will consist of several separate interfaces, i.e.

$$\gamma = \bigcup_j \gamma_j. \quad (2.1)$$

Each of the interfaces $\gamma_j$ encloses a region of $\Omega_A$, possibly together with parts of $\partial \Omega$, the boundary of $\Omega$.

In this chapter we describe three interface tracking techniques, designed to track the evolution of $\gamma$: the segment projection method, the level-set method and the front-tracking method.

2.1 The Segment Projection Method

In the segment projection method, each interface $\gamma_j$ is represented by a union of overlapping curve segments. The segments are chosen such that they can be given as functions of one coordinate variable, i.e. the segments are represented by functions $f_i(x)$ and $g_j(y)$. The domain of the independent variable of such a function is the projection of the segment onto its coordinate axis. The coordinates of points on the curve are given by $(x, y) = (x, f_i(x))$ or $(x, y) = (g_j(y), y)$.

A simple example is the description of a circle. A circle can be described by four segments, two ‘$x$-segments’, i.e. functions $f_1(x)$ and $f_2(x)$, and two ‘$y$-segments’, i.e. functions $g_1(y)$ and $g_2(y)$. A circle $\gamma_1$ with radius 1, centered at the origin, is given by $\gamma_1 = \{ (x, y), \ x^2 + y^2 = 1 \}$. The segment description is

$$\gamma_1 = \bigcup_{k=1}^{4} s_k. \quad (2.2)$$
where the segments are defined by:

\[
\begin{align*}
    s_1 &= \{(x,y), \quad y = f_1(x) = \sqrt{1-x^2}, \quad |x| \leq 1\}, \\
    s_2 &= \{(x,y), \quad y = f_2(x) = -\sqrt{1-x^2}, \quad |x| \leq 1\}, \\
    s_3 &= \{(x,y), \quad x = g_1(y) = \sqrt{1-y^2}, \quad |y| \leq 1\}, \\
    s_4 &= \{(x,y), \quad x = g_2(y) = -\sqrt{1-y^2}, \quad |y| \leq 1\}.
\end{align*}
\]

The domains of the independent variables are \(|x| \leq 1\) for \(f_1(x)\) and \(f_2(x)\), and \(|y| \leq 1\) for \(g_1(y)\) and \(g_2(y)\). A sketch of the segments describing the circle can be seen in Figure 2.1.

![Figure 2.1](image)

**Figure 2.1.** A sketch of the segments describing a circle. The domain of the independent variable for each segment is the projection of the segment onto the coordinate axes.

For each point at the curve \(\gamma\), there is at least one segment describing the curve. To make the description complete, information about the connectivity of segments must be provided.

The number of segments needed to describe a curve depends on the shape of the curve. An extremum of a function \(f_i(x)\) defines a break point in the \(y\)-segments, as no segment given as a function of \(y\), can continue past this point. After this point, we again encounter the same \(y\)-values, and need another segment defined as a function of \(y\) to define this part. Similarly, an extremum of a function \(g_j(y)\) defines a break point in the \(x\)-segments. A sketch of the distribution of segments is shown in Figure 2.2.

The segments are moved by the equation of motion, and after each advection step, the segment representation must be reinitialized. Dynamic creation and destruction of segments is employed to describe the evolution of curves. New segments should be created if new extrema have appeared, and segments should be removed when extrema disappear. The connectivity of segments must be kept updated.
2.1. The Segment Projection Method

For each segment, the domain of the independent variable must be defined. A segment described as a function of $x$, an $x$-segment, is defined from $x_0$ to $x_1$, and a segment described as a function of $y$, a $y$-segment, is defined from $y_0$ to $y_1$. The segments, together with information about connectivity between segments, define the structure that represents the curve.

Given the definition of an $x$-segment, create an ordered set containing the start and end points of the segment, together with the extremum points of the segment. With the number of extremum points equal to $M$, denote the positions of these points by $x^c_0$ to $x^c_{M+1}$ ($x^c_0 = x_0, x^c_{M+1} = x_1$). Then, for each interval

$$ (x_m^c, x_{m+1}^c), \quad m = 0, \ldots, M, \quad (2.7) $$

it is necessary to keep track of which $y$-segment that this part corresponds to. For the $y$-segments, in the same manner we define points $(y_m^c, y_{m+1}^c)$, and corresponding $x$-segments for each part.

Given a velocity field $u = (u, v)$ by which the curve should move, the segments $y = f(x, t)$ and $x = g(y, t)$ are updated according to the partial differential equations

$$ \frac{\partial f}{\partial t} + u \frac{\partial f}{\partial x} = v, \quad (2.8) $$
$$ \frac{\partial g}{\partial t} + v \frac{\partial g}{\partial y} = u. \quad (2.9) $$

Note that there is only one spatial variable present in both these equations.

If the curve is open, boundary conditions must be defined for the segments defining the start and the end of the curve. However, if it is closed, there is an overlap of segments defined in $x$ and $y$, and we can update the boundary values for one segment from the segment in the opposite direction.

After the advection step, we need to review the segment structure. If no new extrema have appeared, and no old have disappeared, no change needs
to be made in the structure of the segments. However, we need to update the positions of the extremum points, and correct the corresponding segment information according to this.

If new extrema have appeared or disappeared, the structure of the segments defined as functions of the other spatial variable needs to be modified. The different scenarios are indicated in Figure 2.3.

![Figure 2.3](image)

(a) Segment with one extremum.  (b) Segment with three extrema.

(c) Segment with one extremum.  (d) Segment with three extrema.

**Figure 2.3.** The four different cases: Development from a) to b) or from b) to a). Development from c) to d) or from d) to c).

Below, we describe the four generic cases. For an $x$-segment, “functions of the opposite variable”, refers to functions of $y$, i.e. $y$-segments, while it refers to functions of $x$ for a $y$-segment.

**Case I:** Two new extrema appear around existing extremum.

Two new segments defined as functions of the opposite variable need to be added in between the extrema.

(Illustrated by $a) \rightarrow b$ in Figure 2.3).

**Case II:** Two extrema vanish. One extremum remains in the same region.

The two segments, defined as functions of the opposite variable in the region between the three old extrema, need to be removed.

(Illustrated by $b) \rightarrow a$ in Figure 2.3).
2.2 The Level-Set Method

Case III: Two new extrema appear away from existing extrema.

The original segment defined as a function of the opposite variable needs to be split into two parts, and a new segment in this direction needs to be added in between the two new extrema, i.e., in between these two parts.
(Illustrated by $c \rightarrow d$ in Figure 2.3).

Case IV: Two extrema disappear. No extremum remains in the same region.
The segment in the opposite direction, defined between the two old extrema, needs to be removed. The two segments in the opposite direction, defined on each side of the segment to be removed, are merged into one segment.
(Illustrated by $d \rightarrow c$ in Figure 2.3).

When we have updated the structure information, we correct the end parts of all segments. In a closed structure (i.e., a structure representing a closed curve), an end point of a segment defined in the $x$-direction, corresponds to an extremum in a segment defined in the $y$-direction. This extremum may have moved, and the domain of definition of the segment in the $x$-direction might need to be extended or decreased. Analogously, the domain of definition of each $y$-segment may possibly need to be modified.

In the case of an open structure (i.e., a structure representing an open curve), the same thing holds for all segments, except for the segments that defines the start and the end of this structure. The first segment in the structure should always be defined starting from a point at a boundary, and the last segment should be defined so that it ends with a point at a boundary.

2.2 The Level-Set Method

In the level-set method, an interface is represented as a zero level set of a continuous function, designed to be of positive sign on one side of the curve, and of negative sign on the other side of the curve. Several interfaces can be represented by the same level-set function $\phi(x)$. For many applications, it is useful to define $\phi(x)$ as a signed distance function. Then $|\phi(x)|$ yields the closest distance to any interface, and the sign of $\phi(x)$ is determined from some sign convention.

As an example, we again define $\gamma$ as the circle centered at the origin with radius 1. We define $\phi(x)$ to be the signed distance function, with a positive sign inside the circle. We have

$$\phi(x) = 1 - \sqrt{x^2 + y^2}. \quad (2.10)$$

This $\phi(x)$ is shown in Figure 2.4.

To move the curve by a law of motion, the full level-set function $\phi(x)$ is updated. Therefore, the vector field determining the movement needs to be defined not only at the curve, but for each point $x$. If the law of motion is such that it only describes the movement of the curve, the full vector field $u$ can be
defined as an extension from this definition, see for example [38]. In the case of multiphase flow, the velocity field \( \mathbf{u} \) is naturally defined for each point \( \mathbf{x} \).

Given a velocity field \( \mathbf{u} \), the advection of the level-set function \( \phi(\mathbf{x}, t) \) is given by

\[
\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0. \tag{2.11}
\]

For multiphase flow simulations, it is critical that \( \phi(\mathbf{x}) \) remains a distance function, in order to obtain acceptable conservation of mass for the different fluids, [43]. This property is however in general not preserved during advection.

If one wishes to restore the level-set function as a distance function in regions close to a zero level set (only there is it an important property), a reinitialization procedure can be applied as part of the calculations.

The following procedure was suggested by Sussman et al. [43]. Let \( \psi_0(\mathbf{x}) = \phi(\mathbf{x}, t^*) \) be the level set function obtained at the time step ending at \( t = t^* \). It will be somewhat distorted from a distance function, and to obtain a true distance function \( \psi \) that has the same zero level set as \( \psi_0 \), the following problem is solved to steady state:

\[
\frac{\partial \psi}{\partial t} = S(\psi_0) (1 - |\nabla \psi|), \tag{2.12}
\]

\[
\psi(\mathbf{x}, 0) = \psi_0(\mathbf{x}). \tag{2.13}
\]
where $S(t)$ is a sign function, given by

$$S(t) = \begin{cases} 
-1 & \text{for } t < 0, \\
0 & \text{for } t = 0, \\
1 & \text{for } t > 0. 
\end{cases} \quad (2.14)$$

Finally, $\phi(x,t^*)$ is replaced by $\psi(x, \infty)$. Alternatively, equation (2.12) can be written as

$$\frac{\partial \psi}{\partial t} + w \cdot \nabla \psi = S(\psi_0), \quad (2.15)$$

where

$$w = S(\psi_0) \frac{\nabla \psi}{|\nabla \psi|}. \quad (2.16)$$

Hence, $w$ is a unit vector, always pointing away from the interface, and information is propagated out from the contour with a speed of one. In practice, the equation does not need to be solved to steady state. The distance function property does only need to be restored in regions close to a zero level-set.

Both the advection and reinitialization equations are hyperbolic partial differential equations. We have used a streamline diffusion method for the discretization of the level-set equations. This is discussed in section 5.3.1.

The most robust and accurate schemes for such equations are only available for the finite difference method, such as the essentially non-oscillatory (ENO) schemes [30]. The equations for some physical problems, in which the level-set technique is used to capture interfaces, are however better discretized starting from their weak form. This was our motivation for using the finite element method for the multiphase flow problem. There are advantages also of using unstructured triangulated meshes for more complex geometries or local mesh refinement. Barth and Sethian [3] have developed numerical schemes, inspired by the techniques used in finite difference methods, for the level-set equations on triangulated domains. One finite element based level-set method for two-phase flows is presented in [26], without including surface tension. In that paper, the author compares different schemes for the level-set equations.

### 2.3 The Front-Tracking Method

In the front-tracking method, each curve is described by a set of discrete points $\{x^{(l)}\}_{l=1}^{N_I} \subset \mathbb{R}^2$, together with a parametric description connecting these points.

Let again $\gamma$ be the circle centered at the origin with radius 1. To represent $\gamma$, we introduce an uniform distribution of points,

$$x^{(l)} = (\cos(\theta_l), \sin(\theta_l)), \quad \theta_l = \frac{2\pi l}{N_I}, \quad l = 1, \ldots, N_I. \quad (2.17)$$
In Figure 2.5, this set of points is shown for \( N_I = 40 \). This is an ordered set of points, and a parametric description, is calculated to connect these points. As will be discussed in section 5.4, a cubic spline has been used in this implementation, but other descriptions are possible as well.

![Figure 2.5](image)

**Figure 2.5.** The discrete points \( \{ x^{(l)} \}_{l=1}^{N_I} \), \( N_I = 40 \), as defined in (2.17). The interface \( \gamma \) is represented by this set of discrete points, together with a parametric description connecting these points.

When the curve should be advected by the law of motion, each point is moved individually, as a Lagrangian marker. Given a vector field \( \mathbf{u}(\mathbf{x}, t) \) by which a curve should be moved, each discrete point is advected by \( \mathbf{u}(\mathbf{x}, t) \),

\[
\frac{d\mathbf{x}^{(l)}}{dt} = \mathbf{u}(\mathbf{x}^{(l)}, t) \quad l = 1, \ldots, N_I.
\]  

(2.18)

After the points have been advected, a new parametric fit is calculated. This is done for each separate curve.

Since there is no constraint on the movements on the Lagrangian markers, modification of the distribution of points is in general needed to avoid clustering or depletion of points. New points can be added where necessary by using the calculated parameterization, and points on other parts of the interface can be removed.
Chapter 3

The Multiphase Flow Problem

In this chapter, the Navier-Stokes equations are presented, and the weak form of these equations is introduced. The requirements on the interface tracking techniques specific to this application are discussed, followed by descriptions of how these requirements are met in the different methods.

3.1 The Navier-Stokes Equations

The equations describing this immiscible multiphase flow are the Navier-Stokes equations for incompressible flow. The contribution of the surface tension forces and the gravity forces are added as source terms.

In this presentation, we assume that we have two different fluids, fluid \( A \) and fluid \( B \). The density and viscosity at a fixed time are given by

\[
(p(x), \mu(x)) = \begin{cases} 
(p_A, \mu_A) & \text{for } x \text{ in fluid } A, \\
(p_B, \mu_B) & \text{for } x \text{ in fluid } B.
\end{cases}
\]  

In general \( \rho_A \neq \rho_B \) and \( \mu_A \neq \mu_B \), so that \( \rho(x) \) and \( \mu(x) \) are discontinuous across each interface separating fluid \( A \) and \( B \). Refer to Figure 1.3 for an example of a configuration of the two fluids \( A \) and \( B \). The Navier-Stokes equations can be written

\[
\rho \left( \frac{\partial u}{\partial t} + u \cdot \nabla u \right) = -\nabla p + \nabla \cdot (\mu \left( \nabla u + \nabla u^T \right)) + f + \rho g, \\
\nabla \cdot u = 0,
\]

in \( \Omega \subset \mathbb{R}^2 \), together with boundary conditions \( u = \nu \) on \( \partial \Omega \). In addition, we need to specify some appropriate initial condition \( u(x, 0) = u_0(x) \). For the velocity field \( u(x, t) \). The pressure field is denoted by \( p(x, t) \). Buoyancy effects
arise from the source term $\varrho \mathbf{g}$, where the gravitational force $\mathbf{g}$ is multiplied by the discontinuous density $\varrho(x,t)$. The source term $\mathbf{f}$ on the right hand side is the surface tension force.

As seen in (2.1), $\gamma$ will in general consist of several separate interfaces $\gamma_j$, where each interface can either be closed or attached to the boundary of the domain. The domain $\Omega_j$ enclosed by $\gamma_j$ is in this case the regions of the domain occupied by fluid $A$. The surface tension force is given by

$$\mathbf{f} = \sigma \kappa \mathbf{n} \delta_{\gamma_j}, \quad (3.4)$$

where $\delta_{\gamma_j}$ is a measure of Dirac delta function type with support on $\gamma_j$, i.e the union of all interfaces $\gamma_j$. Its action on any smooth test function $\varphi(x)$ is given by

$$\int_\Omega \delta_{\gamma_j} \varphi \, d\mathbf{x} = \int_{\gamma_j} \varphi \, d\gamma_j = \sum_j \int_{\gamma_j} \varphi \, d\gamma_j, \quad (3.5)$$

where the last quantity denotes the sum of the line integrals of $\varphi(x)$ along the interfaces $\gamma_j$. The interfaces $\gamma_j$ are advected by the flow field $\mathbf{u}(x,t)$, and change with time. The coefficient $\sigma \in \mathbb{R}$ in (3.4) is the surface tension coefficient, $\kappa \in \mathbb{R}$ is the curvature and $\mathbf{n} \in \mathbb{R}^2$ the normal vector to $\gamma$. The direction of this force is towards the local center of curvature.

In order to give a weak formulation of the equations in (3.2)-(3.3), introduce the spaces

$$\mathcal{V} = H^1(\Omega) \times H^1(\Omega) = \{ \mathbf{v} = \{v_i\}_{i=1}^2 : v_i \in H^1(\Omega) \}, \quad (3.6)$$

$$\Pi = \{ q \in L^2(\Omega) : \int_\Omega q \, d\mathbf{x} = 0 \}, \quad (3.7)$$

where

$$H^1(\Omega) = \{ v : v \text{ is defined on } \Omega \text{ and } \int_\Omega v^2 + |\nabla v|^2 \, d\mathbf{x} < \infty \}, \quad (3.8)$$

$$L^2(\Omega) = \{ v : v \text{ is defined on } \Omega \text{ and } \int_\Omega v^2 \, d\mathbf{x} < \infty \}. \quad (3.9)$$

We define the subspaces

$$\mathcal{V}_0 = \{ \mathbf{v} \in \mathcal{V} : \mathbf{v} = 0 \text{ on } \partial\Omega \}, \quad \mathcal{V}_0 = \{ \mathbf{v} \in \mathcal{V} : \mathbf{v} = 0 \text{ on } \partial\Omega \}, \quad (3.10)$$

where $\nu$ is the boundary condition for $\mathbf{u}(x,t)$.

Multiplying equation (3.2) by $\mathbf{v} \in \mathcal{V}_0$, (3.3) by $q \in \Pi$, and integrating over the domain using Greens' formula, the following variational formulation of (3.2)-(3.3) is obtained:

Find $\mathbf{u}(x,t) \in \mathcal{V}_0$ and $p(x,t) \in \Pi$ such that $\forall t \in [0,T]$

$$m(\rho, \frac{\partial \mathbf{u}}{\partial t}, \mathbf{v}) + a(\mu, \mathbf{u}, \mathbf{v}) + b(\mathbf{v}, \mathbf{p}) + c(\rho, \mathbf{u}, \mathbf{u}, \mathbf{v}) = f_t(\mathbf{v}) + m(\rho, \mathbf{g}, \mathbf{v}), \quad (3.11)$$

$$b(\mathbf{u}, q) = 0, \quad (3.12)$$
hold for all \( \mathbf{v} \in V_0 \), and for all \( q \in Q \), respectively. The bilinear form \( \tilde{a}(\mu, \cdot, \cdot) \) is defined as
\[
\tilde{a}(\mu, \mathbf{u}, \mathbf{v}) = \int_\Omega \mu(x) \left\{ \text{tr}(\nabla \mathbf{u}^T : \nabla \mathbf{v}) + \text{tr}(\nabla \mathbf{u} : \nabla \mathbf{v}) \right\} \, dx
= \int_\Omega \mu(x) \sum_{i,j=1}^{2} \left( \frac{\partial u_i}{\partial x_j} \frac{\partial v_j}{\partial x_i} + \frac{\partial u_i}{\partial x_i} \frac{\partial v_j}{\partial x_j} \right) \, dx, \quad (3.13)
\]
where \( \text{tr}(\nabla \mathbf{u}^T : \nabla \mathbf{v}) \) denotes the trace of the dyadic product of \( \nabla \mathbf{u}^T \) and \( \nabla \mathbf{v} \), and similarly for \( \text{tr}(\nabla \mathbf{u} : \nabla \mathbf{v}) \). Moreover, the forms \( b(\cdot, \cdot, \cdot), c(\rho, \cdot, \cdot, \cdot) \) and \( m(\rho, \cdot, \cdot, \cdot) \) are defined by
\[
b(\mathbf{v}, q) = - \int_\Omega (\nabla \cdot \mathbf{v}) q \, dx, \quad (3.14)
\]
\[
c(\rho, \mathbf{u}, \mathbf{v}, \mathbf{w}) = \int_\Omega \rho(x) (\mathbf{u} \cdot \nabla \mathbf{v}) \cdot \mathbf{w} \, dx, \quad (3.15)
\]
\[
m(\rho, \mathbf{u}, \mathbf{v}) = \int_\Omega \rho(x) \mathbf{u} \cdot \mathbf{v} \, dx. \quad (3.16)
\]
The interfacial force term \( f_\gamma(\mathbf{v}) \) evaluates as a sum of line integrals along the interfaces \( \gamma_j \),
\[
f_\gamma(\mathbf{v}) = \sigma \int_\gamma k \mathbf{n} \cdot \mathbf{v} \, d\gamma = \sigma \sum_j \int_{\gamma_j} k \mathbf{n} \cdot \mathbf{v} \, d\gamma, \quad (3.17)
\]
according to (3.5).

### 3.2 Interface Tracking Methods Applied to the Multiphase Flow Problem

In the multiphase flow problem, interfaces are advected by the flow field. This velocity field is however affected by the configuration of the volumes of different density and viscosity and the surface tension forces, which are all determined by the positions and shapes of the interfaces.

In order to complete the formulation of this problem, we therefore need not only to define how the interface \( \gamma \) is represented and how the evolution of \( \gamma \) is determined. In addition, definitions of the density and viscosity fields using the information from the interface tracking methods are needed, as well as calculation of surface tension forces. These definitions and computations are made somewhat differently depending on which interface tracking method that is used. The definition and advection of \( \gamma \), for each of the three interface tracking methods, have already been discussed in sections 2.1-2.3.
The density and viscosity have one set of values in fluid $A$, $(\rho_A, \mu_A)$ and another set of values in fluid $B$, $(\rho_B, \mu_B)$. At a fixed time, given a characteristic function $I(x)$, where

$$I(x) = \begin{cases} 
1 & \text{for } x \text{ in fluid } A, \\
0 & \text{for } x \text{ in fluid } B.
\end{cases}$$

(3.18)

we can write the density and viscosity as

$$\begin{align*}
\rho(x) &= \rho_B + (\rho_A - \rho_B) \cdot I(x), \\
\mu(x) &= \mu_B + (\mu_A - \mu_B) \cdot I(x).
\end{align*}$$

(3.19)

An algorithm for computation of $I(x)$ must be defined for each interface tracking method.

The definition of the surface tension forces were given in (3.4) as $f = \sigma \kappa n \delta_\gamma$, and in their weak form in (3.17) as a sum of line integrals along each interface. To evaluate this interfacial force term, we need to have a representation of each interface, the curvature $\kappa$ and normal vector $n$ along that interface. In the following sections, we discuss these definitions for the segment projection, level-set and front-tracking methods.

### 3.2.1 The Segment Projection Method

In order to define the characteristic function $I(x)$, two additional variables are assigned to each segment. The coordinates of an $x$-segment is $(x, f(x))$. The variables $Fl_{hc}$ and $Fl_{lc}$, each with value 0 or 1, indicate which fluid an $x$-segment has for coordinates $(x, y)$ with $y > f(x)$, and with $y < f(x)$, respectively. Similarly, for a $y$-segment with coordinates $(g(y), y)$, these variables indicate which fluid one finds for coordinates $(x, y)$ with $x > g(y)$ and $x < g(y)$, respectively. The value of each of $Fl_{hc}$ and $Fl_{lc}$ is set to 1 for fluid $A$ and to 0 for fluid $B$.

The characteristic function $I(x)$ can be defined by using the union of the segments and the indicator variables $Fl_{lc}$ and $Fl_{hc}$ that are defined for each segment. The definition of the characteristic function for a point $x^*$, based on the segment representation of an interface $\gamma_j$ can be described as follows:

1. Given a point $x^* = (x^*, y^*)$, set $l = 0, F_s = 0$.
2. For each $x$-segment whose domain includes $x^*$, set $l = l + 1$.
   The coordinates are $(x^*, f_l(x^*))$.
   If $f_l(x^*) > y^*$, $F_s = F_s + Fl_{hc}$, else $F_s = F_s + Fl_{lc}$.
3. For each $y$-segment whose domain includes $y^*$, set $l = l + 1$.
   The coordinates are $(g_j(y^*), y^*)$.
   If $g_j(y^*) > x^*$, $F_s = F_s + Fl_{hc}$, else $F_s = F_s + Fl_{lc}$.
4. If $F_s > |l/2|$, then $I(x^*) = 1$, else $I(x^*) = 0$. 


3.2. Interface Tracking Methods Applied to the Multiphase Flow Problem

Figure 3.1. The two indicator variables, \( F_{ic} \) and \( F_{he} \), are used when the characteristic function is defined.

If several interfaces exist, define separate characteristic functions \( I_j(x) \) according to the algorithm above. The characteristic function \( I(x) \) is then at each point defined as the maximum over all the \( I_j(x) \)'s. We assume that fluid \( A \) is inside each closed area such that \( I(x) = 1 \) in these regions.

To evaluate the interfacial force term, given in (3.17) we need to have a representation of the interface, and the product of the curvature \( \kappa \) and the normal vectors \( \mathbf{n} \) at the interface. Since we have overlapping segments, we need to define which part(s) of each segment should be used to define the interface, and in which order they should be used. For details about the implementation of this procedure, see section 5.2.3.

The curvature and normal vectors can be computed from the definition of the segments. The product \( \kappa \mathbf{n} \) that is included in the surface tension forces is defined by

\[
\kappa(x) \mathbf{n}(x) = \frac{f''(x)(-f'(x), 1)}{(1 + f'(x)^2)^{3/2}},
\]

(3.20)

for the \( x \)-segments, and

\[
\kappa(y) \mathbf{n}(y) = \frac{g''(y)(1, -g'(y))}{(1 + g'(y)^2)^{3/2}},
\]

(3.21)

for the \( y \)-segments.

The discretization of this method is discussed in section 5.2.
3.2.2 The Level-Set Method

The level-set function $\phi(x, t)$ is initialized as a signed distance function, carrying information about the closest distance to any interface separating the two fluids. It is advected by the velocity field $u$ and reinitialization is applied to restore the distance function property, since this property is not retained during advection.

The sign convention is defined as $\phi(x) > 0$ for $x$ in fluid $A$, and $\phi(x) < 0$ for $x$ in fluid $B$. Since the level-set function $\phi(x)$ is of different sign in the two fluids, the density and viscosity fields are easily defined in terms of $\phi(x)$. We can simply write the characteristic function $I(x)$, defined in (3.18) as

$$I(x) = H(\phi(x)),$$

where $H(t)$ is the Heaviside function,

$$H(t) = \begin{cases} 
0 & \text{for } t < 0, \\
1/2 & \text{for } t = 0, \\
1 & \text{for } t > 0.
\end{cases}$$

(3.23)

The density and viscosity fields are defined through (3.19) using this definition of $I(x)$.

For evaluation of the interfacial force term (3.17), the level-set function $\phi(x)$ is needed to define all parts of the interface $\gamma$. The curvature $\kappa$ and normal vectors $n$ can be calculated as

$$n = \frac{\nabla \phi}{|\nabla \phi|}, \quad \kappa = -\nabla \cdot n.$$  

(3.24)

The unit normal vector $\hat{n}$ always point into fluid $A$ (where $\phi(x) > 0$). The sign of $\kappa$ determines the direction of the product $\kappa \hat{n}$, i.e., the direction of the surface tension force. When $\phi(x)$ is an exact distance distance function, it holds that $|\nabla \phi| = 1$.

The discretization of this method is discussed in section 5.3.

3.2.3 The Front-Tracking Method

In contrast to the level-set method, where the characteristic function $I(x) = H(\phi(x))$, the front-tracking method does not provide any pointwise information about whether a point is inside fluid $A$ or fluid $B$. Instead, the parametric description of each interface $\gamma_j$ needs to be used to determine $I(x)$. This can be done with the notion of the orientation of a curve. Here we assume that each curve, or interface, is closed, with fluid $A$ inside the interface, and fluid $B$ outside.

Denote the parameterization of $\gamma_j$ by $\gamma(s) = (x(s), y(s)), s \in [s_{\text{min}}, s_{\text{max}}]$. To evaluate $I(x_0), x_0 = (x_0, y_0)$, the following algorithm is used:
3.2. Interface Tracking Methods Applied to the Multiphase Flow Problem

i) Find \( s^* \in [s_{\text{min}}, s_{\text{max}}] \) that minimizes

\[
\min_{s \in [s_{\text{min}}, s_{\text{max}}]} \left( (x(s) - x_0)^2 + (y(s) - y_0)^2 \right),
\]

that is, find \( s^* \), such that \( x^* = (x(s^*), y(s^*)) \) is the point on \( \gamma_s \) closest to \( x_0 \).

ii) Define \( \mathbf{x}_d = x_0 - x^* \), and take \( \mathbf{n}_I(s^*) \) to be the normal vector at \( x^* \), pointing into fluid \( A \). This yields

\[
\begin{cases}
I(x_0) = 1 & \text{if } \mathbf{n}_I(s^*) \cdot \mathbf{x}_d \geq 0, \\
I(x_0) = 0 & \text{if } \mathbf{n}_I(s^*) \cdot \mathbf{x}_d < 0.
\end{cases}
\]

If several interfaces exist, define separate characteristic functions \( I_j(x) \), and let the total characteristic function be \( I(x) = \max_j I_j(x) \).

Using the parameterization \( \gamma_s = (x(s), y(s)) \), the force term (3.17) first introduced in equation (3.11), can be evaluated as

\[
f_{\gamma}(v) = \sigma \sum_j \sum_{i=1}^{2} \int_{(x_j)_{\text{min}}}^{(x_j)_{\text{max}}} \kappa_i n_i \kappa v_i d\gamma
\]

\[
= \sigma \sum_j \sum_{i=1}^{2} \int_{(x_j)_{\text{min}}}^{(x_j)_{\text{max}}} \kappa(s) \mathbf{n}_I(s) \kappa(s)q(s) d\gamma.
\]

(3.27)

where the scaling factor \( q(s) = (x'(s)^2 + y'(s)^2)^{1/2} \). Here, \( v \) with components \( \{v_i\}_{i=1}^{2} \) are the test functions from the weak formulation. At any point along the interface, the product of the curvature \( \kappa \) and the unit normal vector \( \mathbf{n} = \{n_i\}_{i=1}^{2} \) points towards the local center of curvature. Introducing the vector \( \mathbf{r}(s) = (x(s), y(s)) \), the tangent \( \mathbf{t}(s) \) to \( \gamma_s \) can be evaluated as

\[
\mathbf{t}(s) = \frac{\mathbf{r}'(s)}{||\mathbf{r}'(s)||}.
\]

(3.28)

The product \( \kappa(s) \mathbf{n}(s) \) determines the surface tension force up to a multiplicative constant. It is given by

\[
\kappa(s) \mathbf{n}(s) = \frac{\mathbf{t}'(s)}{||\mathbf{r}'(s)||}.
\]

(3.29)

In order to be able to calculate curvature and normal vectors, second derivatives \( x''(s) \) and \( y''(s) \) need to be evaluated.

The discretization of this method is discussed in section 5.4.
3.3 Discontinuities and Singularities

The differential form of the Navier-Stokes equations (3.2) includes, in addition to the discontinuous fields $\rho(x)$ and $\mu(x)$, also derivatives of $\mu(x)$. The weak form of the equations has the advantage that the derivatives on $\mu(x)$ can be moved over to the test function by applying Green's formula. The Navier-Stokes equations (3.2) furthermore include the singular surface tension forces as a source term. The localization of these surface tension forces to the interfaces is formalized by the Dirac Delta function $\delta_x$. In the weak form of the equations (3.11), where the equation has been multiplied by a test function and integrated, the contributions of the surface tension forces are included as line integrals along the interfaces (3.17). This reduces the singularity of the problem.

We have not yet discussed how to discretize these quantities. What determines the accuracy of the treatment of the discontinuous density and viscosity is how well we evaluate the integrals where they are included. The errors associated with the numerical integration of discontinuous integrands are analyzed in chapter 4. Regularized approximations to discontinuous functions, designed to minimize these errors, are introduced. The implementation of these results into the three interface tracking techniques defined in this chapter, will be discussed in chapter 5.

The evaluation of the line integral along the interfaces including the surface tension forces will also be discussed in chapter 5 for the segment projection, level-set and front-tracking methods. This approach has been chosen for all three methods. In the level-set method, it would be natural to use instead an approximation of the Dirac delta function, since the interfaces are not discretized explicitly. The construction of such approximations, yielding high order of accuracy in the numerical integration, is similar to the construction of approximations for discontinuous functions (chapter 4). The concerns for using this approach in practice for the level-set method is mainly due to the curvature included in the integral. This is more carefully discussed in section 6.3.
Chapter 4

Numerical Integration of Discontinuous and Singular Functions

4.1 Introduction

The equations governing multiphase flow contain two discontinuous physical variables, the density and the viscosity, as well as singular surface tension forces. This poses extra difficulties when solving these equations. In discretizations based on the differential form of the equations, discontinuities are most commonly smoothed out over a few grid cells, and Dirac delta functions are replaced by mollified approximations, thereby introducing approximation errors to the discretization [43, 44]. We work with a finite element discretization of the equations, and we therefore need to study the weak form of the equations. This formulation includes the integration of functions discontinuous across a curve $\gamma$ in $\Omega \subset \mathbb{R}^2$ and of singular functions with support on $\gamma$.

As in (2.1), we introduce a curve $\gamma$, bounding a region $\Omega_A$, that may be multiply connected. The region $\Omega_A$ is contained in the computational domain $\Omega \subset \mathbb{R}^2$. In general, $\gamma$ will consist of several separate interfaces, i.e.

$$\gamma = \bigcup_j \gamma_j.$$  \hfill (4.1)

Each of the interfaces $\gamma_j$ encloses a region of $\Omega_A$, possibly together with parts of $\partial \Omega$, the boundary of $\Omega$. Define the characteristic function

$$I(x) = \begin{cases} 
1 & \text{for } x \in \Omega_A, \\
0 & \text{for } x \notin \Omega_A.
\end{cases} \hfill (4.2)$$
For our multiphase flow application, $\Omega_A$ is the region occupied by fluid $A$. The weak form of the Navier-Stokes equations (3.11)-(3.12) includes integrals containing the discontinuous density $\rho(x) = \rho_B + I(x)(\rho_A - \rho_B)$ or the discontinuous viscosity $\mu(x) = \mu_B + I(x)(\mu_A - \mu_B)$, with the characteristic function $I(x)$ defined as being 1 in fluid $A$ and 0 in fluid $B$. In general, we want to evaluate

$$\int_{\Omega} \tilde{F}(x) \, dx,$$

(4.3)

where $\tilde{F}(x)$ is a general function that is discontinuous across $\gamma$. This function can be written as

$$\tilde{F}(x) = \begin{cases} f_1(x) & x \in \Omega_A, \\ f_2(x) & x \notin \Omega_A, \end{cases}$$

(4.4)

or, equivalently

$$\tilde{F}(x) = f_1(x) + I(x)(f_2(x) - f_1(x)),$$

(4.5)

using the characteristic function $I(x)$ (4.2). The functions $f_1(x)$ and $f_2(x)$ are assumed to be smooth.

The weak form of the equations also includes the integral over the surface tension forces (3.17), which is of the type

$$\int_{\Omega} f(x) \delta_\gamma \, dx = \int_{\gamma} f(x) \, d\gamma.$$

(4.6)

Since we work with a weak form of the equations, it is not directly the pointwise values of the discontinuous and singular functions that are important, but rather how accurate the integrals including these quantities are evaluated.

The objective of the study in this chapter is to evaluate different methods for the calculation of the type of integrals given above. Generally speaking, we want to analyze the numerical integration of functions discontinuous across curves in $\mathbb{R}^2$ and of singular functions with support on curves in $\mathbb{R}^2$.

Two main approaches for the evaluation of integrals of discontinuous functions are considered. These are: i) keep the characteristic function discontinuous and modify the quadrature procedure, and ii) replace the characteristic function by a more regular approximation.

For the integration of singular functions with support on curves, as formalized by the Dirac delta function $\delta_\gamma$, it is natural to evaluate the line integral that is obtained, instead of explicitly discretizing the delta function. However, if one wants to evaluate integrals containing the Dirac delta functions with the same procedure as is used to evaluate other integrals over the domain, one can replace the delta function by an approximation. The analysis of this procedure is similar to the analysis of approach ii) above. For quadrature rules and smooth approximations to delta functions in the context of vortex methods, see [5, 39].
4.1. Introduction

We perform our integration on a triangulated two-dimensional domain, and in order to easily vary the size of the quadrature triangles close to the discontinuities, adaptive quadrature (to be described in section 4.1.2) has been used.

To define the position of $\gamma$ in (4.1), we use a signed distance function $d(x)$. Denote by $\tilde{d}(x)$ the closest distance from $x$ to any interface $\gamma_i$, $\tilde{d}(x) \geq 0$. The signed distance function $d(x)$ is defined as

$$d(x) = \begin{cases} 
\tilde{d}(x) & \text{if } x \in \Omega_A, \\
-\tilde{d}(x) & \text{if } x \notin \Omega_A.
\end{cases} \quad (4.7)$$

The zero level set of $d(x)$ defines $\gamma$. The characteristic function $I(x)$ (4.2), can be written as $I(x) = H(d(x))$, where $H(t)$ is the Heaviside function,

$$H(t) = \begin{cases} 
0 & \text{for } t < 0, \\
1/2 & \text{for } t = 0, \\
1 & \text{for } t > 0. 
\end{cases} \quad (4.8)$$

Using the signed distance function $d(x)$, the discontinuous function given in (4.5) can be written as

$$F(x) = f_1(x) + H(d(x)) (f_2(x) - f_1(x)). \quad (4.9)$$

For the Dirac delta function, localized to $\gamma$, we can write $\delta_\gamma = \delta(d(x))$. Any function with this type of singularity can be written $f(x) \delta(d(x))$.

The first approach for integration of a discontinuous function, where the characteristic function is kept discontinuous and the quadrature procedure is altered, is studied in section 4.2. The predicted order of accuracy for two different quadrature rules are obtained in numerical tests.

In section 4.3, we study the second approach, where $H(d(x))$ is replaced by an approximation with more regularity. A formula for the error made when replacing the characteristic function by an approximation is derived, and simple conditions on the approximation that reduce this error are deduced. Moreover, we show that the order of accuracy for the quadrature of this approximative characteristic function depends on the regularity of the function, i.e. the number of continuous derivatives. In this framework we also study the approximation of $\delta(d(x))$, and the same type of results are obtained. Numerical results for different approximations are presented in section 4.4.

4.1.1 Preliminaries

For the integration of a function $F(x)$ over a triangulated domain, the integral is simply written as a sum of integrals over each triangle. Denote the approximation to $\int_\Omega F(x) \, dx$ by quad$(F(x))$, i.e.

$$\text{quad}(F(x)) = \sum_\epsilon \sum_\delta w_\epsilon^\delta \, F(\eta_\delta^\epsilon), \quad (4.10)$$
where $\eta^e_i$ and $w^e_i$ are the quadrature points and weights for element $e$, and we sum over all elements and quadrature points. In practice, all triangles are mapped to a reference triangle, where the calculations are performed.

The basic quadrature formula that we use (points and weights are given in Appendix A) is designed to integrate polynomials up to degree 12 exactly. The error made by the quadrature rule will depend on how well $F(x)$ can be approximated by a polynomial of this degree. Using this quadrature rule, the quadrature errors for a general smooth function is kept small. Discontinuous and singular functions are however not well approximated by polynomials and the quadrature error will be large if special care is not taken.

The first part in the expression of the discontinuous function in (4.9) is smooth. Therefore, what we are interested in analyzing is the integration of the second part, $H(d(x))(f_2(x) - f_1(x))$. With $G(x) = f_2(x) - f_1(x)$, this part can be written as

$$F_1(x) = H(d(x))G(x),$$

where $G(x)$ is a smooth function.

In terms of singular functions, we want to study the integration of functions of the form

$$F_2(x) = \delta(d(x))G(x),$$

where $G(x)$ is again a smooth function.

### 4.1.2 Numerical Test Case

The numerical tests have been performed on a regularly subdivided mesh on $[-1,1] \times [-1,1]$, with interval size $\Delta e$. The triangles in the mesh will be isosceles right triangles with the longest side $\sqrt{2}\Delta e$. Since we work with quadratic basis functions, the closest distance between two nodes in the mesh will be $\Delta x = \Delta e / 2$.

In order to easily vary the size of the quadrature triangles, we have used adaptive quadrature, i.e. a local refinement of triangles to which the quadrature rule is applied. At each level of the adaptive procedure, selected elements are split into four sub elements. Denote the (user defined) depth of the adaptive quadrature, i.e. how many splits into sub-triangles that are made, by $l$. Then the triangles at this bottom level are of size $h_{l \perp} = \Delta e / 2^l$. The length of the longest side of a triangle is denoted by $h$, and so $h = \sqrt{2}h_{l \perp}$. The elements selected for refinement in the quadrature procedure are typically elements in some neighbourhood of $\gamma$.

The curve $\gamma$ is initialized as a circle with radius $a$ centered in $(x_c, y_c)$, and the distance function $d(x)$ is defined by

$$d(x) = d(x, y) = a - \sqrt{(x - x_c)^2 + (y - y_c)^2}.$$  

(4.13)

Summarizing the different measures of the element size that we will use, we have $h$ as the length of the longest side of the triangle. In the special case
of a right isosceles triangle, we also use the length of the two shorter sides \( h_\perp = h/\sqrt{2} \). These two measures are the size of a triangle over which the quadrature is performed, and they are reduced by adaptivity. This is not the case for \( \Delta e \) and \( \Delta x \), introduced above, which are parameters of the fixed mesh.

### 4.2 Integration with Modified Quadrature

#### 4.2.1 Integration of the Characteristic Function \( H(d(x)) \)

First, we consider the integration of a piecewise constant function, i.e. let \( G(x) \equiv 1 \) in (4.11), so that \( F_1(x) = H(d(x)) \). Integration of the characteristic function \( H(d(x)) \) over the domain yields the area enclosed by the zero contour of the distance function \( d(x) \), i.e. the area enclosed by \( \gamma \).

In an uncut element, the value of the integral will simply be the value of \( H(d(x)) \) (constant over the element) times the area of the element. For adaptive quadrature, the same holds for uncut sub-triangles. The only sub-triangles (or elements if there is no adaptivity) that will need some special care is therefore the sub-triangles on the bottom level that are still intersected by the zero contour.

The simplest rule is to use a one-point rule for these triangles, i.e. to take the value in this one point and multiply it with the area of the triangle. The local error for one triangle \( T \) is then

\[
E_M = \left| \int_T H(d(x)) \, dx - H(d(x^*)) \right| A_T \leq c_1 A_T + \max |\kappa| \frac{h^3}{12} \tag{4.14}
\]

where \( A_T \) is the area of the triangle, \( \kappa \) is the curvature of the zero contour (which we assume to be bounded), and \( h \) is the longest side of the triangle. Depending on whether we choose \( x_{\text{ref}}^* = (1/4, 1/4) \) or \( x_{\text{ref}}^* = (3/8, 3/8) \) as the evaluation point on the reference triangle, we have \( c_1 = 3/8 \) and \( c_1 = 21/64 \) respectively. The first term is the maximum error made by the one-point rule when a straight line is cutting through the triangle, the second term adds on the error from approximating the zero contour by a straight line. We have \( A_T \leq h^2/2 \), and the local error is of order 2. The number of triangles intersected by the zero level set is proportional to \( 1/h \), hence one would expect a global error of \( O(h) \). However, since the sign of the local error is not monotone, cancellation of errors yields an improvement compared to this estimate.

The next approach is to approximate the part of the zero contour intersecting a triangle by a straight line, and then calculate the area inside the contour. If this is done, only the second term in the error formula (4.14) is retained, and we have

\[
E_L \leq \max |\kappa| \frac{h^3}{12} \tag{4.15}
\]

where \( \max |\kappa| \) is the maximum curvature of the segment of the zero contour intersecting the triangle. With this local error proportional to \( h^3 \), the number
of triangles intersected by the contour being proportional to $1/h$, the predicted 
global error is $O(h^2)$.

In these tests we use the mesh described in section 4.1.2 with $\Delta e = 0.05$ 
and with $d(x)$ as in (4.13) with $a = 0.5$. In order to reduce mesh effects, the 
center of the circle has been shifted $N$ times by small irrational shifts, and it is 
the maximum of the error over all these shifts that is plotted. The error in the 
integral value, i.e., the area of the circle, is plotted versus the number of adaptive 
levels. For each additional level, the size $h$ of the bottom triangle decreases by 
a factor of 2, and so the error should decrease by a factor $2^{p}$, if $p$ is the order of 
the global error.

The result for the one-point rule with the evaluation point $\mathbf{x}^*_{eval} = (3/8, 3/8)$ 
is shown in Figure 4.1a. The dashed line indicates a decay of $(1/2^t)^{1.5}$, and we 
find the order of the global error for this method to be close to 1.5. The 
random distribution of the local errors does really improve the order of the global 
error. To reduce mesh effects, the error plotted in Figure 4.1a is the maximum 
error taken over $N = 144$ shifts of the position $(x_c, y_c)$. In Figure 4.1b the same

![Figure 4.1.](image)

(a) Evaluation by one-point rule. $\mathbf{x}^*_{eval} = (3/8, 3/8)$, $N = 144$. Dashed line indicates the decay rate $(1/2^t)^{1.5}$. 

(b) Evaluation by the “approximation by line” approach, $N = 16$. Dashed line indicates the decay rate $(1/2^t)^2$.

plot is shown for the “approximation by line” approach. Here, the global error 
is of order 2. No improvement compared to the predicted order is achieved here 
since the error in this case is additive – in this special case of a circle, the area 
fraction of the circle will be underestimated in each cut triangle of the mesh.
4.2.2 Integration of the Product $H(d(\mathbf{x})) \ G(\mathbf{x})$

To apply the adaptive quadrature procedure to the product $G(\mathbf{x})H(d(\mathbf{x}))$ instead of to $H(d(\mathbf{x}))$ alone, only the treatment at the bottom triangles need to be modified. For all other elements and sub-triangles, the 13-point quadrature rule is used.

Inside a triangle at the bottom adaptive level that is still intersected by the interface, the intersecting zero contour is approximated by a straight line, cutting two of the edges of the triangle. The case of the zero contour curving in and cutting only one edge is not considered here, it is assumed that the scale of these sub-triangles is small enough for this not to be a significant source of the error. The triangle is then cut in two parts, one triangle and one quadrilateral. Denote the full triangle by $T$, the cut triangle part by $T_A$, and the rest by $T_B$. Denote the corresponding values of the Heaviside function by $H_A$ and $H_B$ respectively. The integral over $T$ can be written

$$\int_T G(x,y) \ H(d(x,y)) \ dx = H_A \int_{T_A} G(x,y) \ dx + H_B \int_{T_B} G(x,y) \ dx$$

$$= (H_A - H_B) \int_{T_A} G(x,y) \ dx + H_B \int_T G(x,y) \ dx. \quad (4.16)$$

Writing the integral in this way, we are able to perform the calculations only on triangles, and $T_A$ can be mapped to the reference element as can any other triangle.

The approximation of the curve by a straight line again yields an $O(h^3)$ local error, although with an error constant depending on $G(\mathbf{x})$. The number of triangles intersected by the zero contour is proportional to $1/h$, so the global error is $O(h^2)$.

![Graph showing the error in integration of $G(x,y)H(d(x,y))$ versus the number of adaptive levels ($l$). $G(x,y) = (x-x_c)^2 + (y-y_c)^2$. $\Delta x = 0.05$, $h = \Delta x^2$. The dashed line indicates the decay $(1/4)^l$. $N = 16$.](image)

**Figure 4.2.** Error in integration of $G(x,y)H(d(x,y))$ versus the number of adaptive levels ($l$). $G(x,y) = (x-x_c)^2 + (y-y_c)^2$. $\Delta x = 0.05$, $h = \Delta x^2$. The dashed line indicates the decay $(1/4)^l$. $N = 16$.}
Figure 4.2 shows the error of the integration with \( d(x) \) as in (4.13) with \( a = 0.5 \) and with \( G(x) = G(x, y) = (x - x_c)^2 + (y - y_c)^2 \). The integration of \( G(x) \) over uncut triangles is performed exactly for this \( G(x) \) with the quadrature rule used, and does not contribute to the total error. The error is plotted versus the number of adaptive levels, and should be proportional to \( (1/2^l)^2 = 1/4^l \) where \( l \) is the number of levels. The error shows the predicted slope.

### 4.3 Integration with Regularized Integrands: Error Analysis

Instead of integrating a discontinuous function or a singular function directly, such functions might be replaced by more regular approximations. We shall analyze the numerical integration of functions of the type \( H(d(x))G(x) \) and \( \delta(d(x))G(x) \), where \( G(x) \) is assumed to be smooth.

As an approximation to \( H(d(x)) \), we introduce the function \( H_w(d(x)) \), where \( H_w(t) \) is defined as

\[
H_w(t) = \begin{cases} 
1 & t > w, \\
\nu(t/w) & |t| \leq w, \\
0 & t < -w, 
\end{cases} \tag{4.17}
\]

where \( \nu(\xi) \) is a smooth transition function such that \( \nu(-1) = 0 \) and \( \nu(1) = 1 \).

Similarly, for the delta function \( \delta(d(x)) \), we introduce the approximating function \( \delta_w(d(x)) \), using \( \delta_w(t) \),

\[
\delta_w(t) = \begin{cases} 
\frac{1}{w} \varphi(t/w) & |t| \leq w, \\
0 & |t| > w, 
\end{cases} \tag{4.18}
\]

where \( \varphi(\xi) \) is a smooth function such that \( \varphi(\pm 1) = 0 \) and

\[
\int_{-1}^{1} \varphi(\xi)d\xi = \int_{-w}^{w} \delta_w(t)dt = 1. \tag{4.19}
\]

The requirement (4.19) ensures that \( \delta_w(t) \) has the same mass as \( \delta(t) \).

In general, introducing an approximating function \( f_w(d(x)) \) to the function \( f(d(x)) \), the total error in the integration of the product of \( f_w(d(x)) \) and a smooth function \( G(x) \) is

\[
E_{\text{tot},G}(f_w) = \int_{\Omega} f(d(x)) G(x) \, dx - \text{quad}(f_w(d(x)) G(x)) \\
= (\int_{\Omega} f(d(x)) G(x) \, dx - \int_{\Omega} f_w(d(x)) G(x) \, dx) + (\int_{\Omega} f_w(d(x)) G(x) \, dx - \text{quad}(f_w(d(x)) G(x))) \\
= E_{w,G}(f_w) + E_{\text{quad},G}(f_w). \tag{4.20}
\]
where \( E_{w,G}(f_w) \) is the analytical error made by replacing \( f(d(x)) \) with \( f_w(d(x)) \), and \( E_{\text{quad},G}(f_w) \) is the numerical error made in the integration of \( f_w(d(x)) \). Both \( E_{w,G}(f_w) \) and \( E_{\text{quad},G}(f_w) \) depend on the particular choice of \( f_w(t) \). In general, the numerical error is large for \( w \) small compared to the mesh size, and decreases with increasing \( w \). Except for special situations when there is no analytical error, this error increases with increasing \( w \), i.e. as the transition zone is made larger. In the case when \( G(x) \equiv 1 \), we denote these errors by \( E_{\text{tot}}(f_w) \), \( E_{w}(f_w) \) and \( E_{\text{quad}}(f_w) \), respectively.

We start by discussing the analytical errors \( E_{w,G}(H_w) \) and \( E_{w,G}(\delta_w) \). In section 4.3.5, the numerical errors \( E_{\text{quad}}(H_w) \) and \( E_{\text{quad}}(\delta_w) \) are analyzed.

### 4.3.1 Introduction of Moments

The analytical error made when replacing a function with an approximation to that function is obtained by taking the difference of their integrals. This error is the first part of the total error in (4.20). For \( f(d(x)) \) approximated by \( f_w(d(x)) \), we have the analytical error

\[
E_{w,G}(f_w) = \int_\Omega f(d(x)) G(x) dx - \int_\Omega f_w(d(x)) G(x) dx.
\] (4.21)

Since \( f_w(d(x)) \) is different from \( f(d(x)) \) only in the region where \( |d(x)| \leq w \), the integrals over \( \Omega \) can be replaced by integrals over \( \Omega_w \subset \Omega \), where

\[
\Omega_w = \{ x \in \mathbb{R}^2 : |d(x)| \leq w \}.
\] (4.22)

To perform the integration of a function \( u(x) = u(x, y) \) over \( \Omega_w \), we parameterize this region. Assume that the zero contour of \( d \) can be parameterized by \( \gamma_s = (x(s), y(s)) \), where \( s \in [0, 2\pi] \) and \( q(s) = \sqrt{x'(s)^2 + y'(s)^2} \neq 0 \). The normal to this curve is given by

\[
\mathbf{n} = \frac{1}{q(s)} (-y'(s), x'(s)).
\] (4.23)

Let \( n_x(s) \) and \( n_y(s) \) denote the \( x \) and \( y \) components of this normal vector. The region \( \Omega_w \) can be parameterized by

\[
\Omega_w = \{ x = (x, y) : x = X(s, t), \ y = Y(s, t), \ s \in [0, 2\pi], \ t \in [-w, w] \},
\] (4.24)

where

\[
X(s, t) = x_c + x(s) + tn_x(s),
Y(s, t) = y_c + y(s) + tn_y(s).
\]

In order to integrate an arbitrary function over this domain, we introduce a change of variables to \((s, t)\), so that

\[
\int_{\Omega_w} U(x, y) dx dy = \int_{s=0}^{2\pi} \int_{t=-w}^{w} U(X(s, t), Y(s, t)) q(s) \left| \det J \right| dt ds.
\] (4.25)
The Jacobian determinant for this transformation is given by

\[
\det(J(s, t)) = X_1 Y_1 - X_2 Y_2
\]

\[
= x'(s)n_y(s) - y'(s)n_x(s) + t(n'_x(s)n_y(s) - n'_y(s)n_x(s)),
\]

(4.26)

where the subscripts \( s \) and \( t \) denote partial derivatives. Using the definition (4.23), the first term in (4.26) evaluates as

\[
x'(s)n_y(s) - y'(s)n_x(s) = q(s),
\]

and the second term as

\[
t (n'_x(s)n_y(s) - n'_y(s)n_x(s)) = -t \frac{x'(s)y''(s) - y'(s)x''(s)}{q(s)^2}.
\]

Identifying the curvature

\[
\kappa(s) = \frac{x'(s)y''(s) - y'(s)x''(s)}{q(s)^2},
\]

(4.27)

the factor \( |\det(J(s, t))| \) can be written as

\[
|\det(J(s, t))| = |q(s) (1 - t\kappa(s))|.
\]

This transformation will be non-singular as long as \( |q(s) (1 - t\kappa(s))| \neq 0 \), i.e. \( 1 - t\kappa(s) \neq 0 \), since we have already assumed \( q(s) \neq 0 \). Therefore, we need to require

\[
w \max_s |\kappa(s)| < 1.
\]

(4.28)

The integral (4.25) can be written

\[
\int_{\Omega_w} U(x, y) \, dx \, dy = \int_{s=0}^{2\pi} \int_{t=-w}^{w} U(X(s, t), Y(s, t)) \, q(s) (1 - t\kappa(s)) \, dt \, ds.
\]

(4.29)

Assume that the function to be integrated is of the form \( U(x) = f(d(x)) \, G(x) \).

In order to write this function as a function \( u(s, t) = U(X(s, t), Y(s, t)) \), we note that \( d(X(s, t), Y(s, t)) = t \), and therefore \( f(d(x)) = f(t) \). Denoting \( g(s, t) = G(X(s, t), Y(s, t)) \), we can write

\[
U(X(s, t), Y(s, t)) = u(s, t) = f(t) \, g(s, t).
\]

(4.30)

The integration formula (4.29) yields the integral over this function as

\[
\int_{\Omega_w} f(t) \, g(s, t) \, dt = \int_{s=0}^{2\pi} \int_{t=-w}^{w} f(t) \, g(s, t) \, q(s) (1 - t\kappa(s)) \, dt \, ds.
\]

(4.31)
4.3. Integration with Regularized Integrand: Error Analysis

Since \( t \in [-w, w] \), \( w \) small, we can expand \( g(s, t) \) in a Taylor series for \( t \), centered in \((s, 0)\), i.e

\[
g(s, t) = \sum_{i=0}^{\infty} \frac{t^i}{i!} g_t(s, 0),
\]

(4.32)

where the index \( i \) in \( g_t \) denotes the number of partial derivatives with respect to \( t \). We have here assumed for convenience that \( g(s, t) = G(X(s, t), Y(s, t)) = G(x, y) \) is analytic. A truncated series with a remainder term could also be used. Inserted into (4.31), this yields

\[
ing_{\alpha}(f(t) g(s, t)) = \int_0^{2\pi} q(s) g(s, 0) \, ds \int_{-w}^{w} f(t) \, dt + \sum_{\alpha=1}^{\infty} C_{\alpha,G} \int_{-w}^{w} f(t) \, t^{\alpha} \, dt,
\]

(4.33)

where

\[
C_{\alpha,G} = \frac{1}{\alpha!} \int_0^{2\pi} q(s) g_{\alpha t}(s, 0) \, ds - \frac{1}{(\alpha - 1)!} \int_0^{2\pi} q(s) \kappa(s) g_{(\alpha-1)t}(s, 0) \, ds.
\]

(4.34)

\( C_{\alpha,G} \) will be bounded, as long as all the partial derivatives of \( g(s, t) \) stay bounded. The requirements made earlier guarantee that \( q(s) \) and \( \kappa(s) \) are bounded. At this point, we introduce what we henceforth call the moments of a function \( f(t) \),

\[
M_{\alpha}(f(t)) = \int_{-w}^{w} f(t) \, t^{\alpha} \, dt.
\]

(4.35)

Using this notation, (4.33) can be written as

\[
ing_{\alpha}(f(t) g(s, t)) = M_{\alpha}(f(t)) \int_0^{2\pi} q(s) g(s, 0) \, ds + \sum_{\alpha=1}^{\infty} C_{\alpha,G} M_{\alpha}(f(t)),
\]

(4.36)

where again \( g(s, t) = G(X(s, t), Y(s, t)) \). This formula will be used to derive the analytical error for the approximations \( H_w(d(x)) \) and \( \delta_w(d(x)) \), where the moments of \( E(t) = H(t) - H_w(t) \) and \( \delta_w(t) \) are important. For another application of moment conditions, see [6].
4.3.2 Analytical Error for Discontinuous Functions

The analytical error in the integration of $H(d(x))G(x)$ that is made when replacing the Heaviside function $H(t)$ by an approximation $H_w(t)$, is given by

$$E_{w,G}(H_w) = \int_{\Omega} (H(d(x)) - H_w(d(x))) G(x) \, dx.$$  \hspace{1cm} (4.37)

With $H_w(t)$ as defined in (4.17), the domain $\Omega$ can be replaced by $\Omega_w$, as given in (4.22). The parameterization of $\Omega_w$ is given in (4.24).

Transforming to the variables $(s, t)$, again using that $d(X(s, t), Y(s, t)) = t$, and therefore $H(d(x)) = H(t)$, we introduce the error function

$$E(t) = H(t) - H_w(t),$$  \hspace{1cm} (4.38)

and $g(s, t) = G(X(s, t), Y(s, t))$, so that the function to be integrated can be written

$$H(d(x))G(x) = (H(t) - H_w(t)) G(X(s, t), Y(s, t)) = E(t) g(s, t),$$  \hspace{1cm} (4.39)

in correspondence to (4.30).

Using the notion of moments of the error function $M_{\nu}(E(t))$ as defined in (4.35), the integral formula (4.36), were $g(s, t)$ has been expanded into a Taylor series yields

$$E_{w,G}(H_w) = \int_{\Omega_w} (E(t)) g(s, t))$$
$$= M_0(E(t)) \int_0^{2\pi} q(s) g(s, 0) \, ds + \sum_{\alpha=1}^{\infty} C_{\alpha, G}(E(t)).$$  \hspace{1cm} (4.40)

If $G(x) \equiv 1$, then $C_{\alpha, G} = 0$ for $\alpha > 1$, and we obtain the simplified error formula

$$E_w = \int_{\Omega_w} (E(t)) = M_0(E(t)) \int_0^{2\pi} q(s) \, ds + C_{1,G} M_1(E(t)).$$  \hspace{1cm} (4.41)

The error formula (4.41) reveals that for any choice of $\nu(\xi)$ in (4.17) such that $M_0(E(t)) = 0$ and $M_1(E(t))$ is no analytic error will be introduced if $G(x)$ is equal to a constant.

The moments of the error function $E(t)$ evaluates as

$$M_{\nu}(E(t)) = \int_{-w}^{w} E(t) t^{\nu} \, dt = w^{\nu+1} \left\{ \frac{1}{\nu+1} - \int_{-1}^{1} \nu(\xi) \xi^{\nu} \, d\xi \right\},$$  \hspace{1cm} (4.42)

where $\nu(\xi)$ is the transition function in the definition of $H_w(t)$ (4.17). This expression, together with the expression for the error in the general case (4.40), shows that the error will be proportional to higher powers of $w$ the more moments of the error function that vanishes. Since $w$ is small, this is a desirable property.
4.3. Integration with Regularized Integrand: Error Analysis

The equality (4.42) gives the conditions for \( \nu(\xi) \) to yield vanishing moments for the error function of the corresponding Heaviside approximation. Considering that the Heaviside approximation will be integrated numerically, the number of continuous derivatives of the approximation will also be important. The number of continuous derivatives of the Heaviside approximation is simply given by the number of derivatives of \( \nu(\xi) \) that are zero at \( \xi = \pm 1 \).

One \( C^1 \)-approximation of the Heaviside function seen in the literature [43] is \( H_w^a(t) \) defined by

\[
\nu(\xi) = \nu^a(\xi) = \frac{1}{2} \left( 1 + \xi + \frac{1}{\pi} \sin(\pi \xi) \right),
\]

together with (4.17). The integral over the error function \( E^a(t) = H(t) - H_w^a(t) \) with respect to \( t \) evaluates to zero \( (M_0(E^a(t))) = 0 \), and

\[
M_1(E^a(t)) = \int_{t-w}^{t+w} E^a(t) \, dt = \left( \frac{1}{6} - \frac{1}{\pi^2} \right) w^2.
\]

Furthermore, \( M_\alpha(E^a(t)) = 0 \) for \( \alpha \) even, and \( M_3(E^a(t)) = O(w^4) \), which together with (4.40) yields

\[
E_{w,G}(H_w^a) = \left( \frac{1}{6} - \frac{1}{\pi^2} \right) w^2 C_1(s) + O(w^4).
\]

For \( G(x) \equiv 1 \),

\[
E_w(H_w^a) = -\left( \frac{1}{6} - \frac{1}{\pi^2} \right) w^2 \int_0^{2\pi} q(s) \kappa(s) \, ds,
\]

from (4.41). This is a rather low order approximation, yielding an analytical error proportional to \( w^2 \), where \( w \) is the width of the transition zone.

There are different classes of functions from which a function \( \nu(\xi) \) with a certain number of vanishing moments and continuous derivatives could be defined. We will study polynomials, and proceed by introducing the definition of a transition polynomial.

**Definition 4.1.** Denote by \( \nu^{m,k}(\xi) \), the transition polynomial of lowest degree such that

\[
\nu^{m,k}(-1) = 0, \quad \nu^{m,k}(1) = 1,
\]

and

\[
(\nu^{m,k})^{(\beta)}(\pm 1) = 0, \quad \beta = 1, \ldots, k,
\]

and furthermore,

\[
\chi^{m,k}_\alpha = \int_{-1}^{1} \nu^{m,k}(\xi) \xi^\alpha \, d\xi - \frac{1}{\alpha + 1} = 0, \quad \alpha = 0, \ldots, m.
\]
Theorem 4.2. The transition polynomial $\nu^{m,k}(\xi)$ exists and is uniquely determined by the conditions in Definition 4.1. It is of degree $r = 2 \left\lfloor (m + 1)/2 \right\rfloor + 2k + 1$. Furthermore, $\nu^{m,k}(\xi) = 1/2 + p(\xi)$, where $p(\xi)$ is a polynomial of degree $r$, containing only odd powers of $\xi$.

Proof Let $\nu(\xi)$ be a polynomial whose derivative satisfies

$$
\nu'(\xi) = c (1 - \xi)^k (1 + \xi)^k Q(\xi) = c (1 - \xi^2)^k Q(\xi),
$$

(4.50)

where $Q$ is a polynomial and $c$ is a constant. This a necessary and sufficient condition for $\nu$ to obey (4.48). Integrating partially using (4.47), we have

$$
\int_{-1}^{1} \nu(\xi) \xi^\alpha d\xi = \left[ \frac{\nu(\xi)\xi^{\alpha+1}}{\alpha+1} \right]_{-1}^{1} - \int_{-1}^{1} \frac{\nu'(\xi)\xi^{\alpha+1}}{\alpha+1} d\xi
$$

$$
= \frac{1}{\alpha+1} - \frac{1}{\alpha+1} \int_{-1}^{1} \nu'(\xi)\xi^{\alpha+1} d\xi,
$$

and this, together with (4.49) yields

$$
\int_{-1}^{1} \nu'(\xi)\xi^{\alpha+1} d\xi = c \int_{-1}^{1} (1 - \xi^2)^k Q(\xi)\xi^{\alpha+1} d\xi = 0, \quad \alpha = 0, \ldots, m, \quad (4.51)
$$

where we have replaced $\nu'(\xi)$ using (4.50). Define an inner product with the positive weight function $(1 - \xi^2)^k$,

$$
\langle f, g \rangle_k = \int_{-1}^{1} (1 - \xi^2)^k f(\xi) g(\xi) \, d\xi, \quad ||f||_k = \langle f, f \rangle_k^{1/2}.
$$

Using this notation, (4.51) yields

$$
\langle Q, \xi^{\alpha+1} \rangle_k = 0, \quad \alpha = 0, \ldots, m
$$

and therefore

$$
Q \perp R_{m+1} = \text{span} \{\xi, \xi^2, \ldots, \xi^{m+1}\},
$$

with respect to this inner product. Note that

$$
\langle \xi^a, \xi^b \rangle_k = 0, \quad \text{when } a + b \text{ is odd.} \quad (4.52)
$$

Let $P_{m+1} = \text{span} \{1, \xi, \xi^2, \ldots, \xi^{m+1}\}$ and note that $R_{m+1} \subset P_{m+1}$. There exists a subspace $R_{m+1}^\perp$, orthogonal to $R_{m+1}$, such that $P_{m+1} = R_{m+1} \oplus R_{m+1}^\perp$. The polynomial $Q$ must be in $R_{m+1}^\perp$, a space of dimension 1, and is therefore defined up to a multiplicative constant. This is the $Q$ of the lowest degree possible, since any other space orthogonal to $R_{m+1}$ contains powers of $\xi$ higher than $m + 1$. 
Finally, to satisfy (4.47), we must have
\[ \int_{-1}^{1} \nu'(\xi) d\xi = c \int_{-1}^{1} (1 - \xi^2)^k Q(\xi) d\xi = 1, \]
and so,
\[ c = \frac{1}{\int_{-1}^{1} (1 - \xi^2)^k Q(\xi) d\xi} = \frac{1}{\langle Q, 1 \rangle_k}. \]
This is well defined since \( 1 \notin R_{m+1} \) and therefore \( \langle Q, 1 \rangle_k \neq 0 \). The product \( c \, Q = Q / \langle Q, 1 \rangle_k \) is hereby uniquely determined. Hence, the desired \( \nu \) exists and is uniquely given by
\[ \nu(\xi) = \int_{-1}^{\xi} \nu'(\eta) d\eta = \frac{1}{\langle Q, 1 \rangle_k} \int_{-1}^{\xi} (1 - \eta^2)^k Q(\eta) d\eta. \tag{4.53} \]

Now, we need to determine the degree of \( Q \). Using the Gram-Schmidt orthogonalization procedure, we can construct an orthonormal basis \( \{ r_j \}_{j=1}^{m+1} \) for \( R_{m+1} \). We define \( r_1 = \xi / \| \xi \|_k \), and proceed by defining
\[ v_{j+1} = \xi^{j+1} - \sum_{i=1}^{j} \langle \xi^{j+1}, r_i \rangle_k r_i, \quad r_{j+1} = \frac{v_{j+1}}{\| v_{j+1} \|_k}, \quad j = 1, \ldots, m. \]
This construction yields that

i) The degree of \( r_j \) is \( j \).

ii) Because of (4.52), the basis functions \( \{ r_{2j+1} \}_{j=0}^{m/2} \) contain only odd powers of \( \xi \), while the basis functions \( \{ r_{2j} \}_{j=1}^{(m+1)/2} \) contain only even powers of \( \xi \).

\( Q \in R_{m+1}^* \) is given by
\[ Q(\xi) = 1 - \sum_{j=1}^{m+1} \langle 1, r_j \rangle_k r_j. \tag{4.54} \]
Splitting the sum over the basis functions in odd and even terms, and noting that \( \langle 1, r_{2j+1} \rangle_k = 0, \forall j = 1, \ldots, \lfloor \frac{m}{2} \rfloor \) due to ii) and (4.52), this yields
\[ Q(\xi) = 1 - \sum_{j=1}^{(m+1)/2} \langle 1, r_{2j} \rangle_k r_{2j}, \]
so \( Q \) is at most of degree \( 2n \), \( n = \lfloor \frac{m+1}{2} \rfloor \). We want to show that the degree of \( Q \neq 0 \) is equal to \( 2n \). Assume \( Q \) is of degree \( \leq 2(n - 1) \). In that case, \( \xi^2 Q \in R_{m+1} \), and therefore we must have \( Q \perp \xi^2 Q \). Since we have
\[ \langle Q, \xi^2 Q \rangle_k = \langle Q, \xi Q \rangle_k = \| \xi Q \|_k^2, \]
this yields \( \|Q\|_k = 0 \), and therefore \( Q = 0 \), a contradiction. This yields that \( \nu'(\xi) \), as defined in (4.50), is of degree \( 2k + 2 \left[ \frac{m+1}{2} \right] \). The degree of \( \nu(\xi) \) is simply one higher than the degree of \( \nu'(\xi) \), i.e. \( r = 2k + 2 \left[ \frac{m+1}{2} \right] + 1 \).

Denote by \( \zeta(\xi) \) the primitive function of \( (1 - \xi^2)^k Q(\xi) \). Using that \( \zeta(\xi) \) is an odd function, we have

\[
\nu(\xi) = \int_{-1}^{\xi} (1 - \eta^2)^k Q(\eta) \, d\eta = \left[ \frac{1}{2} \xi(\xi - 1) + \frac{1}{2} \frac{\zeta(\xi)}{\zeta(1)} \right] = \frac{\zeta(1)}{2} + \frac{\zeta(\xi)}{2\xi(1)} = \frac{\zeta(1)}{2} + \frac{\zeta(\xi)}{2\xi(1)}
\]

The second part of the theorem holds with \( p(\xi) = \frac{\zeta(\xi)}{2 \xi(1)} \).

**Remark 4.3.** In addition to (4.49), the quantity \( \lambda_{\alpha}^{m,k} = 0 \) for all \( \alpha \) even, since \( \nu_{\alpha}^{m,k}(\xi) = 1/2 + p(\xi) \), where \( p(\xi) \) is an odd polynomial. This yields that the transition polynomial \( \nu_{\alpha-1}^{m,k}(\xi) \), \( n \) even, is equal to \( \nu_{\alpha}^{m,k}(\xi) \).

To each transition polynomial \( \nu_{\alpha}^{m,k}(\xi) \), we assign a Heaviside approximation \( H_{\alpha}^{m,k}(t) \), and a corresponding error function \( E_{\alpha}^{m,k}(t) \). We have the following definitions:

**Definition 4.4.** Denote by \( H_{\alpha}^{m,k}(t) \), the Heaviside approximation defined by

\[
H_{\alpha}^{m,k}(t) = \begin{cases} 
1 & t > w, \\
\nu_{\alpha}^{m,k}(t/w) & |t| \leq w, \\
0 & t < -w,
\end{cases} \tag{4.55}
\]

where \( \nu_{\alpha}^{m,k}(\xi) \) was defined in Definition 4.1. Define the corresponding error function \( E_{\alpha}^{m,k}(t) \) by

\[
E_{\alpha}^{m,k}(t) = H(t) - H_{\alpha}^{m,k}(t). \tag{4.56}
\]

Using the definition of \( \nu_{\alpha}^{m,k}(\xi) \), we can show the following:

**Corollary 4.5.** Let \( H_{\alpha}^{m,k}(t) \) and \( E_{\alpha}^{m,k}(t) \) be as in Definition 4.4. Then it follows that the Heaviside approximation \( H_{\alpha}^{m,k}(t) \) has \( k \) continuous derivatives, and

\[
M_{\alpha}(E_{\alpha}^{m,k}(t)) = \int_{-w}^{w} E_{\alpha}^{m,k}(t) t^\alpha \, dt = 0, \quad \alpha = 0, \ldots, 2 \left[ \frac{m+1}{2} \right]. \tag{4.57}
\]

In addition, \( M_{\alpha}(E_{\alpha}^{m,k}(t)) = 0 \) for all \( \alpha \) even.

**Proof** The number of continuous derivatives is simply given by the number of vanishing derivatives of \( \nu_{\alpha}^{m,k}(\xi) \) at \( \xi = \pm 1 \), which is \( k \) by definition. Moreover,

\[
M_{\alpha}(E_{\alpha}^{m,k}(t)) = \int_{-w}^{w} E_{\alpha}^{m,k}(t) t^\alpha \, dt
\]

\[
= w^{\alpha+1} \left\{ \frac{1}{\alpha+1} - \int_{-1}^{1} \nu(\xi) \xi^\alpha d\xi \right\} = -\lambda_{\alpha}^{m,k} w^{\alpha+1}.
\]
4.3. Integration with Regularized Integrands: Error Analysis

From the definition of \( \nu^{m,k}(\xi) \), the quantity \( \lambda^{m,k}_0 = 0 \) for \( 0 \leq \alpha \leq m \), and (4.57) follows for \( 0 \leq \alpha \leq m \). In addition, \( \lambda^{m,k}_\beta = 0 \) for \( \beta = 2 \left\{ (m+1)/2 \right\} \), since \( \beta \) is even (Remark 4.3).

We now have all the necessary tools to state and show the following theorem concerning the analytical error for these polynomial approximations:

**Theorem 4.6.** Let \( H^{m,k}_w(t) \) and \( E^{m,k}_w(t) \) be as in Definition 4.4. Assume that \( \gamma \), the zero level set of \( d(x) \), can be parameterized by \( \gamma = (x(s), y(s)) \), with the curvature \( \kappa(s) \) defined by (4.27). If \( (w \max \{ |\kappa(s)| \} < 1 \), then the analytical error for the integration of \( H(d(x)) G(x) \) made when replacing \( H(d(x)) \) by \( H^{m,k}_w(d(x)) \),

\[
E_{w,G}(H^{m,k}_w) = \int_\Omega \{ H(d(x)) - H^{m,k}_w(d(x)) \} G(x) \, dx, \tag{4.58}
\]

is given by

\[
E_{w,G}(H^{m,k}_w) = \sum_{\alpha=\beta/2}^{\infty} C_{2\alpha+1,G} M_{2\alpha+1}(E^{m,k}_w(t))
\]

\[
= -C_{\beta+1,G} \lambda^{m,k}_{\beta+1} w^{\beta+2} + O(w^{\beta+4}), \quad \beta = 2 \left\{ (m+1)/2 \right\}, \tag{4.59}
\]

with \( C_{\beta,G} \) defined by (4.34) and \( \lambda^{m,k}_\beta \) defined by (4.49).

**Proof** The general error formula is given in (4.40). According to Corollary 4.5, the moments \( M_\alpha(E^{m,k}_w(t)) = 0 \) for \( \alpha = 0, \ldots, \beta \) and all \( \alpha \) even. The terms retained are therefore those in (4.59), and the first non-vanishing moment is

\[
M_{\beta+1}(E^{m,k}_w(t)) = -\lambda^{m,k}_{\beta+1} w^{\beta+2},
\]

with the second non-vanishing moment \( M_{\beta+3}(E^{m,k}_w(t)) \sim w^{\beta+4} \).

\( \square \)

**Remark 4.7.** Theorem 4.6 can be generalized. It refers to the approximation \( H^{m,k}_w(t) \), which is defined using the transition polynomial \( \nu^{m,k}(\xi) \), as defined in Definition 4.1. However, the theorem holds also for approximations based on transition functions \( \nu(\xi) \) from other function classes, that fulfill the same conditions as \( \nu^{m,k}(\xi) \).

**Remark 4.8.** If \( G(x,y) = G(X(s,t),Y(s,t)) = g(s,t) \) is such that \( g(s,t) \) is a polynomial in \( t \) of degree \( n \) for all fixed \( s \), then \( C_{w,G} = 0 \) for \( \alpha \geq n + 2 \). For \( m \) such that \( \beta = 2 \left\{ (m+1)/2 \right\} \geq n + 1 \), it follows that \( E_{w,G}(H^{m,k}_w) = 0 \).
The Heaviside approximations $H^2_k(t)$, with two vanishing moments and $k = 0, 1, 2$ continuous derivatives, are defined using the following transition polynomials

$$
\nu^{2,0}_0(\xi) = \frac{1}{2} + \frac{1}{8}(9\xi - 5\xi^3),
$$

(4.60)

$$
\nu^{2,1}_0(\xi) = \frac{1}{2} + \frac{1}{32}(45\xi - 50\xi^3 + 21\xi^5),
$$

(4.61)

$$
\nu^{2,2}_0(\xi) = \frac{1}{2} + \frac{1}{64}(105\xi - 175\xi^3 + 147\xi^5 - 45\xi^7).
$$

(4.62)

These approximations all have two vanishing moments. The regularity of the functions are important for the numerical error. Comparing the above approximations, $H^2_0(t)$ will have the best numerical properties. If more vanishing moments ($m = 4$) are required, the approximations $H^4_k(t)$, $k = 0, 1, 2$ are defined using the following transition polynomials

$$
\nu^{4,0}_0(\xi) = \frac{1}{2} + \frac{1}{128}(225\xi - 350\xi^3 + 180\xi^5),
$$

(4.63)

$$
\nu^{4,1}_0(\xi) = \frac{1}{2} + \frac{1}{256}(525\xi - 1225\xi^3 + 1323\xi^5 - 495\xi^7),
$$

(4.64)

$$
\nu^{4,2}_0(\xi) = \frac{1}{2} + \frac{1}{2048}(4725\xi - 14700\xi^3 + 23814\xi^5 - 17820\xi^7 + 5005\xi^9).
$$

(4.65)

A plot of these two sets of transition polynomials is shown in Figure 4.3. Note that the transition polynomials are not monotone.

![Plot of transition functions](image1.png)

(a) Transition functions $\nu^{2,k}_0(\xi)$, $k = 0, 1, 2$.  
(b) Transition functions $\nu^{4,k}_0(\xi)$, $k = 0, 1, 2$.

**Figure 4.3.** Plot of different approximations of $\nu(\xi)$ versus $\xi$. The polynomials $\nu^{2,k}_0(\xi)$ and $\nu^{4,k}_0(\xi)$ have two and four vanishing moments, respectively, for all number of continuous derivatives $k$. 

---

Note: The above text is a simplified representation of the original content. The equations have been transformed into plain text for better readability. The figures and diagrams are also simplified and not included in the text.
4.3. Integration with Regularized Integrand: Error Analysis

The exact integration of the functions $H_w^{2,k}(t)$ and $H_w^{1,k}(t)$ yield the same result as the integration of $H(t)$, i.e. there is no analytical error for any of these approximations as long as the domain of integration covers the area in which $H_w(d(x))$ is variable. To achieve this, it is necessary to have at least two vanishing moments, i.e. $E_w(H_w^{m,k}) = 0$ if $m \geq 2$.

According to Theorem 4.6, the errors for the three approximations $H_w^{0,k}(t)$, $H_w^{1,k}(t)$ and $H_w^{2,k}(t)$ (defined through (4.60)-(4.62) together with (4.55)), when multiplied by a general smooth function $G(x)$ are

$$E_w, G(H_w^{2,k}) = \frac{1}{3!} M_3(E_w^{2,k}(t)) C_{3,G} + O(w^6) = \frac{1}{3!} \lambda_3^{2,k} C_{3,G} w^4 + O(w^6),$$

$$E_w, G(H_w^{1,k}) = \frac{1}{5!} M_5(E_w^{1,k}(t)) C_{5,G} + O(w^8) = \frac{1}{5!} \lambda_5^{1,k} C_{5,G} w^6 + O(w^8).$$

(4.66)

where $\lambda_3^{2,k}$, $k = 0, 1, 2$, can be computed from (4.49). For the approximations $H_w^{1,k}$, we have

$$E_w, G(H_w^{1,k}) = \frac{1}{3!} M_3(E_w^{1,k}(t)) C_{3,G} + O(w^6) = \frac{1}{3!} \lambda_3^{1,k} C_{3,G} w^4 + O(w^6).$$

The analytical error for these approximations with four vanishing moments will for a general smooth $G(x)$ be of order 6 in $w$, compared to order 4 for the approximations with two vanishing moments ($m = 2$).

4.3.3 Analytical Error for Singular Functions

The analytical error in the integration of $\delta(d(x)) G(x)$ that is made when replacing the Dirac delta function $\delta(t)$ by a smooth approximation $\delta_w(t)$ is given by

$$E_w, G(\delta_w) = \int_\Omega \delta(d(x)) G(x) \, dx - \int_\Omega \delta_w(d(x)) G(x) \, dx. \tag{4.67}$$

With $\delta_w(t)$ as defined in (4.18), the domain $\Omega$ can be replaced by $\Omega_w$, as given in (4.22). The parameterization of $\Omega_w$ is given in (4.24).

Using the integration formula (4.31), and the definition of $\delta(t)$, the integral over the product $\delta(d(x)) G(x) = \delta(t) g(s, t)$ evaluates as

$$I_\Omega (\delta(t) g(s, t)) = \int_{s=0}^{2\pi} \int_{-w}^{t+w} \delta(t) g(s, t) q(s) (1 - t \kappa(s)) \, dt \, ds \tag{4.68}$$

$$= \int_{s=0}^{2\pi} g(s, 0) q(s) ds.$$

This is the line integral around the zero contour of $d(x)$, parameterized by $\gamma_s = (x(s), y(s))$. The scaling factor $q(s)$ is given by $q(s) = \sqrt{x'(s)^2 + y'(s)^2}$. 
In the integration of the product \( \delta_w(d(x))G(x) = \delta_w(t) g(s, t) \), the integration in \( t \) is retained. To evaluate this integral, we use formula (4.36), obtained by the Taylor series expansion of \( g(s, t) \). We have

\[
I_{\Omega_w}(\delta_w(t)g(s, t)) = M_0(\delta_w(t)) \int_0^{2\pi} q(s)g(s, 0) \, ds + \sum_{\alpha=1}^{\infty} C_{\alpha,G} M_\alpha(\delta_w(t)).
\]  

(4.69)

with \( C_{\alpha,G} \) given in (4.34). The moments of \( \delta_w(t) \) are defined as in (4.35). The mass requirement on \( \delta_w(t) \) (4.19) can be written \( M_0(\delta_w(t)) = 1 \). Subtracting (4.69) from (4.68), we obtain

\[
E_{w,G}(\delta_w) = I_{\Omega_w}(\delta(t)g(s, t)) - I_{\Omega_w}(\delta_w(t)g(s, t)) = - \sum_{\alpha=1}^{\infty} C_{\alpha,G} M_\alpha(\delta_w(t)).
\]  

(4.70)

If \( G(x) \equiv 1 \), the factor \( C_{\alpha,G} = 0 \) for \( \alpha > 1 \), and we obtain the simplified formula

\[
E_w(\delta_w) = I_{\Omega_w}(\delta(t)g(s, t)) - I_{\Omega_w}(\delta_w(t)g(s, t))
= -M_1(\delta_w(t)) \int_0^{2\pi} q(s)\kappa(s) \, ds.
\]  

(4.71)

The moments of \( \delta_w(t) \) evaluate as

\[
M_\alpha(\delta_w(t)) = \int_{-w}^{w} \delta_w(t)t^\alpha \, dt = w^\alpha \int_{-1}^{1} \varphi(\xi)\xi^\alpha \, d\xi,
\]  

(4.72)

where \( \varphi(\xi) \) was introduced in the definition of \( \delta_w(t) \) (4.18). This expression, together with the error formula (4.70), yields that the integration error will be proportional to higher powers of \( w \), the more moments of \( \delta_w(t) \) that evaluates as zero. Since \( w \) is small, it is an advantage to have many zero moments. The equality (4.72) gives the conditions on \( \varphi(\xi) \) to yield zero moments of \( \delta_w(t) \).

One approximation of the delta function that can be found in the literature is defining \( \delta_w(t) \) by

\[
\varphi(\xi) = \frac{1}{2}(1 + \cos(\pi \xi)),
\]  

(4.73)

together with (4.18). This approximation was introduced by Peskin in 1977 [32]. It obeys the mass condition (4.19). Moreover, it has one vanishing moment and one continuous derivative.

To construct approximations with more moments equal to zero, and if one so wish, more continuous derivatives, we now choose \( \varphi(\xi) \) as a polynomial. We make the following definition (similarly to Definition 4.1 for the Heaviside approximation).
**Definition 4.9.** Denote by $\varphi^{m,k}(\xi)$, the delta polynomial of lowest degree such that

$$\varphi^{m,k}(\pm 1) = 0, \quad (4.74)$$

and

$$\int_{-1}^{1} \varphi^{m,k}(\xi) d\xi = 1, \quad (4.75)$$

which are the requirements for any function $\varphi(\xi)$, defining an approximation $\delta_\varphi(t)$ through (4.18). In addition, $\varphi^{m,k}(\xi)$ obeys

$$(\varphi^{m,k})^{(\beta)}(\pm 1) = 0, \quad \beta = 1, \ldots, k, \quad (4.76)$$

and furthermore,

$$\gamma^{m,k}_\alpha = \int_{-1}^{1} \varphi^{m,k}(\xi) \xi^\alpha d\xi = 0, \quad \alpha = 1, \ldots, m. \quad (4.77)$$

**Theorem 4.10.** The delta polynomial $\varphi^{m,k}(\xi)$ exists and is uniquely determined by the conditions in Definition 4.9. It is a polynomial of degree $r = 2 \left(\left\lfloor m/2 \right\rfloor + k + 1\right)$, containing only even powers of $\xi$.

**Proof** This proof is made in analogy to the proof of Theorem 4.2. Many details are however different, and the proof is therefore written in its full length.

Let $\varphi(\xi)$ be a polynomial whose derivative is given by

$$\varphi'(\xi) = c(1 - \xi)^k (1 + \xi)^k Q(\xi) = c(1 - \xi^2)^k Q(\xi), \quad (4.78)$$

where $Q$ is a polynomial and $c$ is a constant. This is a necessary and sufficient condition for $\varphi$ to obey (4.76). Integrating partially, we have

$$\int_{-1}^{1} \varphi(\xi) \xi^\alpha d\xi = \left[ \frac{\varphi(\xi) \xi^{\alpha+1}}{\alpha + 1} \right]_{-1}^{1} - \int_{-1}^{1} \frac{\varphi'(\xi) \xi^{\alpha+1}}{\alpha + 1} d\xi \quad (4.79)$$

where the first part evaluates to zero due to (4.74). We again (as in the proof of Theorem 4.2) use the inner product with the positive weight function $(1 - \xi^2)^k$, i.e.

$$\langle f, g \rangle_k = \int_{-1}^{1} (1 - \xi^2)^k f(\xi) g(\xi) d\xi, \quad ||f||_k = \langle f, f \rangle_k^{1/2}. \quad (4.80)$$

For $\alpha > 0$, (4.79) together with (4.77) yields

$$\int_{-1}^{1} \varphi(\xi) \xi^\alpha d\xi = 0, \quad \alpha = 1, \ldots, m,$$
or, equivalently

$$\langle Q, \xi^{\alpha+1}\rangle_k = 0, \quad \alpha = 1, \ldots, m. \tag{4.81}$$

The condition (4.74) yields

$$\varphi(1) = \int_{-1}^{1} \varphi'(\xi) d\xi = c \int_{-1}^{1} (1 - \xi^2)^k Q(\xi) d\xi = c \langle Q, 1 \rangle_k = 0. \tag{4.82}$$

The orthogonality conditions in equations (4.81) and (4.82) give that

$$Q \perp \tilde{R}_{m+1} = \text{span}\{1, \xi, \ldots, \xi^{m+1}\}, \tag{4.83}$$

with respect to the inner product defined in (4.80). Note that

$$\langle \xi^a, \xi^b \rangle_k = 0, \quad \text{when } \alpha + \beta \text{ is odd}. \tag{4.84}$$

Let $P_{m+1} = \text{span}\{1, \xi, \xi^2, \ldots, \xi^{m+1}\}$. Note that $\tilde{R}_{m+1} \subset P_{m+1}$, and hence there exists a non-empty subspace $\tilde{R}_{m+1} \perp \tilde{R}_{m+1}$, orthogonal to $\tilde{R}_{m+1}$, such that $P_{m+1} = \tilde{R}_{m+1} \oplus \tilde{R}_{m+1}$. The polynomial $Q$ must be in $\tilde{R}_{m+1}$, a space of dimension 1, and is therefore defined up to a multiplicative constant. This is the $Q$ of the lowest degree possible, since any other space orthogonal to $\tilde{R}_{m+1}$ contains powers of $\xi$ higher than $m + 1$.

Using (4.79) (with $\alpha = 0$) together with (4.75), we have that

$$\int_{-1}^{1} \varphi'(\xi) \xi \, d\xi = -1. \tag{4.85}$$

This condition determines the coefficient $c$ in the expression for $\varphi'(\xi)$ in (4.78) as

$$c = -\frac{1}{\langle Q, \xi \rangle_k}.$$

so that

$$\varphi'(\xi) = -\frac{1}{\langle Q, \xi \rangle_k} (1 - \xi^2)^k Q(\xi). \tag{4.86}$$

This is well defined since $\xi \notin \tilde{R}_{m+1}$, and therefore $\langle Q, \xi \rangle_k \neq 0$. Hence, the desired $\varphi$ exists and is uniquely given by

$$\varphi(\xi) = -\frac{1}{\langle Q, \xi \rangle_k} \int_{-1}^{\xi} (1 - \eta^2)^k Q(\eta) \, d\eta. \tag{4.87}$$

Now, we need to determine the degree of $Q$. Using the Gram-Schmidt orthogonalization procedure, we can construct an orthonormal basis $\{r_j\}_{j=1}^{m+1}$ for $\tilde{R}_{m+1}$. We define $r_1 = 1/\|1\|_k$, and proceed by defining

$$r_{j+1} = \xi^{j+1} - \sum_{i=1}^{j} \langle \xi^{j+1}, r_i \rangle_k r_i, \quad r_{j+1} = \frac{r_{j+1}}{\|r_{j+1}\|_k}, \quad j = 1, \ldots, m.$$
This construction yields that
\begin{enumerate}  
  \item[i)] \( r_1 = 1 / \|1\|_k \).
  \item[ii)] The degree of \( r_j \) is \( j \geq 1 \).
  \item[iii)] Because of (4.84), the basis functions \( \{ r_{2j+1} \}_{j=1}^{[m/2]} \) contain only odd powers of \( \xi \), while the basis functions \( \{ r_{2j} \}_{j=1}^{[m+1/2]} \) contain only even powers of \( \xi \).
\end{enumerate}

The lowest order \( Q \) that obeys (4.83) is \( Q \in \tilde{R}_{m+1} \) given by
\[
Q(\xi) = \xi - \sum_{j=1}^{m+1} \langle 1, r_j \rangle_k r_j.
\]  
(4.88)

Splitting the sum over the basis functions in odd and even terms, and noting that \( \langle \xi, r_1 \rangle_k = 0 \) and \( \langle \xi, r_{2j} \rangle_k = 0 \), \( \forall j = 1, \ldots, [m+1/2] \) due to \( i \), \( iii \) and (4.84), this yields
\[
Q(\xi) = \xi - \sum_{j=0}^{[m/2]} \langle 1, r_{2j+1} \rangle_k r_{2j+1},
\]  
(4.89)

so \( Q \) is at most of degree \( 2n + 1 \), \( n = [\frac{m}{2}] \). We want to show that the degree of \( Q \neq 0 \) is equal to \( 2n + 1 \). Assume \( Q \) is of degree \( \leq 2n - 1 \). In that case, \( \xi^2 Q \in \tilde{R}_{m+1} \), and therefore we must have \( Q \perp \xi^2 Q \). Since we have
\[
\langle Q, \xi^2 Q \rangle_k = \langle Q, Q \rangle_k = \|\xi Q\|_k^2,
\]
this yields \( \|\xi Q\|_k = 0 \), and therefore \( Q = 0 \), a contradiction.

This yields that \( \varphi(\xi) \), as defined in (4.78), is of degree \( 2k + 2[\frac{m}{2}] + 1 \).
The degree of \( \varphi(\xi) \) is simply one higher than the degree of \( \varphi(\xi) \), i.e. \( r = 2(k + [\frac{m}{2}] + 1) \).

\( \square \)

**Remark 4.11.** In addition to (4.77), the quantity \( \eta^m_k = 0 \) for all \( \alpha \) odd, since \( \varphi^{m,k}(\xi) \) is an even polynomial. This yields that the polynomial \( \varphi^{m-1,k}(\xi) \), \( n \) odd, is equal to \( \varphi^{m,k}(\xi) \).

To each delta polynomial \( \varphi^{m,k}(\xi) \), we assign a delta function approximation \( \delta^{m,k}_w(t) \). We have the following definition:

**Definition 4.12.** Denote by \( \delta^{m,k}_w(t) \), the delta function approximation defined by
\[
\delta^{m,k}_w(t) = \begin{cases}  
  \varphi^{m,k}(t/w) & |t| \leq w, \\
  0 & |t| > w,
\end{cases}
\]  
(4.90)

where the delta polynomial \( \varphi^{m,k}(\xi) \) was defined in Definition 4.9.

Using the definition of \( \varphi^{m,k}(\xi) \), we can show the following corollary.
Corollary 4.13. Let the delta polynomial \( \varphi^{m,k}(t) \) be as in Definition 4.12. Then it follows that the delta function approximation \( \delta^{m,k}_w(t) \) has \( k \) continuous derivatives, and
\[
M_\alpha(\delta^{m,k}_w(t)) = \int_{-w}^{w} \delta^{m,k}_w(t) t^\alpha \, dt = 0, \quad \alpha = 1, \ldots, 2\left\lceil \frac{m}{2} \right\rceil + 1. \tag{4.91}
\]
In addition, \( M_\alpha(\delta^{m,k}_w(t)) = 0 \) for all \( \alpha \) odd.

**Proof** The number of continuous derivatives is simply given by the number of vanishing derivatives of \( \varphi^{m,k}(\xi) \) at \( \xi = \pm 1 \), which is \( k \) by definition. Moreover,
\[
M_\alpha(\delta^{m,k}_w(t)) = \int_{-w}^{w} \delta^{m,k}_w(t) t^\alpha \, dt = w^{\alpha+1} \int_{-1}^{1} \varphi^{m,k}(\xi) \xi^\alpha \, d\xi = \eta^{m,k}_w w^{\alpha+1}.
\]
From the definition of \( \varphi^{m,k}(\xi) \), the quantity \( \eta^{m,k}_w = 0 \) for \( 1 \leq \alpha \leq m \), and (4.91) follows for \( 1 \leq \alpha \leq m \). In addition, \( \eta^{m,k}_w = 0 \) for \( \alpha = 2\left\lceil (m+1)/2 \right\rceil \), since \( \alpha \) is odd (Remark 4.11).

\( \square \)

We now have all the necessary results in order to give the following theorem concerning the analytical error:

**Theorem 4.14.** Let \( \delta^{m,k}_w(t) \) be as in Definition 4.12. Assume that \( \gamma \), the zero level set of \( d(x) \), can be parameterized by \( \gamma = (x(s), y(s)) \), with the curvature \( k(s) \) defined by (4.27). If \( \langle w \max_s |k(s)| \rangle < 1 \), then the analytical error for the integration of \( \delta(d(x)) G(x) \) made when replacing \( \delta(d(x)) \) by \( \delta^{m,k}_w(d(x)) \),
\[
E_{w,G}(\delta^{m,k}_w) = \int_{\Omega} \{ \delta(d(x)) - \delta^{m,k}_w(d(x)) \} G(x) \, dx. \tag{4.92}
\]
is given by
\[
E_{w,G}(\delta^{m,k}_w) = \sum_{\alpha=\beta+2}^{\infty} C_{2\alpha+\beta} M_{2\alpha}(\delta^{m,k}_w(t))
= -C_{\beta,G} \eta^{m,k}_w w^\beta + O(w^{\beta+2}), \quad \beta = 2\left\lceil \frac{m}{2} \right\rceil + 1, \tag{4.93}
\]
with \( C_{\beta,G} \) defined by (4.34) and \( \eta^{m,k}_w \) defined by (4.77).

**Proof** The general error formula is given in (4.70). According to Corollary 4.13, the moments \( M_\alpha(\delta^{m,k}_w(t)) = 0 \) for \( \alpha = 1, \ldots, 2\left\lceil m/2 \right\rceil + 1 \) and all \( \alpha \) odd. The terms retained are therefore those in (4.93), and the first non-vanishing moment is
\[
M_\beta(\delta^{m,k}_w(t)) = \eta^{m,k}_w w^\beta,
\]
with the second non-vanishing moment \( M_{\beta+2}(\delta^{m,k}_w(t)) \sim w^{\beta+2} \).

\( \square \)
4.3. Integration with Regularized Integrands: Error Analysis

**Remark 4.15.** Theorem 4.14 can be generalized. It refers to the approximation $\delta_{w,k}^m(t)$, which is defined using the delta polynomial $\varphi_{m,k}(\xi)$, as defined in Definition 4.1. However, the theorem holds also for approximations based on functions $\varphi(\xi)$ from other function classes, that fulfill the same conditions as $\varphi_{m,k}(\xi)$.

A few examples of polynomials $\varphi_{m,k}(\xi)$ defining approximations $\delta_{w,k}^m(t)$ through (4.90) are given below. The polynomials with one vanishing moment ($m = 1$) and with $k = 0, 1$ and 2 continuous derivatives, respectively, are

\[
\varphi_{1,0} = \frac{3}{4}(1 - \xi^2), \quad (4.94)
\]
\[
\varphi_{1,1} = \frac{15}{16}(1 - 2\xi^2 + \xi^4), \quad (4.95)
\]
\[
\varphi_{1,2} = \frac{35}{32}(1 - 3\xi^2 + 3\xi^4 - \xi^6). \quad (4.96)
\]

The polynomials with three vanishing moments ($m = 3$) and with $k = 0, 1$ and 2 continuous derivatives, respectively, are

\[
\varphi_{3,0} = \frac{15}{32}(3 - 10\xi^2 + 7\xi^4), \quad (4.97)
\]
\[
\varphi_{3,1} = \frac{105}{64}(1 - 5\xi^2 + 7\xi^4 - 3\xi^6), \quad (4.98)
\]
\[
\varphi_{3,2} = \frac{315}{512}(3 - 20\xi^2 + 42\xi^4 - 36\xi^6 + 11\xi^8). \quad (4.99)
\]

These polynomials are plotted in Figure 4.4. Note the difference in the shape of the functions depending on whether they have one or three vanishing moments. In addition, more continuous derivatives will give a function which is more flat in the beginning and the end of the interval, with its mass squeezed more to the middle.

The analytical error for the corresponding delta function approximations are

\[
E_{w,G}(\delta_{w}^{1,k}) = -C_{2,G} \eta_{2}^{1,k} w^2 + O(w^4),
\]

for the approximations with one vanishing moment, and

\[
E_{w,G}(\delta_{w}^{3,k}) = -C_{4,G} \eta_{4}^{3,k} w^4 + O(w^6),
\]

for the approximations with three vanishing moments. The coefficients $C_{n,G}$ and $\eta_{m,k}^{n}$ are defined in (4.34) and (4.77), respectively.

A different approximation of the Dirac delta function is based on definitions in the Fourier space. Introduce the approximation

\[
\delta_F(t) = \frac{1}{w} \varphi_F(t/w),
\]
with \( \varphi_F(\xi) \) defined by

\[
\varphi_F(\xi) = \frac{1}{2\pi} \int_{-1}^{1} \hat{\varphi}_F(\omega) e^{i\omega \xi} d\omega, 
\]

where \( \hat{\varphi}_F(\omega) \in C^\infty(\mathbb{R}) \) has support on \((-1,1)\). The requirement of the mass of the delta function, and similarly on higher moments of the delta function can be expressed as conditions on \( \varphi_F(\omega) \) in Fourier space.

In correspondence to (4.100), \( \varphi_F(\omega) \) can be found from the Fourier transform

\[
\hat{\varphi}_F(\omega) = \int_{-\infty}^{\infty} \varphi_F(\xi) e^{-i\omega \xi} d\xi. 
\]

If we require \( \varphi_F(0) = 1 \), we have

\[
\int_{-\infty}^{\infty} \delta_F(t) dt = \int_{-\infty}^{\infty} \varphi_F(\xi)d\xi = \varphi_F(0) = 1, 
\]

and so the condition that the mass is 1 is fulfilled.

The number of vanishing moments \( M_p(\delta_F(t)) \) determines the power of the analytical error. These moments evaluates as:

\[
M_p(\delta_F(t)) = \int_{-\infty}^{\infty} \delta_F(t) t^p dt = \tilde{\omega}^p \int_{-\infty}^{\infty} \varphi_F(\xi) \xi^p d\xi. 
\]

Since

\[
\frac{\partial}{\partial \omega^p} \varphi_F(\omega)|_{\omega=0} = (-i)^p \int_{-\infty}^{\infty} \varphi_F(\xi) \xi^p d\xi, 
\]
it follows that
\[
\frac{\partial p}{\partial \omega} \hat{\varphi}_F(\omega) |_{\omega=0} = 0 \iff M_p(\delta_F(t)) = 0. \tag{4.102}
\]
It is possible to define functions \( \hat{\varphi}_F(\omega) \) such that (4.102) is true for all \( p \). Therefore, there exist approximations \( \delta_F(t) \), so that all moments are zero, and no analytic error is introduced by replacing the Dirac delta function by such approximations. The approximation \( \delta_F(t) \) however also needs to decay rapidly to be practically useful. Since we have assumed \( \hat{\varphi}_F(\omega) \) to have compact support on \((-1, 1)\), in addition to be flat on top, i.e have all its derivatives zero at \( \omega = 0 \), the decay of the corresponding delta approximations are in general not satisfactory for practical applications.

### 4.3.4 Relation Between Heaviside and Delta Function Approximations

The Dirac delta function \( \delta(t) \) is defined as the derivative of the Heaviside function \( H(t) \). There exist such relations also between the polynomial approximations \( H_w^{m,k}(t) \) and \( \delta_w^{m,k}(t) \) for specific relations between \( m \) and \( n \) and between \( k_1 \) and \( k_2 \). This is formalized in the following theorem:

**Theorem 4.16.** If \( m \geq 0 \) and \( k \geq 1 \), then
\[
\frac{d}{dt} H_w^{m,k}(t) = \delta_w^{m+1,k-1}(t), \tag{4.103}
\]
with \( H_w^{m,k}(t) \) and \( \delta_w^{m+1,k-1}(t) \) defined in Definition 4.4 and Definition 4.12, respectively.

**Proof** Let \( H_w^{m,k}(t) \) be as in Definition 4.4. Using the polynomial \( \nu^{m,k}(\xi) \) in the definition of \( H_w^{m,k}(t) \), we define
\[
\tilde{\varphi}(\xi) = \frac{d}{d\xi} \nu^{m,k}(\xi), \tag{4.104}
\]
and furthermore,
\[
\tilde{\delta}_w(t) = \begin{cases} \frac{1}{w} \tilde{\varphi}(t/w) & |t| \leq w, \\ 0 & |t| > w. \end{cases}
\]
Since \( \nu^{m,k}(-1) = 0 \) and \( \nu^{m,k}(1) = 1 \), we have
\[
1 = \int_{-1}^{1} \frac{d}{d\xi} \nu^{m,k}(\xi) \, d\xi = \int_{-1}^{1} \tilde{\varphi}(\xi) \, d\xi, \tag{4.105}
\]
which is a requirement for any delta polynomial (4.75).
From $H_w^{m,k}(t) \in C^k(\mathbb{R})$, we get the following conditions on the derivatives of $\nu^{m,k}(\xi)$,

$$(\nu^{m,k})^{(\beta)}(\pm 1) = 0, \quad \beta = 1, \ldots, k. \quad (4.106)$$

For $\tilde{\varphi}(\xi)$ defined in (4.104), it then holds

$$\tilde{\varphi}^{(\beta)}(\pm 1) = 0, \quad \beta = 0, \ldots, k - 1. \quad (4.107)$$

From Definition 4.1, we also have that

$$\chi^{m,k}_\alpha = \int_{-1}^{1} \nu^{m,k}(\xi) \xi^\alpha d\xi - \frac{1}{\alpha + 1} = 0, \quad \alpha = 0, \ldots, m. \quad (4.108)$$

Since

$$\int_{-1}^{1} \nu^{m,k}(\xi) \xi^\alpha d\xi = \left[ \frac{\nu^{m,k}(\xi) \xi^{\alpha + 1}}{\alpha + 1} \right]_{-1}^{1} - \int_{-1}^{1} \frac{d}{d\xi} \nu^{m,k}(\xi) \xi^{\alpha + 1} d\xi$$

$$= \frac{1}{\alpha + 1} - \frac{1}{\alpha + 1} \int_{-1}^{1} \frac{d}{d\xi} \nu^{m,k}(\xi) \xi^{\alpha + 1} d\xi,$$

we have that

$$\chi^{m,k}_\alpha = -\frac{1}{\alpha + 1} \int_{-1}^{1} \frac{d}{d\xi} \nu^{m,k}(\xi) \xi^{\alpha + 1} d\xi = -\frac{1}{\alpha + 1} \int_{-1}^{1} \tilde{\varphi}(\xi) \xi^{\alpha + 1} d\xi, \quad (4.109)$$

using condition (4.104). From this equation, condition (4.108) yields

$$\int_{-1}^{1} \tilde{\varphi}(\xi) \xi^{\tilde{\alpha}} d\xi = 0, \quad \tilde{\alpha} = 1, \ldots, m + 1. \quad (4.110)$$

Equations (4.107) and (4.110), together with (4.105) yield the same conditions as equations (4.74)-(4.77) for $\varphi^{\tilde{m},k}(\xi)$ in Definition 4.9, with $\tilde{m} = m + 1$ and $\tilde{k} = k - 1$. The degree of the polynomial $\nu^{m,k}(\xi)(t)$ is $2\left[ \frac{m + 1}{n} \right] + 2k + 1$. The degree of $\tilde{\varphi}(\xi)$ is one less, i.e. $2\left( \frac{m + 1}{n} \right) + k$.

The lowest possible degree of $\varphi^{\tilde{m},k}$ is (according to Theorem 4.10) $2\left( \frac{\tilde{m} + 1}{n} \right) + \tilde{k} + 1 = 2\left( \frac{m + 1}{n} \right) + k$, and a polynomial of that degree, obeying all conditions of Definition 4.9 is unique.

Therefore, it holds that $\tilde{\varphi}(\xi) = \varphi^{m+1,k-1}(\xi)$ and so,

$$\delta_w(t) = \delta^{m+1,k-1}_w(t),$$

which completes the proof. \qed
4.3.5 Numerical Error

The numerical error in the integration of a function \( f_w(d(x)) \) is given by

\[
E_{\text{quad}}(f_w) = \int_\Omega f_w(d(x)) \, dx - \text{quad}(f_w(d(x))). \tag{4.111}
\]

where \( \text{quad}(f_w(d(x))) \) is evaluated as a sum of integrals over each triangle, and each local integral is approximated by a quadrature rule, see (4.10).

If the function \( f_w(d(x)) \) is smooth, the order of accuracy for the integration depends simply on the order of the quadrature rule that is used. A quadrature rule is designed to integrate a polynomial up to a certain degree exactly, and the order of accuracy of the quadrature rule is determined by how well a general smooth function can be approximated by a polynomial of this degree.

If the function is not smooth, the error will be larger, since this function can not be as well approximated by a polynomial. The numerical integration of a continuous function with a discontinuous \( p \)th derivative will not obtain the full order of accuracy of the quadrature rule applied, as long as \( p \) is too small.

We now study the integration of \( f_w(d(x)) \). The function \( f_w(t) \) is assumed to consist of different parts, pieced together at \( t = \pm w \), as is the case for the Heaviside approximations \( H_w(t) \) and the delta function approximations \( \delta_w(t) \). We assume \( f_w(t) \in C^p, p \geq 0 \). If the \((p+1)\)th derivative of \( f_w(t) \) is discontinuous, then partial derivatives of order \( p + 1 \) of \( f_w(d(x)) \) will be discontinuous across the curves in \( \Omega \subset \mathbb{R}^2 \) defined by \( d(x) = \pm w \).

The quadrature error for integrals over triangles intersected by \( d(x) = \pm w \) must therefore be investigated more carefully. Denote by \( E_T \) the quadrature error made over the triangle \( T \). We have the following theorem:

**Theorem 4.17.** Let

\[
\text{quad}_T(g(x)) = \sum_{k=1}^g w_k g(x_k) \tag{4.112}
\]

be a quadrature rule that is exact for polynomials in \( P^n(T) \), the set of polynomials of degree at most \( n \) on a triangle \( T \). Introduce the quadrature error over \( T \),

\[
E_T(g) = \int_T g(x) \, dx - \text{quad}_T(g(x)).
\]

Then, if

\[
\min_{p_n \in P^n(T)} \max_{x \in T} \| g(x) - p_n(x) \| \leq C_n, \tag{4.113}
\]

it holds that

\[
\| E_T(g) \| \leq C_n(1 + C_q) \text{area}(T).
\]
where

\[ C_q = \frac{1}{\text{area}(T)} \sum_{k=1}^{q} |w_k|. \]

**Proof.** Let \( p_n \in P^n(T) \). We have that

\[
E_T(g) = \int_T g(x) \, dx - \sum_{k=1}^{q} w_k g(x_k) 
= \int_T (g(x) - p_n(x)) \, dx + \int_T p_n(x) \, dx - \sum_{k=1}^{q} w_k g(x_k) 
= \int_T (g(x) - p_n(x)) \, dx + \sum_{k=1}^{q} w_k (p_n(x_k) - g(x_k)),
\]

where the last equality follows from

\[
\int_T p_n(x) \, dx = \sum_{k=1}^{q} w_k p_n(x_k).
\]

This yields,

\[
|E_T(g)| \leq |\int_T (g(x) - p_n(x)) \, dx| + \sum_{k=1}^{q} |w_k (p_n(x_k) - g(x_k))| 
\leq \max_{x \in T} |g(x) - p_n(x)| \left( \int_T \, dx + \sum_{k=1}^{q} |w_k| \right).
\]

From (4.113), we can choose \( p_n(x) \) such that \( \max_{x \in T} |g(x) - p_n(x)| \leq C_n \), and introducing \( C_n \) and \( C_q \), this proves the theorem.

One can note that \( C_q = 1 \) if all the quadrature weights are positive. In order to apply Theorem 4.17 on the integration of a function \( f_w(d(x)) \) which depends on \( x \) through \( d(x) \), we need to estimate \( \min_{p_n \in P^n(T)} \max_{x \in T} |f_w(d(x)) - p_n(x)| \).

Using a auxiliary function \( q(d) \), this quantity can be split in two parts

\[
\min_{p_n \in P^n(T)} \max_{x \in T} |f_w(d(x)) - p_n(x)| 
\leq \max_{x \in T} |f_w(d(x)) - q(d(x))| + \min_{p_n \in P^n(T)} \max_{x \in T} |q(d(x)) - p_n(x)| 
= \max_{d \in [d_{min}, d_{max}]} |f_w(d) - q(d)| + \min_{p_n \in P^n(T)} \max_{x \in T} |q(d(x)) - p_n(x)|.
\]

(4.114)

where the first part is now cast as a function of \( d \). The variables \( d_{min} \) and \( d_{max} \) are the minimum and maximum of \( d(x) \) for \( x \in T \). To proceed, we state the following theorem from Jackson [21]:

56 Chapter 4. Numerical Integration of Discontinuous and Singular Functions
4.3. Integration with Regularized Integrands: Error Analysis

**Theorem 4.18.** Assume that \( f(t) \in C^p(a, b) \), and that \( f^{(p+1)}(t) \) is bounded in \((a, b)\). Then there exists a polynomial \( q_n(t) \) of degree \( n \geq p \) such that, for \( a \leq t \leq b \),

\[
|f(t) - q_n(t)| \leq C(n, p) \max_{\tau \in [a, b]} |f^{(p+1)}(\tau)||b - a|^{p+1},
\]

where

\[
C(n, p) = \frac{G^{p+1}L^p}{p!n^{p+1}}(p + 1).
\]

This theorem can be used to estimate \( \max_{d \in [d_{\text{max}}, d_{\text{min}}]} |f_w(d) - q(d)| \). Since \( d(x) \) is assumed to be a distance function, \( d_{\text{max}} - d_{\text{min}} \leq h \), where \( h \) is the length of the longest side of the triangle. Assuming that \( f_w(d) \) has \( p \) continuous derivatives, the theorem states that we can choose \( q(d) \) as a polynomial of degree \( n \), such that

\[
\max_{d \in [d_{\text{min}}, d_{\text{max}}]} |f_w(d) - q(d)| \leq C(n, p) \max_{t \in \mathbb{R}} |f_w^{(p+1)}(t)| h^{p+1}.
\]

This estimate will however over estimate the error as \( w \to 0 \) since \( |f_w^{(p+1)}(t)| \) contains some power of \( 1/w \). For \( w = 0 \), we can choose \( q(d) = 1/2 \) and the error will never exceed \( 1/2 \). Therefore, we use the estimate

\[
\max_{d \in [d_{\text{min}}, d_{\text{max}}]} |f_w(d) - q(d)| \leq \min(1/2, C(n, p) \max_{t \in \mathbb{R}} |f_w^{(p+1)}(t)| h^{p+1}).
\]

Now, we need to estimate \( \min_{p_n \in P^*} \max_{x \in \omega} |\psi(d(x)) - p_n(x)| \). We have the following lemma:

**Lemma 4.19.** Let \( \omega \) be a simple domain such that \( \max_{x_0, x_1, \in \omega} |x_0 - x_1| \leq \tilde{h} \), and \( \max_{x_0, x_1, \in \omega} |y_0 - y_1| \leq \tilde{h} \), where \( x_i = (x_{i1}, y_{i2}) \), \( i = 0, 1 \). Introduce the multi-index \( \beta = (\beta_1, \beta_2) \), \( |\beta| = \beta_1 + \beta_2 \). Let \( \psi(x) \in C^{n+1}(\omega) \), such that

\[
|\frac{\partial^{\beta} \psi}{\partial x_{\beta_1} \partial y_{\beta_2}}| \leq c, \quad |\beta| \leq n + 1.
\]

Let \( q(\psi) \) be a polynomial in \( \psi \) of degree \( n \). Then there exists a \( p_n(x) \in P^n(\omega) \) such that

\[
\max_{x \in \omega} |q(\psi(x)) - p_n(x)| \leq C_n \max_{|\beta| = n+1} \max_{x \in \omega} \left| \frac{\partial^{\beta} \psi(x)}{\partial x_{\beta_1} \partial y_{\beta_2}} \right| \tilde{h}^{n+1},
\]

where \( C_n \) depends only on \( n \).

**Proof.** The result follows from the Taylor expansion. \( \Box \)
Since \( d(\mathbf{x}) \) is a distance function, and we have assumed \( (w \max_s |\kappa(s)|) < 1 \) (4.28), \( d(\mathbf{x}) \) fulfills the criteria (4.117). Denoting, as before, the length of the longest side of the triangle \( T \) by \( h \), Lemma 4.19 yields

\[
\min_{p_n \in P^n(T)} \max_{x \in T} |g(d(\mathbf{x})) - p_n(\mathbf{x})| \leq C_n \max_{|\beta|=n+1} \left| \frac{\partial^{\beta} g(d(\mathbf{x}))}{\partial x^{\beta}} \right| h^{n+1} = \tilde{C} h^{n+1}.
\]

(4.119)

From (4.114), (4.115) and (4.119) we have

\[
\min_{p_n \in P^n(T)} \max_{x \in T} \left| f_w(d(\mathbf{x})) - p_n(\mathbf{x}) \right| \\
\leq \min(1/2, C(n, p) \max_{t \in \mathbb{R}} |f_w^{(p+1)}(t)| h^{p+1} + \tilde{C} h^{n+1}).
\]

(4.120)

Once we have this estimate, we can use Theorem 4.17 with \( g(\mathbf{x}) = f_w(d(\mathbf{x})) \) to obtain

\[
E_T(f_w(d(\mathbf{x}))) \leq (1 + C_q) \left\{ \min(\frac{h^2}{2}, C(n, p) \max_{t \in \mathbb{R}} |f_w^{(p+1)}(t)| h^{p+1}) + \tilde{C} h^{n+1} \right\}.
\]

(4.121)

since \( \text{area}(T) \leq h^2 \). This is the local error produced over an element which is at least partly inside the transition zone where \( |d(\mathbf{x})| \leq w \). For elements totally contained in this zone, \( f_w(d(\mathbf{x})) \in C^\infty(T) \), and \( E_T = O(h^{n+3}) \). If \( f_w(d(\mathbf{x})) \) has \( p \) continuous derivatives on \( T \), with \( p < n \), the first term will be the dominating error term. This will in general be the case for all elements intersected by the curves described by \( d(\mathbf{x}) = \pm w \).

The global error is the sum over all the local errors, and it will be dominated by the contribution from the elements intersected by the contours \( d(\mathbf{x}) = \pm w \). If \( h \) is representative of the size of all the elements in the mesh, the number of such elements will be proportional to \( 1/h \). Therefore, the global error will be one order lower in \( h \) than the local error in (4.121).

Making the assumption \( p < n \), and assuming also that \( w \) is not too small compared to \( h \), so that it is in the range where estimate (4.115) holds (compare estimates (4.115) and (4.116)), we get the following estimate for the global error:

\[
E_{\text{quad}}(f_w) \sim C \max_{t \in \mathbb{R}} |f_w^{(p+1)}(t)| h^{p+2}.
\]

For the Heaviside approximation \( H_w(t) \) (4.17) defined using the transition function \( \nu(\xi) \), we have

\[
\max_{t \in \mathbb{R}} |H_w^{(p+1)}(t)| = \frac{1}{w^{p+1}} \max_{\xi \in (-1, 1)} |\nu^{(p+1)}(\xi)|.
\]

(4.122)

If we define \( \nu(\xi) \) so that \( H_w(t) \) has \( k \) continuous derivatives, we have

\[
E_{\text{quad}}(H_w) \sim C \frac{1}{w^{k+1}} \max_{\xi \in (-1, 1)} |\nu^{(k+1)}(\xi)| h^{k+2}.
\]

(4.123)
as an estimate of the error for the integration of $H_w(d(x))$ over $\Omega \subset \mathbb{R}^2$.

For the Dirac delta function approximation $\delta_w(t)$ (4.18) defined using the function $\varphi(\xi)$ in the zone where $|d(x)| \leq w$, we get

$$\max_{t \in \mathbb{R}} |\delta_w^{(p+1)}(t)| = \frac{1}{w^{p+2}} \max_{\xi \in (-1,1)} |\varphi^{(p+1)}(\xi)|. \quad (4.124)$$

If we define $\varphi(\xi)$ such that $\delta_w(t)$ has $k$ continuous derivatives, we have

$$E_{\text{quad}}(\delta_w) \sim C \frac{1}{w^{k+2}} \max_{\xi \in (-1,1)} |\varphi^{(k+1)}(\xi)| h^{k+2}, \quad (4.125)$$

as our estimate of the numerical error in the integration of $\delta_w(d(x))$ over $\Omega \subset \mathbb{R}^2$.

### 4.4 Integration with Regularized Integrands: Numerical Results

#### 4.4.1 Integration of the Approximation $H_w(d(x))$

From (4.111), the numerical error for integration of $H_w(d(x))$ is defined as

$$E_{\text{quad}}(H_w) = \int_{\Omega} H_w(d(x)) \, dx - \text{quad}(H_w(d(x))). \quad (4.126)$$

For the approximations $H_w^{2,k}(t)$, $k = 0, 1, 2$, introduced in the previous section, the analytical error $E_w(H_w^{2,k}) = 0$, and so the only contributing error is the numerical error $E_{\text{quad}}(H_w^{2,k})$.

For the polynomial approximations $H_w^{2,k}(t)$, $k = 0, 1, 2$, of the Heaviside function introduced in section 4.3.2, we have the following bounds for the $(k + 1)$'th derivatives. For $t \in \mathbb{R}$:

$$|(H_w^{2,0})^{(1)}(t)| \leq \frac{9}{8} \frac{1}{w^2}, \quad (k = 0),$$

$$|(H_w^{2,1})^{(2)}(t)| \leq \frac{15}{4} \frac{1}{w^2}, \quad (k = 1),$$

$$|(H_w^{2,2})^{(3)}(t)| \leq \frac{105}{4} \frac{1}{w^2}, \quad (k = 2).$$

Using the 13 point quadrature rule given in Appendix A, polynomials up to degree 12 can be integrated exactly, i.e. $n = 12$ and we have $k < n$. Given that the triangles in the decomposition of $\Omega$ is rather similar in size, assume that $w = h$ is in the range where the second part of the estimate (4.116) holds. These were the assumptions used to derive the error estimate (4.123), and the global errors $E_{\text{quad}}(H_w^{2,k})$ are predicted to show the following dependence:

$$E_{\text{quad}}(H_w^{2,0}) \sim \frac{h^2}{w}, \quad E_{\text{quad}}(H_w^{2,1}) \sim \frac{h^3}{w^2}, \quad E_{\text{quad}}(H_w^{2,2}) \sim \frac{h^4}{w^3}. \quad (4.127)$$
In Figure 4.5a the errors \( E_{\text{quad}}^{H^2_{w^0}}, E_{\text{quad}}^{H^2_{w^1}}, \) and \( E_{\text{quad}}^{H^2_{w^2}} \) are plotted versus the quadrature element size \( h \perp \) for \( d(x) \) as in (4.13) with \( \alpha = 0.5 \). The coordinate \((x, y)\) in (4.13) is shifted \( N \) times by small irrational shifts, and the error is taken as the maximum over these shifts. A mesh with \( \Delta e = 0.1 \) has been used, with the number of adaptive levels from \( l = 1 \) to 5, defining \( h \perp = \Delta e/2^l \). Since the analytical error is zero for these approximations, this is the total error, i.e. \( E_{\text{tot}}^{H^2_{w^k}} = E_{\text{quad}}^{H^2_{w^k}}, \) for all \( k \). From the plot, we can deduce that the errors decreases at least by the predicted rates in (4.127). Next, we investigate the dependence on \( w \). In Figure 4.5b, the errors \( E_{\text{tot}}^{H^2_{w^0}}, E_{\text{tot}}^{H^2_{w^1}}, \) and \( E_{\text{tot}}^{H^2_{w^2}} \) are plotted versus \( w/h \perp \). The lines are not straight, but do however approximately follow the slopes predicted in (4.127) when \( w/h \perp \) is larger than 1. Approaching smaller values of \( w \), the error does not increase with this rate as \( w/h \perp \to 0 \). In this region, these estimates overestimate the error made, as was discussed when the predictions were first introduced.

![Figure 4.5](image-url)

(a) Errors plotted versus \( h \perp \) for \( w = 0.025 \). The dashed lines indicate slopes \( h \perp k = 2, 3, 4 \). \( N = 16 \).

(b) Errors plotted versus \( w/h \perp \). \( h \perp = 0.025 \). The dashed lines indicate slopes \( 1/w, 1/w^3 \) and \( 1/w^5 \). \( N = 36 \).

**Figure 4.5**. Plots of errors \( E_{\text{quad}}^{H^2_{w^0}}, E_{\text{quad}}^{H^2_{w^1}}, \) and \( E_{\text{quad}}^{H^2_{w^2}} \). For each set of parameters, the error has been determined as the maximum error over \( N \) irrational shifts of \((x, y)\) in the definition of \( d(x) \) (4.13).

The analytical error for the approximation \( H^s(d(x)) \) is given by (4.46). With our choice of \( d(x), q = a \) and \( \kappa = 1/a \) this formula evaluates as

\[
|E^s| = \left( \frac{\pi}{\alpha} - \frac{2}{\pi} \right) w^2.
\]  

(4.128)

This analytical error will contribute to the total error, which can be seen in Figure 4.6. The numerical error decreases as \( w \) increases, but the analytical error increases with \( w \). This is why the optimal \( w \) to choose is the \( w \) minimizing the sum
of these two errors. This value will depend on $h_L$. Comparing $E_{\text{tot}}(H^w_w)$ to the errors for the polynomial approximations, $E_{\text{tot}}(H^{2,0}_w)$, $E_{\text{tot}}(H^{2,1}_w)$ and $E_{\text{tot}}(H^{2,2}_w)$, we can conclude that the polynomial approximations $H^{2,0}_w(t)$, $H^{2,1}_w(t)$ and $H^{2,2}_w(t)$ are clearly superior to the sine-approximation $H^w_w(t)$ in the sense that these yields a much smaller error $E_{\text{tot}}(H_w(d(x))) = \int H(d(x)) - \text{quad}(H_w(d(x)))$. In the next section, we will study the integration over these different approximations multiplied by an arbitrary smooth function.

\subsection*{4.4.2 Integration of the Product $H_w(d(x)) G(x)$}

In this section, we study the integration of the function $H_w(d(x))G(x)$, where $G(x)$ is a smooth function. The total error for this integration ($E_{\text{tot},G}(H_w)$) is the sum of the analytical error ($E_{w,G}(H_w)$) and the numerical error ($E_{\text{quad},G}(H_w)$), as was defined in (4.20). As was discussed in section 4.3.2, the analytical error is proportional to some power of $w$, and increases as this width of the transition zone gets larger. The numerical error on the other hand, decreases with increasing $w$. The numerical error is also decreased as the size $h_L$ of the quadrature triangles decreases.

We again define $d(x,y) = a - \sqrt{(x-x_c)^2 + (y-y_c)^2}$, with $a = 0.5$.

Two different functions $G(x)$ will be considered. The first one, $G(x,y) = (x-x_c)^2 + (y-y_c)^2$ is a second order polynomial, and can without multiplication of $H_w(d(x))$ be integrated exactly by the 13-point quadrature rule used. The other function is $G(x,y) = e^{\sqrt{(x-x_c)^2 + (y-y_c)^2}}$. The numerical integration of this function will always produce a quadrature error.

The analytical error for the sine approximation $H^w_w(t)$ (4.43) is proportional to $w^2$, as given by (4.45). The numerical error will therefore be negligible compared
to the analytical error for all but very small values of $w$, as was the case already for $G(x) \equiv 1$, see Figure 4.6.

The polynomial approximations $H_{w}^{2,k}$, $k = 0, 1, 2$ introduced no analytical error for $G(x) \equiv 1$. Figure 4.7 shows the total error for these approximations with $G(x, y) = (x-x_c)^2 + (y-y_c)^2$. Two different resolutions have been used, and it is clear that the numerical error decreases as $h_\perp$ is decreased. The analytical error for the approximations $H_{w}^{2,k}(t)$ (4.55) is proportional to $w^4$, as given by (4.66). Since this error increases with $w$, the total error will start to increase with $w$ as soon as the analytical error is dominating. This will happen for a smaller $w$ the smaller the numerical error is.

![Graphs showing errors for polynomial approximations](image)

Figure 4.7. $E_{w, G}(h_{w}^{2,k})$ plotted versus $w$, for the polynomial approximations $H_{w}^{2,0}(t)$, $H_{w}^{2,1}(t)$ and $H_{w}^{2,2}(t)$. $G(x, y) = (x-x_c)^2 + (y-y_c)^2$, $\Delta e = 0.05$, $N = 16$.

In the adaptive procedure, triangles that are intersected by the transition zone where $H_{w}^{2,k}(d(x))$ varies are split into sub-triangles. The triangles in which $H_{w}^{2,k}(d(x))$ takes a value of one contributes however also to the integral value, in difference to the triangles where $H_{w}^{2,k}(d(x))$ is zero. This part of the quadrature is performed over triangles of size $\Delta e$. In the case of the polynomial $G(x)$, which can be integrated exactly by the quadrature, this part introduces no error. Therefore, only the sub-elements intersected by the transition zone contribute to the quadrature error, where the sub-elements intersected by the boundaries of the transition zone, i.e. the level sets for $d(x) = \pm w$, will contribute with the dominating part, since the approximation only has $k$ continuous derivatives in these elements. In conclusion, the numerical error does not depend on the basic element size $\Delta e$, and the only resolution parameter is $h_\perp$, the size of the quadrature triangles in the region where $H_{w}(d(x))$ varies.
4.4. Integration with Regularized Integrands: Numerical Results

For integration of the product of $H_{w}^{2,k}(d(x))$ and the exponential $G(x)$, this previous remark is not valid. In addition to the error produced by triangles that are intersected by the the transition zone, the quadrature over non-intersected triangles where $H_{w}^{2,k}(d(x)) = 1$ produces an error, since this $G(x)$ cannot be integrated exactly. These triangles of size $\Delta e$ in the interior produce an error which defines a lower limit for what numerical error that can be achieved, independently of how small $h_\perp$ is. Therefore, the error depends both on $\Delta e$ and $h_\perp$.

The total error for the approximations $H_{w}^{2,k}(t)$, $k = 0, 1, 2$, with $G(x, y) = e^{\sqrt{(x-x_0)^2 + (y-y_0)^2}}$ is plotted in Figure 4.8. Two different meshes with $\Delta e = 0.05$ and $\Delta e = 0.1$ have been used. The level of adaptivity in the quadrature has been chosen such that $h_\perp$ is the same in both cases. Comparable to Figure 4.7,

![Figure 4.8](image)

(a) $\Delta e = 0.05$. 2 adaptive levels for the quadrature. $h_\perp = \Delta e / 2^2 = 0.0125$.

(b) $\Delta e = 0.1$. 3 adaptive levels for the quadrature. $h_\perp = \Delta e / 2^3 = 0.0125$.

the total error first decreases as the numerical error decreases, but then starts to increase with $w$ as the analytical error becomes dominating. The difference in this case is that the error curve flattens out before it starts to increase with $w$. This is explained in the previous paragraph, there is a lower limit for the numerical error that depends on $\Delta e$. In agreement with this, the flat zone is more pronounced for the larger value of $\Delta e$ in the plots in Figure 4.8.

In section 4.3.2 a second transition polynomial of degree 5, $\nu^4,0(\xi) (4.63)$ defining the Heaviside approximation $H_{w}^{4,0}(t)$ through (4.55) was introduced in addition to $\nu^2,1(\xi) (4.61)$ with corresponding Heaviside approximation $H_{w}^{2,1}(t)$.

In Figure 4.9, we compare the results for these two polynomial approximations, $H_{w}^{2,1}(d(x))$ with an analytical error of $O(w^4)$ and with one continuous
derivative, and $H_w^{1,0}(d(x))$ with an analytical error of $O(w^0)$ with a discontinuous first derivative. This has been done for both the polynomial and the exponential $G(x)$. In the case of the polynomial $G(x)$, the analytical error for $H_w^{4,0}(d(x))$ is actually zero due to vanishing derivatives. It is found that $H_w^{2,1}(d(x))$ yields a lower total error for moderate $w$, even if the analytical error is higher, since the discontinuous first derivative yields a larger numerical error for $H_w^{4,0}(d(x))$. As $w$ increases further however, the total error for $H_w^{2,1}(d(x))$ will eventually grow larger than the total error of $H_w^{4,0}(d(x))$ due to its more rapidly growing analytical error. For the exponential $G(x)$, where $H_w^{4,0}(d(x))$ has a non-vanishing analytical error, the total error $E_{\text{tot},G}(H^{4,0})$ should start to increase, although at a slower pace, as $w$ is increased even more.

The integrals in the variational formulation (3.11) including the discontinuous density and viscosity, will be evaluated with a $G(x)$ which will be a product of a few quadratic Lagrange interpolants, defined on the mesh. Therefore, $G(x)$ is always polynomial, and the quadrature error will only originate from triangles intersected by the transition zone of $H_w(d(x))$. There will therefore be no quadrature error depending on $\Delta x$ of the type present for $G(x, y) = e^{-(x-x_c)^2+(y-y_c)^2}$.

Theoretically, one can define transition polynomials $v^{k,m}(\xi)$ of higher and higher degree, so that more and more moments for the corresponding error function $E^{k,m}(t)$ is zero, thereby decreasing the analytical error. The number of continuous derivatives of the Heaviside approximation $H_w^{k,m}(t)$ can also be increased by using polynomials of higher order, thereby improving the numerical

\[ G(x, y) = (x - x_c)^2 + (y - y_c)^2. \]

\[ G(x, y) = e^{-(x-x_c)^2+(y-y_c)^2}. \]
properties of the approximation. If a higher order transition polynomial is used, one need to make sure that the quadrature rule used is of high enough order.

### 4.4.3 Integration of the Delta Function Approximation

The numerical error for integration of \( \delta_w(d(x)) G(x) \) is defined as

\[
E_{\text{quad}, G}(\delta_w) = \int \delta_w(d(x)) G(x) \, dx - \text{quad}(\delta_w(d(x)) G(x)).
\]  \hfill (4.129)

In section 4.3.5, the error expression for \( G(x) \equiv 1 \), was found to be

\[
E_{\text{quad}}(\delta_w) \sim C \frac{1}{w^{k+2}} \max_{\xi \in (-1,1)} |\varphi^{(k+1)}(\xi)| h^{k+2}.
\]  \hfill (4.130)

where \( k \) is the number of continuous derivatives of the approximative delta function, defined using \( \varphi(\xi) \) as in (4.18). We have that

\[
|\varphi^{(m,k)}(\xi)| \leq C,
\]

for some constant \( C \), independent on \( h \) and \( w \).

This error expression was derived assuming that \( w/h \) is in the range such that the second part of the estimate (4.116) holds. The numerical error for a general \( G(x) \) is expected to have the same asymptotic behavior, that is

\[
E_{\text{quad}, G}(\delta_w^{m,k}) \sim \frac{h^{k+2}}{w^{k+2}}.
\]  \hfill (4.131)

The analytical error for a delta approximation \( \delta_w^{m,k}(d(x)) \) with \( m \) vanishing moments is bounded by

\[
|E_{w,G}(\delta_w^{m,k})| \leq Cw^\beta,
\]

\[
\beta = 2 \left( \left\lceil \frac{m}{2} \right\rceil + 1 \right).
\]  \hfill (4.132)

The total error is the sum of these two errors. The cosine delta approximation, defined by (4.18), together with (7.73), has one vanishing moment, and one continuous derivative, i.e. \( m = 1 \) and \( k = 1 \). It will have no analytic error when integrated by itself, but will in general have a second order analytical error. The error for this approximation is very similar to the error of the polynomial approximation with \( m = 1 \) and \( k = 1 \), as is shown in Figure 4.10, where \( G(x, y) = (x - x_c)^2 + (y - y_c)^2 \). For small values of \( w \), the numerical error dominates, but it decreases as \( w \) gets larger. The analytical error, which is negligible for small \( w \) increases as \( w \) gets larger, and will soon become the dominating source of error. This error is proportional to the width \( w \) of the transition zone, and does not depend on the grid size, which can clearly be seen in the figure.

If we choose a delta approximation with at least three vanishing moments, \( m \geq 3 \), then the integration of \( \delta_w^{m,k}(d(x)) G(x) \), with \( G(x, y) = (x - x_c)^2 + (y - y_c)^2 \) will yield no analytic error. In this case, the total error will be the
Chapter 4. Numerical Integration of Discontinuous and Singular Functions

(a) The cosine delta approximation (has \( m = 1, k = 1 \)).

(b) The polynomial delta approximation \( g^k_{\lambda} (m = 1, k = 1) \).

**Figure 4.10.** \( E_{\text{ast}, G}(\delta_{\lambda}) \) with \( G(x, y) = (x - x_c)^2 + (y - y_c)^2 \) plotted versus \( w \) for two different approximations. In both figures, the error is plotted for \( h_0 = 0.00025 \) and \( h_1 = 0.00125 \). Maximum of error taken over \( N = 36 \) different positions \( (x_c, y_c) \).

(a) The errors \( E_{\text{ast}, G}(\delta_{\lambda}^{1,1}) \) plotted versus \( h_1 \). The slopes indicated are \( h_1^2, h_1^3 \) and \( h_1^4 \) (\( w = 0.05 \)).

(b) The errors \( E_{\text{ast}, G}(\delta_{\lambda}^{2,1}) \) plotted versus \( w/h_1 \). The slopes indicated are \( 1/w^2, 1/w^3 \) and \( 1/w^4 \) (\( h_1 = 0.00025 \)).

**Figure 4.11.** The errors \( E_{\text{ast}, G}(\delta_{\lambda}^{3,1}) \) with \( G(x, y) = (x - x_c)^2 + (y - y_c)^2 \), are plotted for \( k = 0, 1, 2 \). Plots versus \( h_1 \) and \( w/h_1 \). Maximum of error taken over \( N = 36 \) different positions \( (x_c, y_c) \).
4.5. Summary of the Quadrature Analysis

In Figure 4.11, the error is plotted for the approximations \( \delta^3_{w}(t) \), \( k = 0, 1, 2 \). The distance function \( d(x) \) is again as in (4.13), and the error is taken as the maximum over \( N \) shifts of \((x_n, y_n)\).

The slopes marked in the plots are the predicted rates of decay, as a function of \( h \) in Figure 4.11a, and as a function of \( 1/w \) in Figure 4.11b. The errors decay as a function of \( h \) at least by the predicted rates. Due to cancellation of errors, the slopes are not completely straight. In the right plot, the effect of error cancellation is even more apparent, even though the error has been taken over a maximum of 36 shifts in the mesh. Since the mesh is regularly divided, and all triangles are of the same size, some values of \( w \) will yield more cancellation of errors than others. The lines do however approximately follow the slopes indicating the predicted decay rates.

As can be seen, the behavior of the numerical results found for the integration of \( \delta_w(d(x)) \) and \( \delta_w(d(x))G(x) \) is very similar to the behavior of errors in the integration of approximations \( H_w(d(x)) \) and functions \( H_w(d(x))G(x) \), which were thoroughly discussed in the previous section.

4.5 Summary of the Quadrature Analysis

In this chapter, we have studied the integration of discontinuous and singular functions. The study of the integration of discontinuous functions was motivated by the integrals arising in the variational formulation of the Navier-Stokes equations (3.11), including the discontinuous density and viscosity fields. The functions are discontinuous over a curve \( \gamma \) in \( \Omega \subset \mathbb{R}^2 \). The numerical investigation has been applied to the integration of discontinuous functions over a triangulated two-dimensional domain. The analysis could be extended to three dimensions, in which case one would expect similar results, although exact error expressions would be different.

Two main approaches have been studied for the integration of discontinuous functions. In the first approach, a special method is used for the quadrature over triangles intersected by \( \gamma \), as was discussed in section 4.2. In the second approach, the discontinuity is smoothed out over a transition zone of a fixed width by introducing an approximation to the Heaviside function.

The first approach for the discontinuous functions, where the quadrature rule is modified, yields an \( O(h^2) \) error when \( \gamma \) is approximated by a straight line in each triangle. Higher order accuracy could be obtained if instead \( \gamma \) would be approximated by a second degree curve etc. This would however increase the complexity of the algorithm.

The total error for the second approach for the discontinuous function is the sum of the analytical error made when replacing \( H(d(x)) \) in the integral of \( H(d(x))G(x) \) by a smooth approximation \( H_w(d(x)) \) (4.17), and the numerical error made when integrating \( H_w(d(x))G(x) \).

The approximation \( H_w(d(x)) \) is in the transition zone \(-w \leq d(x) \leq w\) defined by a transition function \( \nu(d/w) \) varying smoothly from \( \nu(-1) = 0 \) to \( \nu(1) = 1 \).
Chapter 4. Numerical Integration of Discontinuous and Singular Functions

We have shown that for a general smooth $G(x)$ the analytical error in the integration of $G(x) H(d(x))$ when $H(d(x))$ is replaced by $H_w(d(x))$ is proportional to $w^m$, where the exponent $m$ is determined from the number of vanishing moments of the error function $E(t) = H(t) - H_w(t)$ (Theorem 4.6). The quadrature error for $H_w(d(x)) G(x)$ will contrary to the analytical error decrease with increasing $w$. The important property for $H_w(t)$ concerning the numerical errors produced in the number of continuous derivatives, as was shown in section 4.4.1. The uniqueness of the lowest degree transition polynomial, constructed to yield a corresponding approximation to the Heaviside function with $m$ vanishing moments for the corresponding error function as well as $k$ continuous derivatives, was shown in Theorem 4.2. The construction of this polynomial is straightforward.

Comparing the two approaches used for discontinuous functions, we consider the second approach, where $H(t)$ is replaced by an approximation $H_w(t)$, to be more convenient to use. It can also more easily be adopted to different requirements on the accuracy, by using approximations to the Heaviside function with more vanishing moments of the corresponding error function, reducing the analytical error, and more continuous derivatives, increasing the numerical order of accuracy.

It is not possible to give a choice of the width of the transition zone $w$ yielding the minimum of the total error in every case. To start with, this depends on the Heaviside approximation used, and also on the function $G(x)$ multiplying this approximation. In each such case, the optimal $w$ will also depend on the resolution of the calculations, i.e. the size of the quadrature triangles.

An analytical error proportional to $w^m$ is achieved for a Heaviside approximation for which the corresponding error function has $m = m - 2$ vanishing moments, $m$ even. The numerical error is proportional to $h^{k+2}/w^{k+1}$, where $k$ is the number of continuous derivatives of the Heaviside approximation. If a good choice of $w$ has been found for a certain problem, and one wishes to change the resolution $h$, then $w$ should scale with $h$ as $w \sim h^{(k+2)/(m+k+3)}$, in order to keep the balance between the analytical and numerical error.

A practical recommendation is to use a Heaviside approximation with an analytical error $w^m$ with $m \geq 4$, so that the analytical errors do not grow too fast, and to choose an approximation with at least one continuous derivative. In most cases, $h \leq w \leq 2h$ is a reasonable choice of $w$.

The study of integration of singular functions is motivated by the presence of localized surface tension forces, and thereby the need to evaluate the integral (3.17) in the variational formulation of the Navier-Stokes equations. The natural way to evaluate the integral including the singular forces, is to evaluate the line integral along $\gamma$. The order of accuracy of these calculations depends simply on the order of the 1D quadrature rule used for each segment of the curve. If one prefers to evaluate these integrals over $\Omega$ instead of evaluating the line integral, explicit discretization of the delta function is needed. The analysis of the error made in this case is very similar to the error analysis for discontinuous functions for the second approach described above, where the function is replaced by a more
4.5. Summary of the Quadrature Analysis

regular approximation. This study is also of value for the numerical methods that are based on the strong formulation of the differential equations (3.2), (3.3).

The results for the approximation \( \delta_w(d(x)) \) are similar to the results obtained for the approximation \( H_w(d(x)) \). Also here, the analytical error is proportional to some \( w^m \), where the exponent \( m \) in this case is determined from the number of vanishing moments of the approximation \( \delta_w(t) \). The order of the quadrature error is determined by the number of continuous derivatives of the approximation \( \delta_w(t) \). The uniqueness of the lowest degree delta polynomial, constructed to yield a corresponding approximation to the delta function with \( m \) vanishing moments as well as \( k \) continuous derivatives, was shown in Theorem 4.10.

The construction of this polynomial is straightforward. The relation to the polynomial Heaviside approximation \( H_w^{m,k}(t) \) with \( m \) vanishing moments of the corresponding error function and \( k \) continuous derivatives, was shown to be that the derivative \( \frac{d}{dt} H_w^{m,k}(t) \) defines the polynomial delta function approximation with one more \( (m + 1) \) vanishing moments, and one less \( (k - 1) \) continuous derivatives.

This analysis has been made assuming that the functions multiplying the Heaviside and Dirac delta functions are smooth.
Chapter 5

Discretization

5.1 Discretization of the Navier-Stokes Equations

The discretization of the Navier-Stokes equations is presented in the following way. The time discretization, including iterations for the nonlinearity is presented first. This discretization results in a Stokes-like problem. The iterated penalty method used to enforce the incompressibility constraint when solving this problem is briefly discussed in section 5.1.2. The choice of piecewise quadratic polynomials for the spatial discretization, together with the choice of quadrature rule for evaluation of the integrals, are presented in section 5.1.3. We conclude with a brief discussion about how to solve the resulting linear system of equations by a preconditioned conjugate gradient method.

5.1.1 Time Discretization

The scheme used to advance the Navier-Stokes equations in time includes a combination of implicit and explicit terms and an iteration over the nonlinearity. The discretization of the time derivative is given by:

\[
\frac{\partial \mathbf{u}}{\partial t} \bigg|_{t_{n+1}} = \frac{3 \mathbf{u}^{n+1} - 4 \mathbf{u}^n + \mathbf{u}^{n-1}}{2 \Delta t} + O((\Delta t)^2),
\]

(5.1)

where the indices indicate the time levels. The source terms of equation (3.11) are discretized explicitly. We introduce the notation

\[
F^n(\mathbf{v}) = 2 f_{g^n}(\mathbf{v}) - f_{g^{n-1}}(\mathbf{v}) + 2 m(\rho^n, \mathbf{g}, \mathbf{v}) - m(\rho^{n-1}, \mathbf{g}, \mathbf{v}).
\]

(5.2)

Furthermore, denote

\[
G^n(\rho, \mathbf{v}) = \frac{2}{\Delta t} m(\rho, \mathbf{u}^n, \mathbf{v}) - \frac{1}{2 \Delta t} m(\rho, \mathbf{u}^{n-1}, \mathbf{v}).
\]

(5.3)
and
\[ d(\rho, \mu, u, v) = \frac{3}{2\Delta t} m(\rho, u, v) + \tilde{a}(\mu, u, v). \] (5.4)

Using this notation, the full scheme for the discretization of the variational formulation (3.11) reads
\[ d(\rho^{n+1}, \mu^{n+1}, u^{n+1}, v) - G^n(\rho^{n+1}, v) + b(v, p^{n+1}) + c(\rho^{n+1}, u^{n+1}, u^{n+1}, v) = F^n(v). \] (5.5)

To resolve the nonlinearities, we introduce a fixed point iteration, where we initialize \( u_0^{n+1} = u^n, \rho_0^{n+1} = \rho^n \) and \( p_0^{n+1} = \mu^n \) and consecutive iterates \( u_k^{n+1}, k = 1, 2, \ldots \), are computed from
\[ d(\rho_k^{n+1}, \mu_k^{n+1}, u_k^{n+1}, v) + b(v, p^{n+1}) =
- c(\rho_k^{n+1}, u_k^{n+1}, u_k^{n+1}, v) + G^n(\rho_k^{n+1}, v) + F^n(v). \] (5.6)

After each iteration, the density and viscosity fields are updated by computing a new temporary bubble position. When the difference between two consecutive velocity fields is small enough, the iteration is terminated, i.e. the stopping criterion used for this iteration process is
\[ \frac{||u_{k+1} - u_k||_\infty}{||u_{k+1}||_\infty} < \epsilon, \] (5.7)

where \( \epsilon \) is a tolerance limit.

### 5.1.2 Incompressibility

The time discretization discussed in the previous section results in a Stokes problem (5.6) to solve inside each fixed point iteration. Move all terms that can be computed to the right hand side, and denote the sum of all these terms by \( F(v) \). Dropping the indices, this problem can be stated as: Given \( \rho(x) \) and \( \mu(x) \), find \( u(x) \in V \) and \( p(x) \in \Pi \) such that
\[ d(\rho, \mu, u, v) + b(v, p) = F(v) \quad \forall v \in V, \] (5.8)
\[ b(u, q) = 0 \quad \forall q \in \Pi, \] (5.9)

where \( d(\rho, \mu, u, v) \) is as defined in (5.4).

The divergence-free constraint (5.9) is enforced using an iterated penalty method (described by Brenner and Scott in [8]). With this method, the problem is reformulated as an iterative procedure, where the divergence of the velocity field is removed as the pressure is obtained through iterative updates.

Choose a \( w^0 \in V \), and iterate
\[ d(\rho, \mu, u^l, v) + r(\text{div} u^l, \text{div} v)_{\Pi} = F(v) + (\text{div} v, \text{div} w^l)_{\Pi} \quad \forall v \in V, \]
\[ w^{l+1} = w^l - r u^l, \] (5.10)
5.1. Discretization of the Navier-Stokes Equations

until

$$||\text{div}(u^l)||_H \leq \epsilon||u^l||_V.$$  \hfill (5.11)

The penalty parameter $r$ is a large number to be chosen. The pressure is defined by $p^l = \text{div} \, W^l$.

Let $W_h$ denote the space of piecewise polynomials of degree two on a triangular mesh, and let $V_h$ denote the subspace of $W_h$ with the additional constraint that any member of $V_h$ vanishes at the boundary. Denote by $V_h = V_h \times V_h$ and let $\Pi_h = \text{div} \, V_h$. The space $\Pi_h$ does not need to be constructed, since there are no functions belonging to this space retained in the variational formulation (5.10), and the pressure can be computed from $w \in V_h$ where $w$ is obtained from the penalty iterations. For homogeneous boundary conditions, it holds that $p = \text{div} \, w$ with $w \in V_h$, whereas for inhomogeneous boundary conditions, a $\Pi$-projection of $\text{div} \, w$ has, in general, to be made onto $\Pi_h$ in order to obtain the pressure. This can however be done by introducing a $z \in V_h$ such that $p = \text{div} \, z$, and it is not necessary even in this case to construct a basis for $\Pi_h$.

Since the pressure is not solved for explicitly, the system of equations will be smaller compared to methods requiring direct solution of the pressure, with the number of unknowns determined solely by the dimension of the approximation space for the velocity. The two velocity components are however coupled, both in the divergence operator and in $d(\rho, \mu, ...)$ in (5.10). The coupling in $d(\rho, \mu, ...)$ (5.4) is due to the second part of $a(\mu, \cdot, \cdot)$, defined in (3.13).

Using a large enough penalty parameter, this method usually converges within the first few iterations even with a very restrictive tolerance $\epsilon$. With the equations normalized so that $\mu_H = 1$, the range of values for the penalty parameter $r$ has been $10^4 - 10^6$, and the tolerance $\epsilon$ has been kept in the range $10^{-4} - 10^{-6}$.

5.1.3 Spatial Discretization

As was indicated in the previous section, piecewise quadratic polynomials have been used for the discretization. Assume there is a triangulation of $\Omega$, in which the elements are indexed by $E$, and denote the corresponding triangle by $T_E$. The elements used here are quadratic Lagrange triangles. The number of local nodes is 6, and the set of the local nodes is denoted by $L$. The Lagrange interpolant of any continuous function $f(x)$ in $V_h$ can be written as

$$f_L(x) = \sum_E \sum_{\alpha \in L} f(x_{i(e, \alpha)}) \varphi^E_{\alpha}.$$ \hfill (5.12)

The index $i(e, \alpha)$ relates the local (or element) node number $\alpha$ on a particular element, indexed by $E$, to its global node number $p = i(e, \alpha)$. On triangle $T_E$, \{ $\varphi^E_{\alpha} : \alpha \in L$ \} denotes the set of basis functions. Each basis function $\varphi^E_{\alpha}$ takes the value of 1 at the local node $\alpha$ and 0 at all other nodes.
To illustrate the evaluation of integrals, using this notation, we expand

$$\int_\Omega \mu \operatorname{tr}(\nabla u^T : \nabla v) \, dx = \sum_{e} \sum_{i=1}^{2} \int_{T_e} \mu \nabla u_i \cdot \nabla v_i \, dx =$$

$$2 \sum_{i=1}^{2} \sum_{e} \sum_{i=1}^{2} u_i^{(e, a)} v_i^{(e, b)} \int_{T_e} \mu(x) \frac{\partial \varphi_a(x)}{\partial x_j} \frac{\partial \varphi_b(x)}{\partial x_j} \, dx \quad (5.13)$$

For continued evaluation each element is mapped onto a “reference” element $\tau$, thereby relating all “element” basis functions $\varphi^\epsilon_a$ to a fixed set of basis functions on the reference element $\tau$. This is done via an affine mapping $\xi \to x = J^\tau \xi + \chi_e$ of $\tau$ to $T_e$, so that

$$\varphi^\epsilon_a(x) = \varphi_a((J^\tau)^{-1}(x - \chi_e)) = \varphi_a(\xi). \quad (5.14)$$

The Jacobian for the inverse mapping $(x \to \xi = (J^\tau)^{-1}(x - \chi_e))$ is given by

$$(J^\tau)^{-1}_{m,j} = \frac{\partial \xi_m}{\partial x_j} \quad (5.15)$$

The integral in (5.13) is evaluated as

$$\int_{T_e} \mu(x) \frac{\partial \varphi_a(x)}{\partial x_j} \frac{\partial \varphi_b(x)}{\partial x_j} \, dx =$$

$$2 \sum_{m, m' = 1}^{2} \frac{\partial \xi_m}{\partial x_j} \frac{\partial \xi_{m'}}{\partial x_j} \int_{\tau} \mu(J^\tau \xi + \chi_e) \frac{\partial \varphi_a(\xi)}{\partial \xi_m} \frac{\partial \varphi_b(\xi)}{\partial \xi_{m'}} |\det J^\tau| \, d\xi \quad (5.16)$$

The integrals over the reference element are then evaluated by using a 13-point quadrature formula (cf. Strang and Fix [40], section 4.3). The points and weights for this formula are displayed in Appendix A. Denoting the quadrature points by $\{\eta_{q}\}_{q \in Q}$ and the corresponding weights by $\{\omega_{q}\}_{q \in Q}$, the integral of any function $f(\xi)$ can be evaluated as

$$\int_{\tau} f(\xi) \, d\xi = \sum_{q \in Q} \omega_{q} f(\eta_{q}) \quad (5.17)$$

The evaluation of integrals containing the discontinuous density and viscosity is made following the theory formulated in chapter 4. As in (4.7), the signed distance function $d(x)$ is defined as the closest distance from $x$ to any interface $\gamma_j$, with a positive sign if $x$ is in fluid $A$ and a negative sign if it is in fluid $B$. Introduce the modified Heaviside function

$$H_w(t) = \begin{cases} 
1 & t > w, \\
\nu(t/w) & |t| \leq w, \\
0 & t < -w,
\end{cases} \quad (5.18)$$
where \( \nu(\zeta) \) is a smooth transition function such that \( \nu(-1) = 0 \) and \( \nu(1) = 1 \).

The approximate characteristic function is defined as

\[
I_w(x) = H_w(d(x)),
\]

similarly to the definition of the characteristic function \( I(x) = H(d(x)) \).

To complete this definition, the transition function \( \nu(\zeta) \) in the definition of \( H_w(t) \) must be chosen. The important property for \( H_w(t) \) is its integral properties. The numerical evaluation of integrals with \( I(x) = H(d(x)) \) replaced by \( I_w(x) = H_w(d(x)) \) should yield a result that differs as little as possible compared to the exact evaluation of the integral containing \( H(d(x)) \). This error can be split into two parts, the analytical error made by replacing \( H(d(x)) \) by \( H_w(d(x)) \) and the numerical error made when evaluating the integral containing \( H_w(d(x)) \). Both these parts of the error were analyzed in chapter 4.

The density and viscosity is defined through (3.19), with \( I(x) \) replaced by \( I_w(x) \). To be able to use this definition, the signed distance function \( d(x) \) needs to be defined in a region of width \( w \) on both sides of the zero contour. That is, the closest distance to the zero contour for each point in this region is needed, together with a sign defining which fluid the point is in. Define the modified signed distance function

\[
d_w(x) = \begin{cases} 
  d(x) & \text{for } |d(x)| \leq w, \\
  \text{sign}(d(x)) w & \text{for } |d(x)| > w.
\end{cases}
\]

Then it holds that \( H_w(d(x)) = H_w(d_w(x)) \). The definition and calculation of \( d_w(x) \) are made differently in the three methods.

The integral over the surface tension forces (3.17) is evaluated by calculating the line integral along each \( \gamma_i \), i.e. each interface separating the two fluids. The algorithms for this evaluation are different in the three methods, which will be described in sections 5.2-5.4.

5.1.4 Solution of the Linear Algebraic System

In order to enforce equation (5.10) to hold for all \( v \in V_h \), it needs to be enforced for all basis functions in \( V_h \). Practically, the equations are assembled for all \( \Psi \in \Psi_h \), \( i = 1, 2 \) (x and y direction), \( p = 1, \ldots, N \), where \( N \) is the number of nodes in the mesh, and

\[
\Psi^p = \sum_{e} \sum_{\beta} \varphi_{e, \beta}^p, \quad \text{with } (e, \beta) \text{ such that } i(e, \beta) = p.
\]

The element basis functions \( \varphi_{e, \beta}^p \) were introduced in section 5.1.3. The equations do not include the unknown pressure, but the two velocity components are coupled, as was discussed in section 5.1.2. We therefore solve for the two velocity components in each node, and the resulting algebraic system has \( 2N \) unknowns.

The matrix for this system depend on \( \rho(x) \) and \( \mu(x) \), and will therefore not be constant, since \( \rho(x) \) and \( \mu(x) \) depend on the configuration of the fluids A and B,
which is changing with time. The system is symmetric, and can be solved using a conjugate gradient method. For the system to converge in a reasonable number of iterations, a good preconditioner is however needed. One straight-forward way to define a preconditioner is to assemble the system matrix assuming constant density and viscosity \((\rho_B, \mu_B)\), factor it once, and use it as a preconditioner for the cost of a back solve in each conjugate gradient iteration. This works well when the jumps in density and viscosity are relatively small. Another approach is to assemble a matrix with \(\rho(x, t^*)\) and \(\mu(x, t^*)\) at one instant in time, factor it, and keep it as a preconditioner until the number of iterations needed for convergence exceeds a certain limit. Fortunately, since the structure of the matrix does not change, symbolic information from the first factorization can be used to speed up the refactorization after update of the matrix. This approach is favorable compared to the constant preconditioner approach when the jumps in density and viscosity are large.

5.2 The Discrete Segment Projection Method

5.2.1 Discretization of Segments

We define a uniform \(x\)-discretization from the minimum \(x\)-value of the computational domain \((x_{\text{min}})\), to the maximum \(x\)-value of the domain \((x_{\text{max}})\). The number of intervals is \(N_x\), and the distance between two discrete points is \(\Delta x\). The same is done in the \(y\)-direction, with \(N_y\) intervals, and a grid size of \(\Delta y\). For the \(x\)-discretization, we have

\[
x_k = x_{\text{min}} + k \Delta x, \quad k = 0, \ldots, N_x,
\]

and for the \(y\)-discretization

\[
y_k = y_{\text{min}} + k \Delta y, \quad k = 0, \ldots, N_y.
\]

In the continuous description in 2.1, an \(x\)-segment is defined from \(\bar{x}_0\) to \(\bar{x}_1\). In the discrete case, corresponding integer variables \(k_0\) and \(k_1\) are defined so that \(x_{k_1-1} < \bar{x}_0 \leq x_{k_0}\) and \(x_{k_1} \leq \bar{x}_1 < x_{k_1+1}\). The discretized \(x\)-segment is defined from \(x_{k_0}\) to \(x_{k_1}\). Similarly, we define \(k_0\) and \(k_1\) also for each \(y\)-segment.

For each \(x\)-segment, the points on the curve are then given by

\[
(x_k, f(x_k)) = (x_k, f_k), \quad k = k_0, \ldots, k_1,
\]

and for each \(y\)-segment

\[
(g_j(y_k), y_k) = (g_k, y_k), \quad k = k_0, \ldots, k_1.
\]

The discrete points are in general different for an \(x\)-segment and a \(y\)-segment describing the same part of the curve. The variables \(k_0\) and \(k_1\) are in general different for each segment.
The representation with segments in the continuous case was exemplified with a circle in section 2.1. In Figure 5.1, the discretizations of the segments representing the circle are indicated.

In addition to the discretization of the individual segments, the connectivity of the segments needs to be defined, as has already been described for the continuous case (section 2.1). In the discrete case, the points \( x_m^* \) in (2.7), will be points of the discretization, and not arbitrary \( x \) coordinates.

In the case when two extremum points \( x_m^* \) and \( x_{m+1}^* \) are so close relative to the slope, that a segment discretized in the opposite variable \( y \) would be less than two points long, then no segment in the \( y \)-direction will be discretized for this part, and the corresponding opposite segment is set to be what we call an empty segment. The use of empty segments is discussed further in section 5.2.2.

5.2.2 Advection and Reinitialization

The advection equation for segments \( y = f(x, t) \) and \( x = g(y, t) \) were given in equations (2.8)-(2.9). These equations are on the form

\[
\frac{\partial w}{\partial t} + a \frac{\partial w}{\partial x} = b. \tag{5.26}
\]

Assume that the velocity components, here denoted by \( a(x, y, t) \) and \( b(x, y, t) \), are given for times \( t^n \) and \( t^{n+1} \). Each segment needs to be advected from its position at \( t^n \) to the new position at \( t^{n+1} \). Denote \( a_j^n = a(x_j, w_j^n, t^n) \) for an \( x \)-segment, and \( a_j^n = a(w_j^n, y_j, t^n) \) for a \( y \)-segment. The definition for \( b_j^n \) is analogous.

We use the following Lax-Wendroff scheme to discretize the equations. First compute

\[
\tilde{w}_j^{n+1} = w_j^n + \Delta t \left( b_j^n - a_j^n D_h w_j^n \right). \tag{5.27}
\]
Proceed by computing $\tilde{a}_j^{n+1}$ where $\tilde{a}_j^{n+1} = a(x_j, \bar{w}_j^{n+1}, \bar{\rho}_j^{n+1})$ for an x-segment and $\tilde{a}_j^{n+1} = a(\bar{w}_j^{n+1}, y_j, \bar{\rho}_j^{n+1})$ for a y-segment and in a similar manner, compute also $\tilde{b}_j^{n+1}$. We then define

$$w_j^{n+1} = \bar{w}_j^{n+1} + \frac{\Delta t}{2} \left( a_j^0 \left( (D_0a_j)(D_0w_j^n) + a_j^n D_y w_j^n - D_0b_j^l \right) \right)$$

$$+ \frac{\Delta t}{2} \left( \frac{\bar{w}_j^{n+1} - b_j^n}{\Delta t} - \frac{a_j^{n+1} - a_j^n}{\Delta t} D_0w_j^n \right).$$

(5.28)

Here, $D_0$, $D_y$, and $D_{\bar{u}}$ are the usual discrete divided difference operators. This scheme is second order in both space and time.

The surface tension forces have a regularizing effect on the interfaces. As the interface deforms, the flow will respond to the modified surface tension forces. The flow does however not act on scales smaller than the resolution of the finite element mesh. If the segment discretization is chosen much finer than the resolution of the finite element mesh, then this effect is in general not sufficient to keep the interfaces smooth. In that case, the diffusion term

$$\epsilon D_y D_y w_j^n,$$

(5.29)

where $\epsilon = 0.25 \Delta t \Delta t \|u\|_2$, is added to the right hand side in (5.28). The grid size $\Delta x = \Delta x$ for an x-segment and $\Delta y = \Delta y$ for a y-segment, and $u$ is the velocity field.

The segment structure needs to be updated after each advection step, as was described in section 2.1 for the continuous case. If new extrema have appeared and/or if extrema have disappeared in an x-segment, the structure of the y-segments needs to be modified, and vice versa. A segment might need to be split into two parts, or two segments might need to be merged into one. The difference compared to the continuous case is that no short segments are added or removed in this process. Instead of keeping very short segments, an empty segment is referenced instead. An empty segment is a segment that can be referenced as a corresponding segment, but that contains no information.

Therefore, we need to monitor the use of empty segments. Using the notation introduced around equation (2.7), for each interval

$$\left( x_m^e, x_{m+1}^e \right), \quad m = 1, \ldots, M - 1,$$

(5.30)

we need to check if there should be a segment discretized in the y-direction, or if there should only be a reference to an empty segment. Denote the value at $x_m^e$ by $f_m^e$. If $|f_m^e - f_{m+1}^e| < 2\Delta y$, there should not be an opposite segment (i.e. y-segment) in this interval. If such a segment exists, it should be removed, and an empty segment should be referenced instead. On the other hand, if $|f_m^e - f_{m+1}^e| > 2\Delta y$, there should be an opposite segment corresponding to this interval. If there is no such segment, then it should at this point be created by interpolation from this segment, and added to the structure. For each interval
(\(x_m^*, x_{m+1}^*\)), we make a reference to the corresponding \(y\)-segment, or to an empty segment. The same procedure is made for each \(y\)-segment, where \(|y^*_m - y_{m+1}^*|\) is compared to \(2\Delta x\).

As was described in the end of section 2.1, the domain of definition of a segment might need to be extended or reduced after the advection. The extension of this domain is defined by the position of extrema in segments in the opposite direction. If it is extended, so that new values of the discretization need to be defined, these values are interpolated linearly from the corresponding segment discretized in the opposite direction. In this process, we update \(k_0\) and \(k_1\) for all segments. In the case of the open curve, i.e. a curve that is attached to a boundary at both ends, the same thing holds for all segments, except for the segments that end at the boundary. Those segments should always be defined all the way out to the boundary.

For each part of the structure, there is at least one segment defined. A segment will yield the most accurate description of a part of the interface if the slope of the segment is small. For two overlapping segments, both segments define the same part of the interface, and we can not allow for any discrepancies to develop in the representation. Therefore, we need to interpolate between the overlapping parts of segments discretized in \(x\) and \(y\).

For a given point \(x_k\), the \(x\)-segment has the value \(\tilde{f}_k = f(x_k)\). Denote by \(f_k\), the value at \(x_k\), interpolated from the segment discretized in the \(y\)-direction. Cubic interpolation is used. Compute the slope of the segment function at \(x_k\) by \(s_k = 0.5[D_0f_k + D_0\tilde{f}_k]\). The new value at \(x_k\) is set to

\[
\begin{equation}
\bar{f}_k = f_k + \theta(s_k)(\tilde{f}_k - f_k),
\end{equation}
\]

where the weighting function \(\theta(s)\) is a function of the slope \(s\), and is given by:

\[
\begin{cases}
0 & \text{for } s < 1/\beta, \\
\frac{\beta(s - 1/\beta)}{\beta - 1} & \text{for } 1/\beta \leq s \leq \beta, \\
1 & \text{for } s > \beta.
\end{cases}
\]

Note, that a slope of \(\beta\) in the segment discretized in the \(x\)-direction, corresponds to a slope of \(1/\beta\) in the segment discretized in the \(y\)-direction. We have used \(\beta = 2.0\), in which case no influence is taken from the opposite segment if the slope of the discretization is less than 0.5, and where the value is set to be the value interpolated from the opposite segment if the slope is larger than 2.0. In between, the new value is a weighted sum of the original value and the value interpolated from the opposite segment. The same weighting of original values and interpolated values is done for both the part of the \(x\)-segment and the part of the \(y\)-segment that define the overlap.

The advection and reinitialization process for a structure of segments, representing a curve \(\gamma\), can be summarized as follows:

(i) Given a velocity field \(\mathbf{u} = (u, v)\), advect all \(x\)- and \(y\)-segments.

(ii) Update the segment structure. For each segment do the following.
- Check if any new extrema have appeared, or if any extrema have disappeared. Split or merge corresponding segments in the opposite direction if necessary.

- Review the use of empty segments. Add and/or remove short segments, as described in connection to (5.30). Set references to corresponding segments in the opposite direction, and/or to empty segments.

- Use the position of the extrema to update the start index \(k_0\) and/or the stop index \(k_1\) of opposite segments, (5.24)-(5.25).

(iii) For each segment, whose domain of definition \([x_{k_0} \to x_{k_1}, \ y_{k_0} \to y_{k_1}]\) has been increased, new values need to be defined. These are interpolated from the corresponding segment in the opposite direction.

(iv) Interpolate between overlapping parts of \(x\) and \(y\) segments. The new values are assigned using a weight function based on the slopes of the segments (5.32).

### 5.2.3 Evaluating the Interfacial Force Term

To evaluate the interfacial force term (3.17), we need to have a representation of the interface, and the product of the curvature and normal vector \(\kappa n\) at points around the interface.

Since we have overlapping segments, we need to define which part(s) of each segment that should be used to define the interface, and in which order they should be used. In general, a segment holds a good representation of a part of the interface if the slope of the segment is small at this part.

For this algorithm, we need to distinguish between the interfaces that are closed, and the ones that begin and end at boundaries. For a segment structure that represents a closed curve, we start in the middle of one segment, where we have a small slope. We let the segment define the interface, until the slope gets too large. Then we shift to the segment in the opposite variable. We continue like this, shifting between segments defined in \(x\) and \(y\), respectively, until we are back at the starting point. We have now defined in which order segments should be visited, and which part of the segment that should be used at that visit.

In the case of an open structure, we need to start at a segment connected to a boundary. We start at the boundary point, and the switches between different segments are done as before. We stop when we have reached the end of this curve, at another boundary point.

The output of this procedure is a list, where each object contains a reference to a segment and integers defining an interval, \((p_0, p_1)\), for which the segment should be used. The same segment can appear in the list more than once, with different intervals \((p_0, p_1)\).

The curvature and normal vectors can be computed from the 1D-discretization of the interfaces by divided differences. The formulas for calculation of the prod-
uct \(\kappa\) are given in (3.20)-(3.21). We approximate the derivatives in these formulas by \(f'(x_k) \approx D_0 f_k\) and \(f''(d_k) \approx D_4 D_\perp f_k\) and similarly, \(g'(y_k) \approx D_0 g_k\) and \(g''(y_k) \approx D_4 D_\perp g_k\).

The contribution of the surface tension forces in the weak form of the equations was given in (3.17). The definition of the structure gives us a set of ordered points that describe each interface, and we can also calculate the curvature and normal vectors at these points. The line integral can therefore be calculated by approximating the curve as a straight line between each two points (second order approximation).

However, the basis functions defined on the mesh are included in the integral. Since the basis functions have a discontinuous first derivative across element edges, the integration will only be first order accurate if we integrate across these edges. Therefore, we define the intersection points between the interface and the element edges. At these points, we interpolate the value of the normal vector and curvature linearly from the two closest points. This gives us an overall accuracy of second order for the evaluation of the line integral over the surface tension forces.

### 5.2.4 Density and Viscosity Fields

In the end of section 5.1.3, we introduced the modified characteristic function \(I_w(x) = H_w(d(x)) = H_w(d_w(x))\). In order to define this function, we need to compute the modified distance function \(d_w(x)\) (5.20), and to choose the transition function \(\nu(\xi)\) in the definition of \(H_w(t)\) (5.18).

The transition function used in the calculations is the fifth order polynomial

\[
\nu(\xi) = \frac{1}{2} + \frac{1}{32} \left( 45 \xi - 50 \xi^3 + 21 \xi^5 \right).
\]  

(5.33)

This polynomial has two vanishing moments \((m = 2)\) and one continuous derivative \((k = 1)\), referring to the notation in chapter 4. The integration of a function \(f(x)H_w(x)\) yields a total error as the sum of the analytical error proportional to \(w^4\) and the quadrature error proportional to \(h^3/w^2\), where \(h\) is the size of the quadrature triangle, i.e. the elements in the finite element mesh, and \(w\) is half the width of the transition zone.

Introduce the modified unsigned distance function \(\tilde{d}_w(x)\),

\[
\tilde{d}_w(x) = \begin{cases} 
|d(x)| & \text{for } |d(x)| \leq w, \\
\frac{w}{d(x)} & \text{for } |d(x)| > w,
\end{cases}
\]  

(5.34)

where \(d(x)\) is the signed distance function, such that \(\tilde{d}_w(x)\) gives the closest distance to any interface if the distance is not larger than \(w\). Define also the characteristic function \(I(x)\). The modified signed distance function \(d_w(x)\) (5.20), can be computed by

\[
d_w(x) = (2I(x) - 1)\tilde{d}_w(x).
\]  

(5.35)
where $I(x)$ is the characteristic function. The algorithm for defining $I(x)$ was described in section 3.2.1.

In the computation of $d_w(x)$, we first compute $\hat{d}_w(x)$, where $\hat{w} > w$. The reason for choosing $\hat{w} > w$ will become clear in the description of the algorithm. The calculation of closest distance is made with an algorithm that is second order accurate.

To calculate the closest distance, we make use of the list defined in section 5.2.3. In this list, each object contains a reference to a segment and integers defining an interval $(p_0, p_1)$ for which the segment should be included. We loop over all such segment parts. To have an overlap between segment parts, we define

$$M = \left\lfloor \frac{1.2\hat{w}}{\Delta x} \right\rfloor + 1,$$

where $\Delta d = \Delta x$ for $x$-segments, and $\Delta d = \Delta y$ for $y$-segments. We set

$$s_0 = \max(k_0, p_0 - M),$$

$$s_1 = \min(k_1, p_1 + M),$$

where $k_0$ and $k_1$ are the integers defining the extension of the discretization of the segment (5.24)-(5.25). Without this overlap, we cannot guarantee an accurate definition for points in the area where we switch segment.

Given a point $(x^*, y^*)$, the first step is to define the closest distance to any discrete point on the part from $s_0$ to $s_1$ of each segment. We then choose the segment that yields the closest discrete distance.

If this distance is large enough, so that the true distance is at least larger than $\hat{w}$, then we set $\hat{d}_w(x) = \hat{w}$. Otherwise, we need to do a more accurate calculation. Assume that the selected segment is an $x$-segment. Denote the closest interval of the selected segment by $x_k, x_{k+1}$. The distance of $(x^*, y^*)$ to this segment is approximated by the distance to the straight line through $(x_k, f_k)$ and $(x_{k+1}, f_{k+1})$. The procedure is analogous if the selected segment is a $y$-segment. This is a second order approximation. Another candidate for computing this distance function is the fast marching method, see e.g. [38].

**Definition of Signed Distance Function**

Once we have the characteristic function $I(x)$, and the unsigned modified distance function $d_w(x)$, we can define the modified signed distance function $d_w(x)$ from (5.35).

In practice, we define $I(x)$ and $\hat{d}_w(x)$ on a regular grid, where the discrete points are based on the 1D discretizations in $x$ and $y$ for the segments. The values of $d_w(x)$ will however be needed in the quadrature points of the elements in the finite element mesh, when the evaluation of integrals containing density and viscosity is made. The values of $d_w(x)$ in these points are obtained by linear interpolation from $d_w(x) = (2I(x) - 1)\hat{d}_w(x)$ as defined on the regular grid.
Therefore, we need $\bar{w} > w$, so that the interpolation is accurate also for values of $d_w(x)$ close to $\pm w$. We use $\bar{w} = w + 1.1\sqrt{\Delta x^2 + \Delta y^2}$.

If there are more than one interface, and we have distance functions $d_{\mu_i}(x)$, $i = 1, \ldots, m$, defined for the distance to these interfaces, then the resulting distance function is given by

$$d_w(x) = \max_i d_{\mu_i}(x).$$  \hspace{1cm} (5.39)

Here, we have used that $d_{\mu_i}(x) > 0$ inside closed structures (containing fluid $A$).

### 5.3 The Discrete Level-Set Method

The level-set function $\phi(x)$ is discretized on a triangulation of the computational domain, using piecewise quadratic polynomials.

#### 5.3.1 Advection and Reinitialization

Two different equations need to be considered for the update of the level-set function $\phi(x)$. First, the equation for advecting the level-set function $\phi(x)$ which is given by (2.11), and second, equation (2.15) which reinitializes $\phi(x)$ to reinforce the distance function property. These level-set equations are discretized using piecewise quadratic polynomials, as described in section 5.1.3.

The two partial differential equations are both of the form

$$\frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c = f. \hspace{1cm} (5.40)$$

This equation has the ordinary weak formulation to find $c(x, t) \in H^1(\Omega)$ such that

$$\left( \frac{\partial c}{\partial t}, v \right) + (\mathbf{u} \cdot \nabla c, v) = (f, v) \quad \forall v \in H^1(\Omega), \hspace{1cm} (5.41)$$

where

$$(f_1, f_2) = \int_\Omega f_1 f_2 \, dx. \hspace{1cm} (5.42)$$

The advection and reinitialization are given by hyperbolic equations. The pure Galerkin formulation of these equations (5.41) does not yield stable numerical methods, and in order to stabilize the formulation, the streamline diffusion method has been used, see e.g. [23]. The streamline diffusion method for (5.40) is as follows: Find $c(x, t)$ in some space defined on $\Omega \times I_n$, $I_n = [t_{n-1}, t_n]$ such that

$$\int_{I_n} (L(c), v + \beta L(v)) \, dt + ([c_{n-1}], v_{n-1}^+) = \int_{I_n} (f, v + \beta L(v)) \, dt, \hspace{1cm} (5.43)$$
where \( L(c) = \frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c \), and \( \beta \) is a parameter to be chosen. Here, the basis functions and test functions are discontinuous in time, and \( v_{n-1}^{+} \) indicates that the value of \( v_{n-1} \) from the time interval \((n-1,n]\) should be evaluated. Approximating these functions as being piecewise constant in time, this yields: Given \( c^{n-1}(\mathbf{x}) \in W_{h} \), find \( c^{n}(\mathbf{x}) \in W_{h} \) such that
\[
(\frac{c^{n} - c^{n-1}}{\Delta t}, v) + (\mathbf{u} \cdot \nabla c^{n}, v + \beta \mathbf{u} \cdot \nabla v) = (f, v + \beta \mathbf{u} \cdot \nabla v) \quad \forall v \in W_{h},
\]
(5.44)
where \( W_{h} \) is the space of piecewise polynomials of degree two, defined on a triangulation of \( \Omega \). This method is first order accurate in time. The parameter \( \beta \) is chosen as
\[
\beta = \frac{1}{2\sqrt{(\Delta t)^{-2} + |\mathbf{u}|^{-2} h^{-2}}} = \frac{1}{2\sqrt{(\Delta t)^{-2} + |J^{-1}|^{-2}}},
\]
(5.45)
where \( J^{-1} \) is the inverse Jacobian for the mapping from the reference element, defined in (5.14).

This is an implicit scheme. Since the stability requirements do not pose very restrictive limitations on the time step, we have chosen to do this explicitly. After each advection step, we perform \( M \) reinitialization “time”-steps, in artificial time \( \tau \). In algorithmic form, this reads: Given a velocity field \( \mathbf{u} \in \mathcal{V}_{h} \) and \( \phi^{n-1}(\mathbf{x}) = \phi(\mathbf{x}, t_{n-1}) \in W_{h} \), find \( \phi^{n}(\mathbf{x}) \in W_{h} \) such that
\[
(\frac{\phi^{n} - \phi^{n-1}}{\Delta t}, v) + (\mathbf{u} \cdot \nabla \phi^{n-1}, v + \beta \mathbf{u} \cdot \nabla v) = 0 \quad \forall v \in W_{h},
\]
(5.46)
Set \( \psi^{0}(\mathbf{x}) = \phi^{0}(\mathbf{x}) \), and for \( m = 1, \ldots, M \), solve
\[
(\frac{\psi^{m} - \psi^{m-1}}{\Delta \tau}, v) + (\epsilon \nabla \psi^{m}, \nabla v) =
- (\mathbf{w}^{m-1} \cdot \nabla \psi^{m-1}, v + \beta \mathbf{w}^{m-1} \cdot \nabla v) + (S_{\alpha}(\psi), v + \beta \mathbf{w}^{m-1} \cdot \nabla v)
\quad \forall v \in W_{h}.
\]
(5.47)
Set \( \phi^{n} = \psi^{M} \).

The quantity \( \mathbf{w} = S_{\alpha}(\psi_{0})|\nabla \psi|/|\nabla \psi| \) (2.16). For numerical purposes, the sign function \( S(\psi) \) has been replaced by a smooth approximation \( S_{\alpha}(\psi) \) given by
\[
S_{\alpha}(\psi) = \frac{\psi}{\sqrt{\psi^{2} + \alpha^{2}}},
\]
(5.48)
The parameter \( \alpha \) is on the order of the grid size. Because of this smoothing, \( \mathbf{w}(\mathbf{x}) \) will not be a unit vector for \( \mathbf{x} \) inside the smoothed zone of the sign function, where its magnitude will be given by \( |S(\psi_{0})| < 1 \).
Natural boundary conditions are imposed on the boundaries for the advection. This would normally cause disturbances at the boundaries if only the advection equation was solved, but the reinitialization procedure propagates such disturbances out of the domain, as long as w points outwards at the boundaries. There are cases where this is not true. One simple example is when \( \phi(x) \) is the distance function to a flat horizontal surface, in which case \( w \) is parallel to the vertical boundary. In cases like this, a small modification to \( w \) is added close to the boundary, to ensure that information is propagated out of the domain. If this is ensured, disturbances at the boundaries are not a problem for this coupled advection-reinitialization procedure. At inflow boundaries, the amount of reinitialization needed in each time step to keep the level-set function free from disturbances, depends on the inflow velocity and the size of the advection time step.

In the reinitialization equation (5.47), extra numerical diffusion has been added \( (\epsilon > 0) \). This is needed to stabilize the calculations, since the streamline diffusion modification gives an insufficient diffusion effect close to the zero contour, where \( S(\psi_0) \) is small, and where \( w \) therefore is small in magnitude. As described above, reinitialization is performed after every advection time step. This reinitialization does however not need to be done until a steady state is reached, since we only need for the level-set function to be a distance function in a neighbourhood of each zero level-set. There are two main reasons to keep down the number of reinitialization steps. First, it obviously increases the computational cost. Second, with more reinitialization time steps, more diffusion effects will accumulate, altering the level of \( \phi(x) \). Such a change affects the conservation of mass, i.e. the conservation of the area fractions of fluid A and B, defined by the positions of the zero level sets.

### 5.3.2 Evaluating the Interfacial Force Term.

The interfacial force term in (3.11),

\[
f_s(v) = \sigma \int_{\gamma} \kappa \hat{n} \cdot v \, d\gamma,
\]

is evaluated through a local process. An element will yield a non-zero contribution to this term only if some part of a zero level set of \( \phi(x) \) (i.e. a part of \( \gamma \)) is intersecting the element.

For each element, the values of \( \phi(x) \) at the six nodes determines whether the element is intersected by the zero contour or not. If it is intersected, then the element is split into four sub-elements, and a piecewise linear approximation of the zero level set (linear in each sub element) is defined locally on the present element. The curvature \( \kappa \), normal vector \( \hat{n} \) and the basis functions are evaluated at the end points of each linear part, and the trapezoidal rule is used on each such piece to evaluate the integral. As two interfaces separating fluids A and B get very close, i.e. when two zero level sets get closer than \( \Delta \varepsilon/2 \), \( \Delta \varepsilon \) being
the element size, they might no longer be detected by this procedure. There will always be a discrete effect for when this will happen, since it depends on the positions of the interfaces relative to the discrete nodes. The merging of interfaces should however happen at this time too, since the two zero contours can no longer be resolved.

The curvature is given by (3.24), and it is needed in points around the zero level set. There are difficulties associated with these calculations, emerging from the form of \( \phi(x) \). As an example, assume that \( \phi(x) \) is the distance function to a circle with radius \( a \) with \( \phi(x) > 0 \) inside the circle. This corresponds to a circular bubble or drop of fluid \( A \) immersed in fluid \( B \). Then \( \phi(x) = a \) in the center of the circle, and the gradient of \( \phi(x) \) is discontinuous at this point, and the curvature is singular. The oscillations that this discontinuity cause in the curvature spread over the elements adjacent to the discontinuity. As long as the radius \( a \) is large enough compared to the element size, the curvature at the zero contour of \( \phi(x) \) will not be affected by these oscillations.

There is another effect arising from this discontinuity in the derivative. As \( \phi(x) \) is advected, it will create some high frequency dispersion errors of a small magnitude. All oscillating components of the error will be magnified when derivatives are calculated. To avoid this, the high frequencies in \( \phi(x) \) is filtered out, and (3.24) is applied to that filtered function. We calculate

\[
(\tilde{\phi}, v) + (\epsilon \nabla \tilde{\phi}, \nabla v) = (\phi, v) \quad \forall v \in W_h, \tag{5.49}
\]

and from here,

\[
\mathbf{n} = \frac{\nabla \tilde{\phi}}{|\nabla \tilde{\phi}|}, \quad \kappa = -\nabla \cdot \mathbf{n}. \tag{5.50}
\]

This procedure effectively filters out high frequencies from \( \phi(x) \) to create \( \tilde{\phi}(x) \), and the curvature calculated from \( \tilde{\phi}(x) \) is much smoother than the curvature calculated directly from \( \phi(x) \). The function \( \tilde{\phi}(x) \) is only used as a step in the curvature calculations, and not elsewhere.

The curvature calculations are discussed more in detail in section 6.3.1.

### 5.3.3 Density and Viscosity Fields

The density and viscosity fields can be defined using their values in the different fluids, together with the characteristic function \( I(x) \). In the end of section 5.1.3, the modified characteristic function \( I_w(x) \) (5.20) was introduced. It was defined as \( I_w(x) = H_w(d(x)) = H_w(d_w(x)) \), where \( d_w(x) \) gives the signed closest distance to a zero contour if that distance is less than \( w \), and \( \pm w \) outside of that region.

As in the segment-projection method, we use the transition polynomial

\[
\rho(\xi) = \frac{1}{2} + \frac{1}{32} \left( 45 \xi - 50 \xi^3 + 21 \xi^5 \right), \tag{5.51}
\]
and define $H_w(t)$ through (5.18).

In order to use $H_w(\phi(x))$ for $H_w(d_w(x))$, where $\phi(x)$ is our level-set function, it is important that it closely approximates a distance function in transition zones, i.e. up until a distance of $w$ on both sides of each zero contour. Also for this reason, the reinitialization procedure described in section 5.3.1 is important.

### 5.3.4 Advection and Reinitialization on a Refined Grid

The most time consuming part of the calculations is the solution of the Navier-Stokes equations. The calculations for advection and reinitialization of the level-set function $\phi(x)$ only take a fraction of the total computer time for each time step, and this fraction decreases as the number of unknowns grows.

Therefore, if higher resolution of $\phi(x)$ is wanted, in order to resolve small scales better in a merging process or to increase the quality of the curvature calculations, only the resolution in the advection and reinitialization is increased. This requires much less work than also increasing the resolution in the solution for the velocity field. Of course, this increase in the resolution of $\phi(x)$ is however much more expensive than a comparable increase in the resolution of the one-dimensional interface representations in the segment projection and front-tracking methods.

In addition to the mesh on which we solve the Navier-Stokes equations, we define a refined mesh, which is obtained by regular subdivision of the first mesh, i.e. by splitting each element into four sub-elements. From these meshes, using quadratic basis functions, we define the spaces $W_h$ and $W_{h/2}$. Initially, we define the velocity field $u \in W_h \times W_h$ and the distance function $\phi_I(x) \in W_{h/2}$. The procedure for advection and reinitialization can then be described as follows:

Given $u \in W_h \times W_h$, define $u_I \in W_{h/2} \times W_{h/2}$ by $u_I(x^I) = u(x^I)$, for all nodes $x_i$ in the fine mesh. In the cases where $x_i$ does not coincide with a node on the coarse mesh, the basis functions for $W_h$ are used for evaluation.

The level set function $\phi_I^{n-1}(x) \in W_{h/2}$, is then advected with $u_I$, and reinitialized, as given by (5.46)-(5.47), with $W_h$ replaced by $W_{h/2}$. This defines $\phi_I^n(x) \in W_{h/2}$.

To define $\phi^n(x) \in W_h$ from $\phi_I^n(x) \in W_{h/2}$, we perform a $L_2$-projection. Formally, this reads: Find $\phi^n(x) \in W_h$ such that

$$
\int_{\Omega} \phi^n(x) \nu \, dx = \int_{\Omega} \phi_I^n(x) \nu \, dx \quad \forall \nu \in W_h.
$$

The refinement of the level-set function calculations affects the velocity calculations by the increased accuracy of the force term $f_g(\nu)$. The evaluation of this term was discussed in the previous section. The gain of an increased resolution of $\phi(x)$ is both that the zero level sets of $\phi(x)$ are determined with higher resolution and that the curvature is more accurately calculated. In addition, less diffusion is needed to stabilize the reinitialization procedure (5.47) when the calculations are done on the finer mesh, and it is possible to obtain a better conservation of mass.
5.4 The Discrete Front-Tracking Method

5.4.1 Discretization and Advection

As was described in section 2.3, a set of discrete points \( \{x(l)^i\}_{i=1}^{N_l} \) are introduced to represent each interface separating fluid \( A \) and \( B \). To complete the description, we need a parametric description connecting these points. Using an independent parameter \( s \), \( x(s) \) and \( y(s) \) are defined through a spline-fitting algorithm. This gives an approximation to the interface in parameterized form \( \gamma_s = (x(s), y(s)) \).

The spline is a piecewise polynomial curve, with third order polynomials defined on each interval \( (x(l)^i, x(l)^{i+1}) \). The polynomials are constructed in such a way that the spline, as well as its first and second derivatives, are continuous at all points on the curve. The coefficients for these local third-order polynomials are obtained by solving a system of linear equations. For a closed curve, the system matrix is essentially tridiagonal, but with additional non-zero entries in the corner positions. Exploring the Sherman-Morrison-Woodbury formula (cf. Golub and van Loan [17]), the solution of the linear system can be obtained by an algorithm which includes solving a tridiagonal system with two different right-hand sides. Thus, the system can be solved in \( O(N_l) \) operations.

To yield a good parameterization \( (x(s), y(s)) \) of the curve, the parameter \( s \) must reflect the distance along the curve. Therefore, we want \( s \) to be the arclength of the curve. In an iterative process, we define new spline approximations, with \( s \) defined using the arclength computed from the previous one. The iteration is terminated when the difference between consecutive iterates is small enough.

The discrete interface points are advected by the flow, as was given in (2.18). To define the velocities \( u(l)^i = u(x(l)^i) \) at the discrete interface points, the basis functions defined on the finite element mesh are used to evaluate the vector value of the velocity field at each \( x(l)^i \). In general, \( x(l)^i \) is not the position of any vertex in the triangulation of \( \Omega \).

The time-stepping scheme chosen is based on the implicit Crank-Nicolson scheme,

\[
x(l)^{n+1} = x(l)^n + \frac{\Delta t}{2}(u(l)^n + u(l)^{n+1}), \quad l = 0, \ldots, N_l,
\]

(5.53)

where the second index denotes the time levels, and \( u(l)^{n+1} = u^{n+1}(x(l)^{n+1}) \). In the term \( u^{n+1}(x(l)^{n+1}) \), the interface point positions are unknown \( a \) priori, so this implicit scheme cannot be used as it is written. Instead, an iterative procedure is used. Introducing a subscript iteration index, the initial value is defined as

\[
x_0(l)^{n+1} = x(l)^n,
\]

(5.54)

and, for each \( l = 0, \ldots, N_l \), from \( k = 0 \) to convergence, the iteration

\[
x_{k+1}(l)^{n+1} = x(l)^n + \frac{\Delta t}{2}(u(l)^n + u^{n+1}(x_k(l)^{n+1}))
\]

(5.55)
proceeds to find the new positions of the interface nodes. In each iteration, when
the positions of the interface nodes have been corrected, the velocity field \( u^{n+1} \)
is used to evaluate the velocities at these updated positions if another iteration
is to be made. The convergence criterion is \( \max_t \| x^{(l),n+1}_k - x^{(l),n+1}_k \| < \epsilon \), for an
appropriate choice of \( \epsilon > 0 \). When convergence is reached, set \( x^{(l),n+1}_k = x^{(l),n+1}_k \),
for \( l = 0, \ldots, N \). After the advection of the points is completed, a new spline
fit is calculated. This time-stepping scheme has been found to provide a good
conservation of mass for the two fluids.

In this advection procedure, each discrete interface node is individually ad-
vected by the local flow. No restrictions are made on the movement of the points.
Depending on the flow, points might cluster at parts of the interface, while other
parts might get depleted of points. Two different algorithms have been used to
keep the discrete points more evenly spaced along the curve. The first approach
is local. A point is added in between two others if the spacing is determined to be
too coarse, and a point that is closer than some minimum distance to its neigh-
bors will be removed. The second approach is more global. A specified number
of points are distributed uniformly along curve, using the spline representation
to position the points.

These “repointing” and “redistribution” algorithms have both been used.
Since the first algorithm does not redefine all interface nodes, but just adds or
deletes points in regions as needed, this algorithm has been used more often than
the latter, minimizing the interpolation errors made. Depending on the flow, it
might however be useful to do a full repositioning of the nodes at some point.

The number of interface nodes can be chosen independently, but it is best to
set their number such as to roughly match their spacing to that of the nodes in
the background mesh. As the interface deforms, surface tension forces will adjust,
and the flow will respond to these forces. It is obvious that too few points, i.e. a
course discretization of the interface would cause a loss of information. On the
other hand, with a spacing of the interface nodes much less than the resolution
of the background mesh, the interface representation is given the freedom to
introduce details on a scale so small that the flow cannot respond properly.

### 5.4.2 Evaluating the Interfacial Force Term.

For simplicity of notation, we assume that there is only one interface separating
fluid \( A \) and \( B \). Using the spline approximation \( \gamma_f = (x(s), y(s)) \) of the inter-
faced \( \gamma_f \), the form \( f_f(s) \), as expanded in \( (3.27) \), can be evaluated by splitting
the spline into its segments and calculating the integral over each of these segments.
Denoting the number of segments by \( N_{y,f} \), this yields

\[
    f_f(s) = \sigma \sum_{i=1}^{N_{y,f}} \sum_{j=1}^{N_{x_i}} \int_{x_j}^{x_{j+1}} \kappa(s) \tilde{n}_f(s) |v_f(x(s)) q(s)| ds, \tag{5.56}
\]

where \( N_{y,f} = N_f \) and \( x(s_{N_{y,f}+1}) = x(s_1) \) if \( \gamma_f \) is closed. If \( s \) is close to
the arclength of the curve, then \( q(s) \) should be approximately 1. Each of these
segment integrals are evaluated using a three-point Gauss quadrature formula (Appendix A). Since we do not break integration intervals at intersections with element edges in the finite element mesh, the presence of the test functions \( \mathbf{v}(x) \) will reduce the order of accuracy for the quadrature process.

The product \( \kappa(s) \hat{\mathbf{n}}(s) \), given by (3.29), can be unambiguously calculated at any point along the interface since the second derivatives \( x''(s) \) and \( y''(s) \), determined from the local third degree polynomials, are continuous along the curve.

When \( \gamma \) is a closed curve enclosing fluid \( A \), the area \( A_A \) occupied by this fluid can be calculated as

\[
A_A = \int_{\Omega} I(x) \, dx = \frac{1}{2} \int_{\gamma} x \, dy - y \, dx = \frac{1}{2} \int_{s_{min}}^{s_{max}} (x(s)y'(s) - y(s)x'(s)) \, ds,
\]

The center of mass \( \mathbf{x}_c = (x_c, y_c) \) can then be calculated as:

\[
x_c = \frac{1}{A_A} \int_{\Omega} x I(x) \, dx = \frac{1}{2A_A} \int_{\gamma} x^2 \, dy = \frac{1}{2A_A} \int_{s_{min}}^{s_{max}} x(s)^2 y'(s) \, ds,
\]

\[
y_c = \frac{1}{A_A} \int_{\Omega} y I(x) \, dx = -\frac{1}{2A_A} \int_{\gamma} y^2 \, dx = -\frac{1}{2A_A} \int_{s_{min}}^{s_{max}} y(s)^2 x'(s) \, ds.
\]

The characteristic function \( I(x) \) was defined in (3.18). The practical evaluation of these integrals is made as described above.

### 5.4.3 Density and Viscosity Fields

The algorithm for defining the characteristic function \( I(x) \) was given in section 3.2.3. In practice, the exact solution of the minimization problem (3.25) is however usually not needed for determining \( I(x) \).

The point \( \mathbf{x}^* \) on the curve \( \gamma \), closest to the point \( \mathbf{x}_0 \), only needs to be defined accurately enough to produce the correct sign of \( \mathbf{n}_f(x^*) \cdot \mathbf{x}_d \). Therefore, an \( \mathbf{x}^* \) is defined as the discrete interface node that is closest to \( \mathbf{x}_0 \). With \( \mathbf{x}_d = \mathbf{x}_0 - \mathbf{x}^* \), we evaluate

\[
\eta = \mathbf{n}_f(\mathbf{x}^*) \cdot \frac{\mathbf{x}_d}{||\mathbf{x}_d||},
\]

If the absolute value of \( \eta \) is large enough, this approximation will give the correct \( I(x_0) \), and there is no need to proceed further. On the other hand, if the absolute value of \( \eta \) is less than some \( \eta_0 \), then \( x_0 \) is close to the curve, with \( \mathbf{x}_d \) almost tangential to the curve, and \( \mathbf{x}^* \) is not a good enough approximation to \( \mathbf{x}^* \) to ensure a correct definition of \( I(x_0) \).

Then, a finer search is performed, splitting the spline segment where the closest point is located into four intervals, for a more accurate determination of \( \mathbf{x}^* \). This refined search continues until \( \eta > \eta_0 \). Once a sufficiently large \( \eta \) is obtained, the closest distance between the point and the curve can be taken as \( ||\mathbf{x}_d|| \).
5.5 Interaction Between Interfaces

The characteristic function \( I_w(x) \) is not regularized in the same way as in the segment-projection and level-set methods. Instead, this characteristic function simply varies linearly over a thin transition zone, with a width comparable to some fraction of the node spacing in the mesh. This is done to avoid a discrete change in the integral evaluations as a quadrature point \( x_q \) very close to the interface falls in one fluid in one time step, and in the other fluid in the next. In this way we avoid the need to accurately determine the distance to the interface for all quadrature points in the elements close to the interface. However, the error in the integral evaluation will of course be larger compared to the error if a more careful smoothing was applied, as discussed in chapter 4.

In that framework, this can be seen as performing the calculations with a small \( w \), for the modified characteristic function \( I_w(x) = H_w(d(x)) \), with \( H_w(t) \) defined through (5.18), using the linear transition polynomial

\[

v(\xi) = \frac{1}{2}(1 + \xi).

\]

Only the zeroth moment is vanishing for this choice \((m = 0)\), and the first derivative is discontinuous \((k = 0)\).

5.5 Interaction Between Interfaces

In different applications, there might be different rules for interface interaction. In the case of multiphase flow, two bubbles might merge, or a bubble might burst through a surface. But for some physical parameters, there might not be any merging, even though two volumes of the same fluid (e.g. two bubbles or drops) get very close.

If we do not want to allow merging in a simulation, nothing special needs to be done in the segment projection and the front-tracking methods. Each interface is defined as a separate structure, and can get arbitrarily close without merging. In the level-set method, two interfaces are described as two different zero contours of the same level-set function, and these interfaces will always merge once they get close enough, relative to the spatial resolution of the mesh where the level-set function is defined. At this point, the two different zero contours can no longer be resolved, and the part of the zero contours that are closest together will vanish. The two interfaces will be connected at both sides of this zone, merging into one single interface. To avoid the merging between two interfaces, the algorithm must be modified. The interfaces can for example be defined as the zero level sets of two different level-set functions.

For the same reasons, the level-set method most naturally handles problems where mergers do occur. Once two interfaces get close enough, a part of the zero contour of both interfaces vanishes, and they reconnect. This is only seen as a change in values of the level-set function.

In both the segment projection and the front-tracking methods, each interface is described as a separate structure, which is convenient for non-merging
interfaces. A merging of interfaces on the other hand means that modifications of these structures are needed. This is rather complicated in the front-tracking method, where the discrete points are Lagrangian markers which, during the simulation, move in both spatial directions. In the segment projection method, this is not as complicated thanks to the uniform one-dimensional Eulerian discretizations in the $x$ and $y$-coordinate, respectively. See [13] for a presentation of different front-tracking methods.

For the above mentioned reason, we have not implemented any merging algorithm in the front-tracking method, but we have done so in the segment projection method. The algorithm for merging interfaces in the segment projection method is described in the next section and some comments regarding the level-set method are given in section 5.5.2.

### 5.5.1 Merging Interfaces for the Segment Projection Method

When two interfaces are advected by the same flow field, they can never overlap. Furthermore, when the calculations are performed discretely, the resolution of the calculations will determine how close they can actually get. Once two interfaces are close enough, relative to the resolution, they will move with essentially the same speed, and will not get any closer together.

Therefore, in our calculations, we set a parameter $\tilde{d}$ as the given distance for when two structures merge. To determine if there should be a merger, we compare the $x$-segments with each other, and the $y$-segments with each other, so that the segments that are compared are discretized at the same points. If the difference is less than $\tilde{d}$ for three consecutive points, the segments are determined to be close enough for a merger to take place. The parameter $\tilde{d}$ is chosen proportional to the resolution of the background mesh, usually somewhat smaller than the size of a typical triangle in the mesh. It is possible to replace or complement this distance condition with some other physical condition.

If a merger is indicated both from segments discretized in the $x$-direction and segments discretized in the $y$-direction, we choose the ones that have smallest slope in the merging region to be the reference segments. The segments in the other direction will be the dependent segments. For simplicity of description, assume that the reference segments are the $x$-segments.

When the interval where the $x$-segments should merge is determined, we need to arrange the segment definition so that we have two ends of $x$-segments on each side of the gap, both for the lower and the upper structures. Before the merger, an $x$-segment might be defined over the merge interval and continue on the other side, in which case it needs to be split into two, leaving a gap over the merge interval. If this is not the case, then we instead have a break point inside the gap, and two $x$-segments meet inside the gap. In this case, we just need to redefine the end point of these segments. Figure 5.2 shows the simple segment structure before the merging of two circles. Now, we have four $x$-segments ending at the
5.5. Interaction Between Interfaces

Two segments from the upper and lower structures on the left side, and two segments on the right side of the gap.

Physically, when bubbles merge, we have a singularity at the merging point, where the surface tension forces are infinite. This stage is passed over in an infinitesimal short period of time, resulting in a curve that is smooth, but which still has a high curvature. Therefore, we make no attempt to model this phenomenon with our limited (discrete) resolution. We make our structure smooth by inserting an arc of a circle in the reconnecting zone. The radius of this circle is chosen considering the size of the smallest scales that can be resolved on the background mesh. The resolution of the segment discretization should be so much finer than the resolution of the background mesh, so that such a circle arc can be discretized.

The insertion of the arc has to be done so that the curve is continuous and smooth. Therefore, we start at the upper \( x \)-segment at a certain point, continue with a tangent that has the same slope as the segment at this point, connecting with the arc of the circle such that we get a continuous derivative. At the maximum \( x \)-value of the circle, the definition of this segment ends. The definition of the end of the lower \( x \)-segment is done analogously, and the two \( x \)-segments meet at a point on this arc.

Given a point \( x_{h} \) on the lower \( x \)-segment and a point \( x_{k} \) on the upper \( x \)-segment on the same side of the gap, the two tangents, starting at points \((x_{h}, f_{h})\) and \((x_{k}, f_{k})\), will intersect at some point \((x^{*}, y^{*})\), assuming that they are not parallel. Define a straight line through \((x^{*}, y^{*})\), splitting the angle between the two tangents in half. Then, we call the orientation of the points \((x_{h}, f_{h})\) and \((x_{k}, f_{k})\) orthogonal to the gap, if a straight line through these points is perpendicular to the line through \((x^{*}, y^{*})\) that we just defined, or, in the case where the tangents are parallel, perpendicular to those.

Before the insertion of the circle arc is made, we apply an iterative procedure to find the points at the two \( x \)-segments from which the tangents of the curve should connect to the circle arc. We want the orientation of these points to be orthogonal to the gap in the sense described above. The points are only accepted if they will yield a radius of curvature of the inscribed circle arc in the predetermined range.

Once the insertion of the circle arc is made in the reference segments, we need to update also the segments in the opposite direction. In the two regions where the circle arcs have been inserted, all the values must be redefined. It might be that there are no extrema in the two reference segments in this region where we have inserted the circle arc. If this is the case, the two segments in the opposite direction can merge into one. In the general case, there will however be two \( x \)-segments and two \( y \)-segments meeting in this region. The number of points in the overlap of the \( x \)-segments and \( y \)-segments depend on the resolution of the segment discretizations \( \Delta x \) and \( \Delta y \) relative to the radius of curvature of the inserted circle arc. This relation needs to be such that there is at least two to three points overlap. After these modifications are made, we need to
Figure 5.2. Before merging. There are two segments in the $x$-direction and two segments in the $y$-direction for each interface.

Figure 5.3. After merging. The two structures have merged into one. There are 6 segments in the $x$-direction and 2 segments in the $y$-direction.
Figure 5.4. At a later stage, the structure has become smoother, and there are only 2 segments in both directions.

Figure 5.5. The bubble has become more deformed. New segments have been added in the y-direction. There are now 2 segments in the x-direction, and 4 segments in the y-direction.
update the structure of the segments in this region, and set correct references to corresponding segments in the opposite direction.

Thereafter, the calculation proceeds as usual, with two previously separate structures merged into one. In Figure 5.3, the situation is shown a while after the merger of the structures plotted in Figure 5.2. Figures 5.4 and 5.5 are included to indicate how the distribution of segments can change as this simulation is allowed to proceed.

### 5.5.2 Merging Interfaces for the Level Set Method

In the level-set method, no explicit distance between interfaces, determining when merging should occur, is given. Implicitly, there is however such a distance, that depends on the resolution of the mesh on which the level-set function is defined. The resolution determines the smallest distance between two zero level sets of the level-set function for which they can still be distinguished as separate zero contours.

When the interfaces get close, there is only a thin region where the level-set function is of different sign, such that the two zero contours are still separated. An effect that can make interfaces merge faster is therefore the diffusivity of the numerical methods used for advection and reinitialization of the level-set function.

In practice, this yields a minimum distance \( \tilde{d} > 0 \) between two interfaces, for which they can stay separate. For any distance smaller than \( \tilde{d} \), the two interfaces will merge. This \( \tilde{d} \) will mainly depend on the resolution, directly and through the diffusion, but can also be somewhat affected by other factors, such as the curvature of the interfaces in the region where they get close.

### 5.6 The Algorithm in Summary

The algorithm has been presented in detail in the previous sections. Here, a short summary is made, in order to emphasize the overall structure. For the parts where the calculations differ according to which of the three methods that has been used, we indicate the remarks by an \( S \), \( L \) or \( F \), for the segment projection, level-set and front-tracking methods, respectively.

0. Initially, assume that for the description of \( \gamma \), the union of all interfaces,

\[ S \) the segment discretization,  
\[ L \) the level-set function \( \phi(x) \),  
\[ F \) the front-tracking discretization,

is known, either from initial conditions or the previous time step.
The algorithm can be described as follows:

i) Compute the interfacial force term and the density and viscosity fields.

S) Evaluate \( f(rv) \) using the segment discretization, as discussed in section 5.2.3. The density and viscosity fields are defined through 
\[ I_w(x) = H_w(d(x)), \]
where the computation of \( d(x) \) is discussed in section 5.2.4.

L) Evaluate \( f_r(v) \) using the level-set function \( \phi(x) \), as discussed in section 5.3.2. The density and viscosity fields are defined through 
\[ I_w(x) = H_w(\phi(x)), \]
as described in section 5.3.3.

F) Evaluate \( f_n(v) \) using the spline parameterization \( s \), as discussed in section 5.4.2. The density and viscosity fields are defined based on a
modified indicator function \( I_w(x) \) defined by using the spline
representation \( s \) to determine the closest distance to any interface
(section 5.4.3).

ii) With the information from i) available, solve the time discretized
Navier-Stokes equations (5.5) through an iterative procedure, as given in
(5.6), to obtain the velocity at the next time level. The contributions
from i) are included in these equations.

iii) Adveqt and reinitialize the interface representation.

S) Adveqt the segments in the discretization by the velocity field (5.28).
Go through the reinitialization procedure, as described in section
5.2.2, to update the segment structure.

L) Adveqt the level set function \( d(x) \) by the velocity field (5.46). Do
one or more reinitialization step(s) as defined in (5.47), to restore the
distance function property in the region close to the zero level set.

F) Adveqt the discrete interface points, by the local velocities, as
defined in (5.55). Calculate a new spline fit for each interface, based
on these updated positions. If needed, modify the distribution of
points either by the “repointing” or “redistribution” algorithms,
described in section 5.4.

iv) This completes one time step, repeat i)-iii) to advance further.

If mergers are allowed, there is one more step added in the segment projection
method: After each advection step, check if any mergers should occur. If so, use
the merging algorithm described in section 5.5.1. No merging algorithm has
been implemented for the front-tracking method, and for the level-set method
no specific algorithm is needed, see section 5.5.2.
Some comments regarding ii):

1. Equation (5.5) is obtained by discretizing the variational form of the Navier-Stokes equations (3.11) using a combined implicit-explicit time-stepping scheme. The solution of this equation is obtained by performing fixed-point iterations, until convergence is reached. After each fixed-point iteration (5.6), a temporary advection of the interface representation needs to be done, in order to update the density and viscosity fields included in the left hand side of the equation.

2. In each fixed-point iteration (5.6), a Stokes-like problem is solved. To enforce the incompressibility constraint, the Stokes-like problem is reformulated using an iterated penalty method (5.10).

3. This iterated penalty method is in itself an iterative procedure. With a large enough penalty parameter, it converges within a few iterations. In each penalty iteration, a symmetric system of linear equations must be solved.

4. This system of linear equations is solved using a preconditioned conjugate gradient method.

The basic tools for using high-order finite elements are available in the library *Albert*, written in C++. Some ideas for the implementations in this library were discussed in [2]. For example of numerical simulations performed with this software, see [35].

Many of the features used in our algorithms, such as the streamline diffusion modification, evaluation of the interfacial force term and evaluation of integrals with non-constant density and viscosity are however not available in this library. The software for these algorithms have been written in C++. All simulations presented in this paper have been performed on different Sun workstations, and typical CPU times are 10 – 20 hours.
Chapter 6

Numerical Results

6.1 Convergence Results for a Buoyant Bubble

In this section, we perform runs on three consecutively refined meshes, to test the spatial convergence of the three methods. The runs we make are simulations of a single buoyant bubble, rising in an initially quiescent fluid due to the effects of buoyancy.

This problem can be characterized with a few non-dimensional parameters together with the geometry of the problem. In addition to the viscosity and density ratios, $\mu_B/\mu_A$ and $\rho_B/\rho_A$, we use the Morton number $M$ and the Eötvös number $Eo$. The Morton number and the Eötvös number are defined as

$$M = \frac{\mu_B^4}{\rho_B \sigma^3}, \quad Eo = \frac{\rho_B \rho d^2}{\sigma}.$$  \hspace{1cm} (6.1)

Here, $\mu_B$ and $\rho_B$ denote the viscosity and density of the outer fluid, respectively. $\sigma$ is the gravitational constant and $\sigma$ is the surface tension coefficient. The length $d = 2\sqrt{A/\pi}$, where $A$ is the area of the bubble. For an initially circular bubble, $d$ is the diameter of the bubble. The Eötvös number is also called the Bond number, and is sometimes alternatively defined with the density difference between the two fluids instead of the density of the outer fluid. The typical initial configuration for a single buoyant bubble is shown in Figure 6.1.

The Reynolds number, $Re = \rho_B Ud/\mu_B$, will in this case be a dependent parameter, based on the rise velocity $U$ of the bubble, and might vary with time. The same is true for the Weber number, $We = \rho_B U^2 d/\sigma$, another parameter often used in problems of this kind. We also introduce $Re_\infty = \rho_B U_\infty d/\mu_B$, where $U_\infty$ is the maximum velocity in the domain. Due to the time-dependence of the velocity, it is difficult to give a time scale for this problem, based on the length and velocity scales. We therefore use a viscous time scale $\frac{\rho_B d^2}{\mu_B}$ to normalize the time.

For the simulations in this section, we use a regularly subdivided mesh on $[-1, 1] \times [0, 3]$, with interval size $\Delta e$. The triangles in the mesh will be isosceles
right triangles with the longest side $\sqrt{2}\Delta e$. Since we use quadratic basis functions, the distance between two neighbouring nodes in the mesh is $\Delta x = \Delta e/2$. We have used three meshes with $\Delta e = h$, $h/2$ and $h/4$, where $h = 0.2$. Periodic boundary conditions are imposed for the velocity field in the horizontal direction, and no-slip conditions are imposed on the top and bottom walls.

The time-stepping in the coupled problem includes a splitting between the calculation of the velocity field and the subsequent advection of the interfaces, either represented by the segment discretization, the level-set function $\phi(x)$ or by the front-tracking discretization. This gives a first order effect, and the total order of convergence of the time stepping cannot be expected to be fully second order.

Two sets of runs are presented for each method. In the first sequence, we divide the time step size by a factor of four, as we reduce the mesh size with a factor of two. With $\Delta t_0 = 5 \cdot 10^{-3}$, we use $\Delta t = \Delta t_0$ for $\Delta e = h$, we use $\Delta t_0/4$ for $\Delta e = h/2$ and $\Delta t_0/16$ for $\Delta e = h/4$. In the second set, we reduce both the mesh size and the time step with a factor of two, and use $\Delta t = \Delta t_0/4$ for $\Delta e = h$, $\Delta t = \Delta t_0/8$ for $\Delta e = h/2$ and $\Delta t = \Delta t_0/16$ for $\Delta e = h/4$. That is, the run with the finest resolution is the same in the two sets.

Initially, we start with a circular bubble of fluid $A$ with diameter $d = 1.0$, immersed into fluid $B$, as indicated in Figure 6.1. The velocity is zero $\left( u = 0 \right)$ at $t = 0$. We run the simulations up to $t = 0.15$, where the bubble has moved almost half a diameter and deformed somewhat. At this point we check the convergence.

The physical parameters for these runs are $M = 0.1$, $Eo = 10$, $\rho_B/\rho_A = 100$ and $\mu_B/\mu_A = 2$. The numerical parameters are $h = 0.2$, $\Delta t_0 = 5 \cdot 10^{-3}$. In the segment projection method, we set the resolution of the segment discretizations
to $\Delta x = \Delta y = \Delta e/10$. In the front-tracking method, the initial number of points is set to 50, 100 and 200 points, for the coarsest to finest run. The smoothing parameter $w$ in the Heaviside approximation (5.18), as well as the smoothing parameter for the sign function in the level-set reinitialization procedure (5.48), is set to $\Delta e$.

In the reinitialization procedure for the level-set function, extra diffusion is added (5.47). The coefficient for this diffusion has been kept the same ($\epsilon = 8 \cdot 10^{-3}$) in all three runs, and we study the convergence for this fixed problem.

The results from the runs with the segment projection, level-set and front-tracking methods can be found in Figures 6.2-6.4.

![Figure 6.2](image1.png)

(a) The bubble contours. The dotted contour indicates the initial bubble position.

![Figure 6.3](image2.png)

(b) Velocity field for $\Delta x = h/4$, $Re = 14.55$.

**Figure 6.2.** Results from the segment projection runs at $t = 0.15$. The bubble contours taken from set 1, with $\Delta t = \Delta t_0$ for $\Delta e = h$, $\Delta t = \Delta t_0/4$ for $\Delta e = h/2$, $\Delta t = \Delta t_0/16$ for $\Delta e = h/4$, $\Delta t_0 = 5 \cdot 10^{-3}$, $h = 0.2$.

We define the discrete $L1$-norm of a vector valued function $\mathbf{u}$ as

$$||\mathbf{u}||_1 = \frac{1}{N} \sum_{i=1}^{N} \left( (u_x^i)^2 + (u_y^i)^2 \right)^{1/2},$$

(6.2)

where $N$ is the number of discrete nodes. Denote the velocity fields at time $t = 0.15$ by $\mathbf{u}_h$, $\mathbf{u}_{h/2}$ and $\mathbf{u}_{h/4}$, indexed by the spatial resolution. In Table 6.1 we study the convergence of the velocity fields from the runs in set 1. The difference in $L1$-norm of consecutive velocity fields, and the order of convergence
obtained are given for all three methods. The convergence rate for the velocity field is closest to second order for the segment projection method, and lowest for the level-set method. In Table 6.2, the convergence of the velocity fields for the runs in set 2 is studied. The order \( p \) obtained is slightly lower compared to the results in Table 6.2, where the time step was divided with a factor of 4 as the spatial resolution was refined with a factor of 2. For the level-set method, the obtained order is however slightly higher in this second case.

![Bubble Contours](image1)

![Velocity Field](image2)

**(a)** The bubble contours. The dotted contour indicates the initial bubble position.

**(b)** Velocity field for \( \Delta t = h/4 \).

\( Re_{\infty} = 14.02 \).

**Figure 6.3.** Results from the level-set runs at \( t = 0.15 \). The bubble contours taken from set 1.

The results from these two set of runs indicate that spatial errors are dominating. The time step used is small compared to the element size \( \Delta t \).

It is difficult to point out the different sources of errors in the three methods, since there are many parts put together. Essentially all parts should however be second order accurate in space. This can however not be expected in the curvature calculations in any of the three methods, and the contribution of this effect yield a somewhat lower total order of convergence. The curvature calculations in the level-set method was discussed in section 6.3.1.

It is somewhat surprising that the front-tracking method yields as good convergence rate as it does, given the fact that the integration over the discontinuous density and viscosity fields are less accurate than in the other methods.

This convergence study has been based on the velocity fields. It is more complicated to measure the error in the position of the bubble. In the front-
6.1. Convergence Results for a Buoyant Bubble

Figure 6.4. Results from the front-tracking runs at \( t = 0.15 \). The bubble contours taken from set 1.

Figure 6.5. The velocity field plotted in the reference frame of the bubble. Reynolds number based on rise velocity \( \tilde{R} = 2.33 \). Result taken from finest front-tracking run. \( \Delta t = \Delta t_0/16, \Delta x = h/4 \) with \( \Delta t_0 = 5 \cdot 10^{-3}, h = 0.2 \).
<table>
<thead>
<tr>
<th>Method</th>
<th>( |u_h - u_{h/2}|_1 )</th>
<th>( |u_{h/2} - u_{h/4}|_1 )</th>
<th>( \frac{|u_h - u_{h/2}|<em>1}{|u</em>{h/2} - u_{h/4}|_1} )</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment projection</td>
<td>4.06 \cdot 10^{-2}</td>
<td>1.07 \cdot 10^{-2}</td>
<td>3.78</td>
<td>1.92</td>
</tr>
<tr>
<td>Level-set</td>
<td>3.29 \cdot 10^{-2}</td>
<td>1.03 \cdot 10^{-2}</td>
<td>3.20</td>
<td>1.68</td>
</tr>
<tr>
<td>Front-tracking</td>
<td>3.61 \cdot 10^{-2}</td>
<td>1.04 \cdot 10^{-2}</td>
<td>3.48</td>
<td>1.80</td>
</tr>
</tbody>
</table>

**Table 6.1.** Measured order of convergence \( p \) for the three different methods. Runs from set 1, such that \( \Delta t = \Delta t_0 \) for \( \Delta x = h \), \( \Delta t = \Delta t_0 / 4 \) for \( \Delta x = h / 2 \), \( \Delta t = \Delta t_0 / 16 \) for \( \Delta x = h / 4 \). \( \Delta t_0 = 5 \cdot 10^{-3} \), \( h = 0.2 \).

<table>
<thead>
<tr>
<th>Method</th>
<th>( |u_h - u_{h/2}|_1 )</th>
<th>( |u_{h/2} - u_{h/4}|_1 )</th>
<th>( \frac{|u_h - u_{h/2}|<em>1}{|u</em>{h/2} - u_{h/4}|_1} )</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment projection</td>
<td>3.73 \cdot 10^{-2}</td>
<td>1.04 \cdot 10^{-2}</td>
<td>3.58</td>
<td>1.84</td>
</tr>
<tr>
<td>Level-set</td>
<td>3.28 \cdot 10^{-2}</td>
<td>1.01 \cdot 10^{-2}</td>
<td>3.26</td>
<td>1.70</td>
</tr>
<tr>
<td>Front-tracking</td>
<td>3.38 \cdot 10^{-2}</td>
<td>1.00 \cdot 10^{-2}</td>
<td>3.37</td>
<td>1.75</td>
</tr>
</tbody>
</table>

**Table 6.2.** Measured order of convergence \( p \) for the three different methods. Runs from set 2, such that \( \Delta t = \Delta t_0 / 4 \) for \( \Delta x = h \), \( \Delta t = \Delta t_0 / 8 \) for \( \Delta x = h / 2 \), \( \Delta t = \Delta t_0 / 16 \) for \( \Delta x = h / 4 \). \( \Delta t_0 = 5 \cdot 10^{-3} \), \( h = 0.2 \).

...tracking method, the discrete points describing the contour will be distributed differently in the three runs, and it is difficult to get any meaningful measure of the difference in the bubble position. In the level-set method, the convergence of the level-set function \( \phi(x) \) can be studied. However, the largest difference between consecutive solutions is usually found in the interior of the bubble, around the discontinuity in the gradient of \( \phi(x) \). That is not a good measure of the convergence of the problem, since it is only the zero level set of \( \phi(x) \) that determines the bubble contour.

In the segment projection method, some comparison can be made if we select the parts of the interface from each run that are based on the same discrete \( x \)-points, and the parts that are based on the same discrete \( y \)-points. Each such part from the three runs can be compared, since the values \( f(x_j) \) or \( g(y_j) \) should converge for a given \( x_j \) or \( y_j \). We compare four different parts of the bubble interface for the three resolutions. The norm that has been used is the \( L_1 \) norm.
of the Euclidean distance, and the distance for part of a segment $\mathbf{I}$, with $M$
points $(\ell^i_x, \ell^i_y)$, to part of a segment $\mathbf{I}$ with $M$ points $(\tilde{\ell}^i_x, \tilde{\ell}^i_y)$ is measured by

$$
||\mathbf{I} - \mathbf{I}||_1 = \frac{1}{M} \sum_{i=1}^{M} (\ell^i_x - \tilde{\ell}^i_x)^2 + (\ell^i_y - \tilde{\ell}^i_y)^2)^{1/2}.
$$

(6.3)

For set 1, this yields values for $p$ between 1.88 and 2.05. For set 2, this yields values for $p$ between 1.63 and 2.05, as measured for different segments.

### 6.2 Comparison for Single Buoyant Bubble

The convergence of the methods for the case of a single buoyant bubble was studied in the previous section. The comparison was done at an early time, when the bubble had not risen very far.

Now, we change the mesh to an irregular mesh defined on a taller domain, and let the bubble rise further. The results from the segment projection, the level-set and the front-tracking methods are compared.

The initial configuration is the same as in Figure 6.1, where an initially circular bubble of fluid $A$ with diameter $d = 1.0$ is immersed in fluid $B$. The velocity is zero initially, i.e. $\mathbf{u} = 0$ at $t = 0$. The computational domain is $[-1.5, 1.5] \times [0, 8.0]$, and the mesh is shown in Figure 6.6. It is constructed with finer resolution in the middle of the domain where the bubble rises, and large triangles are used to extend the domain. Periodic boundary conditions are imposed for the velocity field in the horizontal direction, and no-slip conditions are imposed on the top and bottom walls. We use the following physical parameters: $M = 0.2$, $Eo = 10.0$, $\rho_B / \rho_A = 100$ and $\mu_B / \mu_A = 10$. We use the time step $\Delta t = 10^{-3}$. In the simulations presented here, the advection and reinitialization of $\phi(x)$ are made on a refined mesh, as was described in section 5.3.4.

The segment discretization is made with $\Delta x = \Delta y = 0.005$ in the segment projection method. The smoothing parameter in the Heaviside approximation (5.18) is set to 0.1. The results shown for the level-set method are obtained with the time step in the reinitialization set to $\Delta \tau = 2.5 \cdot 10^{-3}$, with the diffusion parameter $\epsilon = 5 \cdot 10^{-4}$, performing two reinitialization time steps for each advection time step. In addition to these regular reinitialization steps, we perform two additional time steps with the sign function in (5.48) multiplied by $H^\alpha_n(-2\alpha - \phi)$, $\alpha = 0.15$, where $H^\alpha_n$ is defined by (4.17) together with (4.43). These boundary reinitialization time steps affects only the level-set function outside of the zero contour, and is made to ensure a smoother level-set function free from boundary disturbances. The smoothing parameter for the sign function in (5.48) is set to 0.075.

The fluid interfaces support dispersive interface waves. These phenomena may introduce a CFL-type restriction on the time step, see [7]. In our computations, other conditions also limit the time step size and we have not seen any instabilities that could be traced to CFL interface conditions.
Figure 6.6. The mesh used in the simulation. The domain is $[-1.5, 1.5] \times [0, 8.0]$. There are 1755 elements in the mesh, with 6 nodes in each element.

In Figure 6.7, the bubble contours from the three different methods are plotted at a sequence of different times. A close up of the contours from the three methods at time $t = 1.25$ is shown in Figure 6.8. The contours from the three methods are in good agreement, both when it comes to position and shape of the contours. The bubble in the level-set simulation has however lost most mass (i.e. area since $\rho_\Lambda$ is constant). At the last time plotted, the decrease compared to the initial area is 2.26%. For the segment projection method, we have a decrease of 0.76%. The front-tracking method yields the best mass conservation for this run, the area is increased by 0.15%.

In general, the mass conservation (or equivalently the area conservation) of the bubble is good for the front-tracking method. It is also in general satisfying for the segment projection method, even though it was not as excellent as for the front-tracking method in this case. In the level-set simulations, it is more difficult to get a good conservation of mass. The area of the bubble will shrink to an extent determined by the numerical diffusion present in the calculations. The main part of this artificial viscosity originates from the reinitialization procedure, and
6.2. Comparison for Single Buoyant Bubble

Figure 6.7. The bubble interface plotted at different times. The dotted contour at the bottom indicates the initial position. Results for all three methods are plotted on top of each other. $M = 0.2$, $Ec = 10.0$, $\rho_B/\rho_A = 100$ and $\mu_B/\mu_A = 10$. $\Delta t = 10^{-3}$.

Figure 6.8. A close up of the bubble interface at $t = 1.25$, the last solution plotted in Figure 6.7.
is determined by the amount of dissipation added in the reinitialization equation (5.47) together with how much reinitialization that is performed. The artificial diffusion needed to stabilize the equations decreases as the spatial resolution is increased, which results in improved mass conservation with reduction in mesh size.

The velocity fields for all methods at this time are shown in Figures 6.9-6.11. The velocity fields have been plotted both in an absolute frame, and in the reference frame of the bubble. The velocity fields are very similar for all three methods. We have \( Re_\infty = 3.93, 3.92 \text{ and } 3.94 \) for the segment projection, the level-set and the front-tracking methods respectively. The difference is a bit larger for the rise velocity based Reynolds number \( Re \). The rise velocities at a certain time are computed by calculating the movement of the center of mass in the last time step.

![velocity fields](image)

(a) \( Re_\infty = 3.93 \).

(b) In the reference frame of the bubble. \( Re = 2.87 \).

**Figure 6.9.** The velocity field at time \( t = 1.25 \) for the segment projection method. Only a part of the domain is shown.

Note that the periodic boundary conditions in the horizontal direction and the symmetry effectively restrict the flow to be vertical at these boundaries. The allowed size of the recirculation zone with fluid turning to move downwards will therefore be limited by the size of the domain. To see the recirculation zones inside the bubble, the velocity field has been plotted in the reference frame of the bubble in Figures 6.9b, 6.10b, and 6.11b. Since the velocities are smaller inside the bubble than outside, we have only plotted the interior flow to create a better visualization. This state is a relatively steady state, i.e. the bubble continues to rise with approximately the same shape and the same velocity. If we study the velocity field at time \( t = 1.25 \) (Figures 6.9-6.11), the maximum velocity and the rise velocity of the bubble is only slightly lower compared to the values at
6.2. Comparison for Single Buoyant Bubble

Figure 6.10. The velocity field at time \( t = 1.25 \) for the level-set method. Only a part of the domain is shown.

Figure 6.11. The velocity field at time \( t = 1.25 \) for the front-tracking method. Only a part of the domain is shown.
$t = 0.5$, where we have $Re_{\infty} = 4.04$, $Re = 2.87$ for the segment projection method, $Re_{\infty} = 4.03$, $Re = 2.85$ for the level-set method, and $Re_{\infty} = 4.05$, $Re = 2.85$ for the front-tracking method.

The results we have presented here show good agreement between all three methods. In order to obtain good area conservation with the level-set method with the present reinitialization technique, high resolution is needed to reduce the amount of artificial viscosity required to keep these calculations stable. The front-tracking method is well suited for this kind of simulations, where a single bubble is treated, and there is no merging or splitting. The area conservation is good, and in these two-dimensional calculations, the insertion and deletion of points along the interface that is needed to keep the interface evenly resolved is quite simple.

The front-tracking method is however not as suitable for simulations including changes in topology, and such an algorithm has not been implemented. For simulations including mergers of bubbles etc, only the segment projection method and the level-set method will be used.

### 6.3 Remarks on Force Calculations in the Level-Set Method

The interfacial force term $f_\gamma(\mathbf{v})$ (3.17) can either be calculated as an integral over $\Omega \subseteq \mathbb{R}^2$, 

$$ f_\gamma(\mathbf{v}) = \sigma \int_{\Omega} (\mathbf{n} \cdot \mathbf{v}) \delta_\gamma \, dx, \quad (6.4) $$

where $\delta_\gamma$ is the Dirac delta function with support on $\gamma$, the union of the interfaces. It can also be evaluated as a line integral along $\gamma$,

$$ f_\gamma(\mathbf{v}) = \sigma \int_{\gamma} \mathbf{\kappa} \cdot \mathbf{v} \, d\gamma. \quad (6.5) $$

The second formulation has been used in all practical simulations, and this algorithm was discussed in section 5.3.2. In the level-set method, it would however be natural to use the first formulation with $\delta_\gamma$ replaced by an approximation $\delta_n(\phi(x))$, since $\gamma$ is not discretized explicitly. Such approximations were derived in the analysis in chapter 4. The problems of using this approach arise mainly due to the non-smoothness of the curvature. In this section, we compare the numerical implementation of the two formulations. We will however begin by studying the effect of advection on the accuracy in the location of the interface $\gamma$ and in the value of the computed curvature $\kappa$. 
6.3. Remarks on Force Calculations in the Level-Set Method

6.3.1 Curvature Calculations

In this section, we study the convergence of the curvature calculations in connection to the advection of the level-set function $\phi(x, t)$. To advect the level-set function, a velocity field $\mathbf{u} = (u, v)$ for rigid body rotation is imposed,

\begin{align}
  u(x, y) &= -c \sin(\theta), \\
  v(x, y) &= c \cos(\theta),
\end{align}

with $c$ defined as $c = 2\pi R / T_{isp}$ with $R = 1.5$ and $T_{isp} = 8.0$. The parameter $\theta$ is the argument of the complex number $x + iy$. The level-set function $\phi(x)$ is initialized as the signed distance function to a circle of radius $a = 0.3$, centered in $(x_c, y_c) = (-1 + R \cos(\pi / 8), -1 + R \sin(\pi / 8))$ ($\phi(x) > 0$ inside the circle). The initial level-set function, together with the time-independent velocity field, has been plotted in Figure 6.12. Regularly subdivided grids on $[-1, 1] \times [-1, 1]$ with $2 / \Delta x$ intervals in each direction are used to perform the simulations. Since the basis functions are quadratic, the closest distance between two nodes in the mesh will be $\Delta x / 2$. As the level-set function is advected by this velocity field, the zero contour should remain a circle of radius $a$, only translated. Therefore, the curvature at any point of the zero contour should at all times be $\kappa_E = 1/a$.

![Figure 6.12](image-url)  
(a) The initial level-set function.  
(b) The velocity field.

**Figure 6.12.** The level-set function $\phi(x)$ is initially defined as the distance function to a circle with radius $a = 0.3$ centered in $(x_c, y_c)$, and advected by the time independent velocity field given in (6.6)-(6.7).

Natural boundary conditions are specified on the boundaries when the level-set function is advected by the velocity field, and to keep the level-set function free from disturbances at inflow boundaries, we need to use the reinitialization procedure. In the reinitialization calculations, information is propagated toward the boundaries and out of the domain. The amount of reinitialization needed to
yield the correct values of \( \phi(\mathbf{x}) \) at the boundaries subject to inflow in the advection calculations depends on the inflow velocity and the size of the advection time step. No reinitialization is however needed elsewhere to keep \( \phi(\mathbf{x}) \) as a distance function, since this is a rigid body rotation. Therefore, we only reinitialize outside of the zero contour. This is done by multiplying the sign function in (5.48) by \( H^{\frac{3}{2}}(\alpha - \phi(\mathbf{x})) \). \( \alpha = 0.1 \), where \( H^{\frac{3}{2}} \) is defined by (4.17) together with (4.43). Before we start the advection, we calculate the curvature and measure the error of the curvature along the zero contour. The curvature \( \kappa(\mathbf{x}) \) is calculated according to (3.24), i.e. \( \kappa = -\nabla \cdot (\nabla \phi | \nabla \phi |) \). To define a linear approximation to the zero level-set locally in each element, we use the approach described in section 5.3.2. Denote the points at the interface, defining this approximation, by \( \{ \mathbf{x}_i \}_{i=1}^{N_h} \). The number of discrete points \( N_h \) will mainly depend on the spatial resolution \( h \), but also on the explicit position of the zero contour in the mesh. The curvature \( \kappa_h \) is evaluated in each of these points, and the error is defined as

\[
||\kappa_h - \kappa_E||_1 = \frac{1}{N_h} \sum_{i=1}^{N_h} |\kappa_h(\mathbf{x}_i) - \kappa_E|.
\]

The exact value of the curvature, \( \kappa_E = 1/\alpha \) is constant since the zero contour is a circle. The error in the curvature will arise partly from the error in the position of the discrete points defining the approximation of the zero level-set and partly from the curvature calculations. The curvature evaluated at the discrete points \( \{ \mathbf{x}_i \}_{i=1}^{N_h} \) approximating the zero contour has been plotted for two different resolutions (\( \Delta e = h \) and \( \Delta e = h/2 \)) in Figure 6.13. We denote the curvatures calculated at \( t = 0 \) by \( \kappa_h^0 \) and \( \kappa_h^0/2 \).

Now, we advect \( \phi(\mathbf{x}) \). In the run where \( \Delta e = h = 0.1 \), we use \( \Delta t = 0.01 \), and perform 100 time steps up to \( t = 1.0 \). In each advection time step, \( 4 \) reinitialization time steps with \( \Delta \tau = 0.01 \) are performed. To make sure that time stepping errors are not a dominating source of error, the size of the time step is reduced with a factor of \( 2 \) for the higher resolution run. (\( \Delta e = h/2 = 0.05 \), \( \Delta t = 2.5 \cdot 10^{-3} \)). The time-step for the reinitialization is reduced by a factor of \( 2 \) in order not to change the stability of those calculations (\( \Delta \tau = 5 \cdot 10^{-3} \)). No extra diffusion is added (\( \epsilon = 0 \) in (5.47)). Denote the center position of the circle defining the zero contour by \( (x_c, y_c) \). The segments are plotted starting at \( (x_c + a, y_c) \), following the contour counter clockwise. We find that \( ||\kappa_h^0 - \kappa_E||_1 = 48 \cdot 10^{-2} \), and \( ||\kappa_h^0/2 - \kappa_E||_1 = 1.1 \cdot 10^{-2} \). This yields a convergence order for the curvature of \( h^p \) with \( p = 2.18 \), at the initial time.

The exact solution \( \phi_E(\mathbf{x}) \) at \( t = 1.0 \) is known, and we measure the error of the computed \( \phi_h(\mathbf{x}) \) by

\[
||\phi_h - \phi_E||_1 = \frac{1}{N} \sum_{i=1}^{N} |\phi_h(\mathbf{x}_i) - \phi_E(\mathbf{x}_i)|,
\]

where the sum is over all the nodes \( \mathbf{x}_i \) in the mesh. We find that \( ||\phi_h - \phi_E||_1 = 8.70 \cdot 10^{-4} \), and \( ||\phi_h/2 - \phi_E||_1 = 1.54 \cdot 10^{-4} \). This yields the error for \( \phi(\mathbf{x}) \)
6.3. Remarks on Force Calculations in the Level-Set Method

is proportional to $h^p$ with $p = 2.50$. This order of convergence is supported by the theory of the streamline diffusion modification, see [23].

The magnitude of the error in $\phi(x)$ converges well and is small, but it contains an oscillating component. This is illustrated in Figure 6.14, where the error is plotted along the line $x = -0.1$. Since the level-set function $\phi(x)$ is defined as the signed distance function to a circle with its center in $(x_c, y_c)$, it is a circular cone with its top in $(x_c, y_c)$. The derivatives of $\phi(x)$ will be discontinuous at this point. The advection of the non-smooth $\phi(x)$ introduces oscillatory errors.

If the oscillating error component in the error of $\phi(x)$ is not filtered out, it will be magnified in the curvature calculations. The curvature along the zero contour at time $t = 1$ is plotted in Figure 6.15. The errors due to this oscillating component are most pronounced for the higher resolution, where other errors are smaller. The errors in the curvature are larger on parts of the zero contour that have been advected behind the top of the level-set function, as could be expected, since that is where the dispersion errors spread most. Denote the calculated curvatures at time $t = 1.0$ by $\kappa_h$ and $\kappa_{h/2}$. We find that $||\kappa_h - \kappa_{h/2}||_1 = 6.9 \cdot 10^{-2}$, and $||\kappa_{h/2} - \kappa_{h/2}||_1 = 2.9 \cdot 10^{-2}$. This yields a convergence order for the curvature of $h^p$ with $p = 1.24$. Due to the errors that arise when we advect the level-
Figure 6.14. A contour plot of the level-set function at $t = 1.0$ is shown in a). Figures b)–d) show plots versus $y$ for $x = x^* = 0.1$ (the dashed line in a)). The exact solution $\phi_E(x^*, y)$, and the computed solutions $\phi_h(x^*, y)$ and $\phi_{h/2}(x^*, y)$ have been plotted on top of each other in b). In c) and d) we plot the error of $\phi_h(x^*, y)$ and $\phi_{h/2}(x^*, y)$. Note the different scales on the axes.
set function \( \phi(x) \), we have lost the second order convergence for the curvature calculations that we had before the advection started.

Now, we apply the filtering described in (5.49) to \( \phi(x) \) to define a function \( \tilde{\phi}(x) \), from which the curvature \( \tilde{\kappa} \) is calculated. The result is shown in Figure 6.16, where \( \tilde{\kappa} \) is plotted for the discrete points along the zero contour. The large oscillating errors are greatly reduced, while the average level of the curvature is well preserved. Denote the mean value of the calculated curvatures

\[
\{ \kappa_{h/2}(x_i) \}_{i=1}^{N_h/2} \quad \text{and} \quad \{ \tilde{\kappa}_{h/2}(x_i) \}_{i=1}^{N_h/2}
\]

by \( \bar{\kappa}_{h/2} \) and \( \bar{\tilde{\kappa}}_{h/2} \). The deviation from these mean values to the exact curvature \( \kappa_E \) are \( |\bar{\kappa}_{h/2} - \kappa_E|/\kappa_E = 2.0 \cdot 10^{-4} \) and \( |\bar{\tilde{\kappa}}_{h/2} - \kappa_E|/\kappa_E = 4.7 \cdot 10^{-5} \). The error of \( \tilde{\kappa}_{h/2} \) is \( ||\tilde{\kappa}_{h/2} - \kappa_E||_1 = 1.2 \cdot 10^{-2} \), as compared to \( ||\bar{\kappa}_{h/2} - \kappa_E||_1 = 2.9 \cdot 10^{-2} \), which was the error without any smoothing of \( \phi(x) \). The error for \( \tilde{\kappa}_{h/2} \) is actually only slightly larger than the error in the calculated curvature before advection \( ||\kappa_{h/2}^0 - \kappa_E||_1 = 1.1 \cdot 10^{-2} \).

The definition of \( \tilde{\phi}(x) \) is done only as a part of the curvature calculations, and \( \tilde{\phi}(x) \) is not used elsewhere.

---

**Figure 6.15.** The curvature \( \kappa(x_i) \) at \( t = 1.0 \) plotted versus \( i \) for \( i = 1, \ldots, N_{\Delta x} \). The segments are plotted starting at \( (x_i + a, y_i) \), following the contour counterclockwise.
6.3.2 Force Evaluations

In our implementation, the interfacial force term is evaluated by computing the line integral (6.5), as was described in section 5.3.2. In this procedure, a linear approximation of the zero level sets of the level set function is determined on a sub element level in each element that they intersect. Since the zero level-set is not explicitly described by the level-set function, it might seem more natural to evaluate the integral (6.4), if we could only do that with reasonable numerical accuracy.

In chapter 4, we studied how to choose approximations to \( \delta \gamma \), the Dirac delta function with support on a curve \( \gamma \), such that the sum of the analytical error and the numerical error (quadrature error) is kept small. Therefore, approximations of \( \delta \gamma \), which are candidates to yield any desired accuracy in the integration of the surface tension forces are available.

The study was however carried out assuming that the function multiplying \( \delta \gamma \) is smooth. In the integral (6.4), the delta function is multiplied by the normal vector and the curvature of the interface. These quantities are obtained by differentiating the level-set function. In the previous section, the evaluation of the curvature was discussed, proving it necessary to apply filtering to \( \phi(x) \) as a step in the curvature calculations, to avoid magnification of small oscillatory components present in \( \phi(x) \). Even with this procedure for the curvature calculations, the curvature will not be smooth. Therefore, the evaluation of (6.4) with some otherwise good choice of delta function approximation, will not yield as good a result as was indicated in our study, where the functions multiplying the delta approximation were smooth.

The assembly of the system of equations when solving the Navier-Stokes equations was discussed in section 5.1.4. The equations are assembled for all \( \phi_{P,i} \), \( i = 1, 2 \) (x and y direction), \( p = 1, ..., N \), where \( N \) is the number of nodes in the mesh. The function \( \phi_{P} \) is the sum of all element basis functions that is 1 in node \( p \), as was given in (5.21). Therefore, to evaluate the contribution of the surface tension forces to resulting system of equations, we need to evaluate \( f_{T}(\phi_{P,i}) \) for \( i = 1, 2 \) and \( p = 1, ..., N \).

Given a \( \phi(x) \) obtained in a simulation of a single rising buoyant bubble, we
compute these quantities both by evaluating the line integral (6.5), and by evaluating the integral over \( \Omega \) (6.4). The line integral is evaluated as was discussed in section 5.3.2. The integral over \( \Omega \) is computed using the delta approximation \( \delta^{3,1}_w(\phi(x)) \), as described in section 4.3.3 (equation (4.98) together with (4.18)). We use two different values of the width \( w \) of the zone of support of \( \delta^{3,1}_w(\phi(x)) \). In Figure 6.17, the results from these calculations are plotted.

\[ \begin{align*}
\text{(a) The level set function.} \\
\text{(b) Evaluation of line integral. Maximum} \\
\text{arrow length 0.31.}
\end{align*} \]

\[ \begin{align*}
\text{(c) Using } \delta^{3,1}_w(\phi(x)) \text{ with } w = 0.05. \text{ Maximum} \\
\text{arrow length 0.32.} \\
\text{(d) Using } \delta^{3,1}_w(\phi(x)) \text{ with } w = 0.1. \text{ Maximum} \\
\text{arrow length 0.26.}
\end{align*} \]

**Figure 6.17.** Integrated force contributions \( \{f_1(\Psi\hat{e}_1), f_2(\Psi\hat{e}_2)\} \) for a given function \( \phi(x) \). The contributions are defined to be node values for a Lagrange interpolant \( F \), which in turn interpolated onto a regular grid and plotted.

The results of the evaluation of the integrated forces, displayed in Figure 6.17 look very similar for the different methods of evaluation. Of course, they will never be identically the same, since the buffer zone defined by \( |\phi(x)| \leq w \)
will intersect more elements than the zero contour defined by $\phi(x) = 0$, thereby yielding non-zero contributions for more node numbers $p$.

However, if we sum all the $f_i(\Psi^p e_i)$ together, the result should ideally be the same for the two different approaches. In the case where $\phi(x)$ has just been initialized and contains no dispersion errors, this works fine. But in the case plotted in Figure 6.17, this yields a positive contribution in the $y$-direction for the evaluation with the delta function approximation, while it is close to zero for the line integral evaluation. We compare the results from the evaluation of the line integral and the delta function approximation with $w = 0.1$. The difference in the sum of the $y$-components is $2.8 \cdot 10^{-1}$ at this later time, compared to $5 \cdot 10^{-4}$ for the initial $\phi(x)$.

Errors like this come into the calculations at many consecutive time steps, and will have an effect on the final result. In Figure 6.18, the solution of the Navier-Stokes equations are compared for two different ways of evaluating the force contributions. Reinitialization is performed in every time step, to ensure that $\phi(x)$ is a good approximation to the distance function $d(x)$ in the support zone of the delta approximation, $|d(x)| \leq w$.

![Figure 6.18](image)

**Figure 6.18.** The full Navier-Stokes equations have been solved to simulate the rise of a buoyant bubble. The solid line is the contour obtained when the force contribution was calculated by evaluation of the line integral, and the dashed line is the contour obtained for the delta approximation $\phi^{-1}(\phi(x))$ (equation (4.36) together with (4.18)) with $w = 0.1$. The bubble was circular initially with its center in $(0,0,1.0)$. The mesh is shown in Figure 6.6.
6.4 Bubble Dynamics with Topology Changes

In this section, we present results from numerical experiments including topology changes. The same simulation has been performed with both the level-set and the segment projection methods.

In the initial configuration, we have three bubbles of fluid $A$ of different sizes, immersed in fluid $B$, with a layer of fluid $A$ on top of fluid $B$. The configuration is shown in Figure 6.19. The mesh that has been used in the simulations is shown in the same figure. We impose no-slip conditions on the top and bottom walls, and periodic boundary conditions in the horizontal direction.

![Image of mesh](image1)

![Image of initial configuration](image2)

(a) The mesh used in the simulations.
(b) The initial configuration.

**Figure 6.19.** The mesh used for the simulations has 1554 elements. Each element has 6 nodes since quadratic basis functions are used. The velocity field is quiescent initially, $u(x, 0) = 0$.

The parameters for this run are the Morton number $M = 0.1$, and the Eötvös number $Eo = 10.0$. These numbers were defined in (6.1), here they are based on the diameter of the largest bubble. The density ratio $\rho_B/\rho_A = 100$ and the viscosity ratio $\mu_B/\mu_A = 2$.

The fluids are quiescent initially. The bubbles rise due to buoyancy. The smallest bubble will rapidly merge with the largest bubble. Eventually, the lowest bubble will catch up with this merged bubble, and yet another merge will occur, creating one single larger bubble. The merged bubble deforms as it continues to rise, approaching the surface. The drainage of fluid $B$ from the
region between the two bubble and the surface starts. Finally, the bubble merges with the surface.

The numerical parameters for the runs are as follows: The time step is $\Delta t = 5 \cdot 10^{-4}$, the width of the transition zone for the Heaviside function is set to $w = 0.075$. For the level-set run, the equations for the level-set function are solved on a refined mesh, obtained by splitting each triangle in the mesh (Figure 6.19) into four triangles. This procedure was described in section 5.3.4. In each advection step, one reinitialization step with step size $\Delta \tau = 0.01$ is taken, with diffusion parameter $\epsilon = 4 \cdot 10^{-3}$. In the simulation by the segment projection method, the segments are discretized on uniform one-dimensional grids with grid size $h = 5 \cdot 10^{-4}$, both in the $x$ and $y$ directions.

The results from the level-set run are shown in Figures 6.20-6.22, and the results from the run with the segment projection method are plotted in Figures 6.23-6.26. The segment representation for the interfaces in Figure 6.23b (at $t = 0.25$), is shown in Figure 6.24.

![Figure 6.20](image)

(a) At $t = 0.10$, $\|u\|_\infty = 7.36$.
(b) At $t = 0.20$, $\|u\|_\infty = 9.31$.

**Figure 6.20.** Run with initial configuration as in Figure 6.19. Results from the level-set run.

In the level-set method, different interfaces are represented as zero level-sets of the same function, and merging occurs when two interfaces get so close, relative to the resolution of the mesh, that they cannot be distinguished anymore. When two volumes of fluid $A$ get close, there is a thin region of fluid $B$ between two interfaces. The level-set function is positive in fluid $A$, negative only in this thin region occupied by fluid $B$. In this region, the gradient of the level-set function
6.4. Bubble Dynamics with Topology Changes

Figure 6.21. Results from the level-set run.

(a) At $t = 0.30$, $\|u\|_\infty = 5.02$.
(b) At $t = 0.35$, $\|u\|_\infty = 5.53$.

Figure 6.22. Results from the level-set run.

(a) At $t = 0.45$, $\|u\|_\infty = 2.53$.
(b) At $t = 0.55$, $\|u\|_\infty = 1.13$. 
Figure 6.23. Run with initial configuration as in Figure 6.19. Results from the segment projection run.

(a) At $t = 0.10$, $|u|_{\infty} = 6.11$.

(b) At $t = 0.25$, $|u|_{\infty} = 8.45$.

Figure 6.24. The segments used to represent the two interfaces in Figure 6.23.

(a) The $x$-segments.

(b) The $y$-segments.
will be discontinuous along the line of equal distance to the two interfaces. Such a region will be subject to more diffusion of the numerical method, which speeds up the merging process. The change in topology is a natural part of the level-set method, the level-set function $\phi(x)$ simply evolves continuously during this process.

In the segment projection method, some criterion must be used to determine if and when merging should occur. Here, we have used the the distance between the interfaces. When a merger occurs, a special algorithm needs to be evoked, connecting the two structures representing the involved interfaces, creating one new modified structure. This algorithm was described in section 5.5.1.

What distance $\delta$ that is set as the merging distance will affect when merging takes place. It might also affect the merging shape, since with a smaller merging distance, the interfaces merge later and might have deformed more before this happens. The merging distance $\delta$ should be set with consideration of the resolution of the mesh, and should never need to be larger than the size of an element. Since the diffusive effect in the level-set method, that was described above, has no correspondence in this method, the mergers generally occur at a later point in time in simulations with the segment projection method.

Compare Figures 6.20(b) and 6.23(b), showing the interfaces and velocity fields after the second merger for the two methods. The bubbles in the level-set method have merged earlier, and surface tension effects have made the remaining large bubble approach a more regular shape. In the segment projection run, the bubbles deformed more before the second merger, which occurred later in time. Another difference is found after the bubble has merged with the surface. The filaments left after this merger get thinner and disappears more quickly in the level-set method. The results of the two methods are however qualitatively similar.

The curvature calculations are not very accurate along the filaments after the bubble-surface merge. In the level-set method, the curvature calculated is cut off at a maximum value of 15.0. This maximum value is motivated from the fact that structures with larger curvatures, i.e. with such small scale details, can not be represented with the resolution of the mesh used in the simulation. The high frequencies of this cut curvature is thereafter filtered out equivalent to (5.49).

In the merging process in the segment projection method, circular arcs have been inserted to reconnect merging structures. The radius of curvature of these arcs have been chosen considering the smallest scales that can be represented on the mesh, and because of this, there are resolution effects in the curvature calculations. The curvature is calculated using divided differences for the one dimensional segment discretizations. Since the resolution of the segment discretizations is finer than the resolution of the mesh and the interface is smooth, these calculations are however rather accurate.

When a merger takes place, there are sub-grid scale effects. The curvature gets infinitely large at a point connecting the two previously separated interfaces. The surface tension forces however regularize the interface in an infinitesimal
Figure 6.25. Results from the segment projection run.

Figure 6.26. Results from the segment projection run.
short time. No numerical method can fully resolve these sub-grid phenomena, and some approximation must be made.

We have discussed the merging process for the level-set and segment projection methods. The approximation of the merging process is different in the two methods, and it is difficult to know what the correct behavior should be. The results from the two runs are however qualitatively similar, showing the same type of behavior after each merger.

6.5 Sedimentation of Viscous Drops

In this section, we present the results of simulations of ten viscous drops, settling in a box due to the effects of buoyancy. The segment projection method has been used, and no merging between drops is allowed.

We use a regularly subdivided mesh on $[-1,1] \times [0,3]$. The element size is 0.05, i.e. the distance between two neighbouring nodes in the mesh is 0.025, since we use quadratic elements. The segments are discretized on uniformly divided one-dimensional grids, with grid size $h = 6.25 \cdot 10^{-3}$ in both directions. We impose no-slip conditions on all four walls. Initially, ten drops with diameter $d = 0.25$ are immersed into the surrounding fluid. The configuration is shown in Figure 6.27.

![Figure 6.27. Initial condition. Ten drops of fluid A immersed into fluid B. All ten drops have radius 0.125. The velocity field is quiescent initially, $u(x,0) = 0$.](image-url)
We present two runs, with different physical parameters. In the first run, the viscous drops deform as they settle at the bottom of the box. In the second run, the viscosity and surface tension of the drops are high enough for the drops to resist deformation.

Denote the fluid inside the drops by fluid $A$, and the surrounding fluid by fluid $B$. For the first run we use the Morton number $M = 0.1$ and the Eötvös number $Eo = 0.02$. These numbers were defined in (6.1), and are based on the drop diameter $d = 0.025$. The ratio of the density of the drops to the density of the surrounding fluid is $\rho_A/\rho_B = 100$ and the viscosity ratio is $\mu_A/\mu_B = 10$. For the second run, the parameters are $M = 0.1$, $Eo = 0.002$, $\rho_A/\rho_B = 100$ and $\mu_A/\mu_B = 20$.

![Image](image-url)

**Figure 6.28.** Numerical simulation with initial configuration as in Figure 6.27. The physical parameters are $M = 0.1$, $Eo = 0.02$, $\rho_A/\rho_B = 100$, $\mu_A/\mu_B = 10$. The time step is $\Delta t = 0.025$.

The time scales of the two runs are quite different. Although the first run is very viscous, the drops do deform somewhat. In the second run, the viscosity of the drops is increased, as well as the surface tension. The drops resist any
6.5. Sedimentation of Viscous Drops

deformation, and move essentially like rigid bodies. This slows down the time scale of the problem substantially. The settling times are about 30 times longer than in the first simulation. In the first run, we use $\Delta t = 0.025$, and in the second run $\Delta t = 0.5$.

Results from the first simulation are presented in Figures 6.28-6.29, and results from the second run are presented in Figures 6.30-6.31.

All drops are of equal size, and have the same density and viscosity. The settling velocity of a drop will however depend on the configuration of drops. A drop might move with a higher velocity if it travels in the wake of another drop. It might also be slowed down by fluid recirculating due to the fall of other drops. The three drops in the top of Figure 6.28 move slower than the other drops to begin with. As the other drops starts to settle close to the bottom, and the velocities in this region decrease, the settling velocities of the three drops increase, and they fall faster towards the bottom.

![Figure 6.29](image-url)  
(a) At $t = 18.75$, $\|u\|_\infty = 0.106$.  
(b) At $t = 25.0$, $\|u\|_\infty = 0.046$.  

**Figure 6.29.** Plots at later times for the same simulation as in Figure 6.28.
In Figure 6.29, all drops have reached the lower part of the domain. The velocities decrease, as the drops slowly move to obtain a more densely packed configuration.

The results for the stiffer drops are shown in Figures 6.30-6.31. Also here, the drops fall with different speed. The time scale in this simulation is however much slower compared to the previous run. As in the first simulation, three drops fall behind, but increase their settling speeds as the fluid velocities created by the other drops get smaller.

![Diagram](image_url)

(a) At $t = 125$, $\|u\|_\infty = 5.90 \cdot 10^{-3}$.
(b) At $t = 375$, $\|u\|_\infty = 7.43 \cdot 10^{-3}$.

**Figure 6.30.** Numerical simulation with initial configuration as in Figure 6.27. The physical parameters are $\mathcal{M} = 0.1$, $Eo = 0.002$, $\rho_A/\rho_B = 100$, $\mu_A/\mu_B = 20$. The time step is $\Delta t = 0.5$.

The drops in this simulation do hardly deform at all, mainly due to the high surface tension. They are very viscous, and more essentially like rigid bodies. Because of this, the velocity field inside a drop in the reference frame of the drop is almost zero. The effective boundary conditions on these drops therefore very much resembles the no-slip conditions associated with solid particles. In the case of solid particles, the detailed tracking of interface shapes is redundant.
6.5. Sedimentation of Viscous Drops

The fictitious domain technique has been successfully applied to the simulation of particulate flows in [15]. The existence of solutions for the motion of solid particles immersed in a viscous fluid is studied in [11].

The configuration of the drops in the end is somewhat different in the two runs. Since the drops in the second run resist deformation, they will not be packed as densely as the drops in the first run. The mass conservation is very good in the last run, each drop has lost less than 0.5\% of its mass. In the first run, the mass conservation is not as good. A typical drop has lost 3 – 5\% of its mass in the end of the simulation.

\begin{figure}[h]
\centering
\includegraphics[width=\linewidth]{figure6.31.png}
\caption{Plot at later times for the same simulation as in Figure 6.30.}
\end{figure}

In many important physical problems, the number of drops in the flow is so large that it is not feasible to perform simulations resolving each drop. To simulate such problems, effective equations, based on a macroscopic description of the problem, needs to be formulated. Such formulations make use of a function defining the volume fraction of drops. The objective is to study the behavior of the mixture, rather than the interaction of individual drops. To complete this
formulation, one must define some bulk variables, such as the viscosity of the mixture.

Bulk models are used in many simulations, in one, two and three spatial dimensions. In many cases, it is difficult to perform physical experiments such that bulk variables can be measured. Numerical experiments where a relatively large, yet limited, number of drops is considered, could be used to measure such bulk variables. Simulations of the type presented here could therefore be used to verify physical models formulated for different properties of the mixture.
Appendix A

Quadrature Rules

The quadrature rule used for evaluation of integrals in the variational formulations of the Navier-Stokes equations and the level-set advection and reinitialization equations is given in Table A.1.

<table>
<thead>
<tr>
<th>Quadrature points</th>
<th>Quadrature weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3333333333333333</td>
<td>-0.149570044467670</td>
</tr>
<tr>
<td>0.4793089067841923</td>
<td>0.175615257433204</td>
</tr>
<tr>
<td>0.260345966079038</td>
<td>0.175615257433204</td>
</tr>
<tr>
<td>0.869739794195568</td>
<td>0.053347235608839</td>
</tr>
<tr>
<td>0.65130102902216</td>
<td>0.053347235608839</td>
</tr>
<tr>
<td>0.63844188569809</td>
<td>0.077113760890257</td>
</tr>
<tr>
<td>0.63844188569809</td>
<td>0.077113760890257</td>
</tr>
<tr>
<td>0.48690315425316</td>
<td>0.077113760890257</td>
</tr>
<tr>
<td>0.48690315425316</td>
<td>0.077113760890257</td>
</tr>
<tr>
<td>0.312865496004876</td>
<td>0.077113760890257</td>
</tr>
<tr>
<td>0.312865496004876</td>
<td>0.077113760890257</td>
</tr>
</tbody>
</table>

Table A.1. Quadrature points and weights for the 13-point quadrature formula. This formula integrates polynomials of degree 12 exactly on the triangle with nodes (0, 0), (1, 0) and (0, 1).

The three-point Gauss quadrature rule, used to evaluate the line integral in the front-tracking method (section 5.4.2) is given in Table A.2.
<table>
<thead>
<tr>
<th>Quadrature points</th>
<th>Quadrature weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1127016653792582</td>
<td>5/18</td>
</tr>
<tr>
<td>0.5</td>
<td>8/18</td>
</tr>
<tr>
<td>0.8872983346207417</td>
<td>5/18</td>
</tr>
</tbody>
</table>

**Table A.2.** Quadrature points and weights for the 3-point Gauss-quadrature formula. This formula integrates polynomials of degree 5 exactly on the interval (0, 1).
Bibliography


