JYVÄSKYLÄ STUDIES IN COMPUTING
58

Kari Kärkkäinen

Shape Sensitivity Analysis for Numerical Solution of Free Boundary Problems

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 58

Kari Kärkkäinen

Shape Sensitivity Analysis for Numerical Solution of Free Boundary Problems

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston vanhassa juhlasalissa (S212)
joulukuun 20. päivänä 2005 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in Auditorium S212, on December 20, 2005 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2005

Shape Sensitivity Analysis
for Numerical Solution
# of Free Boundary Problems

JYVÄSKYLÄ STUDIES IN COMPUTING 58

Kari Kärkkäinen

Shape Sensitivity Analysis for Numerical Solution of Free Boundary Problems

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2005

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN:9513923959
ISBN 951-39-2395-9 (PDF)

ISBN 951-39-2354-1 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2005

ABSTRACT

Kärkkäinen, Kari
Shape Sensitivity Analysis for Numerical Solution of Free Boundary Problems
Jyväskylä: University of Jyväskylä, 2005, 83 p. (+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 58)
ISBN 951-39-2395-9
Finnish summary
Diss.

This work is devoted to the development of efficient and robust solution algorithms for a class of free boundary problems. This consists of mathematical analysis of different model problems and the description of numerical implementation to generic free boundary problems along with the numerical results.

The free boundary problems that are investigated in this work are elliptic stationary boundary value problems with overdetermined boundary conditions. Overdetermined boundary conditions are satisfied only in a special geometry which is a solution to the free boundary problem. The free boundary problems are nonlinear and can not be solved straightforwardly. Algorithms to solve this kind of free boundary problems are iterative, the solution is sought by iterating geometries.

The suggested algorithm is based on the combination of continuous shape sensitivity analysis and automatic differentiation of discrete equations. The discrete linearized equations are derived tuning the finite element method to correspond to the continuous shape linearization of the problem.

Efficiency of the presented algorithms are tested and illustrated through numerous numerical examples.

Keywords: free surfaces, free boundary problems, automatic differentiation, finite element method, convergence.

Author

Kari Kärkkäinen
Information Technology Research Institute
University of Jyväskylä
Finland

Supervisor

Professor Timo Tiihonen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers

Professor N. V. Banichuk
The Institute for Problems in Mechanics
Russian Academy of Science
Russia

Reviewers

Professor V. Arnautu
University “Al I Cuza” Iasi,
Romania

Opponent

Professor Bertrand Maury
Laboratoire de Mathématiques
Université Paris-Sud
France

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere gratitude to Prof. Timo Tiihonen for introducing me into this field of research and for his excellent guidance and continuous support throughout my research. I am grateful to Prof. Arnautu and Prof. Banichuk for reviewing the manuscript and giving encouraging feedback and to Prof. Pekka Neittaanmäki for the tenacious motivation to my work. I would also like to acknowledge my appreciation to prof. Haslinger for fruitful discussions of the topic. I would like to thank my colleagues and friends Mika Laitinen and Markus Inkeroinen for their inspiring ideas and thoughts. Gary Littler deserves special thanks for his linguistic comments. I want to express my thanks to my colleagues at the Faculty of Information Technology for creating the friendly and fertile atmosphere to work.

This work was financially supported by Tekes, the Academy of Finland, The Artturi Nyyssönen Foundation, COMAS Graduate School of the University of Jyväskylä, Department of Mathematical Information Technology and Information Technology Research Institute.

Finally, I must express my appreciation to my parents Esko and Kaisa, my family and friends for their support, especially to Satu for her love. I thank my children Lotta and Konsta for their love and patience, they always succeed to delight their father.

Jyväskylä, 1 December, 2005

Kari Kärkkäinen

CONTENTS

LIST OF INCLUDED ARTICLES

I Kari Kärkkäinen and Timo Tiihonen, Free surfaces: shape sensitivity analysis and numerical methods, *Internat. J. Numer. Methods Engrg., 44 (1999), pp. 1079–1098,* 1999.

II Kari Kärkkäinen, Bernoulli's free boundary problem with curvature dependent boundary conditions, *Free Boundary Problems: Theory and Applications, vol. 14 of GAKUTO International Series, Gakkotosho, Tokyo, 2000, pp. 239–254,* 2000.

1 INTRODUCTION

Free boundary problems are boundary value problems in which some parts of the boundaries of the considered domains have to be determined as a part of the solution. A free boundary value problem consists of differential equations with overdetermined boundary conditions in some parts of the geometrical boundaries that are assumed to be free boundaries. Overdetermined means formally that the boundary conditions in the free boundary can be divided into two parts, in which the first part can be used to solve differential equations and the second part to find the geometry of free boundaries.

1.1 Free boundaries in continuum mechanics

Let us first introduce a few examples from real life, which illustrate some boundaries, or discontinuities with respect to the geometry. Consider, for example, a liquid–gas surface, which evidently is a free surface. On the surface viscosity, density and heat conductivity change rapidly thus producing material properties to be different on both sides of the surface. Further, different adhesion forces in the air and in water produce forces on the interface, which appear as surface tension forces [Ada67, MT68].

The water–air surface may remain in one position in time and the free surface can be considered as stationary. For example, water in the capillary pipe can be considered as a representative of a static free boundary, where liquid is in a static position and the free boundary is in an equilibrium state. Here the liquid does not flow. Hence, the free boundary can only be defined by modelling the surface position.

The second example, fluid flowing at a (low) constant speed from the pipe can be considered as a representative of a stationary free boundary, as long as the free boundary remains in its position in time. However, with higher speed the flow field becomes unstable and so the free boundary eventually becomes dynamic.

Between stationary and dynamic flows there is a region, which can be considered as quasi–stationary. Quasi–stationary means, that by moving the coordinate system in some speed, depending on the system itself, the system in this reference coordinate remains stationary. Here we shall sketch two different quasi–stationary free boundary problems. An example of a quasi–stationary free boundary problem is a rising air bubble in water. In this case the small time scale is measured in seconds, and in this small scale time interval the rising bubble can be quasi–stationary, but as the external forces, like hydrostatic pressure, change during the rise of a bubble, the bubble can evolve as time goes on. This evolution appears, however, so slowly that the dynamic effects can be ignored. Hence by this approach we can take snapshots of the dynamical system. To model the rising bubble perfectly in a larger time–scale, one must take into account the change in the physical quantities in time.

Our second physical example is the melting or solidification of a material. There is quite an extensive collection of references of the books and articles collected by Tarzia [Tar00] on the topic. In this case the free boundary is the boundary of a solid material and therefore it can be characterized as a liquid–solid free boundary. From the melting temperature of the material we know the temperature on the free surface and it is relevant to consider the temperature of the material to provide an insight into this dynamical system.

Consider now the enthalpy of the material. Near the melting point of a material the enthalpy grows rapidly as a function of temperature, see figure 1. In particular, enthalpy can be a set valued function of temperature and may vary in single temperature depending on the state of the material.

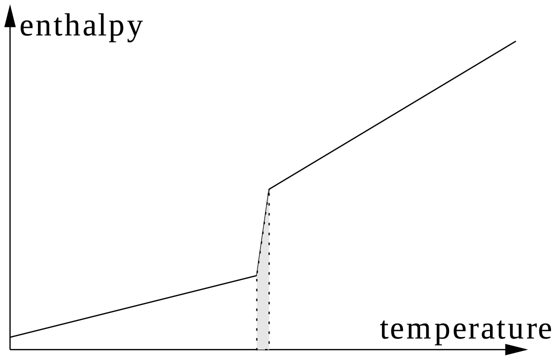


FIGURE 1 Relation between enthalpy and temperature.

Again we can consider the quasi–stationary approach to obtain an insight into the time–dependent process. One such quasi–stationary process is a continuous casting of steel. In this case the melted steel is formally fed into the pipe and then the solidified steel is continuously pulled out from the other head of the pipe while the pipe is cooled to obtain the solidification inside the pipe. To obtain as good steel as possible, it is eventually important to know the possible properties that influence the formation of the steel.

As well as the water–air and the water–ice surface both may be thick when considered on a microscopic scale we may in a macroscopic view consider these boundary layers as thin boundaries where physical quantities may rapidly change. This enables us to treat physical quantities as continuous functions elsewhere but on this thin boundary, which is consequently a free boundary.

1.2 Model free boundary problem

In this thesis we shall consider general mathematical or engineering approaches to solve the free boundary problems of type

$$A(u, \Omega) = 0, \tag{1.1a}$$
$$B(u, \Omega) = 0. \tag{1.1b}$$

Here A corresponds to a well posed elliptic boundary value problem in domain Ω, and B respectively operates on the functions supported at the free boundary Σ. It is supposed that function u can be solved from equation (1.1a) for any given suitable domain Ω. Our aim in this thesis is to study the systematic and efficient ways to solve the system (1.1) and thus provide tools to solve general stationary free boundary problems numerically.

The solution of an elliptic boundary value problem usually depends highly nonlinearly on the shape of the domain. Thus the geometry can not be solved straightforward from a linear equation. In this thesis we consider approaches where this system of equations is solved iteratively. We further restrict our attention to finding solutions with known topology.

In the following sections we shall briefly review the solution strategies to solve the free boundary problems in the above form and formulate them to general principles.

We shall formulate the solution strategies for the abstract problem (1.1) and for the convenience of the reader will then apply the strategies to the following model free boundary problem, so called Bernoulli free boundary problem [FR97, Zol86],

$$-\Delta u = 0 \qquad \text{in } D_\Omega, \tag{1.2a}$$
$$u = 1 \qquad \text{on } \partial D, \tag{1.2b}$$
$$u = 0 \qquad \text{on } \partial\Omega, \tag{1.2c}$$
$$\frac{\partial u}{\partial n} = \lambda \qquad \text{on } \partial\Omega, \tag{1.2d}$$

where $D_\Omega = D \setminus \bar{\Omega}$. This equation models for example the electro-chemical galvanization or two dimensional inviscid irrotational fluid. For simplicity we assume here that the unknown free boundary is contained within a larger domain D. This assumption enables us to study the so called interior Bernoulli free boundary problem.

Exterior Bernoulli FBP is defined by changing the requirement $\Omega \subset D$ to $\Omega \supset D$. For this model problem (1.2) the existence and regularity of the solution are studied for example in [AC81].

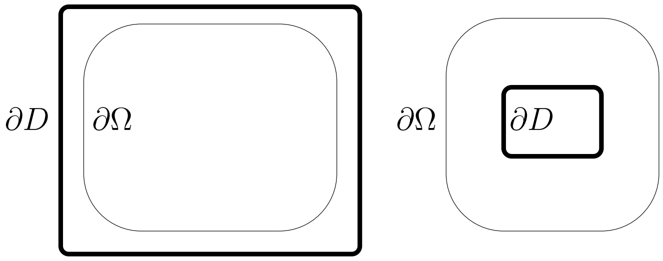


FIGURE 2 Interior (left) and exterior (right) Bernoulli free boundary problems.

To obtain a general abstract form for the above model free boundary problem we rewrite equations (1.2) in a weak form:

$$a_\Omega(u,\phi) := \int_{D_\Omega} \nabla u \cdot \nabla \phi \, dx - \int_{\partial\Omega} \lambda \phi \, d\sigma = 0, \quad \phi \in \mathcal{V}, \tag{1.3a}$$

$$b_\Omega(u,\psi) := \int_{\partial\Omega} u\psi \, d\sigma = 0, \quad \psi \in \mathcal{W}. \tag{1.3b}$$

Here we seek u from space $H^1(D)$ so that $u|_{\partial D} = 1$. Loosely speaking, above formulation is equivalent with (1.1), if we define $A(u,\Omega) : \mathcal{V}(\Omega) \to \mathcal{V}(\Omega)' : \langle A(u,\Omega), \phi \rangle = a_\Omega(u,\phi)$ and $B(u,\Omega) : \mathcal{W}(\Omega) \to \mathcal{W}(\Omega)' : \langle B(u,\Omega), \phi \rangle = b_\Omega(u,\phi)$ respectively. At the moment we will leave function spaces $\mathcal{V}$ and $\mathcal{W}$ undefined. Later we will discuss the different choices for these function spaces.

1.3 Shape optimization approach

In the shape optimization approach the given free boundary problem is rewritten so that the minimum of some cost functional is attained at the solution of the free boundary problem. Roughly speaking, optimization based approaches can be divided into two subclasses. The first class assumes that the free boundary equations (1.1) are first order optimality conditions for some “energy” functional. Formally this means that there exists a functional $E(u,\Omega)$ such that

$$\frac{\partial E(u,\Omega)}{\partial u} = 0 \text{ and } \frac{\partial E(u,\Omega)}{\partial \Omega} = 0 \Rightarrow A(u,\Omega) = 0 \text{ and } B(u,\Omega) = 0.$$

However, this approach only applies for some restricted subclass of free boundary problems. There is quite an extensive presentation of energy–based free boundary problems in reference [Fri82]. One subclass in this approach is such that the free boundary problem is reformulated in such a way that the free boundary disappears from the formulation. From a mathematical point of view this helps with the analysis of the qualitative properties of the free boundary problem, since the geometry is determined as a posterior from the solution of a boundary value problem. In this work we shall only consider solution methods where the free boundary is solved as one unknown in the solution procedure.

In the second class of shape optimization based approaches the additional free boundary condition is reformulated as a functional which attains its minimum at the solution of the free boundary problem. The other free boundary condition is then taken into account as a constraint for this shape optimization problem. Later we shall call this the least squares approach. For our abstract model problem this can read as

$$\min_{\Omega \in \mathcal{O}} \|B(u,\Omega)\|^2 \text{ with subject to } A(u,\Omega) = 0, \tag{1.4}$$

where $\mathcal{O}$ is a suitable set of domains and $\|\cdot\|$ is some norm supported only on the free boundary. This formulation is much weaker than the original free boundary problem (1.1) since the second condition is relaxed and need not to be fulfilled exactly at the solution. Hence we can have a solution for the above shape optimization problem although we do not have a solution for the free boundary problem. Therefore this formulation can be useful only from a numerical point of view.

This approach gives us, however, an opportunity to play with suitable domains $\mathcal{O}$. At the solution of the free boundary problem we should have $\|B(u,\Omega)\|^2 = 0$. However, if we restrict our admissible set of domains too much we can not reach the optimal set and thus we may observe that $\|B(u,\Omega)\|^2 > 0$ for all domains in $\mathcal{O}$. On the other hand, if we enrich the space $\mathcal{O}$ too much we can end up with a minimizing sequence that converges to a fractal–like domain, which is in most practical cases an incorrect solution. We further have freedom to choose a suitable norm for boundary condition $B(u,\Omega)$. In particular, using proper norm results usually a well posed shape optimization problem if there exists a smooth enough solution for the free boundary problem.

In the following sections we shall briefly apply shape optimization [HN96] to our model free boundary problem.

1.3.1 Variational formulations

For general free boundary equations (1.1) we can not always construct an "energy" formulation [Luk67, Zol94, Fri82, Cra84]. However, for our model problem it is well known that the solution of the model problem is a critical point of the functional [Tii97],

$$E(u) = \int_D \nabla u \cdot \nabla u + \lambda^2 I_{\{u>0\}}\, dx,$$

where $I_{\{u>0\}} := \{x \in D \,|\, u(x) > 0\}$. This energy is minimized over all functions in space $\mathcal{V} := \{u \in H^1(D) \mid u = 1 \quad \text{on } \partial D\}$. In this formulation the geometry is hidden into $I_{\{u>0\}}$ so that the geometry is not a variable in this definition. However, the characteristic function is non-smooth thus producing trouble in minimization. The actual benefit of this formulation is thus in the mathematical analysis, although with the help of optimal control theory efficient numerical implementations can be achieved [AN03, HHM93].

To obtain a shape optimization approach for the above problem we rewrite the energy as a functional of two variables,

$$E_1(u,\Omega) = \int_D \nabla u \cdot \nabla u + \int_{D_\Omega} \lambda^2 \, dx.$$

Now Ω appears as an unknown variable in the energy. Thus we seek for the minimizer of $E_1(u,\Omega)$ in some space $\mathcal{V} \times \mathcal{O}$.

If the minimization is succesfull the minimizer describes the free boundary $\partial\Omega = \{u = 0\}$. Alt and Caffarelli [AC81] showed for first energy formulation that a nontrivial minimizer exists if and only if $\inf E(u) < E(0) = \lambda^2|D|$. This is valid only if λ is larger than some critical value λ_c.

1.3.2 Least squares approach

Let us apply least squares approach for our model problem. Now choosing as the cost function the L^2-norm for the solution on the free boundary, we end up with the following shape optimization problem:

$$\min_{\Omega\in\mathcal{O}} J(\Omega), \quad J(\Omega) := \int_{\partial\Omega} u^2 \, d\sigma \text{ where } u \text{ is such that } u = 1 \text{ on } \partial D \text{ and} \tag{1.5}$$

$$\int_{D_\Omega} \nabla u \cdot \nabla \phi \, dx = \int_{\partial\Omega} \lambda\phi \, d\sigma \quad \text{for all } \phi \in H_0^1(D). \tag{1.6}$$

Clearly, if the FBP has a solution (u,Ω^*) and $\Omega^* \in \mathcal{O}$, then $u = 0$ on $\partial\Omega^*$ and therefore $J(\Omega^*) = 0$.

This formulation is by far not a unique shape optimization formulation for our model problem. We can take another boundary condition to be minimized, so that the state equation in the above contains the Dirichlet boundary condition $u = 0$ on free boundary $\partial\Omega$ and the shape optimization problem is in this case

$$\min_{\Omega\in\mathcal{O}} J_1(\Omega), \quad J_1(\Omega) := \left\| \frac{\partial u}{\partial n} - \lambda \right\|^2_{Z(\partial\Omega)} \quad \text{where } u \in H^1(D_\Omega) \text{ is such that} \tag{1.7}$$

$$\int_{D_\Omega} \nabla u \cdot \nabla \phi \, dx = 0 \quad \text{for all } \phi \in H_0^1(\Omega) \quad \text{with} \tag{1.8}$$

$$u = 1 \quad \text{on } \partial D \text{ and } u = 0 \quad \text{on } \partial\Omega. \tag{1.9}$$

Here the norm $\|\cdot\|_{Z(\partial\Omega)}$ is chosen properly, for example $Z = H^{-\frac{1}{2}}$ or $Z = L^2$. The norm $Z = H^{-\frac{1}{2}}$ seem to be more natural, since the solution u relies on $H^1(D_\Omega)$ and therefore the trace of u on $\partial\Omega$ is in $H^{-\frac{1}{2}}$. However, this choice leads to difficulties in the analysis of derivatives of the cost functional, that is essential in the study of optimal conditions for the shape optimization problem. Haslinger et al has studied the above formulation for the model free boundary problems in reference [HKKP03]. They use genetic algorithms in the optimization procedure which does not require shape sensitivity analysis.

1.4 Fixed point approach

In the fixed point approach to the free boundary problem the nonlinear problem is usually solved by constructing a sequence of linear problems. Fixed point algorithms do not commonly use gradient information but the algorithms can be based on the shape sensitivity analysis of the continuous problem.

1.4.1 Trial methods

Trial methods can be characterized as Picard–type iterations for free boundary problems. A trial method to solve a free boundary problem of type (1.1) can be written in algorithmic form:

1. Set $k = 0$, choose initial domain Ω_0.
2. Solve u_k from $A(u_k, \Omega_k) = 0$.
3. Construct $\Omega_{k+1} = F(u_k, \Omega_k)$, where F is chosen so that $B(u_k, \Omega_{k+1}) \approx 0$.
4. Update $k = k + 1$ and continue from step 2 until converged.

In particular, the above solution algorithm is generally simple to implement, but it is not always obvious to construct the updating step so that the method converges or so that the convergence is fast.

For our model problem (1.2) we can set up an algorithm as follows:

1. Set $k = 0$, choose initial domain Ω_0.
2. Solve $\Delta u_k = 0$ with Neumann boundary condition $\frac{\partial u}{\partial n} = \lambda$ on the free boundary and $u_k = 1$ on the fixed boundary ∂D.
3. Move free boundary with updating rule
$$\partial\Omega_{k+1} = \partial\Omega_k + \frac{u_k}{\lambda}\tilde{\mathbf{n}}.$$
4. Update $k = k + 1$ and continue from step 2 until converged.

Here $\tilde{\mathbf{n}}$ is a smoothed normal vector field on the free boundary. This is required, since updating the boundary with an unsmoothed normal vector field would decrease the smoothness of the free boundary.

The updating rule in the above algorithm was derived by taking into account the derivative of the solution $\frac{\partial u}{\partial n} = \lambda$ at the free boundary $\partial\Omega$. However, it does not take into account the change in the solution of the boundary value problem as the geometry is changed. The change in the solution u due to change of Ω can be minimized by changing the boundary condition in the state problem appropriately, as proposed by Garabedian [Gar56] for a model free boundary problem. Cryer [Cry70] further calculated the optimal combination of complementing boundary conditions for a general elliptic boundary value problem.

For our model problem (1.2) we can set up an algorithm as follows:

1. Set $k = 0$, choose initial domain Ω_0.
2. Solve $\Delta u_k = 0$ with Neumann boundary condition $\kappa u + \frac{\partial u}{\partial n} = \lambda$ on the free boundary $\partial\Omega$ and $u = 1$ on the fixed boundary ∂D.
3. Move free boundary with updating rule
$$\partial\Omega_{k+1} = \partial\Omega_k + \frac{u_k}{\lambda}\tilde{\mathbf{n}}.$$
4. Update $k = k + 1$ and continue from step 2 until converged.

Here κ is the mean curvature of the free boundary. This has been studied by Tiihonen [TJ91] with the viewpoint of second order shape derivatives and later Flucher and Rumpf [FR97] proved rigorously the convergence in spaces of Hölder continuous functions. Earlier Duprét [Dup81] constructed a modified state problem for a similar free boundary problem that models inviscid irrotational flow.

Detailed analysis of the algorithm shows that the convergence is less than quadratic but still superlinear [FR97]. For a two dimensional case the convergence factor is $\frac{3}{2}$ and for n–dimensional case $\frac{6}{5}$. In our studies the convergence was investigated in the presence of a curvature term in the boundary condition in paper [II]. It was shown that the the convergence factor is $\frac{4}{3}$ in a two dimensional case.

1.4.2 Shape linearization

In shape linearization the nonlinear system of equations is solved by Newtons method in functional spaces. The derivatives are calculated with respect to the state u and geometry Ω. Newton iteration assumes that we have an initial guess for the solution (u_k, Ω_k) and then we solve a linear equation to find a new iterate (u_{k+1}, Ω_{k+1}) close to the solution. In one Newton step we calculate the change δu_k to the solution u_k and the change $\delta\Sigma_k$ to the geometry Ω_k, and then update $(u_{k+1}, \Omega_{k+1}) = (u_k, \Omega_k) + (\delta u_k, \delta\Sigma_k)$. Formally we obtain the system of equations

$$A_{,u}(u_k, \Omega_k)\,\delta u_k + A_{,\Sigma}(u_k, \Omega_k)\,\delta\Sigma_k = A(u_k, \Omega_k), \tag{1.10a}$$
$$B_{,u}(u_k, \Omega_k)\,\delta u_k + B_{,\Sigma}(u_k, \Omega_k)\,\delta\Sigma_k = B(u_k, \Omega_k). \tag{1.10b}$$

Here $\cdot_{,u}$ means derivative with respect to u and $\cdot_{,\Sigma}$ derivative with respect to the geometry.

There is no easy way to construct a Banach space for general geometries. A detailed description of differentiation of functionals with respect to the geometry is given by Zolésio and Delfour [DZ01]. In addition, they process extensively different ways to construct valid topologies for geometries. Under appropriate smoothness assumptions the differentiation with respect to the geometry results to an operator which is supported only on the functions which rely on the boundary of the domain that is differentiated. In particular, for our purpose the derivatives $A_{,\Sigma}$ and $B_{,\Sigma}$ are supported only on the free boundary. Thus the new unknown $\delta\Sigma_k$ is also supported only on the free boundary.

Not much can be said about the solvability of the above equation. Intuitively, if the system (1.10) is solvable in the appropriate functional spaces, the solution of the FBP should be stable and behave well. In solving this equation we obtain a pair $(\delta u_k, \delta\Sigma_k)$. $\delta\Sigma_k$ corresponds to the change of the domain on the free boundary to the normal direction. δu corresponds respectively to the change in the state solution u. Iterate u_k and update δu_k is defined on geometry Ω_k and thus to obtain the next iterate $u_{k+1} = u_k + \delta u_k$ on Ω_k it must be extended to geometry Ω_{k+1}.

Let us now study the shape linearization for our model free boundary problem. The derivative with respect to a state variable u reads as

$$A_{,u}(u, \Omega)\delta u = \int_\Omega \nabla \delta u \cdot \phi \, dx$$

and the derivative with respect to geometry reads as

$$A_{,\Sigma}(u, \Omega)\delta\Omega = \int_{\partial\Omega} (\nabla_\Gamma u \cdot \nabla_\Gamma \phi - \lambda\kappa\phi)\, \delta\Sigma \, d\sigma.$$

Here κ is the mean curvature of the free boundary. In derivation of the above formula it is assumed that $\frac{\partial\phi}{\partial n} = 0$ on $\partial\Omega$ in weak sense.

The derivatives of B reads as

$$B_{,u}(u, \Omega)\delta u = \int_{\partial\Omega} \delta u \psi \, d\sigma,$$

and

$$B_{,\Sigma}(u, \Omega)\delta\Omega = \int_{\partial\Omega} \left(\frac{\partial u}{\partial n}\psi + \kappa u \psi \right) \delta\Sigma \, d\sigma.$$

As a result we end up with following iteration scheme: Given $(u_k, \Omega_k) \in H^2(\Omega) \times C^{2,\alpha}$, find $(\delta u_k, \delta\Sigma_k) \in H^1(D_\Omega) \times H^{\frac{1}{2}}(\partial\Omega)$ such that

$$\begin{aligned} \int_{D_{\Omega k}} \nabla \delta u_k \cdot \nabla\phi \, dx - \int_{\partial\Omega_k} (\lambda\kappa\phi - \nabla_\Gamma u_k \cdot \nabla_\Gamma \phi)\, \delta\Sigma_k \, d\sigma = \\ \int_{D_\Omega} \nabla u_k \cdot \nabla\phi \, dx - \int_{\partial\Omega_k} \lambda\phi \, d\sigma, \\ \int_{\partial\Omega_k} \delta u_k \psi \, d\sigma + \int_{\partial\Omega_k} \left(\frac{\partial u_k}{\partial n}\psi + \kappa u_k \psi \right) \delta\Sigma_k \, d\sigma = \int_{\partial\Omega_k} u_k \psi, \end{aligned} \tag{1.11}$$

for all $(\phi, \psi) \in H^1(D_\Omega) \times H^{\frac{1}{2}}(\partial\Omega)$. Newton update after solving $(\delta u_k, \delta\Sigma_k)$ from the above weak equation consists of an update of the geometry by $\partial\Omega_{k+1} = \partial\Omega_k + \tilde{\mathbf{n}}\delta\Sigma_k$, and an update of the state by $u_{k+1} = \tilde{u_k} + \widetilde{\delta u_k}$, where $\tilde{\cdot}$ means smooth H^2-extension to the updated geometry. Existence of such extension is guaranteed since $\delta u_k \in H^2(D_\Omega)$ under above smoothness assumptions.

The above system can be even more simplified by taking into account the properties of the solution u in the solution of the free boundary. As it holds that $u = 0$ and $\frac{\partial u}{\partial n} = \lambda$ we can substitute these to u_k to get

$$\begin{aligned} \int_{D_{\Omega k}} \nabla \delta u_k \cdot \nabla\phi \, dx - \int_{\partial\Omega_k} \lambda\kappa\phi \, \delta\Sigma_k \, d\sigma = \int_{D_\Omega} \nabla u_k \cdot \nabla\phi \, dx - \int_{\partial\Omega_k} \lambda\phi \, d\sigma, \\ \int_{\partial\Omega_k} \delta u_k \psi \, d\sigma + \int_{\partial\Omega_k} \lambda\psi \, \delta\Sigma_k \, d\sigma = \int_{\partial\Omega_k} u_k \psi \, d\sigma. \end{aligned} \tag{1.12}$$

In the above linearized system we can eliminate $\lambda\delta\Sigma_k$ by setting $\lambda\delta\Sigma_k = u_k - \delta u_k$. In particular, this results in the same algorithm that was presented in the previous section as a superlinearly convergent algorithm. What makes the situation more interesting here is that the above derivation was done in the usual Newton–linearization and the resulting system of equations can be set up for a general free boundary problem also. Thus we claim that the previous approach can be applied only to a subclass of free boundary problems whereas this approach seems to be more general.

1.5 Main results of the thesis

So far we have only reviewed some solution strategies for free boundary problems. At this point we shall formulate several questions to be answered in this thesis.

1.5.1 Questions

The first questions arise from the shape optimization approach. It has already been pointed out that from the point of view of the solution algorithm the shape variational principle is mainly a tool for mathematicians. Does the shape variational principle lead to efficient solution strategies? For the least squares approach, which norm would be the best one and which formulation should be used to solve a free boundary problem? How should we formulate the state equation to obtain the solution easily.

After the shape optimization based approaches we reformulated our model free boundary problem as a fixed point iteration and constructed a rapidly converging algorithm. Which one is better, the least squares approach or the fixed point type approach? Can we always construct fixed point type iterations that converge superlinearly?

Finally we presented a generic form of a shape linearization procedure. Open question is that is it really generic? Can it be applied to any stationary free boundary problem? What about practical issues then? Is shape linearization the best approach to create optimal solution procedures for generic free boundary problems? How fast does it converge?

As our approach is numerical we want to address implementation aspects as well. If the optimal approach to solve free boundary value problems can not be implemented with regular finite element software, the results of our work has less impact. Therefore we raise the question, is it possible to automate the construction of a solution procedure to solve a free boundary problem? In particular, an optimal idea would be, that we could gain a solution method for any solvable free boundary problem by just describing the equations $A(u, \Omega) = 0$ and $B(u, \Omega) = 0$. This is for the time being state of the art in the finite element software for normal elliptic boundary value problems.

1.5.2 Answers

The first question deals with the shape variational principle and the efficiency of the solution algorithm. In general, gradient–based solution algorithms based on the variational principle lead to unoptimal solution algorithms. In a shape variational principle the solution to a free boundary problem is sought by minimizing an energy functional. Gradient-based methods to solve free boundary problems formulated with shape variational principle optimization-based solution algorithm are studied in references [KT96, Tii98]. By analyzing second order optimality condition in the solution of the free boundary problem we find out that the shape Hessian operates in $H^{\frac{1}{2}}(\Sigma) \times H^{\frac{1}{2}}(\Sigma)$ for elliptic $A(u, \Omega)$. This means that at each iteration the update to the geometry has to be smoothed to obtain a convergent algorithm based on the gradient of the cost functional. Therefore to obtain better convergence the gradient based formulation has to be preconditioned by an operator that affects spectrally like Hessian of the energy formulation. This aspect is studied in section 2.2.3, where the question of choosing a norm to the least squares formulation is also considered. With preconditioning the convergence of the solution algorithm can be improved but the optimal preconditioning requires evaluation of the shape Hessian.

The question of construction of a fixed point type iteration that converge superlinearly is related to shape differentiability of the state equation and the structure of the shape derivative. If the state solution appears to depend linearly on the small displacement of the free boundary the fixed point type methods can be easily derived. However, if the state solution depends on the derivatives of the displacements to the free boundary the fixed point type iterations are hard to derive.

The next question was about the shape linearization procedure. The shape linearization procedure can be treated as a Newton method in metric spaces and therefore it is generic. However, the differentiability required for the Newton method to work brings limits to the applicability of the shape linearization. Shape linearization collects all the good properties from fixed point type iteration and from the shape optimization based approaches. Although the shape linearization approaches asymptotically the Newton method the linearization procedure cannot converge quadratically as it requires smoothing to preserve the regularity of the geometry. The convergence is studied in the continuous case in paper [II].

Shape linearization procedure can be automated but the discrete variant looses some convergence properties compared to continuous calculations. Shape linearization is based on the knowledge that the shape gradient is supported on the free boundary. It turns out that the normal derivative of the test function plays an important role in the numerical implementation. Wheather as in the continuous analysis it is assumed that the normal derivative of the test function vanishes in the distributional sense the same assumption in the discrete implementation can not be enforced. However, it turns out that by post-processing the derivatives (linear matrices) of the shape linearized equations the assumption of a vanishing normal derivative of the test function can be implemented.

1.5.3 Authors contribution

In this section I report my contribution in this research subject. Our work started with research report [KT96] where we analysed systematically different shape optimization formulations for a model free boundary problem. The ideas in this paper was mainly from prof. Tiihonen and my duty was to implement the ideas by applying the technical shape sensitivity calculations. The ideas were then published in a conference proceeding paper [KT97]. In paper [I] we studied an impingement of a jet to a wall and made a first step towards an automated calculation of shape sensitivities. My main contributions in this paper were the systematic analysis of continuous formulation, calculation of possible wave lengths produced by surface tension, the discrete implementation of mixed calculation of shape sensitivities and one important contribution was to notice that the solution depended on the inflow position in the presence of surface tension forces. The paper was made in co-operation with prof. Tiihonen.

In paper [II] I studied the convergence of the Bernoulli free boundary problem when the curvature appears in the boundary term. This work is mainly an extension of a convergence proof of Flucher and Rumpf [FR97].

Some of the results presented in this thesis are published in the proceedings of Eccomas 2004 conference [KT04].

2 SHAPE SENSITIVITY ANALYSIS

Shape sensitivity analysis is a fundamental tool in shape optimization. Its goal is to analyse the relationship between the geometry and some property of the state solution. From a pragmatic point of view the shape sensitivity analysis corresponds to the calculation of partial derivatives with respect to the parameters that specify the geometry. However, it can also provide valuable theoretical insight of shape related problems.

In this chapter we shall give a short glance into the analysis of shape functionals using the universal approach, based on the work several authors, [Zol79, Sim80, SZ92, CH83, Del90, DZ91a, DZ91b, Zol92, DZ97, Ban90] and introduce the basic concepts of shape calculus. A major part of this chapter is devoted to different examples demonstrating the use of shape calculus in analysis of free boundary problems.

2.1 Differentiation with respect to shape

In this section we will present the toolbox to analyse functions and functionals that depend on the shape of the geometry.

2.1.1 Differential Geometry

In this section we will focus on some basic geometrical properties of domains, surfaces and curves.

We say that $\Omega \in C^{k,\gamma}$ if the boundary of Ω can be locally represented by $C^{k,\gamma}$–functions and that Ω is locally an epigraph of these functions, see [BC84]. Here we denote by $C^{k,\gamma}(\Omega)$ the space of Hölder continuous functions in Ω equipped with a norm

$$\|\phi\|_{C^{k,\gamma}(\Omega)} = \sup_{\substack{x\in\Omega \\ l\leq k}} |D^l\phi(x)| + \sup_{x\neq y} \frac{|D^k\phi(x) - D^k\phi(y)|}{|x-y|^\gamma}$$

For completeness we define $C^{k,0} = C^k$, k–times continuously differentiable functions.

The oriented distance function describes the oriented distance from a given domain. In the context of shape optimization the oriented distance function was introduced by Delfour and Zolésio in [DZ94]. It has several good properties especially when handling quantities defined only on the boundary. Oriented distance function b for domain Ω is defined by

$$b(x) = \begin{cases} \operatorname{dist}(x, \Omega), & \text{if } x \in \Omega^C, \\ 0, & \text{if } x \in \partial\Omega, \\ -\operatorname{dist}(x, \Omega^C), & \text{if } x \in \Omega. \end{cases} \tag{2.1}$$

With the distance function we define a neighbourhood for $(n-1)$–dimensional manifold Γ,

$$\Gamma_\mu = \{x \in \mathbb{R}^n \mid \operatorname{dist}(x, \Gamma) < \mu\}.$$

Oriented distance function is Lipschiz–continuous with Lipschiz–constant one:

$$|b(x) - b(y)| \leq \|x - y\|.$$

For C^k–domains with boundary $\Gamma = \partial\Omega$, $k \geq 2$, we know that there exists a constant $\mu > 0$ depending on Ω such that the oriented distance function is an element of space $C^k(\Gamma_\mu)$, see Gilbarg–Trudinger [GT77] page 282. In particular, the oriented distance function has the following properties:

Lemma 2.1. *Let Ω be a C^2–domain. Then*

(i). $\nabla b = \mathbf{n}$,

(ii). $\Delta b = \kappa$,

(iii). If further Ω is C^3, $\frac{\partial}{\partial n}(\Delta b) = -\sum_{i=1}^{n-1} \kappa_i^2$,

on boundary $\partial\Omega$, where κ_i are the principal curvatures of the boundary. Here mean curvature κ is defined by $\sum_{i=1}^{n-1} \kappa_i$.

Proof. For (i) and (ii) see [DZ94] for (iii) see paper [II] and [Kim00]. □

The oriented distance function is a powerful tool in studying functions defined only on the boundary [DZ94]. For a smooth domain we can define a projection $p(x) = x - b(x)\nabla b$, and we see, that for x close enough to $\partial\Omega$, $p(x) \in \partial\Omega$. This provides a tool to study functions defined only on the boundary and to estimate, how these functions depend on the change to the geometry. The oriented distance function is also used in tracking the position of a free surface in the numerical implementation. It is especially useful in dynamic free boundary problems and named as level–set method [OS88, Set90, SSO94, Bur03, Set96, Tor00, Smo01, BCT05].

2.1.2 Material derivative

In this section we will concentrate on the material derivative method developed by Zolèsio et al [Zol79, SZ92, Del90, DZ91a, DZ91b, Zol92, DZ97]. In the material derivative method the idea is to analyse how the quantities that depend on the geometry change when the geometry is perturbed with a velocity field such as in fluid mechanics. The material derivative method has its background in the continuum mechanics with the use of Lagrangian coordinates. In brief, the material derivative method gives the tools to answer the question: how does the solution of a PDE change when the geometry is varied with a given velocity field.

Let us first define a few notations. In the following we will denote by $\mathbf{x}$ a vector in $\mathbb{R}^n$, n is a dimension of the space, for us it is sufficient to consider $n = 2, 3$. Let us introduce an artificial time t, $t \in \mathbb{R}$ and a velocity field $V(t, \mathbf{x})$. We say that V is autonomous, if $V(t, \mathbf{x}) = V(0, \mathbf{x})$, so that the velocity field does not depend on a "time" variable t. Assume further that we have an open domain $D \subset \mathbb{R}^n$, and a domain $\Omega \subset D$. It is natural to define transformation $T_t(V)\mathbf{x} = X(t, \mathbf{x})$ with a velocity field V through differential equation

$$\frac{\partial X}{\partial t}(t, \mathbf{x}) = V(t, \mathbf{x}), \qquad X(0, \mathbf{x}) = \mathbf{x}.$$

This transformation is quite close to a perturbation of the identity [Del90], where the transformation was chosen by

$$T_t(V) = I + tV(\mathbf{x}).$$

For small perturbations these two transformations are close. However, if we apply a condition $V \cdot \mathbf{n} = 0$ on ∂D, the former transformation yields to domains that are contained in D. That is, the condition $T_t(V)(D) \subset D$ is fulfilled. The image of Ω under T_t is denoted by Ω_t.

Remark For the perturbation of the identity this property is not fulfilled as can be seen from the following two dimensional example: let D be a unit disk and use a velocity field $V(t, \mathbf{x}) = (-x_2, x_1)$. This velocity field rotates the unit disk but does not make any deformations to it if we use the deformation through the differential equation, so that $T_t(V)D = D$ for any t. For the perturbation of the identity we have for example $\mathbf{x} = (1, 0)$, $T_t(V)\mathbf{x} = (1, 0) + t(0, 1) = (1, t)$, which is obviously outside of the domain D.

Let there now be defined a domain functional $J : \Omega \to \mathbb{R}$. We say that the functional has a *directional Eulerian shape derivative* to direction V at Ω if the limit

$$\lim_{t \to 0+} \frac{J(\Omega_t) - J(\Omega)}{t} =: dJ(\Omega; V)$$

exists. If further $dJ(\Omega; V)$ is linear and continuous with respect to V, we say that J is *shape differentiable* at Ω.

For function $y(\Omega) \in W^{s,p}(\Omega)$, Ω in C^k, $k \geq 1$, $s \in [0, k]$, we define the material derivative as a limit

$$\dot{y}(\Omega; V) := \lim_{t \to 0} \frac{y(\Omega_t) \circ T_t(V) - y(\Omega)}{t}.$$

This limit may exist in a weak or a strong sense, and the material derivative is called respectively a weak or strong material derivative [SZ92]. We define also *shape derivative* y',

$$y'(\Omega; V) = \dot{y}(\Omega; V) - \nabla y(\Omega) \cdot V(0),$$

whenever it exists in a weak or a strong sense. Shape derivative y' represents the change of function y with respect to the geometry.

If function y and a vectorfield $V(x,t)$ are defined in a neighborhood of $\Gamma \subset \partial\Omega$ we define

$$\begin{aligned}
\nabla_\Gamma y &= \nabla y - (\nabla y \cdot \mathbf{n})\mathbf{n}, \\
\nabla_\Gamma \cdot V &= \nabla \cdot V - \mathbf{n} \cdot \nabla V \cdot \mathbf{n}, \\
\Delta_\Gamma y = \nabla_\Gamma \cdot \nabla_\Gamma y &= \Delta y - \kappa \frac{\partial y}{\partial n} - \frac{\partial^2 y}{\partial n^2}, \\
y'_\Gamma &= \dot{y}\big|_\Gamma - \nabla_\Gamma y \cdot V.
\end{aligned}$$

Here $\Delta_\Gamma \cdot$ is the Laplace–Beltrami operator [DL85].

If y is defined only on the surface Γ we can extend (for Ω smooth) y to the small neighborhood of Γ by taking it constant to the direction of the normal. Therefore $\frac{\partial y}{\partial n} = \frac{\partial^2 y}{\partial n^2} = 0$ and $\nabla_\Gamma y = \nabla y$, $\Delta_\Gamma y = \Delta y$ on Γ. These definitions for surface derivatives can be weakened in the usual distributional sense.

Further, a shape boundary derivative $\cdot'_\Gamma$ has following properties [Des95, DZ97]:

$$\begin{aligned}
(y_1\, y_2)'_\Gamma &= y_1{}'_\Gamma\, y_2 + y_1\, y_2{}'_\Gamma, \\
\mathbf{n}'_\Gamma &= -\nabla_\Gamma \langle V, \mathbf{n} \rangle, \\
\kappa'_\Gamma = (\Delta b)'_\Gamma &= -\Delta_\Gamma \langle V, \mathbf{n} \rangle + \frac{\partial \Delta b}{\partial n} \langle V, \mathbf{n} \rangle\,.
\end{aligned}$$

2.1.3 Derivatives of integrals

We are now ready to formulate the basic formulas for shape differentiation of integrals [CH53, GF63, SZ92]. In the following, we assume that Ω is bounded.

Lemma 2.2 ([SZ92]). *Let $f(\Omega_t) \in L^1(\Omega_t)$ be shape differentiable and $f'(\Omega_t) \in L^1(\Omega_t)$, $t \in [0,T]$, $T > 0$. If Ω_t is $C^{0,1}$–domain, ie Lipschiz–domain, then*

$$\left(\frac{d}{dt} \int_{\Omega_t} f(\Omega_t)\ dx \right)\bigg|_{t=0} = \int_\Omega f'(\Omega)\ dx + \int_{\partial\Omega} f(\Omega)\, \langle V, \mathbf{n} \rangle\ d\sigma. \tag{2.2}$$

For boundary integrals, we need more regularity for the boundary:

Lemma 2.3 ([SZ92]). *Let $f(\partial\Omega_t) \in L^1(\partial\Omega_t)$ be shape differentiable and $f'(\partial\Omega_t) \in L^1(\partial\Omega_t)$, $t \in [0,T]$, $T > 0$ and let Ω_t be a C^k–domain, $k \geq 2$. Then*

$$\frac{d}{dt} \int_{\partial\Omega_t} f(\partial\Omega_t)\ d\sigma \bigg|_{t=0} = \int_{\partial\Omega} f'_\Gamma(\partial\Omega)\, d\sigma + \int_{\partial\Omega} f(\partial\Omega) \kappa \, \langle V, n \rangle\ d\sigma, \tag{2.3}$$

where κ is the mean curvature of Γ.

In the above lemma the regularity of the domain is quite restrictive. Consider for example a trivial question: how does the arclength of a polygon change, when one cornerpoint is perturbed? We can not use the above lemma, since the regularity assumption is not fulfilled. In the next lemma this assumption is weakened so that the corner points are covered by the analysis. However, we first consider a two dimensional case, and then the three dimensional case. A three dimensional case is more difficult, since it contains both edges and corners.

Lemma 2.4 ([SZ92]). *Suppose that $\Omega \in \mathbb{R}^2$, $f \in H^{\frac{3}{2}}$, $\partial\Omega$ is piecewice C^k–domain, $k \geq 2$ and let a_i, $i = 1, \ldots, m$ be the cornerpoints of the boundary. Then*

$$\frac{d}{dt}\int_{\partial\Omega_t} f\ d\sigma\bigg|_{t=0} = \int_{\partial\Omega\setminus\{a_1,\ldots,a_m\}} \left(\frac{\partial f}{\partial n} + \kappa f\right)\langle V, \mathbf{n}\rangle\ d\sigma \\ + \sum_{i=1}^{m} f(a_i)\, V(a_i)\cdot(\tau^-(a_i) - \tau^+(a_i)), \quad (2.4)$$

where τ^- is the left - and respectively $\tau^+(a_i)$ right limit of tangential vector τ of the surface. Tangent vector is assumed to be oriented to counterclockwise direction.

For a three dimensional case we the have following proposition:

Proposition 2.1 ([SZ92]). *Suppose that $\Omega \in \mathbb{R}^3$ $f \in H^{\frac{3}{2}}$, $\partial\Omega$ is divided into two C^k, $k \geq 2$ boundaries $\partial\Omega_1$ and $\partial\Omega_2$ by a C^k, $k \geq 2$ path γ so that $\partial\Omega_1 \cap \partial\Omega_2 = \emptyset$ and $\partial\Omega = \partial\Omega_1 \cup \partial\Omega_2 \cup \gamma$. Then*

$$\frac{d}{dt}\int_{\partial\Omega_t} f\ d\sigma\bigg|_{t=0} = \int_{\partial\Omega\setminus\gamma} \left(\frac{\partial f}{\partial n} + \kappa f\right)\langle V, \mathbf{n}\rangle\ d\sigma + \int_{\gamma} f\, V\cdot(\nu^-(a_i) + \nu^+(a_i))\, d\gamma, \quad (2.5)$$

where ν^- is normal of the boundary of the surface Ω_1 on γ – and respectively ν^+ normal of the boundary of the surface Ω_2 on γ.

2.1.4 Structure theorems

Structure theorem of the shape gradient tells that the support of the shape gradient is contained in the boundary [DZ01].

Theorem 2.1 ([SZ92]). *Let $J(\Omega)$ be a real–valued shape function. Assume that for some $k \geq 0$ shape gradient $G(\Omega)$ exists for an open domain $\Omega \subset \mathbb{R}^n$, $G(\Omega)$ is continuous in $\mathcal{D}^k(\mathbb{R}^n, \mathbb{R}^n)$–topology and boundary of Ω is C^{k+1}. Then*

(i) there exist a scalar distribution $g(\partial\Omega)$ with support in $\partial\Omega$ such that $g(\partial\Omega) \in C^k(\partial\Omega)'$ and for all V in $D^k(\mathbb{R}^n, \mathbb{R}^n)$,

$$dJ(\Omega; V) = \langle g(\partial\Omega), \gamma_\Omega(V)\cdot\mathbf{n}\rangle_{C^k(\partial\Omega)}.$$

(ii) If further $g(\partial\Omega) \in L^1(\partial\Omega)$,

$$dJ(\Omega; V) = \int_{\partial\Omega} gV\cdot\mathbf{n}\, d\sigma$$

and $G = \gamma_\Omega^*(g\mathbf{n})$, where γ_Ω is the trace operator on $\partial\Omega$.

So the above theorem states that if we have a continuous shape gradient it can be represented by a distribution supported only on the boundary of the domain studied. As an application to the free boundary problems we recall the example in the introduction. Assume that u is given from space $H^{\frac{3}{2}}(D)$ and let $\Omega \subset D$ be the open domain with C^2–boundary. We shall use the variational functional from section 1.3.1

$$E^u(\Omega) := \inf_{u \in H_0^1(D)} E_1(u,\Omega) := \inf_{u \in H_0^1(D)} \int_D \nabla u \cdot \nabla u \, dx - \int_{D_\Omega} \lambda^2 \, dx.$$

Then

$$dE^u(\Omega; V) = \int_{\partial\Omega} \left(\nabla u \cdot \nabla \phi - \lambda \phi \kappa - \lambda \frac{\partial \phi}{\partial n} \right) V \cdot \mathbf{n} \, d\sigma$$

so that the shape gradient can be represented by a distribution

$$g(\partial\Omega) = \gamma_{\partial\Omega} \left(\nabla u \cdot \nabla \phi - \lambda \phi \kappa - \lambda \frac{\partial \phi}{\partial n} \right).$$

Let us now study the second order shape derivatives. Second order derivatives are important especially when analysing optimality conditions for shape functions. To introduce the second order shape derivative of a shape function we assume that there are smooth vector fields V and W defined in $[0,\tau] \times D$. Denote $\Omega_t(W) = T_t(W)(\Omega)$ with $T_t(W)$ being the transformation related to field W. Assume that $dJ(\Omega_t(W); V(t))$ exists for all $t \in [0,T]$. The second order Eulerian semiderivative at Ω in directions V, W is defined as [BZ97]

$$d^2 J(\Omega; V; W) := \lim_{t \to 0} \frac{DJ(\Omega_t(W); V(t)) - DJ(\Omega; V(0))}{t}$$

whenever the limit exists. Function J is twice shape differentiable at Ω if for all $V, W \in \mathcal{D}(\mathbb{R}^n, \mathbb{R}^n)$, $d^2 J(\Omega; V; V)$ exists and the map

$$h(V,W) \mapsto d^2(\Omega; V; W) : \mathcal{D}(\mathbb{R}^n, \mathbb{R}^n) \times \mathcal{D}(\mathbb{R}^n, \mathbb{R}^n) \to \mathbb{R}$$

is bilinear and continuous. Associate notation $H(\Omega)$ for the vector distribution in $(\mathcal{D}(\mathbb{R}^n, \mathbb{R}^n) \times \mathcal{D}(\mathbb{R}^n, \mathbb{R}^n))'$,

$$d^2(\Omega; V; W) = \langle H(\Omega), V \otimes W \rangle = h(V,W).$$

H is the shape Hessian of the shape function J. Here

$$(V \otimes W)_{ij}(x,y) = V_i(x) \, W_j(y), \quad i,j \in \{1, \ldots, n\}.$$

In the following we shall quote the structure theorem for the shape Hessian in smooth case [BZ97]. V and W are assumed to be autonomous fields and suppose that $g(\partial\Omega) \in H^{\frac{3}{2}}(\partial\Omega)$. J is supposed to be regular enough to guarantee regular extension for g say $Q(\Omega) \in H^2(\Omega)$ and that the map $\Omega \mapsto Q(\Omega)$ is shape differentiable, i.e. shape derivative $Q'(\Omega; V)$ exists for any $V \in \mathcal{D}(\mathbb{R}^n, \mathbb{R}^n)$. Since shape derivative depends only on the normal component of the perturbation, we can use this and define $\tilde{Q} : \mathcal{D}(\partial\Omega) \to L^2(\partial\Omega)$, $\tilde{Q}(v) = Q'(\Omega; V)$, where V is such that $\langle V, \mathbf{n}\rangle = v$. Now the structure of the shape Hessian can be written by

$$d^2J(\Omega; V; W) = \left\langle \tilde{Q}(W \cdot \mathbf{n}), V \cdot \mathbf{n} \right\rangle_{L^2(\partial\Omega)\times L^2(\partial\Omega)} + \mathcal{B}(V, W) + \int_{\partial\Omega} Q(\Omega) W_i \frac{\partial V_j}{\partial x_i} n_j \, d\sigma$$

where $\mathcal{B}(V, W)$ is a symmetric bilinear form,

$$\mathcal{B}(V, W) = \int_{\partial\Omega} \nabla \cdot (Q(\Omega)V) \langle W, \mathbf{n}\rangle - Q(\Omega) W_i \frac{\partial V_j}{\partial x_i} n_j \, d\sigma.$$

Symmetricity of $B(V, W)$ can be verified by transforming the above representation to the domain integrals [BZ97].

Calculation of the shape Hessian for shape optimization based approaches for free boundary problems at the solution gives an insight into the behavior of the solution. Second order optimality conditions require the Shape Hessian to be positive definite. Indefinite shape Hessian states that the critical point is a saddle point and therefore the solution of the free boundary problem may not be a solution of the optimization problem. In the following section a shape sensitivity analysis is applied for modified model problems to show different ways in which to analyze shape optimization based approaches for free boundary problems.

In general shape Hessian is not symmetric but it can be shown that for critical domains ie. domains in which the shape gradient is zero the shape Hessian is symmetric. A nonsymmetric part can be in particular presented by a shape gradient applied to a velocity field $(DV)W$ [DZ01]

2.2 Use of shape calculus in free boundary problems – an example

In this section we shall give a more detailed analysis of the free fluid surface problem studied in paper [I]. Let us next study two dimensional fluid flows under gravitation. The flow is assumed to be inviscid, incompressible and irrotational so that the potential flow formulation can be used. The problem to be studied can be formulated either by potential formulation,

$$\Delta u = 0 \quad \text{in } \Omega, \qquad \tfrac{\partial u}{\partial n} = 0 \quad \text{on } \Sigma, \tag{2.6a}$$

$$\nabla_\Gamma u = \theta \quad \text{on } \Sigma, \qquad \tfrac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma_b, \tag{2.6b}$$

$$\frac{\partial u}{\partial n} = -1 \quad \text{on } \Gamma_i, \qquad u = 0 \quad \text{on } \Gamma_o, \tag{2.6c}$$

or by stream function formulation,

$$\Delta \psi = 0 \quad \text{in } \Omega, \qquad \psi = 1 \quad \text{on } \Sigma, \tag{2.7a}$$

$$\frac{\partial \psi}{\partial n} = \theta \quad \text{on } \Sigma, \qquad \psi = 0 \quad \text{on } \Gamma_b, \tag{2.7b}$$

$$\frac{\partial \psi}{\partial n} = 0 \quad \text{on } \Gamma_i, \qquad \psi = y \quad \text{on } \Gamma_o. \tag{2.7c}$$

Here Γ_i is the inflow boundary, Γ_o is the outflow boundary, Σ is the free boundary and Γ_b is a fixed boundary (bottom of the geometry). In above equations θ depends on the height of the free boundary point,

$$\theta := \sqrt{2C_0 + \frac{2}{Fr}y},$$

where Fr is the Froude number, C_0 is the Bernoulli constant and y is the height at given point.

Let us consider the following functional $J(\Omega)$,

$$J(\Omega) := \inf_{u \in \mathcal{V}} E(\Omega, u), \quad E(\Omega, u) := \frac{1}{2}\int_\Omega |\nabla u|^2 \, dx + \frac{1}{2}\int_\Omega \theta^2 \, dx, \tag{2.8}$$

which depends both on the geometry of the fluid surface and on the flow velocity (norm of the gradient of potential/stream function). $E(\Omega, u)$ can be considered to be the sum of kinetic and potential energies. This energy formulation can be used both for stream function formulation and for potential function formulation, depending on the choice of space $\mathcal{V}$ that we vary the energy. It turns out that the critical points of J are the solutions of our free boundary problem when Ω is varied in a suitable set. In what follows we shall show this and study the nature of the critical points.

2.2.1 Potential flow formulation

The potential formulation can be derived from the energy E by choosing the function space $\mathcal{V} = \{v \in H^1(\Omega) \,|\, v = 0 \text{ at } \Gamma_o\}$. In this case the definition of energy E has to be modified to take into account the inflow boundary condition. Let us now introduce functional J_1,

$$J_1(\Omega) := \inf_{u \in \mathcal{V}} E_1(\Omega, u), \quad E_1(\Omega, u) := \frac{1}{2}\int_\Omega |\nabla u|^2 \, dx + \int_{\Gamma_i} u \, ds + \frac{1}{2}\int_\Omega \theta^2 \, dx. \tag{2.9}$$

The first order optimality conditions for E_1 (with respect to u) reads

$$dE_1(\Omega, u; v) = \int_\Omega \nabla u \cdot \nabla v \, dx + \int_{\Gamma_i} v \, ds = 0, \quad \text{for all } v \in \mathcal{V}, \tag{2.10}$$

and for $J_1(\Omega)$,

$$dJ_1(\Omega; V) = \int_\Omega \nabla u'_V \cdot \nabla u \, dx + \frac{1}{2}\int_\Sigma \left(\nabla u \cdot \nabla u + \theta^2\right) \langle V, \mathbf{n} \rangle \, ds = 0 \tag{2.11}$$

Proposition 2.2. *Let Ω be a critical point of J satisfying $C_0 + \frac{y}{Fr} > \epsilon > 0$ for all $(x, y) \in \Omega$, Ω is smooth enough. Then (Ω, u) is a solution of the free boundary problem* (2.6).

Proof. As u solves equation (2.10) for all $v \in \mathcal{V}$ we get that u solves the potential formulation (2.6) except the condition $\nabla u \cdot \nabla u = \theta^2$ has to be verified. First we perturb the equation (2.10) in both sides with respect to the geometry to get

$$\int_\Omega \nabla u'_V \cdot \nabla v \, dx + \int_\Sigma \nabla u \cdot \nabla v \, \langle V, \mathbf{n} \rangle \, dx = 0 \quad \text{for all } v \in \mathcal{V}.$$

As $u \in \mathcal{V}$ we substitute above $u \to v$ to get

$$\int_\Omega \nabla u'_V \cdot \nabla u \, dx = -\int_\Sigma \nabla u \cdot \nabla u \, d\sigma. \tag{2.12}$$

Substituting this to equation (2.11) we get

$$dJ_1(\Omega; V) = \frac{1}{2} \int_\Omega (\theta^2 - |\nabla u|^2) \langle V, \mathbf{n} \rangle = 0 \quad \text{for all suitable } V. \tag{2.13}$$

Therefore $\nabla u \cdot \nabla u = \theta^2$ must hold at critical point. □

Thus, to find one solution to FBP, it is sufficient to find a critical point of E_1. In order to be able to do that we have to analyse the second order optimality conditions so that we know how E_1 behaves near the critical points. Let V and W be two autonomous vector fields so that $\langle V, \mathbf{n} \rangle = \langle W, \mathbf{n} \rangle = 0$ on $\partial\Omega \setminus \Sigma$ and $W = V = 0$ at point $\Sigma \cap \Gamma_o$. We get from (2.13)

$$\begin{aligned} dE_1(\Omega, u; V) &= \frac{1}{2} \int_\Sigma \left(\theta^2 - |\nabla u|^2\right) V \cdot \mathbf{n} \, ds \\ &= \frac{1}{2} \int_\Omega \nabla \cdot \left(V \left(\theta^2 - |\nabla u|^2\right)\right) \, dx \end{aligned}$$

by applying the Stokes theorem. For the second order shape derivative we first have

$$\begin{aligned} dJ_1(\Omega, u; V, W) = &- \int_\Omega \nabla \cdot (V (\nabla u \cdot \nabla u'_W)) \, dx \\ &+ \frac{1}{2} \int_\Sigma \nabla \cdot \left(V \left(\theta^2 - |\nabla u|^2\right)\right) \langle W, \mathbf{n} \rangle \, ds \\ =& I_1 + I_2. \end{aligned} \tag{2.14}$$

Applying once more the Stokes theorem and integrating by parts on Σ we get

$$I_1 = -\int_\Sigma \nabla u \cdot \nabla u'_W \langle V, \mathbf{n} \rangle \, ds = \int_\Sigma \nabla_\Gamma \cdot (\langle V, \mathbf{n} \rangle \nabla_\Gamma u) \, u'_W \, ds,$$

For u'_W we can find a boundary condition from (2.12) by integrating by parts,

$$\frac{\partial u'_W}{\partial n} = \nabla_\Sigma \cdot (\langle W, \mathbf{n} \rangle \nabla_\Gamma u) \quad \text{on } \Sigma. \tag{2.15}$$

We now denote by P the Neumann to Dirichlet map which is defined by

$$P\mu = v\Big|_\Sigma$$

for v being solution of

$$\begin{aligned} -\Delta v &= 0 \quad \text{in } \Omega, \\ \frac{\partial v}{\partial n} &= \mu \quad \text{on } \Sigma, \\ v &= 0 \quad \text{on } \Gamma_b. \\ \frac{\partial v}{\partial n} &= 0 \quad \text{on } \partial\Omega \setminus (\Sigma \cup \Gamma_b), \end{aligned}$$

Then we can write at the solution

$$u'_W = P\left(\frac{\partial u'_W}{\partial n}\right) = P\left(\nabla_\Sigma \cdot \left(\langle W, \mathbf{n}\rangle \nabla_\Gamma u\right)\right) \quad \text{at } \Sigma. \tag{2.16}$$

I_2 can be splitted to

$$\begin{aligned} I_2 = &\frac{1}{2}\int_\Sigma (\nabla\cdot V)\left(\theta^2 - |\nabla_\Gamma u|^2\right)\langle W, \mathbf{n}\rangle \, ds \\ &+ \frac{1}{2}\int_\Sigma \langle V, \mathbf{n}\rangle \frac{\partial}{\partial n}\left(\theta^2 - |\nabla_\Gamma u|^2\right)\langle W, \mathbf{n}\rangle \, ds \\ &+ \frac{1}{2}\int_\Sigma (V_\Sigma)\cdot\nabla_\Gamma\left(\theta^2 - |\nabla_\Gamma u|^2\right)\langle W, \mathbf{n}\rangle \, ds, \end{aligned}$$

where $V_\Sigma = V - \langle V, \mathbf{n}\rangle \mathbf{n}$. The first and the third term disappear at the critical point so that we have for I_2

$$I_2 = \frac{1}{2}\int_\Sigma \frac{\partial}{\partial n}\left(\theta^2 - |\nabla_\Gamma u|^2\right)\langle V, \mathbf{n}\rangle \langle W, \mathbf{n}\rangle \, d\sigma. \tag{2.17}$$

We continue by differentiating

$$\frac{\partial \theta^2}{\partial n} = \frac{2}{Fr}\frac{\partial y}{\partial n} = \frac{2\mathbf{n}_y}{Fr},$$

where $\mathbf{n}_y$ is the upward pointing component of the normal vector $\mathbf{n}$. Finally, it follows from $\frac{\partial u}{\partial n} = 0$ on Σ that we have

$$\frac{\partial}{\partial n}|\nabla u|^2 = -2\kappa|\nabla_\Gamma u|^2 \quad \text{on } \Sigma, \tag{2.18}$$

where κ is the mean curvature of the surface Σ. Thus we get the final expression for the shape Hessian:

Proposition 2.3. *At any critical point of the energy E_1 the shape Hessian of E_1 has the expression*

$$\begin{aligned} d^2 J_1(\Omega, u; V, W) = &\int_\Sigma \nabla_\Gamma\cdot\left(\langle V, \mathbf{n}\rangle \nabla_\Gamma u\right) P\left(\nabla_\Sigma\cdot\left(\langle W, \mathbf{n}\rangle \nabla_\Gamma u\right)\right) ds \\ &+ \int_\Sigma \left(\theta^2\kappa + \frac{\mathbf{n}_y}{Fr}\right)\langle W, \mathbf{n}\rangle \langle V, \mathbf{n}\rangle \, ds. \end{aligned} \tag{2.19}$$

We observe that the Hessian is a continuous mapping from $H^{\frac{1}{2}}(\Sigma) \times H^{\frac{1}{2}}(\Sigma) \to \mathbb{R}$. This implies, among other things, that straightforward discretisation of the problem leads to a discrete problem where the condition number is inversely proportional to grid size.

A closer look at the shape Hessian tells us how the functional J_1 behaves near the solution. For big Froude numbers, say $Fr \gg 1$, we can see that the last term $\frac{\mathbf{n}_y}{Fr}$ is small and the first term majorizes the Hessian if the curvature of the surface Σ is near zero. Thus the operator is positive definite. Then we have a local minimum at every critical point. So we can use optimization to achieve the critical point. For small Froude numbers the last term gets bigger and we lose the positive definiteness so the critical point turns to a saddle point. This means that the free boundary is not a minimiser of the 'energy' of the system.

This can be in particular analysed by studying Hessian for a simple case, for a rectangular domain $\Omega := (0, 2\pi) \times (-1, 0)$. Then for Bernoulli constant $C_0 = 2$ the potential function is $u = 2\pi - x$. The shape Hessian can then be written in a simplified form,

$$d^2 J_1(\Omega; V, W) = \int_0^{2\pi} \frac{\partial V_y}{\partial x}(x,1) P\left(\frac{\partial W_y}{\partial x}(x,1)\right) + \frac{1}{Fr} V_y(x,1)\, W_y(x,1)\, dx.$$

Denote now by $v(x) = V_y(x, 1)$ and $w(x) = W_y(x, 1)$ respectively. Now we try to construct a velocity field W such that $d^2 J_1(\Omega; V, W) = 0$ for all V. This means that the shape Hessian is not strictly positive definite and therefore there may exist a solution to the free boundary problem near the given geometry. We shall search this in the form $w(x) = \sin(t_0\, x)$. Now as in paper [I] we obtain by separation of the variables $S^{-1}(v'(x)) = \frac{1}{\tanh(t_0)} \cos(t_0\, x)$, and we get

$$\begin{aligned} d^2\, J_1(\Omega; V, W) &= \int_0^{2\pi} v'(x) \frac{1}{\tanh(t_0)} \cos(t_0 x) + \frac{1}{Fr}\, v(x) \sin(t_0\, x)\, dx \\ &= \int_0^{2\pi} v(x) \sin(t_0\, x) \left(-\frac{t_0}{\tanh(t_0)} + \frac{1}{Fr} \right) dx. \end{aligned}$$

Thus $d^2 J_1(\Omega; V, W) = 0$ for all V by given form of W if

$$\tanh(t_0) = t_0\, Fr \quad \text{holds.} \tag{2.20}$$

For $Fr > 1$ there exists no such t_0 but for $Fr < 1$ there exists exactly one $t_0 > 0$ such that (2.20) holds. This corresponds to the formation of gravity waves [LK96], for which the above equation descibes the possible wavelengths as a function of the Froude number. This agrees with the linear theory for water waves [Sto57, Whi74].

Remark 2.1. *In a two dimensional case the gravity waves for this particular free boundary problem require use of nonreflecting boundary conditions. This is considered in references [Bai75, CL83, LM81, LT88, SZ96, JAY79, Clé96, Giv91].*

2.2.2 Stream function formulation

We now formulate the problem to a stream function form in a bounded region Ω. The mathematical formulation differs from the potential function formulation only in the boundary conditions. Our aim here is to check how this affects the shape Hessian. Let Ω now be a region with the boundary $\partial\Omega = \bar{\Sigma} \cup \bar{\Gamma}_i \cup \bar{\Gamma}_o \cup \bar{\Gamma}_b$. Let $\mathcal{V}(\Omega) = \{v \in H^1(\Omega) \,|\, v = 0 \text{ on } \Gamma_b \text{ and } v = 1 \text{ on } \Sigma\}$. We now formulate the free boundary problem by

$$J_2(\Omega) := \inf_{u \in \mathcal{V}} E(\Omega, u) \tag{2.21}$$

The variation of E with respect to parameter u gives us

$$dE(\Omega, u; v) = \int_\Omega \nabla u \cdot \nabla v \, dx \quad \text{for all } v \in H_0^1(\Omega).$$

So in the critical point of E u is a weak solution of

$$\begin{aligned} -\Delta u &= 0 \quad \text{in } \Omega, \\ \frac{\partial u}{\partial n} &= 0 \quad \text{on } \Gamma_i \cup \Gamma_o, \\ u &= 1 \quad \text{on } \Sigma, \\ u &= 0 \quad \text{at } \Gamma_b. \end{aligned} \tag{2.22}$$

The following proposition can be easily obtained repeating the steps taken in section 2.2.1:

Proposition 2.4. *Let V and W be two autonomous vector fields so that $\langle V, \mathbf{n}\rangle = \langle W, \mathbf{n}\rangle = 0$ at $\partial\Omega \setminus \Sigma$, $V = W = 0$ at point $\Sigma \cap \Gamma_i$. At any critical point of J_2 the shape Hessian of J_2 has the expression*

$$d^2 J_2(\Omega; V, W) = \int_\Sigma \langle V, \mathbf{n}\rangle \, \theta \, (S + I\kappa) \, \langle W, \mathbf{n}\rangle \, \theta \, d\sigma + \int_\Sigma \langle V, \mathbf{n}\rangle \, \langle W, \mathbf{n}\rangle \, \frac{\mathbf{n}_y}{Fr} \, d\sigma, \tag{2.23}$$

where $S = P^{-1}$ is the Steklov-Poincare operator on Σ (i.e. the Dirichlet to Neumann map).

As the operator S maps the Sobolev space $H^l(\Sigma)$ into $H^{l-1}(\Sigma)$, the minimal regularity for (2.23) to make sense is that V and W belong to $H^{\frac{1}{2}}(\Sigma)$. A closer look at the shape Hessian again tells us that the shape optimization works only for big Froude numbers for this formulation. Thus the situation is completely analogous to potential formulation.

2.2.3 Least squares formulations for free boundary problems

Let us now consider the shape analysis of least squares formulation of FBPs. To make the analysis transparent we shall first consider an abstract formulation. So let us introduce two conditions $f = f(u, \Sigma) = 0$ and $g = g(u, \Sigma) = 0$ so that the free boundary conditions (kinematic and dynamic) are satisfied if and only if $f = 0$ and $g = 0$. Moreover we assume that the condition $g = 0$ can be used as a boundary condition in the state problem. Then we introduce a functional J,

$$J(\Sigma) = \frac{1}{2} \int_\Sigma f^2 \, ds, \tag{2.24}$$

where u is a solution of the boundary value problem

$$\begin{aligned} \Delta u &= 0 \quad \text{in } \Omega, \\ g(u, \Sigma) &= 0 \quad \text{on } \Sigma, \end{aligned}$$

combined with the boundary conditions in the outflow and inflow boundaries. That is, the condition $f = 0$ is to be satisfied in the least squares sense. The Eulerian shape derivative of J to direction V reads now

$$dJ(\Sigma; V) = \int_\Sigma f'_V f \, ds + \int_\Sigma \left(\frac{\partial f}{\partial n} f + \frac{1}{2} f^2 \kappa \right) \langle V, \mathbf{n} \rangle \, ds.$$

Obviously this is zero at the solution of the FBP. The second order derivative at the solution of our FBP (where $f = 0$) is

$$d^2 J(\Sigma; V, W) = \int_\Sigma \left(f'_V + \frac{\partial f}{\partial n} \langle V, \mathbf{n} \rangle \right) \left(f'_W + \frac{\partial f}{\partial n} \langle W, \mathbf{n} \rangle \right) ds. \tag{2.25}$$

If f is defined only on the free boundary the above formula can be presented by shape boundary derivatives,

$$d^2 J(\Sigma; V, W) = \int_\Sigma f'_{\Gamma V} \, f'_{\Gamma W} \, ds. \tag{2.26}$$

Minimisation of the dynamic condition

In the potential function formulation it is difficult to impose the dynamic condition as a boundary condition for the state problem. Hence we choose

$$f_1 = |\nabla u|^2 - \theta^2$$

and

$$g_1 = \frac{\partial u}{\partial n} = 0 \quad \text{at } \Sigma$$

and define

$$J_1(\Sigma) = \frac{1}{2} \int_\Sigma {f_1}^2 \, ds. \tag{2.27}$$

which is minimised under constraint

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_{\Gamma_i} v \, ds \quad \text{for all } v \in \mathcal{V},$$

where $\mathcal{V} = \{ v \in H^1(\Omega) \, | v = 0 \text{ at } \Gamma_o \}$. The measure of Σ is also bounded below with some constant. This problem can be considered as an optimal control problem, where the control is the shape of Ω and the state is the velocity potential.

A straight calculation gives us

$$f_1{}'_V = 2 \nabla u \cdot \nabla u'_V = 2 \nabla_\Gamma u \cdot \nabla_\Gamma P \left(\nabla_\Sigma \cdot \left(\langle W, \mathbf{n} \rangle \nabla_\Gamma u \right) \right)$$

and

$$\frac{\partial f_1}{\partial n} = -2\kappa |\nabla_\Gamma u|^2 + \frac{2\mathbf{n}_y}{Fr}.$$

So we conclude the following proposition:

Proposition 2.5. *At any critical point of the cost functional J_1 the shape Hessian of J_1 has the expression*

$$d^2J_1(\Sigma;V,W) = 4\int_\Sigma \left[\nabla_\Gamma u\cdot\nabla_\Gamma P(\nabla_\Gamma\cdot\langle V,\mathbf{n}\rangle\,\nabla_\Gamma u) - \left(\kappa\theta^2 - \frac{\mathbf{n}_y}{Fr}\right)\langle V,\mathbf{n}\rangle\right]\cdot \\ \left[\nabla_\Gamma u\cdot\nabla_\Gamma P(\nabla_\Gamma\cdot\langle W,\mathbf{n}\rangle\,\nabla_\Gamma u) - \left(\kappa\theta^2 - \frac{\mathbf{n}_y}{Fr}\right)\langle W,\mathbf{n}\rangle\right]. \tag{2.28}$$

Clearly, the Hessian is positively semidefinite. However, it is still unbounded requiring $V,W\in H^1(\Sigma)$ to be well defined. Hence, in the discrete case, the conditioning behaves proportionally to the inverse of the grid size squared.

Minimisation of the kinematic condition of the stream function

As in previous sections we introduce a shape functional for a stream function and try to find a form for the Shape Hessian. Now we choose

$$f_2 = u - 1$$

and

$$g_2 = \frac{\partial u}{\partial n} + \alpha(u-1) - \theta. \tag{2.29}$$

So we minimise

$$J_2(\Sigma) = \frac{1}{2}\int_\Sigma {f_2}^2\,ds \tag{2.30}$$

under constraint

$$\int_\Omega \nabla u\cdot\nabla v\,dx = \int_\Sigma [\alpha(u-1)-\theta]\,v\,ds \quad \text{for all } v\in\mathcal{V},$$

where $\mathcal{V} = \{v\in H^1(\Omega)\,|\,v=0 \text{ on } \Gamma_b\}$. J_2 is well defined for all u in $\mathcal{V}$ and for regular enough Σ. Parameter α is here an additional parameter which can be chosen freely as $u=1$ at the solution. Clearly we have for $\frac{\partial f_2}{\partial n}$,

$$\frac{\partial f_2}{\partial n} = \frac{\partial u}{\partial n} = \theta$$

at the solution. For ${f_2}'_V$ we have

$${f_2}'_V = u'_V.$$

The boundary condition for a shape derivative u'_V on Σ reads in this case

$$\frac{\partial}{\partial n}[u'_V] + \alpha u'_V = \nabla_\Gamma\cdot(\langle V,\mathbf{n}\rangle\,\nabla_\Gamma u) + \kappa\theta\,\langle V,\mathbf{n}\rangle + \frac{\partial\theta}{\partial n}\langle V,\mathbf{n}\rangle - \alpha\frac{\partial u}{\partial n}\langle V,\mathbf{n}\rangle$$

but as we have $u=1$ at the solution the first term in the right hand side vanishes at the solution. Moreover we can now choose the parameter α so, that the right hand side and, consequently u'_V, vanishes also. That is, if we choose α,

$$\alpha = \kappa - \frac{\mathbf{n}_y}{Fr\theta^2}, \tag{2.31}$$

we have $u'_V = 0$ for all V at the solution.

We conclude here with the following propositions:

Proposition 2.6. *If we choose $\alpha = 0$ in* (2.29) *and we have $J_2(\Sigma) = 0$ the shape Hessian of the cost J_2 reads*

$$\begin{aligned} d^2 J_2(\Sigma; V, W) = \int_\Sigma &\left[P \left(\theta\kappa \langle V, \mathbf{n} \rangle + \frac{\partial \theta}{\partial n} \langle V, \mathbf{n} \rangle \right) + \theta \langle V, \mathbf{n} \rangle \right] \\ &\cdot \left[P \left(\theta\kappa \langle W, \mathbf{n} \rangle + \frac{\partial \theta}{\partial n} \langle W, \mathbf{n} \rangle \right) + \theta \langle W, \mathbf{n} \rangle \right] ds \end{aligned} \tag{2.32}$$

This Hessian is bounded in $L^2(\Sigma) \times L^2(\Sigma)$. Moreover, the 'main' part of the Hessian, i.e. the term arising from a shape derivative of the state is now regularising. This means that Hessian will remain bounded even if the condition $u = 1$ will be replaced with the true kinematic condition $\nabla_\Gamma u = 0$ on Σ.

Proposition 2.7. *If we choose $\alpha = \kappa - \frac{\mathbf{n}_y}{Fr\theta^2}$ in* (2.29) *and we have $J_2(\Sigma) = 0$ the shape Hessian of the cost J_2 reads*

$$d^2 J_2(\Sigma; V, W) = \int_\Sigma \theta^2 \langle V, \mathbf{n} \rangle \langle W, \mathbf{n} \rangle \, ds \tag{2.33}$$

So here we have gained the best possible Hessian for the problem – θ^2 times identity operator in $L^2(\Sigma)$.

Another way to formulate the stream function model is to set up a cost functional for the dynamic boundary condition. This formulation again leads to an unbounded Hessian as in the potential flow case, so a better way is to formulate the problem by this kinematic condition.

2.2.4 Internal free boundaries

In previous sections we analysed optimization based approaches to solve free boundary problems, where the free boundary was part of the boundary of the system. In this section we shall consider free boundary problems where the free boundary is between two materials. Let us in particular study the following free boundary problem,

$$-\Delta u = 0 \qquad \text{in } D_\Omega, \tag{2.34a}$$

$$u = 1 \qquad \text{on } \partial D, \tag{2.34b}$$

$$u = 0 \qquad \text{on } \partial\Omega, \tag{2.34c}$$

$$[\nabla u]_\Sigma \cdot \mathbf{n} = \lambda \qquad \text{on } \Sigma \tag{2.34d}$$

$$u = 1 \qquad \text{on } \Sigma. \tag{2.34e}$$

Here $[\nabla u]_\Sigma$ means the jump in the gradient on the boundary to the exterior direction from Ω. The geometry of the studied problem is described in Figure 3. Now domain Ω and D are fixed but between these domains there exists a free boundary which is formed as a discontinuity in the gradient of u. Let us first study a one dimensional case,

$$u''(x) = 0 \quad \text{for } x \in (0, 2) \setminus \{s\}, \tag{2.35a}$$

$$[u'(x)]_{x=s} = \lambda, \tag{2.35b}$$

$$u(s) = 1, \quad u(0) = 0 \quad \text{and } u(2) = 1, \tag{2.35c}$$

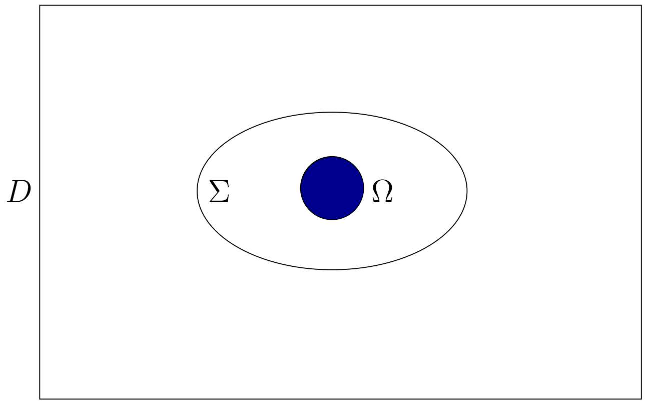


FIGURE 3 Geometry for the internal free boundary problem.

where

$$[u'(x)]_{x=s} := \lim_{x\to s^-} u'(x) - \lim_{x\to s^+} u'(x).$$

For $\lambda > \frac{1}{2}$ we have a solution to this free boundary problem,

$$u(x) = \begin{cases} \lambda x & \text{for } x \in [0, \frac{1}{\lambda}], \\ 1 & \text{for } x \in (\frac{1}{\lambda}, 2]. \end{cases}$$

Here the free boundary is at position $s = \frac{1}{\lambda} \in (0,2)$.

Let us first consider the following shape optimization formulation of this transmission type problem

$$\min_{s\in(0,2)} (u(s) - 1)^2 \quad \text{with } u - v \in H_0^1(0,2),\ v(x) := \frac{1}{2}x \tag{2.36}$$

$$\int_0^2 u'(x)\,\varphi'(x)\,dx = \lambda\varphi(s) \quad \text{for all } \varphi \in H_0^1(0,2). \tag{2.37}$$

Now equation (2.37) is uniquely solvable and the solution is

$$u(x) = \begin{cases} \frac{x - s\,(x-2)\,\lambda}{2} & \text{for } x \in [0,s], \\ \frac{x\,(1-(s-2)\,\lambda)}{2} & \text{for } x \in (s,2]. \end{cases} \tag{2.38}$$

By substituting the value of the solution at point s to the cost function, we get

$$J(s) = \frac{(s-2)^2\,(s\,\lambda - 1)^2}{4}.$$

We see that $J \in C^\infty(\mathbb{R})$. Let us study the optimality conditions at the minimum point ($s^* = \frac{1}{\lambda}$). The first order derivative of J with respect to free boundary position is

$$J'(s) = \frac{(s-2)\ (2\ (-1+s)\ \lambda - 1)\ (s\,\lambda - 1)}{2}.$$

The second order derivative is

$$J''(s) = \frac{1}{2} + \lambda\ (4 - 3\,s + (2 + 3\ (s-2)\ s)\ \lambda)$$

By substituting $s^* \to s$ we get

$$J''(s^*) = \frac{1}{2}(1-2\lambda)^2.$$

Obviously $J''(s^*) > 0$ for $\lambda > \frac{1}{2}$ and the second order optimality condition guarantees the existence of a local minima for the functional. Thus we can, for example, apply the quadratically convergent Newton method to obtain the solution.

As above we could solve this shape optimization problem straightforwardly but we could not observe the actual difficulty arising in this transmission type problem. The basic difficulty arises from the treatment of the function presenting the shape derivative of the state function. Consider an iterative solution algorithm where we first solve the state equation (2.37) and get a solution (2.38). Now the derivative of the solution is discontinuous on the free boundary and therefore we can not use a normal derivative of the solution for finding a new trial point for the free boundary. As we know the solution we can analyze that the left derivative of the solution is equal to λ. This suggests the following algorithm:

1. Set $k = 0$, choose initial guess $s_0 \in (0,2)$.
2. Evaluate $u_k(s_k) = \frac{1}{2}(s_k - \lambda {s_k}^2 + 2\lambda\, s_k)$.
3. Set $s_{k+1} = s_k - \frac{u_k(s_k)-1}{\lambda}$.
4. Update $k = k+1$ and continue from step 2 until converged.

This iteration converges, but not quadratically. We can calculate the convergence speed by differentiating the update $\delta(s) := \frac{u_k(s_k)-1}{\lambda}$ with respect to s,

$$\delta'(s)|_{s=\frac{1}{\lambda}} = 1 - \frac{1}{2\lambda}.$$

Hence $|s_{k+1} - s^*| \approx \frac{1}{2\lambda}|s_k - s^*|$. From here we can see that if λ is near $\frac{1}{2}$ the method converges very slowly, even near the solution. For bigger λ, say $\lambda >> 1$ method converges faster but still only linearly. Thus we have to consider the problem in more detail.

Let us now analyse the shape derivative of the solution u. Let us now study how we can apply the shape linearization method to this model problem. As the solution is not continuous we can not use normal derivatives on the boundary. One remedy in this case is to separate the solution to $u_1(x)$ and $u_2(x)$ and use the shape linearization method separately in each part. Now we use a weak formulation

$$\int_0^s {u_1}'(x)\,\varphi'(x)\,dx + \int_s^2 {u_2}'(x)\,\varphi'(x)\,dx - \lambda\,\varphi(s) = 0,$$
$$u_1(0) = 0, \quad u_1(s) - 1 = 0, \quad u_2(s) - 1 = 0 \text{ and } u_2(2) = 1.$$

From these equations we get the following shape linearized equations for $(\delta u_1, \delta u_2, \delta s)$

$$\int_0^s \delta u_1'(x)\,\varphi'(x)\,dx + \int_s^2 \delta u_2'(x)\,\varphi'(x)\,dx = \lambda\,\varphi(s) - \int_0^s u_1'(x)\,\varphi'(x)\,dx - \int_s^2 u_2'(x)\,\varphi'(x)\,dx, \quad (2.39)$$

$$\delta u_1(0) = -u_1(0), \quad \delta u_1(s) + u_1'(s)\,\delta s = 1 - u_1(s), \quad (2.40)$$

$$\delta u_2(s) + u_2'(s)\,\delta s = 1 - u_2(s) \quad \text{and} \quad \delta u_2(2) = 1 - u_2(2). \quad (2.41)$$

This gives us the following solution algorithm

1. Set $k = 0$, choose initial guesses $s_0 \in (0, 2)$, $u_{01}(x) \in H^1(0, s)$ and $u_{02}(x) \in H^1(s, 2)$ such that $u_1'(s_0) - u_2'(s_0) = \lambda$.
2. Solve $(\delta u_1, \delta u_2, \delta s)$ from equations (2.39)-(2.41).
3. Set $s_{k+1} = s_k + \delta s$, $u_{k+1i} = u_{ki} + \delta u_i$, $i = 1, 2$.
4. Update $k = k + 1$ and continue from step 2 until converged.

The above algorithm converges quadratically if the initial guess is near the solution.

Previous calculations with a one dimensional transmission-type free boundary problem gave us a small hint about the linearization procedure. First of all, the solutions were separated but connected via linearized equations. In the following we shall extend the result for a multidimensional case. We shall study the continuous casting of steel. In this case the physical situation is as follows. The melted steel comes into the domain from boundary Γ_i (see Figure 4). The steel solidifies inside the domain and the steel is pulled out from the tube by a constant velocity from boundary Γ_o. The temperature is controlled on boundary Γ_b with cooling or heating (temperature gradient is known in the normal direction).

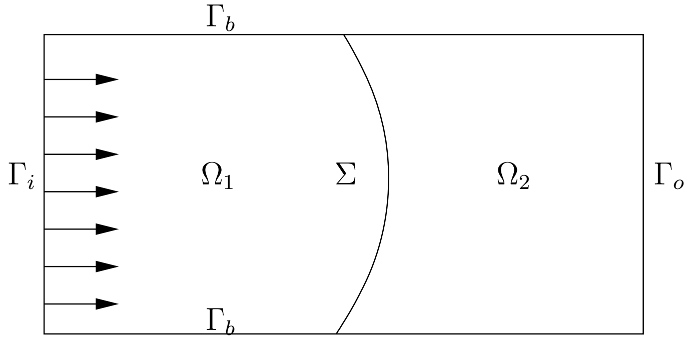


FIGURE 4 Geometry for the model transmission problem. Γ_b in the bottom and the top are the wall boundaries, Γ_i and Γ_o are the “inflow” and “outflow” boundaries.

Neglecting all but temperature diffusion and temperature convection by constant velocity we end up with the following simplified equations

$$-\Delta u + \vec{S}\cdot\nabla u = 0 \qquad \text{in } \Omega\setminus\Sigma, \tag{2.42a}$$

$$u = 0 \qquad \text{on } \Gamma_i, \tag{2.42b}$$

$$u = 2 \qquad \text{on } \Gamma_o, \tag{2.42c}$$

$$\frac{\partial u}{\partial n} = a \qquad \text{on } \Gamma_n, \tag{2.42d}$$

$$u = 1 \qquad \text{on } \Sigma, \tag{2.42e}$$

$$[\nabla u]_\Sigma \cdot \mathbf{n}_\Sigma = \lambda \qquad \text{on } \Sigma. \tag{2.42f}$$

Here $\Omega = \Omega_1 \cup \Omega_2$, u denotes the scaled temperature, Σ is the freezing position (at temperature $u = 1$) and a denotes the rate of boundary cooling.

We shall linearize the following form of the equations (we first set $\vec{S} = 0$ for simplicity without loss of generality).

$$\sum_{k=1}^{2}\left(\int_{\Omega_k}\nabla u_k\cdot\nabla\phi\,dx\right) - \int_\Sigma \lambda\phi\,d\sigma - \int_{\Gamma_n} a\phi\,d\sigma = 0,$$
$$\int_\Sigma (u_1-1)\psi\,d\sigma = 0,$$
$$\int_\Sigma (u_2-1)\xi\,d\sigma = 0.$$

For these equation we can apply shape linearization to achieve the following equations for δu_1, δu_2 and $\delta\Sigma$,

$$\sum_{k=1}^{2}\left(\int_{\Omega_k}\nabla\delta u_k\cdot\nabla\phi\,dx\right) + \sum_{k=1}^{2}\delta\Sigma(s_k)a(s_k)\varphi(s_k) = \int_\Sigma \lambda\phi\,d\sigma - \sum_{k=1}^{2}\left(\int_{\Omega_k}\nabla u_k\cdot\nabla\phi\,dx\right), \tag{2.43}$$

$$\int_\Sigma\left(\delta u_1 + \frac{\partial u_1}{\partial n}\delta\Sigma\right)\phi\,d\sigma = \int_\Sigma (1-u_1)\psi\,d\sigma, \tag{2.44}$$

$$\int_\Sigma\left(\delta u_2 + \frac{\partial u_2}{\partial n}\delta\Sigma\right)\phi\,d\sigma = \int_\Sigma (1-u_2)\psi\,d\sigma, \tag{2.45}$$

where s_k, $k = 1,2$ are the intersection points of the free boundary Σ and the wall Γ_n. Both derivatives $\frac{\partial u_1}{\partial n}$ and $\frac{\partial u_2}{\partial n}$ are taken in an outward normal direction from domain Ω_1. If $\frac{\partial u_1}{\partial n} \neq 0$ or $\frac{\partial u_2}{\partial n} \neq 0$ on free boundary Σ the linearized equation is solvable. Later we will study the convergence properties of the above algorithm by numerical implementation.

As a conclusion we have seen that the shape derivatives can be used to analyze the qualities of the different formulations of free boundary problems. Shape calculus provide the tools to derive efficient algorithms to solve FBPs. For internal free boundary problems shape calculus suggests that for optimal formulation the state solution is not continuous during the iteration but instead the solution is connected by a transmission–type condition on the free boundary.

3 IMPLEMENTATION ASPECTS

In this chapter we will describe the implementation of the shape linearization method. In the first section of this chapter we shall study the approximation of different geometrical quantities defined in discrete geometry. We shall propose approximations for curvature and normal vector although these quantities are not defined pointwise for polyhedral geometries.

In the second section we shall focus on the automatic calculation of the sensitivities for discrete equations. First we shortly review the automatic differentiation and discuss the applicability of the automatic differentiation to a discrete form of a free boundary problem. This is studied in detail for a one dimensional model of a free boundary problem. We observe the differences between continuous and discrete linearization.

In the third section we shall present numerical results. First we build a test bench to verify that the automatic differentiation produces the correct linearized equations. The shape linearization is applied to a simple model problem and the convergence properties of the algorithm is studied with a smooth case. The automatic shape linearization is applied to a die–swell problem and an internal free boundary problem.

3.1 Numerical approximation of the geometry

In this section we shall consider approximation of different geometrical quantities of the discrete geometry.

3.1.1 Approximation of geometric quantities

In a finite element method an infinite dimensional function space is approximated using a finite set of functions. These functions that form the basis of a discrete finite dimensional subspace of some functional space are usually defined in simple geometrical elements. A domain Ω is approximated by a collection of elements, $\Omega_h = \cup_i e_i$, where e_i denotes a single element (triangle, quadrilateral or tetrahedra etc). For a suitable approximation for the geometry it usually holds [KN90]

$$\|\Omega - \Omega_h\|_\infty \leq ch^2,$$

Let us now study the approximation of some geometric quantities of the boundary, like a normal vector field $\mathbf{n}$ and the mean curvature κ. The mean curvature appears in the shape derivative of the boundary integrals and it is essential in the shape linearization of free boundary problems.

First of all we shall sketch the smoothing of a normal vector field. For this we shall set up a finite dimensional subspace $\mathcal{W}_h$ defined on a discrete free boundary Σ_h. $\mathcal{W}_h$ might be for example the space of piecewise linear functions. Now we project the discrete normal vector field into the space $\mathcal{W}_h = \{\mathrm{span}\{\phi_i\}, i = 1, ..., N\}$ by solving the projection equation

$$\sum_{i=1}^{N} \mathbf{r}_i \int_{\Sigma_h} \phi_i \phi_j \, dx = \int_{\Sigma_h} \mathbf{n}_{\Sigma_h} \phi_j \, dx, \quad j = 1, ..., N. \tag{3.1}$$

Here $\int_{\Sigma_h} \phi_i \phi_j \, d\sigma$ is a boundary mass matrix, $\mathbf{n}_{\Sigma_h}(x) \in \mathbb{R}^n$ is the normal vector of the element at point $x \in \Sigma_h$ oriented outward of Ω_h and $r_i \in \mathbb{R}^n$ are the multipliers for a smoothed normal vector for basis functions in space $\mathcal{W}_h$. From this linear equation we can solve $\mathbf{r}$ to get a discrete smoothed normal vector field $\tilde{\mathbf{n}}_{\Sigma_h} = \sum_i \mathbf{r}_i \phi_i$.

Let us now study the approximation of the mean curvature. We shall use the following properties from differential geometry [Bän01, BMN04a]:

$$\Delta_\Gamma \vec{x} = \vec{\kappa}, \quad (n-1)\,\kappa = \vec{\kappa} \cdot \mathbf{n}, \tag{3.2}$$

where Δ_Γ is the Laplace-Beltrami operator on the surface and n is the dimension of the space so that (n-1) is the dimension of the boundary. $\vec{x} = (x_1, \ldots, x_n)$ are the coordinates of the boundary at a given point and $\vec{\kappa} = (\kappa_1, \ldots, \kappa_n)$ is curvature vector. We can multiply this equation by a smooth test function ϕ and integrate over the boundary to get for each x_i,

$$\int_\Sigma \Delta_\Gamma x_i \phi \, d\sigma = -\int_\Sigma \nabla_\Gamma x_i \cdot \nabla_\Gamma \phi \, d\sigma + \int_{\partial\Sigma} (\nu_\Sigma \cdot \nabla_\Gamma x_i)\, \phi \, d\gamma, \quad i = 1, \ldots, n.$$

Here ν_Σ is the normal of the boundary of Σ. ν_Σ is oriented outward of the boundary Σ and is orthogonal to the normal vector $\mathbf{n}$ of Σ.

In what follows we shall discretize equation (3.2) to achieve the following equation for a smoothed discrete curvature vector $\vec{\kappa}^h = \sum_{i=1}^{M} \mathbf{q}_i \phi_i$,

$$\sum_{i=1}^{M} \int_{\Sigma_h} \mathbf{q}_i \, \phi_i \, \phi_j \, d\sigma = -\int_{\Sigma_h} \nabla_\Gamma \vec{x} \cdot \nabla_\Gamma \phi \, d\sigma + \int_{\partial\Sigma_h} (\nabla_\Gamma \vec{x} \cdot \nu_{\Sigma_h})\, \phi \, d\gamma, \tag{3.3}$$

for $j = 1, \ldots, M$ and

$$(n-1) \sum_{i=1}^{M} \int_{\Sigma_h} q_i \, \phi_i \, \phi_j - \mathbf{q}_i \cdot \mathbf{n}_{\Sigma_h} \, \phi_i \, \phi_j \, d\sigma = 0, \quad \text{for } j = 1, \ldots, M. \tag{3.4}$$

As a matrix equation in a discrete case we get the following system of equations for the curvature:

$$\begin{bmatrix} \mathbf{M} & 0 \\ \mathbf{C} & (n-1)M \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ q \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}, \tag{3.5}$$

where matrices are

$$\mathbf{M}_{ij} := M_{ij} Id_{N\times N}, \quad M_{ij} := \int_{\Sigma_h} \phi_i \, \phi_j \, d\sigma \text{ and } \mathbf{C}_{ij} := Id_{N\times N} \int_{\Sigma_h} \mathbf{n}_{\Sigma_h} \phi_i \, \phi_j \, d\sigma,$$

and the force vector is

$$\mathbf{f}_j = -\int_{\Sigma_h} \nabla_\Gamma \vec{x} \cdot \nabla_\Gamma \phi_j \, d\sigma + \int_{\partial\Sigma_h} (\nabla_\Gamma \vec{x} \cdot \nu_{\Sigma_h}) \, \phi_j \, d\gamma.$$

After solving this linear equation we get an approximation for the mean curvature of the boundary Σ defined on Σ_h, $\tilde{\kappa}_h = \sum_i q_i \phi_i$. In reference [BMN04b] there is an error analysis for surface diffusion flows and especially they proved second order convergence for the error of the curvature in a curvature driven flow, if the surface can be represented locally by a graph and the surface is smooth enough. In [Smo01] the convergence of the approximated curvatures are estimated in less restrictive smoothness conditions. Clearly, as the regularity of the boundary reduces the convergence estimate of the curvature also reduces.

Remark 3.1. *By using smooth approximation (like Bezier-curves or Bezier-surfaces) for the boundaries the normal vector and the curvatures can be calculated straightforwardly using the smooth boundary approximation.*

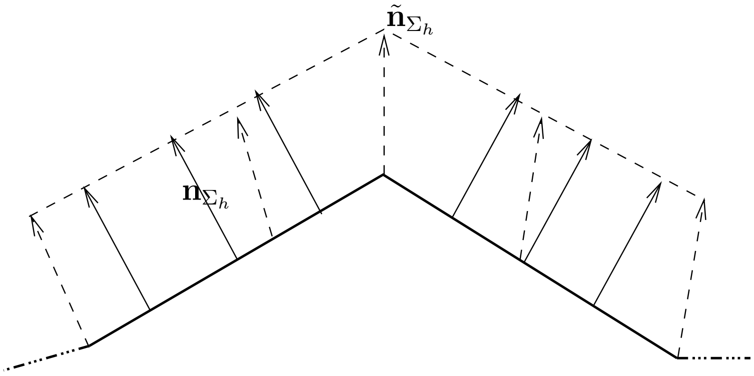


FIGURE 5 Smoothed normal vector field ($\tilde{\mathbf{n}}_{\Sigma_h}$, dashed) and actual normal vector field ($\mathbf{n}_{\Sigma_h}$, continuous line) for piecewise linear approximation.

3.1.2 Updating of the geometry

One important issue in the solution procedure of a free boundary problem is the updating scheme for the discrete geometry. We know the update for the geometry in each iteration only on the free boundary. For the finite element approximation the mesh would distort if we would move only the boundary nodes. Therefore we must extend the transformation of the boundary inside the domain.

One possible approach would be remeshing. By remeshing we mean that the finite element mesh is created in each iteration as the geometry is changed. This approach is computationally expensive but can be necessary in applications where the free boundary changes dramatically and the error in the solution is controlled by adaptation. Figure 6 shows the initial mesh and the final meshes for a two dimensional free boundary problem, where the mesh is created during the iteration steps. In a remeshing strategy the mesh topology changes and produces small discontinuity between the state solution and the geometry. The changing mesh topology can be for example a reordering of the elements or even new nodes and elements within the mesh.

In the following we shall discuss some issues to be taken into account when applying the remeshing strategy. Regular mesh generators take the position of the boundary as an input and produce the created mesh (i.e. nodal positions etc). As we get movement to the free boundary from the solution of shape linearized equations at each iterate we can pass this change to the mesh generator. Following aspects are important for implementation

- The change of mesh topology induces discontinuous change in the solution.
- When using a fixed point type iteration to solve a free boundary problem the remeshing should be frozen during the iteration. Freezing means that the internal nodal points will stop to move and the last few movements are done only to free boundary nodes.
- If the solution method requires data from the previous mesh, the solution has to be interpolated between two different meshes.

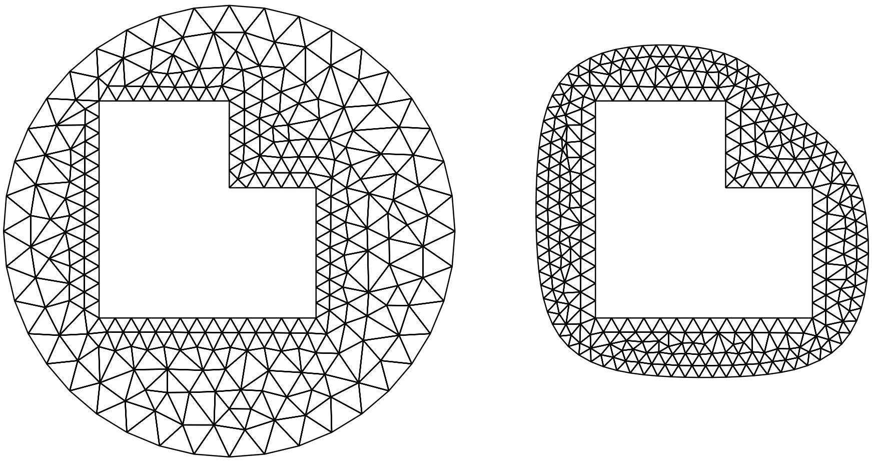


FIGURE 6 Initial mesh (left) and final mesh (right) with remeshing solution strategy.

Another way to update the finite element mesh geometry is to expand the boundary update inside the domain and thus deform the finite elements, see Figure 7. This is computationally more efficient since the mesh topology does not change during the iteration. However, in this approach the mesh can be distorted during the iteration, if the expansion is not chosen properly, see Figure 8. Let us now consider the extension of the free boundary update inside the finite element mesh.

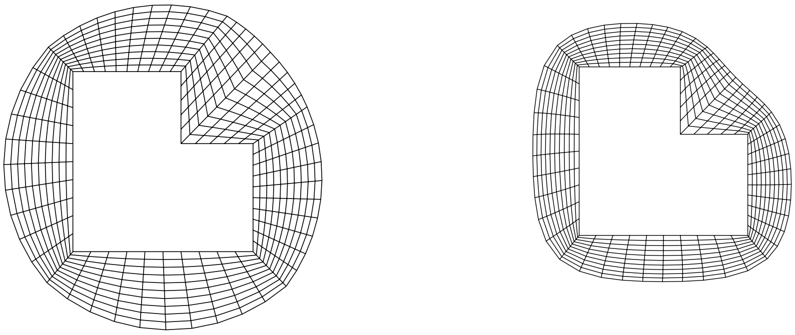


FIGURE 7 Initial mesh (left) and final mesh (right) with mesh perturbation strategy.

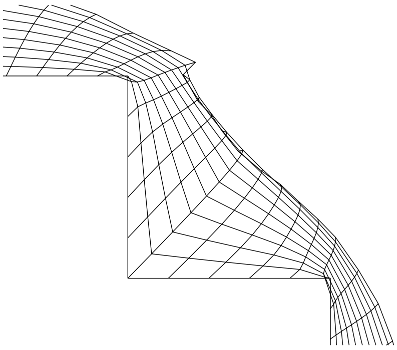


FIGURE 8 Distorted mesh when free boundary is moved in normal direction.

At each iteration step in the solution procedure we know the change to the current geometry on the free boundary in the normal direction. What is required now is that we extend this update to the inner part of the discrete domain Ω_h. We could decide the lines along which the geometry would move during the iteration and use a so called arbitrary Lagrangian–Eulerian (ALE)–method in the update of the geometry [SFDO91, Kno04, B.M96] or to use curvilinear coordinate system to handle the changes in the geometry [CS92].

In this thesis we shall use simple linear mapping to extend the change in the geometry from the free boundary to the internal parts of the geometry. Here we shall give the formulation for star–shaped initial geometry Ω for which it holds that there exists a point $\mathbf{x}_p$ for which all points $\mathbf{x} \in \partial\Omega$, $\mathbf{n}(\mathbf{x}) \cdot (\mathbf{x}_p - \mathbf{x}) \neq 0$. For this geometry we shall construct linear mapping $G_h : \mathbb{R}^{2m} \rightarrow \mathbb{R}^{2n}$ (in two dimensional case), where m is the number of freedoms of free boundary geometry and n is the number of node points of the mesh. This linear mapping is calculated from the initial mesh. In the following we shall give the algorithm to calculate this mapping for a star–shaped domain:

1. For each nodal point $\mathbf{x}_j$ find point $\mathbf{x}_f$ on a free boundary and point $\mathbf{x}_b$ in the radial direction of some fixed point in the middle of the star–shaped domain. Determine the element e_f for which point $\mathbf{x}_f$ belongs to.

2. For each basis function φ_k supported on e_f describing the free boundary geometry, define

$$\mathbf{g}_{jk} = \frac{\|\mathbf{x}_k - \mathbf{x}_b\|}{\|\mathbf{x}_f - \mathbf{x}_b\|^2}(\mathbf{x}_f - \mathbf{x}_b)\varphi_k(\mathbf{x}_b).$$

Matrix $(\mathbf{g}_{jk})$ is clearly very sparse and therefore the update of the geometry is efficient.

Remark 3.2. *Above linear mapping G_h can also be extended to simple movements to one coordinate direction and also for three dimensional case.*

3.2 Automatic shape calculus

In this section we consider automatic derivation of the equations in order to obtain the sensitivity equations for the iteration of the geometry. In the following diagram we sketch the approaches leading to discrete equations for iterate in the solution algorithm for a free boundary problem:

$$\begin{tikzcd}
\begin{array}{r} A(u,\Omega) = 0 \\ B(u,\Omega) = 0 \end{array} \arrow[r, "\text{discretization}"] \arrow[d, "\text{shape calculus}"] & \begin{array}{r} A_h(u_h,\Omega_h) = 0 \\ B_h(u_h,\Omega_h) = 0 \end{array} \arrow[d, "\text{automatic differentiation}"] \\
\begin{array}{rl} A_{,u}(u,\Omega)\,\delta u + A_{,\Sigma}(u,\Omega)\,\delta\Sigma & = A(u,\Omega), \\ B_{,u}(u,\Omega)\,\delta u + B_{,\Sigma}(u,\Omega)\,\delta\Sigma & = B(u,\Omega). \end{array} \arrow[r, "\text{discretization}"] & \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \mathbf{dq} \\ \mathbf{d}\boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1(\mathbf{q},\boldsymbol{\alpha}) \\ \mathbf{R}_2(\mathbf{q},\boldsymbol{\alpha}) \end{bmatrix}
\end{tikzcd}$$

In particular, choosing a different order (discretization → shape calculus or shape calculus → discretization) results in different discrete equations in the standard approach. We will see however that with suitable modifications we can change the order and end up with practically similar discrete equations.

3.2.1 Automatic differentiation

Automatic differentiation is a toolbox to achieve derivatives from functions without calculation of the derivatives by hand. It has developed during recent years mostly together with the improvement of the object oriented programming languages. In the numerical simulation of different real–life situations automatic differentiation can produce more exact and correct implementations [MKMA03]. In particular, the calculation of the derivatives can be hidden from the implementation by overloading the operations that evaluate the functions to be differentiated. However, there are still a few important issues to be left into consideration for the programmer in order to achieve efficient programs.

By definition directional derivative of a function f is a limit

$$df(x;y) := \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon y) - f(x)}{\varepsilon}.$$

In a computer program one can estimate this by difference, instead of taking a real limit, to take ε as a small parameter and approximating

$$df(x;y) = \frac{f(x+\varepsilon y) - f(x)}{\varepsilon} + \mathcal{O}(\varepsilon),$$

or by central difference

$$df(x;y) = \frac{f(x+\varepsilon y) - f(x-\varepsilon y)}{2\varepsilon} + \mathcal{O}(\varepsilon^2).$$

The latter gives better approximation but requires almost twice as many evaluations of function f at different points. Approximation of the derivatives by finite differences has been applied for solving a free boundary problem in [KT99], where we calculated the derivative of the smoothed mean curvature with respect to the geometry.

The above approach, although it seems to be natural by definition, has eventually some drawbacks. Firstly, it requires an evaluation of function f several times, and, if applied to the free boundary problems for example, it could require a solution of a state equation during each evaluation step. Secondly, the evaluation of the derivative includes a division within a small parameter ε, which results in an increase of the approximation errors. Thus the resulting derivative is less accurate than the function value $f(x)$.

In automatic differentiation the idea is to provide derivatives based on the computer code, say the automatic differentiation results a program that calculates the derivative of f analytically. A computer program can abstractly be represented by elementary unary and binary operators. A computer program that calculates some values from the given data can be represented by a function $\Theta : \mathbb{R}^n \to \mathbb{R}^m$, $\Theta(\mathbf{a}) = \mathbf{b}$, where vector $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{R}^n$ is the given data, and $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{R}^m$ is the output of the calculation. The calculation can be presented by

$$\begin{aligned}
t_1 &= f_1(\mathbf{a}),\\
t_2 &= f_2(\mathbf{a}, t_1),\\
t_3 &= f_3(\mathbf{a}, t_1, t_2),\\
&\vdots\\
t_p &= f_p(\mathbf{a}, t_1, t_2, \ldots, t_{p-1}),\\
\mathbf{b} &= \mathbf{F}(\mathbf{a}, t_1, t_2, \ldots, t_p),
\end{aligned}$$

where t_i, $i = 1, \ldots, p$ are temporary variables to store data.

The aim in automatic differentiation is the calculation of the Jacobian

$$J = \begin{bmatrix}
\frac{\partial b_1}{\partial a_1} & \frac{\partial b_1}{\partial a_2} & \cdots & \frac{\partial b_1}{\partial a_n}\\
\frac{\partial b_2}{\partial a_2} & \frac{\partial b_2}{\partial a_2} & \cdots & \frac{\partial b_2}{\partial a_n}\\
\cdots & \cdots & \cdots & \cdots\\
\frac{\partial b_m}{\partial a_2} & \frac{\partial b_m}{\partial a_2} & \cdots & \frac{\partial b_m}{\partial a_n}
\end{bmatrix},$$

which characterises the dependence of the result from the given data in the sense of exact derivative of the program code. Applying this to a free boundary problem, $\mathbf{a}$ could consist of representation of the free boundary (node points, spline control points etc) and $\mathbf{b}$ could be, for example,

$$\mathbf{b} \equiv (b_1), \quad b_1 \equiv \|B(u, \Sigma)\|_*^2$$

so that the calculation of the Jacobian J is essentially calculation of a gradient $\nabla_{\mathbf{a}} b_1$.

Automatic differentiation has two approaches to calculate derivatives: forward and reverse modes [Gri89, HM03]. In brief, a forward mode collects the needed derivatives during the calculation and a reverse mode stores the operations to calculate the derivatives after the calculation is done. A reverse mode can be compared to the adjoint state technique for optimal control problems, since it can be formulated by using an adjoint state variable see Pironneau in [KPTZ00] for example.

In what follows we study how to retain the relative 'compactness' of the continuous approach and still exploit the ease of automatic differentiation. In the case of a sensitivity analysis for a finite difference method this has been studied by Borggaard et al [BV00], who proposed to modify the functional that was to be differentiated. Here we shall use a different approach. We shall study how the finite element software should be modified to produce continuous like shape derivatives of the coded residuals R_1 (discrete version of A) and R_2 (discrete version of B).

From a numerical point of view, we apply the automatic differentiation to calculate the derivative of residual $(\mathbf{R}_1(\mathbf{q}, \boldsymbol{\alpha}), \mathbf{R}_2(\mathbf{q}, \boldsymbol{\alpha})) \in \mathbb{R}^{n \times m}$, where m is the degrees of freedom in the geometry design and n is the degrees of freedom of state. Here $\mathbf{q}$ denotes the state solution and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m)$ stands for geometry design vector, which defines the geometry of the finite element mesh. A Newton step to find the zero point of the residual consists of solving a linear system

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \mathbf{dq} \\ \mathbf{d}\boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1(\mathbf{q}, \boldsymbol{\alpha}) \\ \mathbf{R}_2(\mathbf{q}, \boldsymbol{\alpha}) \end{bmatrix}. \tag{3.6}$$

Here A_{11} is the derivative of residual $\mathbf{R}_1$ with respect to $\mathbf{q}$, A_{12} is the derivative of residual $\mathbf{R}_1$ with respect to $\boldsymbol{\alpha}$ and respectively A_{21} derivative of $\mathbf{R}_2$ with respect to $\mathbf{q}$, A_{22} derivative of $\mathbf{R}_2$ with respect to $\boldsymbol{\alpha}$. Now the efficiency of above method depends how easily this system is solved. The most important issue in this context is the sparsity of these matrices blocks.

The derivatives in the above with respect to $\mathbf{q}$ are standard and usually these matrices are sparse. Here we shall focus on the differentiation with respect to the geometry. As was initially introduced with the continuous formulation of residuals A and B the residual A stands for elliptic BVP. A quite common way to implement a finite element program is to use a mesh generation to produce a finite element mesh. To obtain nodal movement of the finite element mesh the mesh generation procedure must then be differentiated with respect $\boldsymbol{\alpha}$. In the worst case this derivative can be a large matrix which is not so sparse. The derivative of the mesh generation routine can then be used in the calculation of a derivative of residuals with respect to the geometry ($\boldsymbol{\alpha}$). Again, in the worst case this can result in matrix A_{12} which is not sparse thus leading to an inefficient solution procedure.

The continuous shape sensitivity analysis proposes that the shape gradient only depends on the normal component of the perturbation velocity field. However, for a discrete formulation the movement of internal nodal points also affect the solution. In general, internal nodal movements affect less on the solution than a boundary node movement, i.e. the solution is more sensitive to the boundary node changes.

In this thesis we study an approach where the discrete shape sensitivities are modified to correspond to continuous shape sensitivities. To obtain more insight in this we shall first study a model one dimensional free boundary problem.

3.2.2 One dimensional model free boundary problem

Our aim is to automate the calculation of the equations for the shape linearization procedure. However, we approach this in a semi discrete way, that is, let us first study the following one dimensional free boundary problem and its discretization: find solution $u(x)$ and position s such that

$$\begin{aligned} -u''(x) &= 1 \quad x \in (0, s), \\ u(0) &= 0, \\ u(s) &= 1, \\ u'(s) &= \frac{1}{2}. \end{aligned}$$

This problem has a unique solution assuming $s > 0$, $u(x) = -\frac{1}{2}x^2 + \frac{3}{2}x$ and $s = 1$.

Applying shape linearization for the above problem we get an equation for the iterate $(\delta u, \delta s)$:

$$\int_0^s \delta u'(x)\phi'(x)\,dx - \delta s\,\phi(s) = \int_0^s u'(x)\phi'(x) - \phi(x)\,dx - \frac{1}{2}\phi(s), \tag{3.7a}$$

$$\delta u(s) + u'(s)\,\delta s = u(s) - 1. \tag{3.7b}$$

Here we have assumed without loss of generality that $\phi'(s) = 0$. Assuming, that u solves $u''(x) = 1$ for $x \in (0, s)$ we get that $\delta u''(x) = 0$ for $x \in (0, s)$, $\delta u(0) = 0$ and $\delta u'(s) = \delta s - u'(s) - \frac{1}{2}$. Substituting $\delta u(x) = ax$ we get the following system for pair $(a, \delta s) \in \mathbb{R}^2$:

$$\begin{bmatrix} 1 & -1 \\ s & u'(s) \end{bmatrix} \begin{bmatrix} a \\ \delta s \end{bmatrix} = \begin{bmatrix} u'(s) - \frac{1}{2} \\ u(s) - 1 \end{bmatrix}$$

we see that this equation is solvable, if the determinant $u'(s) + s \neq 0$. Solving this we get the following algorithm:

1. Set k=0, choose initial guess s_0 and $u_0(x)$ so that $u_0''(x) = 1$ for $x \in (0, s_0+1)$, $u_0(0) = 0$ and $u_0'(s_0) = \frac{1}{2}$

2. Solve

$$\delta s_k = \frac{s_k - 2 + 2\,u_k(s_k) - 2\,s_k\,u_k'(s_k)}{2\,\left(s_k + u_k'(s_k)\right)}$$

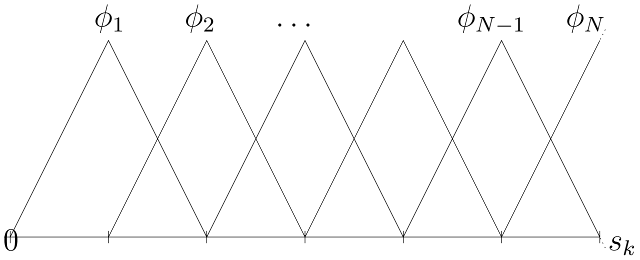


FIGURE 9 Numbering of piecewise linear basis functions in one dimensional case.

and

$$\delta u_k = \frac{u_k(s_k) - 2 + 2\,u_k(s_k)\,u_k'(s_k)}{2\,\left(s_k + u_k'(s_k)\right)}\,x.$$

3. Update $s_{k+1} = s_k + \delta s_k$, $u_{k+1} = u_k + \delta u_k$ and $k = k+1$. Repeat steps 2 and 3 until $\|(\delta u_k, \delta s_k)\|$ small enough.

It is easy to see, that if we are near the solution, we obtain a quadratic convergence rate for the iteration $(s_k, a_k) \to (s_{k+1}, a_{k+1})$:

Lemma 3.1. *Assume that* $u_k(x) = \left(-\frac{1}{2}x^2 + a_k x\right)$, $|s_k - 1| < \epsilon$ *and* $|a_k - \frac{3}{2})| < \epsilon$ *with* $\epsilon < \frac{1}{2}$. *Then* $|s_{k+1} - 1| < \epsilon^2$ *and* $|a_{k+1} - \frac{3}{2}| < \epsilon^2$.

Next our aim is to discretize the above iterative scheme and study the convergence of this discretized problem. We shall see that fast convergence can be obtained only by careful implementation.

First we approximate the solution of our one dimensional model free boundary problem by $u_h = \sum_{i=1}^{N} q_i \phi_i$, where ϕ_i, $i = 1, \dots, N$ are piecewise linear basis functions and q_i, $i = 1, \dots, N$ are the multipliers for the basis functions (see Figure 9). Now the discrete version of (3.7) reads as

$$\int_0^s \delta u_h{}' \phi_j{}'\,dx - \delta s\,\phi_j(s) = \int_0^s u_h{}'\phi_j{}' - \phi_j\,dx - \frac{1}{2}\phi_j(s),$$
$$\delta u_h(s) + u_h{}'(s)\,\delta s = u_h(s) - 1,$$

where $\delta u_h = \sum_{i=1}^{N} \delta q_i\,\phi_i$.

The resulting system of equations can be written in matrix form,

$$A\,\delta\mathbf{q}^k + \mathbf{b}\delta s_k = A\mathbf{q}^k - \mathbf{f}, \tag{3.8a}$$
$$\mathbf{c}^T\,\delta\mathbf{q}^k + d\,\delta s^k = \mathbf{c}^T\,\mathbf{q}^k - 1, \tag{3.8b}$$

where superscript $\cdot^k$ denotes the kth iteration. Here $A_{ij} = \int_0^s \phi_i{}'\phi_j{}'\,dx$, $\mathbf{b}_i = -\phi_i(s)$, $C_i = \phi_i(s)$, $d = u_h{}'(s)$ and $\mathbf{f}_i = \int_0^s \phi_i\,dx + \frac{1}{2}\phi_i(s)$. Now one iteration step in discrete case consists of evaluation of A, $\mathbf{b}$, $\mathbf{c}$ and d and solving of pair $(\delta\mathbf{q}^k, \delta s_k)$ from system (3.8). The new candidates for the solution are then sought by setting $(\mathbf{q}^{k+1}, s_{k+1}) := (\mathbf{q}^k + \delta\mathbf{q}^k, s_k + \delta s_k)$.

Let us now study the convergence of the above discrete equation near the solution. We shall consider the convergence of this problem separately, first with respect to the state.

Lemma 3.2. *Assume that $s_k = 1$ and interval $[0, s_k]$ is divided into N intervals with length $h = \frac{s_k}{N}$. Then $\delta s_k = 0$ for any $\mathbf{q}^k$ such that $u_h{}'(s_k) \neq -s_k$.*

Proof. Assuming that we have divided the interval with N elements with length h we get that matrix A has a form

$$A = \frac{1}{h} \begin{bmatrix} 2 & -1 & 0 & \multicolumn{2}{c}{\dots\dots\dots\dots} \\ -1 & 2 & -1 & 0 & \dots\dots \\ \multicolumn{5}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots} \\ \dots\dots & 0 & -1 & 2 & -1 \\ \dots\dots\dots\dots & & 0 & -1 & 1 \end{bmatrix}.$$

The inverse of matrix A is

$$A^{-1} = \begin{cases} j\,h & \text{for } j \leq i \\ i\,h & \text{for } j > i \end{cases}$$

Vectors $\mathbf{f}$, $\mathbf{b}$ and $\mathbf{c}$ are

$$\mathbf{f} = \begin{pmatrix} h \\ \vdots \\ h \\ \frac{h}{2} + \frac{1}{2} \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix} \quad \text{and} \quad \mathbf{c} = -\mathbf{b}$$

and the scalar $d = u_h{}'(s_k)$. Next we shall show that the full linear system is solvable. This is accomplished by noticing that the determinant of the matrix

$$\det \begin{bmatrix} A & \mathbf{b}^T \\ \mathbf{c} & d \end{bmatrix} = \frac{d + s_k}{h^N} \neq 0 \quad \text{by assumption.}$$

Solving δs_k from system (3.8),

$$\delta s_k = \frac{\mathbf{c}^T A^{-1} \mathbf{f} - 1}{d - \mathbf{c}^T A^{-1} \mathbf{b}}.$$

Now $\mathbf{c}^T A^{-1} = h\,(1, 2, \dots, N)$ and therefore

$$\mathbf{c}^T A^{-1} \mathbf{f} = \frac{(Nh)^2 + Nh}{2} = \frac{s_k{}^2 + s_k}{2},$$

from the mesh setup $Nh = s_k$. But as $s_k = 1$ we have $\mathbf{c}^T A^{-1} \mathbf{f} - 1 = 0$ which concludes the proof with the remark that the divisor

$$d - \mathbf{c}^T A^{-1} \mathbf{b} = d + hN = d + s_k \neq 0.$$

□

The above lemma is quite obvious, since we have a linear state equation under consideration. The convergence with respect to the free boundary point is more delicate. In particular, we note that the approximation of the derivative of the discrete solution plays an essential role in the convergence of the free boundary point. For a simple approximation we can only obtain a linear convergent rate which depends on the mesh size h:

Lemma 3.3. *Assume that*

$$u_h'(s) := \frac{\mathbf{q}_N - \mathbf{q}_{N-1}}{h}, \tag{3.9}$$

$\mathbf{q}^k{}_i = \left(-\frac{1}{2}\left(\frac{i}{N}\right)^2 + \frac{3}{2}\frac{i}{N}\right)$ *and that* $|s_k - 1| < \epsilon$. *Then* $|s_{k+1} - 1| < c_1\,\epsilon\,h + c_2\,\epsilon^2$.

Proof. For given $\mathbf{q}^k$ we have $\frac{\mathbf{q}^k{}_N - \mathbf{q}^k{}_{N-1}}{h} = \frac{h+s_k}{2{s_k}^2}$. Substituting this to the formula calculated in the proof of lemma 3.2 we get

$$\delta s(s_k) = \frac{{s_k}^2 + s_k - 2}{2s_k + \frac{h+s_k}{{s_k}^2}}.$$

We see that $\frac{\partial \delta s}{\partial s_k}(1) = \frac{3}{3+h}$. Using Taylor series we conclude the estimate. □

Using some post–processing for the solution we can obtain a quadratically convergent iteration:

Lemma 3.4. *With the assumptions of lemma 3.3, except approximate the derivative of* u_h *by*

$$u_h'(s) := \frac{(3\mathbf{q}_N - 4\mathbf{q}_{N-1} + \mathbf{q}_{N-2})}{2h}. \tag{3.10}$$

Then $|s_{k+1} - 1| < c\,\epsilon^2$.

Proof. Now $u_h'(s_k) = \frac{1}{2Nh} = \frac{1}{2s_k}$. Thus

$$\delta s_k(s_k) = \frac{{s_k}^2 + s_k - 2}{3s_k}, \quad \text{and} \quad \delta s_k'(1) = 1.$$

□

Lemma 3.4 shows that careful implementation is needed to obtain better than linear convergence in a discrete case. Post processing information was required - even in the most simple, one dimensional free boundary problem. Therefore much attention must be paid into implementation of a multidimensional free boundary problem solver.

Remark 3.3. *In a general case the convergence can not be independent of the mesh size since the finite element error itself does not obey the super convergence property, which is the case in this simple one dimensional example, where the solution can be presented by a quadratic function.*

3.2.3 Residual based finite element implementation of 1D problem

Our next aim is to build a finite element implementation for the previous one dimensional free boundary problem. We shall use a residual based finite element implementation, where only the residual form of the equations is coded and the matrices are calculated using automatic differentiation. We shall try separately two different approaches. The first one uses a straightforward implementation where the movement of the inner nodal points is also taken into account. In the second approach we only use movement of the last nodal point to calculate the shape derivatives. This approach suggests modifications to the calculation of the shape sensitivities to tune up the automatic calculation of shape linearised equations.

In a residual based finite element calculation the form of the weak equations is coded in a straightforward manner. We now assume that we have a geometry parameter α that describes the position of the free boundary. The finite element nodal points for the one dimensional free boundary problem are given by

$$x_k := k\frac{\alpha}{N}.$$

Thus the nodal points of the mesh are moved as α is changed. The residuals that describe the solution are now

$$r_i(\mathbf{q},\alpha) := \int_0^\alpha \sum_{k=1}^N q_k {\varphi_k}'(x){\varphi_i}'(x) - \varphi_i(x)\,dx - \frac{1}{2}\,\varphi_i(\alpha), \tag{3.11}$$

$$r_{N+1}(\mathbf{q},\alpha) := \sum_{k=1}^N q_k\varphi(\alpha) - 1. \tag{3.12}$$

We further denote by

$$\begin{aligned} \mathbf{r} &= (r_1, r_2, \ldots, r_{N+1}), \\ \mathbf{s} &:= (q_1, q_2, \ldots, q_N, \alpha) \quad \text{and} \\ \delta\mathbf{s} &:= (\delta q_1, \delta q_2, \ldots, \delta q_N, \delta\alpha). \end{aligned}$$

In a standard finite element implementation the shape functions $\varphi_i(x)$ depend on the parameter α. This results in the following residuals:

$$\mathbf{r}(\mathbf{s}) = \begin{pmatrix} \frac{N(2q_1-q_2)}{\alpha} - \frac{\alpha}{N} \\ \frac{N(-q_1+2q_2-q_3)}{\alpha} - \frac{\alpha}{N} \\ \vdots \\ \frac{N(-q_{N-2}+2q_{N-1}-q_N)}{\alpha} - \frac{\alpha}{N} \\ \frac{N(-q_{N-1}+q_N)}{\alpha} - \frac{\alpha}{2N} - \frac{1}{2} \\ q_N - 1 \end{pmatrix}. \tag{3.13}$$

In the second approach we perturb only the last nodal point of the finite element mesh. Further, the basis functions on the last element are assumed to be fixed with respect to the geometry change and the basis functions are continued continuously over the free boundary. Then the residual reads as

$$\bar{\mathbf{r}}(\mathbf{s}) = \begin{pmatrix} \frac{(2q_1-q_2)}{h} - h \\ \frac{(-q_1+2q_2-q_3)}{h} - h \\ \vdots \\ \frac{(-q_{N-3}+2q_{N-2}-q_{N-1})}{h} - h \\ \frac{(-q_{N-2}+q_{N-1})}{h} - \frac{(q_N-q_{N-1})\,(h-h\,N+\alpha)}{h^2} - \frac{h-h\,N^2+2\,N\,\alpha-\frac{\alpha^2}{h}}{2} - \frac{1}{2}\,\frac{Nh-\alpha}{h} \\ \frac{(q_N-q_{N-1})\,(h-h\,N+\alpha)}{h^2} - \frac{(h-h\,N+\alpha)^2}{2\,h} - \frac{1}{2}\,\frac{\alpha-(N-1)\,h}{h} \\ q_{N-1}\frac{(N\,h-\alpha)}{h} + q_N\frac{(\alpha-(N-1)\,h)}{h} - 1 \end{pmatrix}. \tag{3.14}$$

Here we denote by $\bar{\mathbf{r}}$ the modified residual and by h a fixed element size that does not depend on α at fixed geometry.

Remark 3.4. *It holds that* $\bar{\mathbf{r}}(\mathrm{s}) = \mathbf{r}(\mathrm{s})$ *for given* s*, but* $\nabla_{\mathbf{s}}\bar{\mathbf{r}}(\mathrm{s}) \neq \nabla_{\mathbf{s}}\mathbf{r}(\mathrm{s})$.

Now a straightforward application of the Newton method to solve zero-point of residual (3.13) converges quadratically in a neighbourhood of the solution. The Newton iteration can be formulated by the following algorithm:

1. Pick a initial guess s_0, set $k = 0$.
2. Solve $\delta\mathrm{s}_k$ from equation
$$\nabla_{\mathbf{s}}\mathbf{r}(\mathbf{s}_k)\,\delta\mathbf{s}_k = \mathbf{r}(\mathbf{s}_k)$$
3. Update $\mathrm{s}_{k+1} = s_k - \delta\mathrm{s}_k$ and $k = k+1$ and continue from 2 until converged.

Lemma 3.5. *Assume that there exist a point* $\mathrm{s}^* \in \mathbb{R}^{N+1} \setminus \mathbf{0}$*, for which* $\mathbf{r}(\mathrm{s}^*) = \mathbf{0}$ *and* $\det(\nabla_{\mathbf{s}}\mathbf{r}(\mathrm{s}^*) \neq 0$*. Then the Newton algorithm converges quadratically in the neighbourhood of* s^*.

Proof. q is twice continuously differentiable at the solution of the free boundary and the second order derivative is bounded. Further, the determinant of matrix $\nabla_{\mathbf{s}}\mathbf{r}(\mathrm{s}) \neq 0$ for s neighbourhood of s^*. Therefore the statement follows. □

Studying the second residual $\bar{\mathbf{r}}(\mathrm{s})$ we notice that a straightforward implementation of the Newton method yields a slowly converging or even divergent method. This is because the internal nodal point movement is not taken into account in the computation, as in practice h should depend on α and this is not taken into account in the computation of the Jacobian of the residual.

The derivative of residual $\bar{\mathbf{r}}$ with respect to the geometry parameter α reads

$$\frac{\partial\bar{\mathbf{r}}}{\partial\alpha}(\mathrm{s}) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -\frac{(q_N - q_{N-1})}{h^2} - N + \frac{\alpha}{h} + \frac{\alpha}{2h} \\ \frac{(q_N - q_{N-1})}{h^2} + N - \frac{\alpha}{h} - 1 - \frac{\alpha}{2h} \\ \frac{q_N - q_{N-1}}{h} \end{pmatrix}. \tag{3.15}$$

Comparing (3.15) to discretization of the shape linearised equations from the previous section we notice that they differ in the second and third row from below. A careful look at the calculations reveal that the extra terms shown up here comes from the integrals involving the normal derivatives of the test functions, which do not disappear at point α as it was assumed in the continuous formulation of the shape linearised equations. The derivative $\frac{\bar{\mathbf{r}}}{\partial\alpha}$ should be calculated only for the part of $\bar{\mathbf{r}}$ which corresponds to the smaller trial space (where $\varphi'(s) = 0$). The correct formula can be achieved by replacing the second row from below by the sum of the second and third row from below. This corresponds to calculation of shape derivative of the discrete formula for a test function $\tilde{\varphi}_N(x) := \varphi_{N-1}(x) + \varphi_N(x)$ in the last element, with perturbation that moves only the free boundary point. Same contribution can be achieved by multiplying the above vector by $(N+1) \times (N+1)$ projection matrix P:

$$P = \begin{pmatrix} 0 & \dots & 0 & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & 0 \\ 0 & \dots & 1 & 1 & 0 \\ 0 & \dots & 0 & 0 & 1 \end{pmatrix}. \tag{3.16}$$

That is,

$$P \frac{\partial \bar{\mathbf{r}}}{\partial \alpha}(\mathbf{s}) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -1 \\ \frac{q_N - q_{N-1}}{h} \end{pmatrix}.$$

Now the shape linearised equations can be obtained by the above procedure by also applying an automatic differentiation in the calculations of the derivatives. The calculated derivative is then modified by multiplying the shape sensitivity part of the equations by matrix P. Instantly we notice that Lemma 3.3 suggests that we do not have a quadratic convergence rate for the above discrete variant of the one dimensional shape linearization but instead a linear convergence rate depending on the mesh size h.

Remark 3.5. *To obtain a better convergence rate we should obtain formula* (3.10) *from the discrete shape sensitivity calculation. This can be obtained by using $P2$ elements to approximate the boundary value of the discrete solution.*

3.2.4 Isoparametric approach for automatic differentiation

In this section we shall study automatic differentiation of boundary integrals in an isoparametric finite element approach [HM03, Mäk90]. In an isoparametric approach the local integrals are transformed to integrals over the a reference element. We shall study this for a single element at first. In this context we shall adopt the notations used in [HM03], $\hat{\cdot}$ indicates that the variable is defined on reference element. In the following T_e is a single element, $\hat{T}$ is the reference element and $F_e : \hat{T} \to T_e$ is a one-to-one mapping of $\hat{T}$ onto T_e. Then the integrals defined on T_e can be computed by integrating over the reference element

$$\int_{T_e} f(x)\,dx = \int_{\hat{T}} f(F_e(\xi))\,|J| d\xi,$$

where $|J|$ is the determinant of the Jacobian J of mapping F_e. The standard way to calculate local integrals in the isoparametric finite element method is to use local matrices and vectors in computation. First define

$$\widehat{N} := \begin{pmatrix} \widehat{\varphi}_1 \\ \vdots \\ \widehat{\varphi}_p \end{pmatrix} \quad \text{and} \quad \widehat{G} := \begin{pmatrix} \frac{\partial \widehat{\varphi}_1}{\partial \xi_1} \cdots \frac{\partial \widehat{\varphi}_p}{\partial \xi_1} \\ \frac{\partial \widehat{\varphi}_1}{\partial \xi_2} \cdots \frac{\partial \widehat{\varphi}_p}{\partial \xi_2} \end{pmatrix},$$

where φ_k are shape functions and $\partial\widehat{\varphi}_k/\partial\xi_j$ are derivatives of the shape functions for p-noded Lagrangian finite element in reference element $\widehat{T}$. F_e is now given by

$$F_e(\xi) = \sum_{i=1}^{p} \widehat{\varphi}(\xi) X_i \quad \text{for } \xi \in \widehat{T},$$

where X is a vector of nodal points,

$$X = \begin{pmatrix} X_{11} & X_{12} \\ X_{22} & X_{22} \\ \vdots & \vdots \\ X_{p1} & X_{p2} \end{pmatrix}.$$

Here X_{i1} is the first coordinate point and X_{i2} is the second coordinate point of ith element nodepoint in dimension two. In the following we have collected standard results for the finite element calculation [HM03]:

$$\varphi(x) = \widehat{\varphi}({F_e}^{-1}(x)), \tag{3.17}$$

$$\nabla\varphi(x) = J^{-1}\nabla_\xi\widehat{\varphi}({F_e}^{-1}(x)) \quad \text{and} \tag{3.18}$$

$$J = \widehat{G}X \tag{3.19}$$

for $x \in T_e$.

Now in practical finite element computations with an automatic differentiation approach we have a design parameter vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m)$, which defines the geometry of the finite element mesh. Quite a standard way of calculating the shape derivatives in finite element calculations seems to be to split the differentiation with respect to the design variable by a chain rule [HM03, HLNT03, Num03], i.e.

$$\frac{\partial f(x(\boldsymbol{\alpha}))}{\partial \alpha_i} = \frac{\partial f(x(\boldsymbol{\alpha}))}{\partial x}\frac{\partial x}{\partial \alpha_i}.$$

Here $\partial x/\partial\alpha_i$ can be provided in advance for finite elements by calculating the dependence of each nodepoint with respect to the geometry design variable vector $\boldsymbol{\alpha}$. This approach has, however, a few drawbacks. First of all we need to compute the derivative of each nodal point with respect to the design variable $\boldsymbol{\alpha}$. Thus in the worst case we could have a full $m \times n$ matrix, for which m is the length of $\boldsymbol{\alpha}$ and n is the number of nodal points in the finite element mesh.

Haslinger and Mäkinen [HM03] have provided calculations of derivatives of local finite element integrals with respect to the state variable $\boldsymbol{\alpha}$ assumed that the shape functions depend on the geometry. Then for isoparametric finite elements they show that $\frac{\partial N}{\partial \alpha_i} = 0$ for all $i = 1, \ldots, M$. This means that the shape functions follow the perturbation of the finite element mesh.

The objective here is to enable the calculation of shape sensitivities only on the free boundary. The main idea is presented in the following lemma

Lemma 3.6. *Assume that V is a velocity field supported interior of discrete domain Ω_h, i.e. $V(x_k) = 0$ for $x_k \in \partial\Omega_h$. Assume further that the finite element mesh is perturbed by speed method, i.e. $x'_k(t) = V(x_k)$ and that $f_h(x)$ is a function defined on Ω_h, $f_h \in \breve{\mathcal{V}}_h(\Omega_h)$, where $\breve{\mathcal{V}}_h$ is a discrete finite element function space of piecewise polynomial continuous functions for which $\varphi'(\Omega_h; V) = 0$ for all $\varphi \in \breve{\mathcal{V}}_h(\Omega_h)$ for any smooth enough velocity field V. Then*

$$\frac{\partial}{\partial t}\int_{\Omega_h} f_h(x)\varphi\,dx\bigg|_{t=0} = 0$$

and

$$\frac{\partial}{\partial t}\int_{\Omega_h} \nabla f_h(x)\cdot\nabla\varphi\,dx\bigg|_{t=0} = 0$$

for all $\varphi \in \breve{\mathcal{V}}_h$.

Proof. Since $f_h\varphi \in L^1(\Omega_h)$ and $\nabla f_h \cdot \nabla\varphi \in L^1(\Omega_h)$ we can use Lemma 2.2 and the statement follows. □

The above Lemma states that by fixing the shape functions with respect to the geometry change we can restrict our attention to the boundary of discrete geometry in the variation of the geometry. The next question is, can this modified isoparametric technique be used in the calculation of a shape gradient of a given functional.

Now to mimic the shape differentiation on a continuous level we require that the shape functions do not depend on the perturbation of the geometry. For this we need new one-to-one mapping

$$\breve{F}_e(\xi) = \sum_{i=1}^{p} \widehat{\varphi}(\xi)\breve{X}_i,$$

where notation $\breve{X}$ means that we have fixed the positions of the nodal coordinates with respect to the design variable $\boldsymbol{\alpha}$. In practice, this can be handled by defining a new variable $\breve{X}$, which is not an AD-variable (AD stands for automatically differentiated) in the finite element program. Now define

$$\varphi_i(x) = \widehat{\varphi}_i(\breve{F}_e^{-1}(x)).$$

Now the test functions do not depend on the design parameter $\boldsymbol{\alpha}$ and therefore we can compute using the derivative of a domain integral [SZ92]

$$\frac{\partial}{\partial t}\int_{T_e} \varphi_i(x)\,dx\bigg|_{t=0} = \int_{T_e} \nabla\cdot(\varphi_i(x)V)\,dx = \int_{\partial T_e} \varphi_i(s)V\cdot\mathbf{n}\,d\sigma$$

by the Stokes formula. Here we have applied transformation of identity, $X(t) = X + tV(X)$. We see that summing this over the finite element mesh only leaves the boundary of the mesh the exterior normal vector of each element boundary and is opposite to the exterior normal vector of the adjacent element boundary.

Consider now shape sensitivity of a local integral

$$\int_{T_e(t)} \nabla\phi_i(x) \cdot \nabla\phi_j(x)\, dx.$$

Now the shape functions do not depend on the perturbation and we get

$$\frac{\partial}{\partial t} \int_{T_e(t)} \nabla\phi_i \cdot \nabla\phi_j \, dx \bigg|_{t=0} = \int_{\partial T_e} \nabla\phi_i \cdot \nabla\phi_j V \cdot \mathbf{n}\, d\sigma.$$

Here derivatives $\nabla\phi_i$ are calculated at each element using isoparametric mapping. The derivatives of test functions have a jump over each element boundary and therefore summing the above over all elements does have some contribution from the inner part of the finite element mesh.

Remark 3.6. *In Lemma 3.6 it is supposed that the shape functions do not continue over the boundaries of the element. In practical computations of integrals the values of shape functions are evaluated only at integration points. The local elementwise integrals are then calculated using an integration rule that gives sufficient approximation to the continuous integrals. In this case the shape functions are continued continuously outside local elements and these values contribute to the shape sensitivities in the modified elements.*

Assuming that V is supported only on the free boundary leads then to

$$\int_{\partial T_e} \nabla\phi_i \cdot \nabla\phi_j V \cdot \mathbf{n}\, d\sigma = \int_{\partial T_e \cap \Sigma_h} \nabla\phi_i \cdot \nabla\phi_j V \cdot \mathbf{n}\, d\sigma.$$

Now the situation is similar to the one dimensional example, we have a contribution of a normal derivative of the test function to the derivative. The question now is, how can this contribution be subtracted from the calculated derivative? If we only consider quadrilateral elements that are constructed so that the elements adjacent to the free boundary are orthogonal to the free boundary we can apply the projection defined for one dimensional example. Then we can construct the projection mapping by summing up the rows calculated from the discrete shape sensitivity analysis. In this thesis we shall restrict our attention only to Q_1 elements although this can be generalised for different elements also.

The idea proposed here will be the extension of the test function appearing in integral by a constant value to the inside of the domain. This is basicly a mimic of the calculations in the continuum, where the derivatives of the test functions are assumed to vanish in a smooth case [SZ92].

In particular, we propose that we can use an element layer near the boundary for which we calculate the shape derivatives of the integrals. This element layer is defined so that at each nodepoint $\mathbf{x}_k$ the neighbouring node point is given by $\mathbf{x}_k + \delta(h)\tilde{\mathbf{n}}_k(x_k)$, where δ is small enough and depends on size h locally. A sketch of this configuration is given in Figure 10. Now we replace test functions ϕ_j in the calculations by summing the test function which corresponds to the adjacent nodepoints in a generated element layer to each other. Furthermore, the new nodepoints are assumed to be fixed so that the element layer is moved only

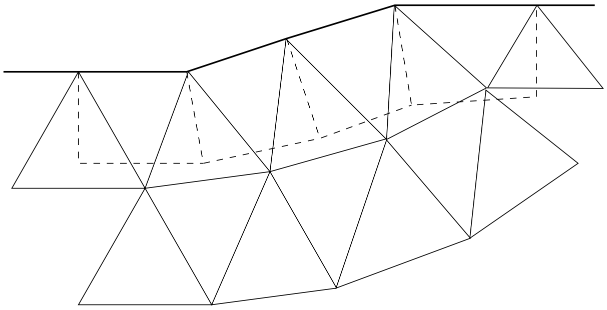


FIGURE 10 Q_1–element layer generated for P_1–element mesh.

on the boundary. Denoting $\tilde{\phi}_j$ the test function that is constructed with the above method we get

$$\frac{\partial}{\partial t}\int_{\tilde{T}_e(t)} \nabla\phi_i \cdot \nabla\tilde{\phi}_j \, dx\bigg|_{t=0} = \int_{\partial\tilde{T}_e\cap\partial\Omega_h} \nabla_\Gamma\phi_i \cdot \nabla_\Gamma\tilde{\phi}_j V \cdot \mathbf{n}\, d\sigma.$$

This corresponds to the calculation of shape sensitivities for continuous integrals.

For boundary integrals the integral itself is already supported on the boundary of the finite element mesh so that the shape sensitivity can be calculated by just modifying the calculation of the values of shape functions. However, the modified shape functions are continued continuously over the free boundary and thus the discrete shape sensitivity calculation involves normal derivatives. This can be seen from the sensitivity calculation of

$$\frac{\partial}{\partial t}\int_{\partial T_e(t)} \varphi \, d\sigma\bigg|_{t=0} = \int_{\partial T_e} \left(\frac{\partial\varphi}{\partial n} + \kappa\right)\langle V, \mathbf{n}\rangle \; d\sigma \quad \text{(+ contributions of corners).}$$

Now the situation is more difficult compared to the interior integral. In practical computations we can not use the projection since in standard finite element implementation the boundary integrals are supposed to vanish in shape functions that are not supported on the boundary and these zero integrals are not therefore evaluated and thus the nonzero shape derivatives are not evaluated either. Then we do not get a contribution from the shape functions from the interior nodal points that would cancel the normal derivatives of the test functions supported on the free boundary. The remedy for this is presented in the following. In the calculation of shape sensitivies we define a new variable $\psi(x)$ defined on the free boundary only. We define a new test function on the free boundary, $\tilde{\varphi}(x) := \varphi(x) - \frac{\partial\varphi}{\partial n}(x)\psi(x)$. Setting $\psi(x) = 0$ for $x \in \Sigma$ during the calculation we get

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\Sigma(t)} \tilde{\varphi}\, d\sigma\bigg|_{t=0} + \left(\frac{\partial}{\partial\psi}\int_{\Sigma} \tilde{\varphi}\, d\sigma\right)\langle V, \mathbf{n}\rangle &= \int_{\Sigma}\left(\frac{\partial\tilde{\varphi}}{\partial n} + \kappa\tilde{\varphi}\right)\langle V, \mathbf{n}\rangle \; d\sigma - \int_{\Sigma}\frac{\partial\varphi}{\partial n}\langle V, \mathbf{n}\rangle \\ &= \int_{\Sigma} \varphi\,\kappa\,\langle V, \mathbf{n}\rangle \; d\sigma.\end{aligned}$$

Here differentiation with respect to the state variable ψ can be automated and this can be done simultaneously with the shape sensitivity calculation.

3.3 Numerical results

In this section we shall study the numerical implementation of different Newton–type solution methods to solve a free boundary problem. We shall compare the convergence between five different implementations:

DSL (*Discrete Shape Linearization method*) The derivatives with respect to the geometry are calculated using a modified implementation of an isoparametric finite element, automatic differentiation is used to obtain the shape linearized equations and shape differentiation is restricted only on the free boundary.

CSL (*Continuous Shape Linearization method*) The shape differentiation is first performed for a continuous form of the equations and then the continuous equations are discretized. Automatic differentiation is applied only with respect to the state variable.

ISL (*Implicit Shape Linearization method*) This corresponds to CSL but the shape linearized equations are further processed by applying conditions at the solution of the free boundary to the state equation.

FAD (*Full Automatic Differentiation*) The mesh movement is not restricted only on the free boundary, the internal boundary nodes are taken into account in the shape differentiation.

BAD (*Boundary restricted Automatic Differentiation*) The automatic differentiation with respect to the geometry is performed only for nodal points relying on the free boundary.

The finite element program that was modified was Numerrin 2.0 [Num03]. In this finite element library automatic differentiation is already implemented but for our case the kernel routines had to be modified to implement DSL. The finite element mesh movement was handled by linear mapping that extended the boundary movements to the internal nodes. The tests were performed on a Linux Debian-workstation with AMD K-7 CPU 550 MHz processor with 256Mb memory.

3.3.1 Convergence near the solution

At first we study the convergence near the known solutions of the FBP. The first FBP is exterior Bernoulli's free boundary problem with fixed boundary $\Gamma_b = \partial B(0; \frac{1}{2})$ (circle with radius $\frac{1}{2}$). The boundary conditions for the problem are $u = 0$ on Γ_b, $u = 1$ on Σ and $\frac{\partial u}{\partial n} = 1$ on Σ. There exists an axisymmetric solution for this FBP, $u(x, y) = \frac{\log(2\sqrt{x^2+y^2})}{\omega(2)}$, where ω is the Lambert W-function, i.e. the inverse of function $f(x) = x\,e^x$. The solution geometry is a disc with radius $r^* = \frac{1}{\omega(2)}$.

The convergence is studied by perturbing the solution geometry by small sinusoidal perturbations. The algorithm to test the different methods is given in the following.

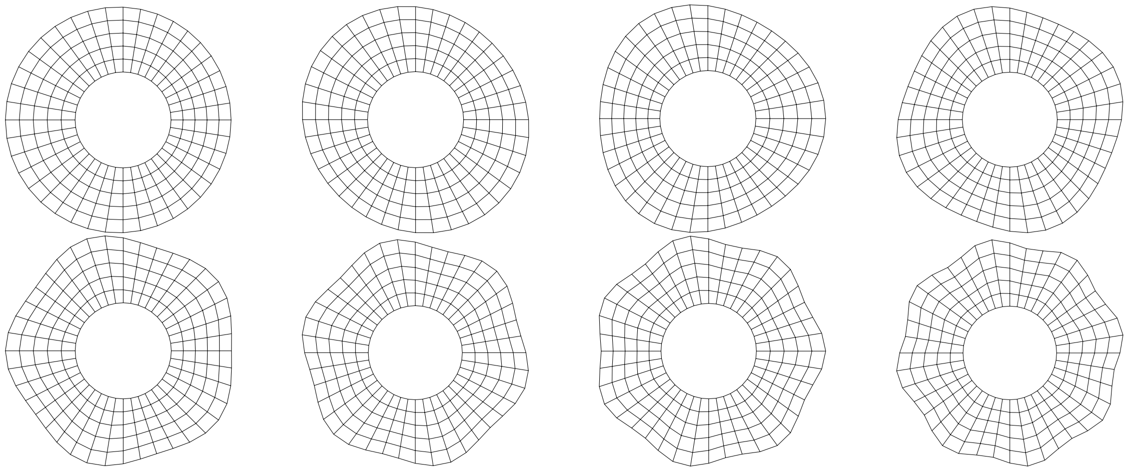


FIGURE 11 Perturbations for the mesh with wave numbers $k = 1, \ldots, 8$, amplitude $a = 0.05$

1. Initialize the mesh to correspond to the solution of the FBP.

2. Calculate the perturbation to the mesh on a free boundary by

   $$p(\theta) = \omega \frac{\sin(k\theta)}{\| \sin(k\theta) \|_{H^s}},$$

   where θ is the angle of the point in polar coordinates, ω is the magnitude of the perturbation and s is the spectral norm index ($s = 0$ corresponds to L^2-norm, $s = \frac{1}{2}$ to $H^{\frac{1}{2}}$-norm and $s = 1$ to H^1-norm), and k is the wave number. Perturb the free boundary by perturbation p, $\Sigma_p = (r_h^* + p(\theta))(\cos\theta, \sin\theta)$.

3. Set initial guess u_h for the state solution to

   $$u_h(x_i, y_i) = \frac{\log(2\sqrt{x_i{}^2 + y_i{}^2})}{\omega(2)},$$

   at each nodal point (x_i, y_i).

4. Solve one iteration to get an update for the geometry. Update the geometry $\Sigma_p \mapsto \Sigma_1$.

5. Calculate the error between the updated geometry and the solution geometry

   $$e(\theta) = \Sigma_1(\theta) - r_h^*.$$

6. Calculate the contraction rate of the error

   $$\vartheta_b^a := \frac{\|e\|_{H^a}}{\|p\|_{H^b}}, \quad a,\, b \in \{0, \frac{1}{2}, 1\}.$$

In Figure 11 we can see perturbations with eight different wave numbers for the geometry. Figures 12– 14 show graphically the contraction rates ϑ_0^0 for 40 wave lengths and two amplitudes and $\vartheta_{\frac{1}{2}}^0$ for one amplitude. We used four different meshes, the free boundary was divided into 20, 40, 80 and 160 elements and respectively in the radial direction the mesh contained 3, 5, 10 and 20 element layers. The calculation of the fractal norm was done by using fourier multipliers.

We can see that the contraction rate gets better with low frequencies. When the amplitude of the perturbation increases the convergence slows down or fails. From Figure 12 we see that the different methods fail to get closer to the solution in L^2–norm when the wavelength of the perturbation gets smaller (with large wave numbers). However, when studying the high frequencies of the error by calculating the H^1 seminorm of the error one finds that the error becomes smaller in the H^1 norm. Studying this more carefully it can be seen that the sawtooth–like error is smoothed and then after one iteration there appears a constant displacement in the radial direction. This constant value appears in L^2–norm but not in H^1 seminorm.

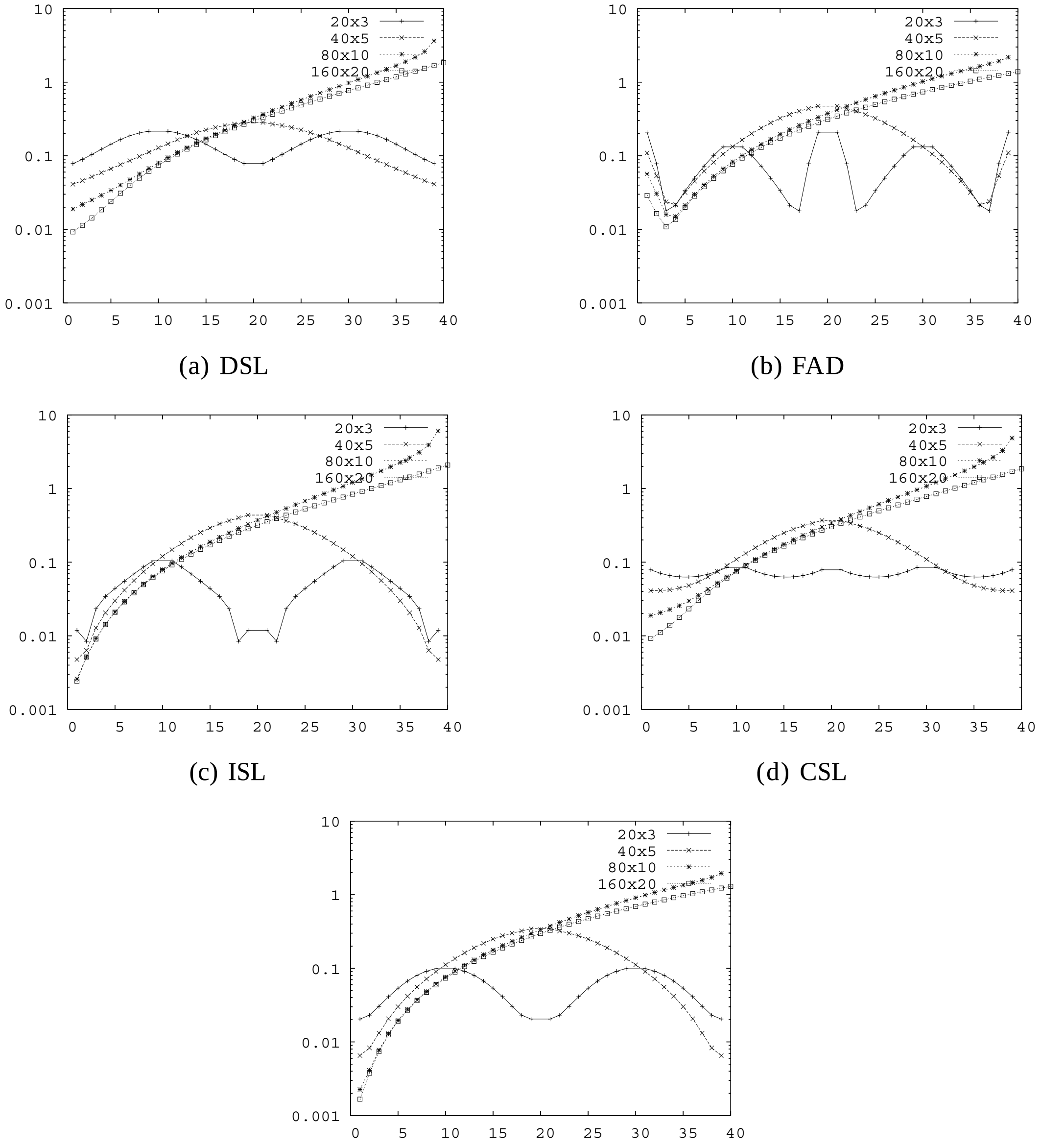
Figure 15 presents the contraction rates for 160×20–mesh. We can observe spectral behaviour for the solution methods. DSL and CSL seem to behave similarly, these methods seem to operate like $H^s \to L^2$, where $0 < s < \frac{1}{2}$. For ISL, BAD and FAD spectral parameter s is most likely closer to 1 than $\frac{1}{2}$.

3.3.2 Model problem

Let us next study a model problem for which the solution is not known a priori. The problem under investigation is the model problem presented in equation (1.2). The geometry of the domain can be seen in Figure 16. In the Neumann condition (1.2c) we used value $\lambda = 1$ and we changed the free boundary to be an outer boundary, and a fixed boundary to be the inner boundary of the geometry. We further used a zero Dirichlet value for the fixed boundary and $u = 1$ on the free boundary. The fixed geometry consists of four segments of a circle with a radius of 0.316, corner points of the fixed boundary are located at $(-0.4, 0.0)$, $(0.0, 0.2)$, $(0.4, 0.0)$ and $(0.0, -0.2)$.

We tested the convergence of the algorithm with different meshes and compared the results against different implementations. Execution times of different algorithms are shown in Table 1. We observed that a discrete shape linearization method (denoted DSL) is slower due to the automatic differentiation of variables with respect to the geometry, but the slow down is only approximately 6% from the continuous form of the algorithms. The full automatic differentiation takes the longest time for one iteration but on the other hand the FBP is practically solved after four iterations. The most competitive solution algorithm seems to be ISL. The worst results come from boundary restricted automatic differentiation (BAD) which do not even get close to the solution in five iterations. The solution times for BAD are the same as in full automatic differentiation since the sparsity of the nodal derivatives is not used in the calculations.

In Figure 17 we plot the sum of L^2-norms of the updates. The L^2-norm for the geometry update is calculated on the free boundary. We can observe that the convergence of the implicit shape linearization method (ISL) accelerates as the mesh is refined. This can not be observed from the other two algorithms.



(a) DSL

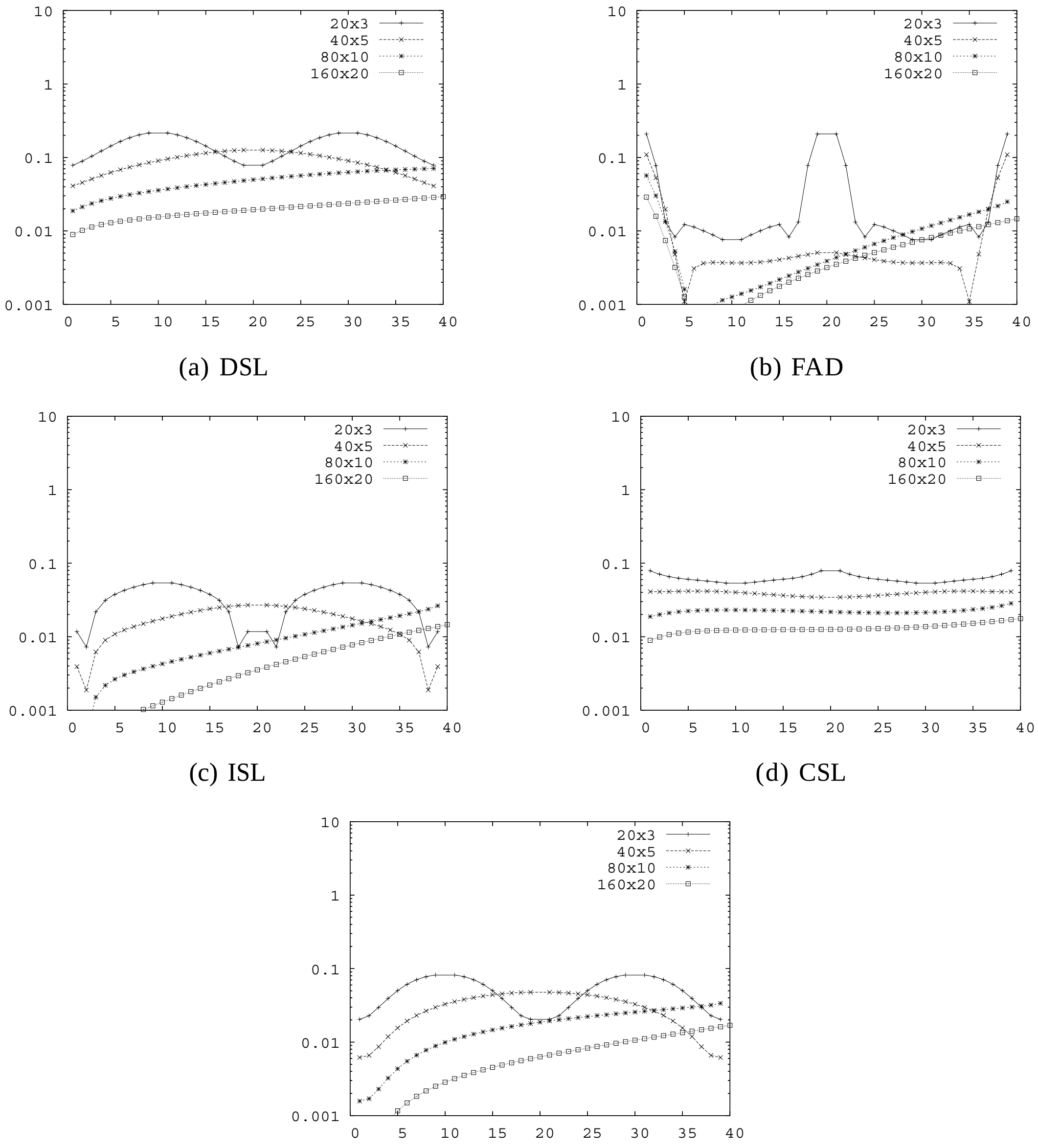
(b) FAD

(c) ISL

(d) CSL

(e) BAD

FIGURE 12 Contraction rate ϑ_0^0 for different wave numbers, $k = 1, \ldots, 40$, $\omega = 0.01$ for four meshes.



(a) DSL

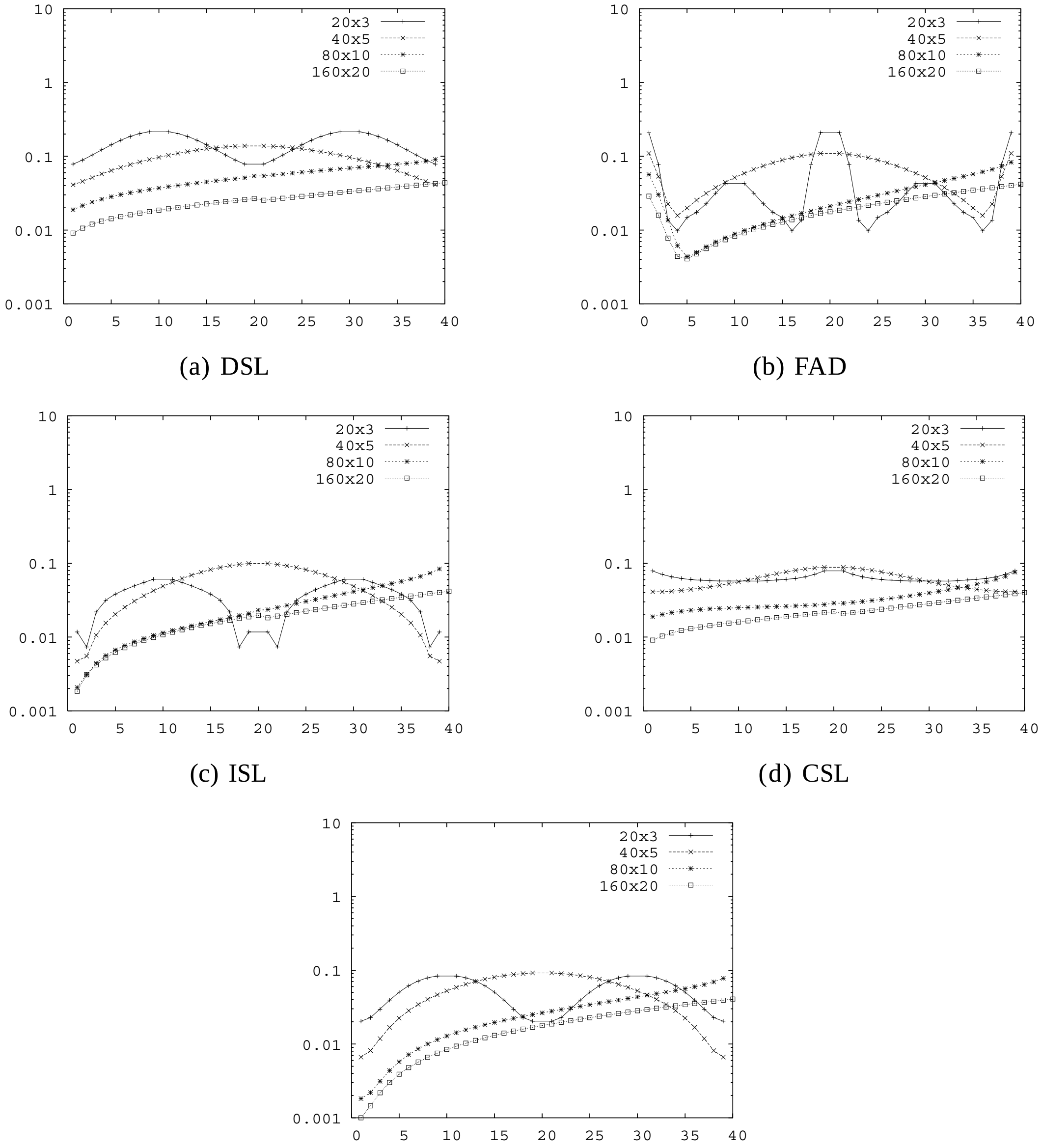
(b) FAD

(c) ISL

(d) CSL

(e) BAD

FIGURE 13 Contraction rate ϑ_0^0 for different wave numbers, $k = 1, \ldots, 40$, $\omega = 0.001$ for four meshes.



(a) DSL

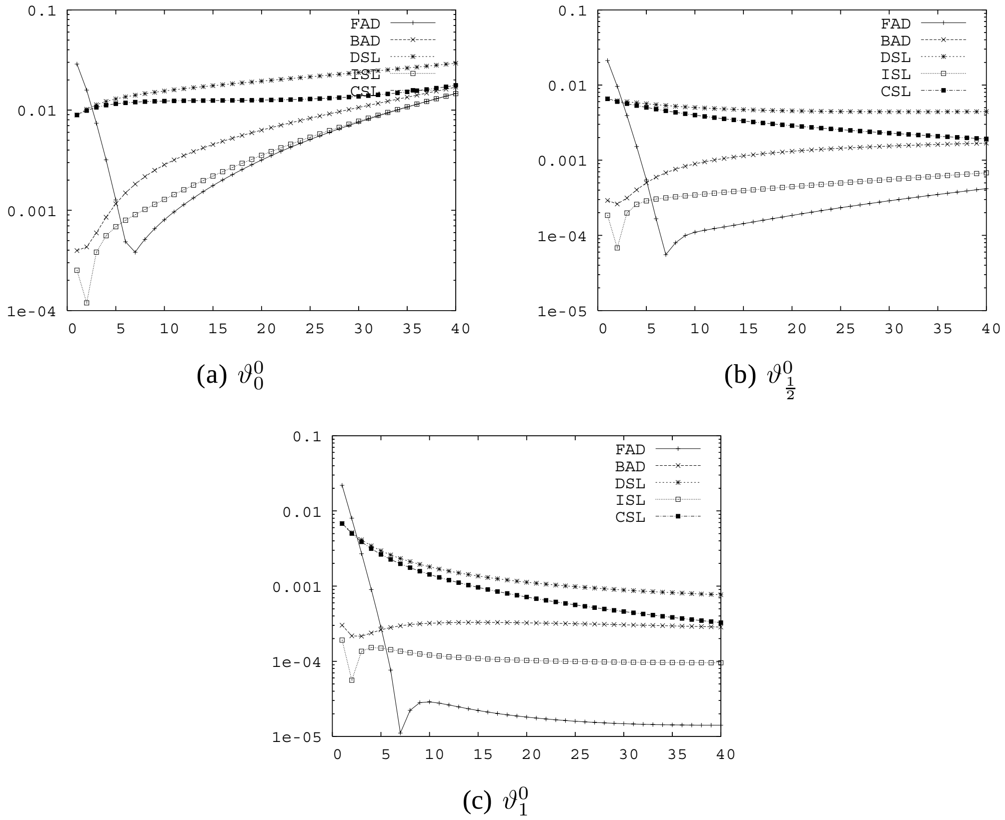
(b) FAD

(c) ISL

(d) CSL

(e) BAD

FIGURE 14 Contraction rate $\vartheta_{\frac{1}{2}}^{\frac{1}{2}}$ for different wave numbers, $k = 1, \ldots, 40$, $\omega = 0.01$ for four meshes.



(a) ϑ_0^0

(b) $\vartheta_{\frac{1}{2}}^0$

(c) ϑ_1^0

FIGURE 15 Contraction rates for 160×20-mesh, $\omega = 0.001$.

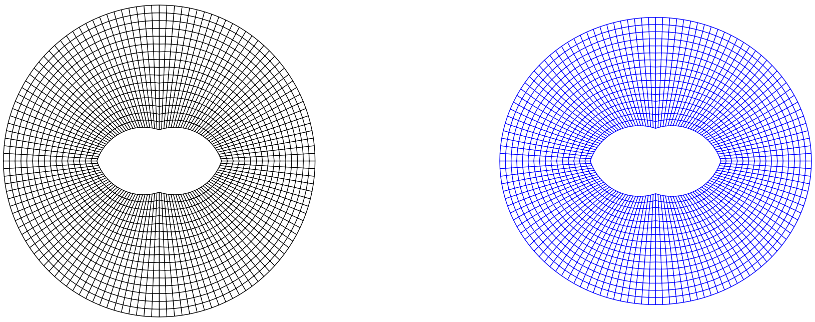


FIGURE 16 Initial 128×16 mesh (left) and final mesh (right) for model problem.

TABLE 1 Execution times in seconds, updates and residuals after five iterations for 128×16 mesh.

	one iteration	5 iterations	$\|\delta u_5\|$	$\|\delta\Sigma_5\|$
FAD	3.4	19.2	$0.504{*}10^{-14}$	$0.247{*}10^{-14}$
DSL	2.6	14.2	$0.330{*}10^{-2}$	$0.486{*}10^{-4}$
ISL	2.5	13.6	$0.219{*}10^{-5}$	$0.188{*}10^{-8}$
CSL	2.5	13.6	$0.434{*}10^{-3}$	$0.223{*}10^{-5}$
BAD	3.3	17.8	0.119	$0.918{*}10^{-2}$

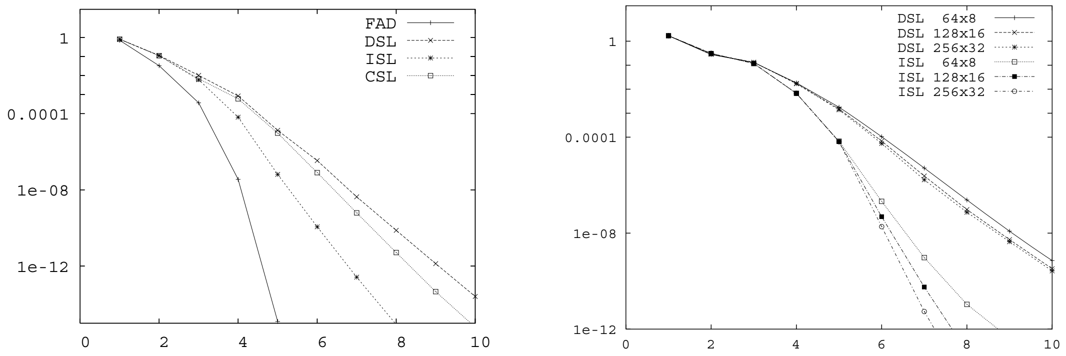


FIGURE 17 Convergence of different shape linearization methods for 256×32 mesh (left) and the convergence of the discrete shape linearization method and implicit shape linearization method as the mesh is refined (right) for the model problem.

3.3.3 Die–swell problem

In this section we shall study the solution of Navier–Stokes equations with a free boundary. We shall study a bench–mark problem and compare the results for known implementations. Cuvelier et al [KCSV88, CS90] and Duprèt [Dup82] introduced total linearization for the Navier–Stokes equations. They linearized the nonlinear equations in a continuous case and calculated the solution of the nonlinear free boundary problem by a sequence of geometries.

Let us first introduce the problem under consideration. The Navier–Stokes equations are

$$-\nabla \cdot \bar{\sigma} + \rho \mathbf{u} \cdot \nabla \mathbf{u} = \rho \mathbf{g}, \tag{3.20}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{3.21}$$

where $\bar{\sigma} = 2mu\bar{\varepsilon} - p\mathbb{I}$, and $\bar{\varepsilon} = \frac{1}{2}\left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right)$ is the transformation rate tensor. Here $\mathbf{u}$ denotes the velocity of the fluid in a given point and p is the pressure of the fluid.

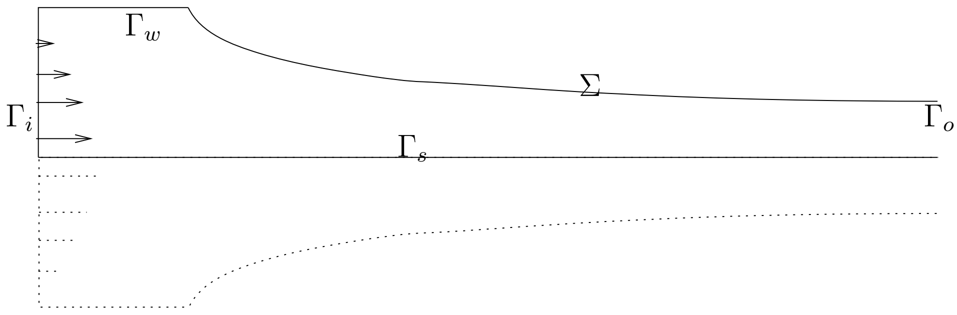


FIGURE 18 Die–Swell problem.

The boundary conditions for this free boundary problem are as follows

$$\mathbf{n}\cdot\mathbf{u} = 0 \quad \text{on } \Gamma_s, \tag{3.22a}$$

$$\mathbf{u} = \mathbf{0} \quad \text{on } \Gamma_w, \tag{3.22b}$$

$$\mathbf{p} = p_e \quad \text{on } \Gamma_o, \tag{3.22c}$$

$$\mathbf{u} = (\frac{3}{2}U*(y-1)*(1-y) \text{ and } ,0) \quad \text{on } \Gamma_i, \tag{3.22d}$$

$$\bar{\sigma}\cdot\mathbf{n} = \lambda\kappa\mathbf{n} + p_e\mathbf{n}, \quad \mathbf{n}\cdot\mathbf{u} = 0 \quad \text{on } \Sigma. \tag{3.22e}$$

Here λ is the surface tension coefficient and p_e is the external pressure on the fluid surface, here it is assumed to be zero as it is determined up to constant. The finite element approximation uses Franca–Frey stabilisation [FF92] to give feasible linear element approximation. Inflow boundary is positioned so that the upper corner of the inflow boundary is at point $(x, y) = (0, 1)$ and the corner point on the symmetry axis is located in the origin $(0, 0)$. The free boundary is fixed in position in which it touches the pipe (point where Σ and Γ_w meet). In the calculations we used the following dimensionless parameters

Re Reynolds number, $\frac{\rho UL}{\mu}$, ratio between inertia and viscosity of the fluid,

Ca^{-1} Inverse capillary number, $\frac{\lambda}{\mu U}$, ratio between surface tension and viscous forces.

The free boundary problem can be written in a weak form

$$\int_\Omega \bar{\sigma}_1\cdot\nabla\phi\,dx + \int_\Omega \rho\,\mathbf{u}\cdot\nabla u_1\phi\,dx + \int_\Sigma \lambda\kappa\mathbf{n}_1\phi\,d\sigma = 0, \tag{3.23a}$$

$$\int_\Omega \bar{\sigma}_2\cdot\nabla\phi\,dx + \int_\Omega \rho\,\mathbf{u}\cdot\nabla u_2\phi\,dx + \int_\Sigma \lambda\kappa\mathbf{n}_2\phi\,d\sigma = 0, \tag{3.23b}$$

$$\int_\Omega \nabla\cdot\mathbf{u}\,dx = 0, \tag{3.23c}$$

$$\int_\Sigma \mathbf{u}\cdot\mathbf{n} = 0 \tag{3.23d}$$

In the numerical implementation we used Numerrin 2.0 software. In our calculations we used each free boundary node as a control point of the geometry and

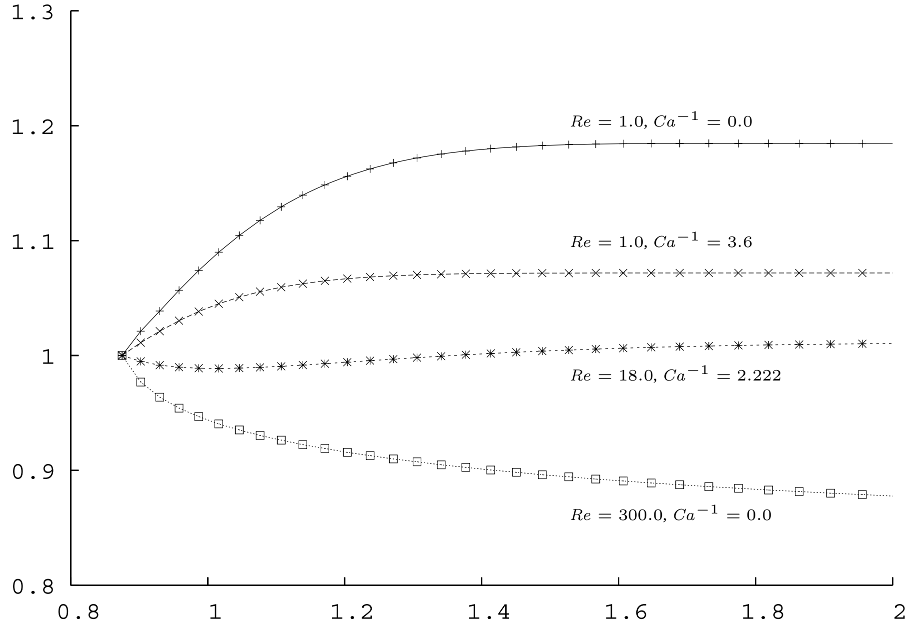


FIGURE 19 Die–Swell free boundary solutions for four cases.

the rest of the mesh was moved along the free boundary nodes. The calculations were made in a regular rectangular mesh. In reference [TM05] there is also considered a triangular mesh and especially the optimal restriction of the nodal movement with the viewpoint of sparsity of the linearized matrices.

The following algorithms were implemented:

FAD *Full Automatic Differentiation*, all inner nodal point movement was taken into account during the calculations.

DSL *Discrete Shape Linearization*, the derivatives were calculated only for moving free boundary nodes.

DSLI *Discrete Shape Linearization with Interpolation*, same as above but at each iteration the solution was interpolated between two meshes.

BAD *Boundary restricted Automatic Differentiation*, same as DSL but without projection of the test function contribution.

In the calculation we used 100×12 –mesh. From Figure 20 we can see the behaviour of the residual against the execution time with $Re = 1.0$ and $Ca^{-1} = 0.4$. The first step is taken without a geometry update and the time starts from this point. The maximum iterations was set to 10. We see that after 20 seconds the FAD method gets closer to the solution of the free boundary problem, but in the beginning the discrete shape linearization takes more steps in the same execution time (5 vs 4 iterations). The reduced computation time becomes from the reduced calculations of the shape sensitivities. After five iterations DSL and as well as DSLI turn out to be a linear convergence rate. This was already suggested for one dimensional model problem.

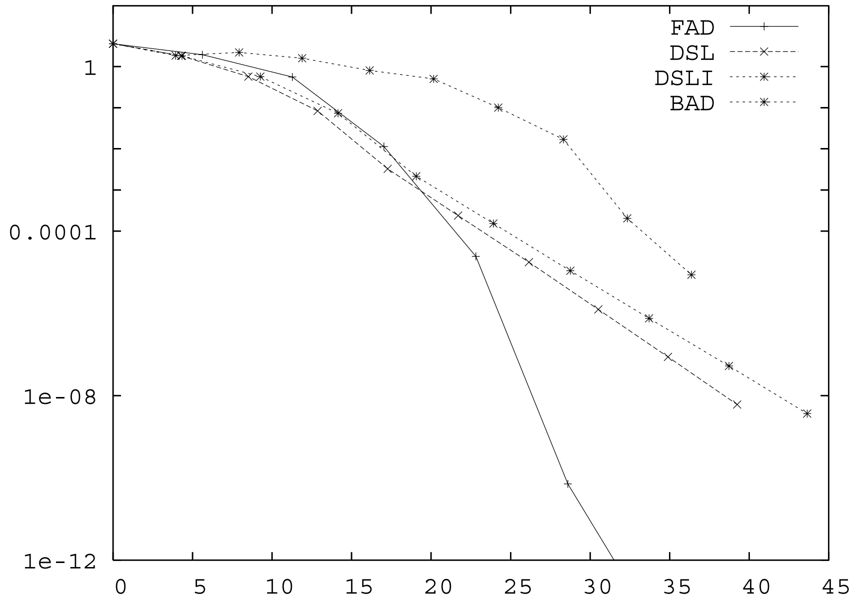


FIGURE 20 Convergence of the residual in execution time.

TABLE 2 Execution times in seconds for Die–Swell problem.

	one iteration	4 iterations
FAD	5.7	22.8
DSL	4.3	17.2
DSLI	4.8	19.1
BAD	4.0	16.1

As a conclusion we can see, that the discrete shape linearization method converges and is competitive against the full automatic differentiation of the whole system. The execution time could be reduced even more by using iterative methods that can take advantage of the sparsity of the linearized system. The convergence is not superlinear in the last iteration steps. This could be improved by post–processing the state solution in the finite element layer near the free boundary as was suggested in the calculation of the one dimensional example.

3.3.4 Internal free boundary

In this section we will test the applicability of the shape linearization method to an internal free boundary problem. The first test is made for a model free boundary problem (2.34) with axisymmetric interior and exterior boundaries. The interior boundary Γ_i is a circle with radius $r_i = \frac{1}{2}$ and the exterior boundary ∂D is a

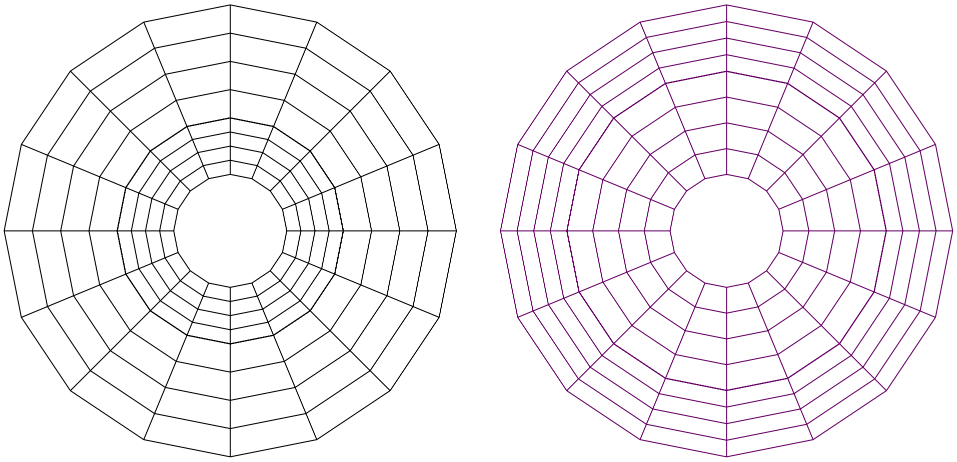


FIGURE 21 Initial 16×8 mesh (left) and final mesh (right) for model internal FBP ($\lambda = 0.8$).

TABLE 3 Numerical results of the axisymmetric internal FBP with coarse mesh and dense mesh various λ.

λ	r^*	16×8	64×8
0.6	1.509	1.531	–
0.8	1.304	1.320	–
1.0	1.173	1.186	1.173
1.2	1.081	1.092	1.082
1.4	1.012	1.022	1.013
1.6	0.959	0.968	–
1.8	0.917	0.924	0.917
2.4	0.827	0.833	0.827

circle with radius $r_e = 2$. The solution free boundary takes its position at radius

$$r^* = \frac{1}{\lambda\,\omega\left(\frac{2}{\lambda}\right)}.$$

In Table 3 we have collected the numerical results for two different meshes. In this case only a Discrete Shape Linearization method was used. The convergence was observed to be linear in the last few iterations, see Figure 22. Further, the convergence required the initial guess to be close to the solution. The convergence accelerated slightly as the mesh size was refined. For a dense mesh the convergence turned out to be more sensitive to the initial guess.

In the last numerical example we tested the internal free boundary problem by adding a source term to the Laplacian $-\Delta u = 1$ on the exterior part of the domain. Thus the solution did not remain constant any more. Further the domain Ω was chosen to be a rectangle. The observed convergence was again only linear but the solution was attained in 10 iterations. Initial guess and the solution can be seen from Figure 23.

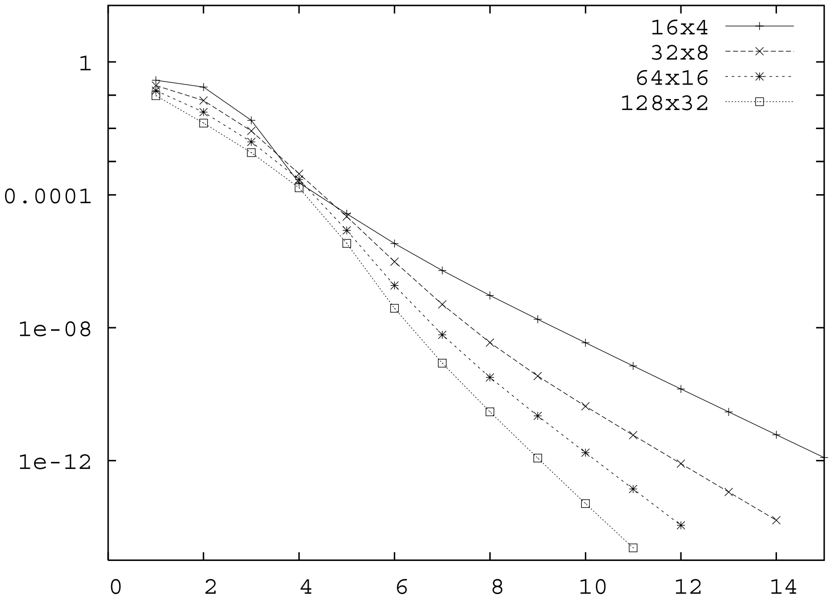


FIGURE 22 Convergence of the residual for axisymmetric internal free boundary problem with four different meshes.

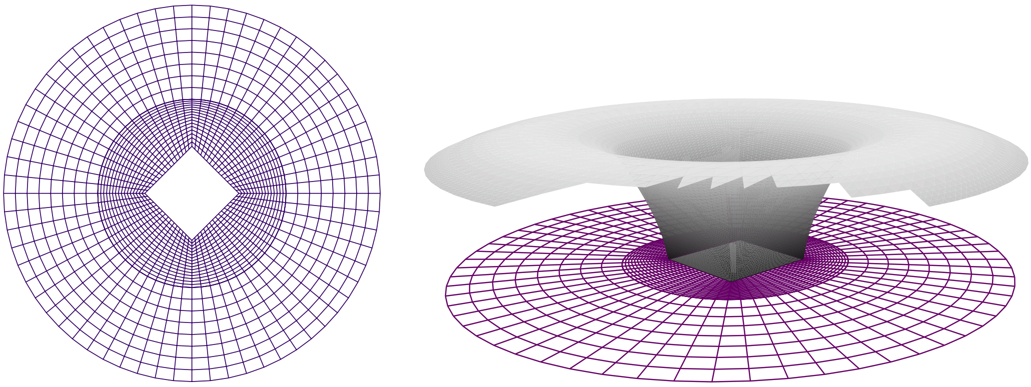


FIGURE 23 Initial 64×32 mesh (left) and the solution (right) for model internal FBP ($\lambda = 1.1$) with force term. Some of the rendering triangles are removed to show the solution.

4 CONCLUSIONS

Shape calculus provides tools for systematic study and construction of algorithms for solving stationary free boundary problems. In this thesis we studied different formulations of free boundary problems and analysed solution methods using shape sensitivity analysis. We constructed a simple fixed point type iteration schemes. The iteration scheme was numerically verified to converge superlinearly for a model problem.

The shape linearization method was introduced as an approximation of the Newton method. The technique of the shape linearization was confirmed to work also for numerical approximations. In particular, the automatic differentiation could be applied to calculate the shape sensitivities and to linearize free boundary problems. However, the implementation of the discrete shape linearization was not straightforward but required careful treating of normal derivatives of discrete shape functions.

The discrete shape linearization method was tested numerically for three different examples. The first example illustrated the convergence properties of different solution methods. Numerical results showed that the discrete shape linearization method converges characteristically like discrete continuous linearization methods. That is, the iteration starts with superlinear rate but the last iteration steps converge in a linear rate which depends on the mesh size. The convergence accelerates slightly by refining the mesh. The second example was a Die–Swell free boundary problem with the Navier–Stokes equations. It showed out that the discrete shape linearization method works also for a free boundary problem with nonlinear and non scalar state equation. The third example demonstrated the applicability of the discrete shape linearization method to an internal free boundary problem. When applying the method the state solution was relaxed to allow discontinuity of the solution on the free boundary. This approach seems to be more robust and generic compared to methods presented in the known literature. For example Tiihonen [Tii98] proposed a trial method for an internal free boundary problem, where the state solution remains continuous over the free boundary. The proposed method was however restricted to cases where the derivative of the state solution is continuous over the free boundary. For our approach the continuity of the derivative is not necessary.

Finally we want to point out some research issues that were left open in this thesis.

Treatment of corner points. In this thesis we assumed and analyzed only sufficiently smooth cases. However, in practical calculations there often exist stagnation points of edges of three materials or fluids [VB97, BS97]. The question is, how this can be treated efficiently?

Convergence of the numerical method. It would be interesting to prove the actual convergence rates for the discrete shape linearization method. One dimensional example and the numerical examples indicate that the convergence rate is $\mathcal{O}(h)$, i.e. the convergence depends on the mesh size. This could be turned to superlinear convergence rate by post–processing the solution at each step to obtain smoother approximation to the solution near the free boundary.

3D cases. In the thesis only two dimensional models were presented. The presented methods are also applicable to three dimensional cases. Limiting the shape sensitivity calculations only on the free boundary affects to the efficiency of the assemblation of local matrices in a three dimensional case even more than in the two dimensional case. This is because the matrices corresponding to a three dimensional mesh contain more nonzero elements since there are more shape functions supported on the same element. Further, the computational cost of elementwise discrete shape sensitivity analysis is increased as there are more nodal points that affect to the geometry and for each nodal point there is one direction more to vary. For example, $Q1$-element has 8 degrees of geometric freedom in 2D (4 nodal points, 2 coordinate directions) and 24 degrees of geometric freedom in 3D. On the other hand Flucher and Rumpf [FR97] showed that the theoretical convergence rate for shape linearization in three dimensional case becomes slower. Thus we can foresee that the discrete shape linearization will converge a bit slower in 3D compared to full Newton method but the full Newton method will take longer time for one iteration.

Time–dependent problems. Most common time dependent free boundary solvers are based on the implicit solver with respect to state equation and explicit solver with respect to the geometry. The velocity of the mesh is usually calculated along with the state equation. In this case there is no need to use shape sensitivity calculations during the time steps. However, to obtain unconditionally stable time discretizations for free boundary problems the geometrical quantities have to be treated implicitly. This concerns especially the curvature terms like surface tension forces [Bän01]. This necessitates the shape sensitivity analysis.

BIBLIOGRAPHY

[AC81] H.W. Alt and L.A. Caffarelli. Existence and regularity for a minimum problem with free boudary. *J. Reine Angew. Math.*, 325:105–144, 1981.

[Ada67] Arthur W. Adamson. *Physical Chemistry of Surfaces.* Interscience Publishers, second edition, 1967.

[AHK92] S.N Antontsev, K.-H. Hoffmann, and A.M. Khludnev, editors. *Free boundary problems in continuum mechanics*. Birkhäuser–Verlag, 1992.

[AN03] Viorel Arnăutu and Pekka Neittaanmäki. *Optimal control from theory to computer programs*, volume 111 of *Solid Mechanics and its Applications.* Kluwer Academic Publishers Group, Dordrecht, 2003.

[Bai75] Kwang June Bai. A localized finite–element method for steady, two dimensional free surface flow problems. In *First International Conference on Numerical Ship Hydrodynamics,* pages 209–229. Gaithersburg, 1975.

[Ban83] N. V. Banichuk. *Problems and methods of optimal structural design*, volume 26 of *Mathematical Concepts and Methods in Science and Engineering.* Plenum Press, New York, 1983. Translated from the Russian by Vadim Komkov, Translation edited by Edward J. Haug, With a preface by Haug and Komkov.

[Ban90] N. V. Banichuk. *Introduction to optimization of structures.* Springer-Verlag, New York, 1990. Translated from the Russian by Vadim Komkov.

[Bän01] Eberhard Bänsch. Finite element discretization of the Navier-Stokes equations with a free capillary surface. *Numer. Math.*, 88(2):203–235, 2001.

[BBS97] Morton Brøns, Martin Philip Bendsøe, and Mads Peter Sørensen, editors. *Progress in industrial mathematics at ECMI 96.* European Consortium for Mathematics in Industry. B. G. Teubner, Stuttgart, 1997. Papers from the 9th European Conference for Mathematics in Industry held at the Technical University of Denmark, Lyngby, June 1996.

[BC84] Claudio Baiocchi and Antóni Capelo. *Variational and Quasivariational Inequalities.* Wiley, 1984.

[BCT05] F. Bouchon, S. Clain, and R. Touzani. Numerical solution of the free boundary Bernoulli problem using a level set formulation. *Comput. Methods Appl. Mech. Engrg.*, 194(36-38):3934–3948, 2005.

[B.M96] B.Maury. Characteristics ALE method for the unsteady 3D navier–stokes equations with a free surface. *IJCFD*, 6:175–188, 1996.

[BMN04a] Eberhard Bänsch, Pedro Morin, and Ricardo H. Nochetto. Finite element methods for surface diffusion. In *Free boundary problems (Trento, 2002)*, volume 147 of *Internat. Ser. Numer. Math.*, pages 53–63. Birkhäuser, Basel, 2004.

[BMN04b] Eberhard Bänsch, Pedro Morin, and Ricardo H. Nochetto. Surface diffusion of graphs: variational formulation, error analysis, and simulation. *SIAM J. Numer. Anal.*, 42(2):773–799 (electronic), 2004.

[BS97] Ivan B. Bazhlekov and Peter J. Shopov. Numerical simulation of dynamic contact-line problems. *J. Fluid Mech.*, 352:113–133, 1997.

[Bur03] Martin Burger. A framework for the construction of level set methods for shape optimization and reconstruction. *Interfaces and Free Boundaries*, 5:301–329, 2003.

[BV00] Jeff Borggaard and Arun Verma. On efficient solutions to the continuous sensitivity equation using automatic differentiation. *SIAM J. Sci. Comput.*, 22(1):39–62 (electronic), 2000.

[BZ97] Dorin Bucur and Jean-Paul Zolésio. Anatomy of the shape Hessian via Lie brackets. *Ann. Mat. Pura Appl. (4)*, 173:127–143, 1997.

[CH53] R. Courant and D. Hilbert. *Methods of mathematical physics. Vol. I.* Interscience Publishers, Inc., New York, N.Y., 1953.

[CH83] Kyung K. Choi and Edward J. Haug. Shape design sensitivity analysis of elastic structures. *J. Structural Mech.*, 11(2):231–269, 1983.

[CL83] Jacques Cahouet and Marc Lenoir. Rèsolution numèrique du problème non linéaire de la résistance de vagues bidimensionnelle. *C.R. Acad. Sci. Paris Sér. II*, 297(12):819–822, 1983.

[Clé96] A. Clément. Coupling of two absorbing boundary conditions for 2D time–domain simulations of free surface gravity waves. *J. Comput. Phys.*, 126:139–151, 1996.

[Cra84] John Crank. *Free and moving boundary problems.* Oxford Science Publications. The Clarendon Press Oxford University Press, New York, 1984.

[Cry70] C. W. Cryer. On the approximate solution of free boundary problems using finite differences. *Journal of the Association for Computing Machinery*, 17(3):397–411, July 1970.

[CS90] C. Cuvelier and R.M.S.M. Schulkes. Some numerical methods for the computation of capillary free boundaries governed by the Navier-Stokes equations. *Siam Review*, 32:355–423, 1990.

[CS92] K. N. Christodoulou and L. E. Scriven. Discretization of free surface flows and other moving boundary problems. *J. Comput. Phys.*, 99(1):39–55, 1992.

[Del90] M.C. Delfour. Shape derivative and differentiability of min max. In Delfour and Sabidussi [DS92], pages 35–111.

[Des95] Fabrice Desaint. *Dérivées par rapport au domaine en géometrie intrinsèque: Application aux équations de coques.* PhD thesis, De l'université de Nice–Sophia Antipolis, 1995.

[DL85] Robert Dautray and Jacques-Lois Lions. *Mathematical analysis and Numerical Methods for Science and Tecnology*, volume 2. Springer–Verlag, 1985.

[DS92] M.C. Delfour and Gert Sabidussi, editors. *Shape optimization and free boundaries.* Kluwer Academic Publishers, 1992.

[Dup81] F. Dupret. Étude numérique d'écoulements irrotationnels incompressibles à surface libre avec tension superficielle par une méthode variationelle. *Journal de Mécanique*, 20(4):659–690, 1981.

[Dup82] F. Dupret. A method for the computation of viscous flow by finite elements with free boundaries and surface tension. In Kawai [Kaw82].

[DZ91a] M.C. Delfour and J.P. Zolésio. Anatomy of the shape Hessian. *Ann. Mat. Pura Appl.*, 158(4):315–339, 1991.

[DZ91b] M.C. Delfour and J.P. Zolésio. Velocity method and Lagrangian formulation for the computation of the shape Hessian. *SIAM J. Control Optim.*, 29:1414–1442, 1991.

[DZ94] Michel C. Delfour and Jean-Paul Zolésio. Shape analysis via oriented distance functions. *J. Funct. Anal.*, 123(1):129–201, 1994.

[DZ97] F. R. Desaint and Jean-Paul Zolésio. Manifold derivative in the Laplace-Beltrami equation. *J. Funct. Anal.*, 151(1):234–269, 1997.

[DZ01] M. C. Delfour and J.-P. Zolésio. *Shapes and geometries.* Advances in Design and Control. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001. Analysis, differential calculus, and optimization.

[FF92] Leopoldo P. Franca and Sérgio L. Frey. Stabilized finite element methods. II. The incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 99(2-3):209–233, 1992.

[FR97] M. Flucher and M. Rumpf. Bernoulli's free-boundary problem, qualitative theory and numerical approximation. *J. Reine Angew. Math.*, 486:165–204, 1997.

[Fri82] Avner Friedman. *Variational Principles and Free-Boundary Problems.* Wiley, 1982.

[Gar56] P.R. Garabedian. The mathematical theory of three dimensional cavities and jets. *Bull Amer. Math. Soc.*, 62:219–235, 1956.

[GF63] I. M. Gelfand and S. V. Fomin. *Calculus of variations*. Revised English edition translated and edited by Richard A. Silverman. Prentice-Hall Inc., Englewood Cliffs, N.J., 1963.

[Giv91] Dan Givoli. Non-reflecting boundary conditions. *Journal of computational physics*, 94:1–29, 1991.

[Gri89] Andreas Griewank. On automatic differentiation. In *Mathematical programming (Tokyo, 1988)*, volume 6 of *Math. Appl. (Japanese Ser.)*, pages 83–107. SCIPRESS, Tokyo, 1989.

[GT77] D. Gilbarg and N.S. Trudinger. *Elliptic Partial Differential Equations of Second Order*. Springer-Verlag, 1977.

[HHM93] J. Haslinger, K.-H. Hoffmann, and R. A. E. Mäkinen. Optimal control/dual approach for the numerical solution of a dam problem. *Adv. Math. Sci. Appl.*, 2(1):189–213, 1993.

[HKKP03] J. Haslinger, T. Kozubek, K Kunisch, and G. Peichl. Shape optimization and fictitious domain approach for solving free boundary value problems of Bernoulli type. *Computational Optimization and Applications*, 26:231–251, 2003.

[HLNT03] Kai Hiltunen, Mika Laitinen, Antti Niemistö, and Pasi Tarvainen. Using mathematical concepts in software design of computational mechanics. In *Proceedings of the VIII Finnish Mechanics Days*, 2003.

[HM03] J. Haslinger and R. A. E. Mäkinen. *Introduction to shape optimization*. Advances in Design and Control. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003. Theory, approximation, and computation.

[HN96] J. Haslinger and P. Neittaanmaki. *Finite element approximation for optimal shape, material and topology design*. John Wiley & Sons Ltd., Chichester, second edition, 1996.

[JAY79] C.C. Mei J.A. Aranha and K.P. Yue. Some properties of a hybrid element method for water waves. *Internat. J. Numer. Methods Engrg.*, 14(11):1627–1641, 1979.

[Kaw82] Tadahiko Kawai, editor. *Finite Element Flow Analysis*. University of Tokyo Press, Tokyo, 1982.

[KCSV88] N.P. Kruyt, C. Cuvelier, A. Segal, and J. Van der Zanden. A total linearization method for solving viscous free boundary flow problems by the finite element method. *Internat. J. Numer. Methods Fluids*, 8:351–363, 1988.

[Kim00] Masato Kimura. Shape derivative and linearization of moving boundary problems. In *Free boundary problems: theory and applications, I (Chiba, 1999)*, volume 13 of *GAKUTO Internat. Ser. Math. Sci. Appl.*, pages 167–179. Gakkōtosho, Tokyo, 2000.

[KN90] M. Křížek and P. Neittaanmäki. *Finite element approximation of variational problems and applications*, volume 50 of *Pitman Monographs and Surveys in Pure and Applied Mathematics*. Longman Scientific & Technical, Harlow, 1990.

[Kno04] E. M. Knobbe. A FEM-ALE approach for continuous domains with arbitrary moving boundaries. In Neittaanmäki et al. [NRMP04].

[KPTZ00] B. Kawohl, O. Pironneau, L. Tartar, and J.-P. Zolésio. *Optimal shape design*, volume 1740 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2000. Lectures given at the Joint C.I.M./C.I.M.E. Summer School held in Tróia, June 1–6, 1998, Edited by A. Cellina and A. Ornelas, Fondazione C.I.M.E.. [C.I.M.E. Foundation].

[KT96] Kari Kärkkäinen and Timo Tiihonen. Trial methods for a nonlinear Bernoulli problem. Report 6, University of Jyväskylä, Department of Mathematics, Laboratory of Scientific Computing, 1996.

[KT97] Kari Kärkkäinen and Timo Tiihonen. Trial methods for nonlinear Bernoulli problem. In Brøns et al. [BBS97], pages 260–267. Papers from the 9th European Conference for Mathematics in Industry held at the Technical University of Denmark, Lyngby, June 1996.

[KT99] Kari Kärkkäinen and Timo Tiihonen. Free surfaces: shape sensitivity analysis and numerical methods. *Internat. J. Numer. Methods Engrg.*, 44(8):1079–1098, 1999.

[KT04] Kari Kärkkäinen and Timo Tiihonen. Shape calculus and free boundary problems. In Neittaanmäki et al. [NRMP04].

[LK96] B. V. Loginov and A. O. Kuznetsov. Capillary-gravity waves over a flat surface. *European J. Mech. B Fluids*, 15(2):259–280, 1996.

[LM81] M. Lenoir and D. Martin. An application of the principle of limiting absorption to the motions of floating bodies. *J. Math. Anal. Appl.*, 79(2):370–383, 1981.

[LT88] M. Lenoir and A. Tounsi. The localized finite element method and its application to the two-dimensional sea-keeping problem. *SIAM J. Numer. Anal.*, 25(4):729–752, 1988.

[Luk67] J. C. Luke. A variational principle for a fluid with a free surface. *J. Fluid Mech.*, 27:395–397, 1967.

[Mäk90] R. A. E. Mäkinen. Finite-element design sensitivity analysis for nonlinear potential problems. *Comm. Appl. Numer. Methods*, 6(5):343–350, 1990.

[MKMA03] R. A. E. Mäkinen, Hiltunen K., Laitinen M., and Niemistö A. Exact linearization in numerical simulation and optimization. In *Proceedings of the VIII Finnish Mechanics Days*, 2003.

[MT68] L. M. Milne-Thomson. *Theoretical hydrodynamics*. The Macmillan Co., New York, fifth edition, 1968.

[NRMP04] Pekka Neittaanmäki, Tuomo Rossi, Kirsi Majava, and Olivier Pironneau, editors. *European Congress on Computational Methods in Applied Sciences and Engineering*, http://www.mit.jyu.fi/eccomas2004, 2004.

[Num03] Numerola Oy. *Numerrin 2.0 käyttäjän opas*, 2003. (in finnish).

[OS88] Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.

[Pir04] Olivier Pironneau. Derivatives with respect to piecewise constant coefficients of a PDE with applicationt to calibration. In Neittaanmäki et al. [NRMP04].

[Set90] J. A. Sethian. Numerical algorithms for propagating interfaces: Hamilton-Jacobi equations and conservation laws. *J. Differential Geom.*, 31(1):131–161, 1990.

[Set96] J. A. Sethian. *Level set methods*, volume 3 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 1996. Evolving interfaces in geometry, fluid mechanics, computer vision, and materials science.

[SFDO91] Azzeddine Soulaïmani, Michel Fortin, G. Dhatt, and Y. Ouellet. Finite element simulation of two– and three–dimensional free surface flows. *Comput. Methods Appl. Mech. Engrg.*, 86(2):265–296, 1991.

[Sim80] J. Simon. Differentiation with respect to the domain in boundary value problems. *Numer. Funct. Anal. Optim.*, 2(7-8):649–687 (1981), 1980.

[Smo01] Anton Smolianski. *Numerical modeling of two-fluid interfacial flows*. PhD thesis, University of Jyväskylä, 2001.

[SSO94] Mark Sussman, Peter Smereka, and Stanley Osher. A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.*, 114:146–159, 1994.

[Sto57] J.J. Stoker. *Water waves*. Interscience publishers, INC., New York, 1957.

[SZ92] Jan Sokolowski and J.P. Zolésio. *Introduction to shape optimization*. Springer-Verlag, 1992.

[SZ96] M. Souli and J.P. Zolésio. Finite element method for free surface problem. *Comput. Methods Appl. Mech. Engrg.*, 129:43–51, 1996.

[Tar00] Domingo Alberto Tarzia. A bibliography on moving-free boundary problems for the heat-diffusion equation. the stefan and related problems. Technical report, Departamento de Matemática - CONICET, Facultad de Ciencias Empresariales, Universidad Austral, Paraguay 1950, S2000FZF Rosario, ARGENTINA., 2000.

[Tii97] Timo Tiihonen. Shape optimization and trial methods for free boundary problems. *RAIRO Modél. Math. Anal. Numér.*, 31(7):805–825, 1997.

[Tii98] T. Tiihonen. Fixed point methods for internal free boundary problems. *Numer. Funct. Anal. Optim.*, 19(3-4):399–413, 1998.

[TJ91] T. Tiihonen and J. Järvinen. On fixed point (trial) methods for free boundary problems. In Antontsev et al. [AHK92].

[TM05] Jukka Toivanen and Janne Martikainen. A new method for creating sparse design velocity fields. Computer Methods in Applied Mechanics and Engineering (submitted), 2005.

[Tor00] Anna-Karin Tornberg. *Interface Tracking Methods with Application to Multiphase Flows.* PhD thesis, Royal Institute of Technology, Department of Numerical Analysis and Computing Science, Stockholm, 2000.

[VB97] Jean-Marc Vanden-Broeck. Local gravity-capillary flows near the intersection of a free surface with a vertical wall. *European J. Appl. Math.*, 8(2):177–184, 1997.

[Whi74] G.B. Whitham. *Linear and nonlinear waves.* John Wiley & sons, 1974.

[Zol79] J.P. Zolésio. *Identification de domaines par deformations.* PhD thesis, Univ. Nice, 1979.

[Zol86] J.P. Zolésio. Shape variational solution for a Bernoulli like stedy free boundary problem. In F.Kappel, K.Kunisch, and W.Schappacher, editors, *Distributed Parameter Systems.* Springer–Verlag, 1986.

[Zol92] J.P. Zolésio. Introduction to shape optimization problems and free boundary problems. In Delfour and Sabidussi [DS92], pages 397–457.

[Zol94] J.P. Zolésio. Weak shape formulation of free boundary problems. *Annali della Scuola Normale Superiore di Pisa*, XXI(1):11–44, 1994.

YHTEENVETO (FINNISH SUMMARY)

Tässä työssä kehitetään tehokkaita ja käyttökelpoisia ratkaisumenetelmiä vapaan reunan tehtäville. Työ koostuu mallitehtävien matemaattisesta analyysistä ja siihen pohjautuvasta menetelmän konvergenssitarkastelusta, vastaavan numeerisen toteutuksen esittelystä sekä numeerisista esimerkeistä.

Työssä tutkittavat vapaan reunan tehtävät ovat ylimääriteltyjä elliptisiä osittaisdifferentiaaliyhtälöitä. Ylimääritellyt reunaehdot voivat toteutua vain erityisessä geometriassa, joka ratkaisee yhtälöillä kuvatun vapaan reunan tehtävän. Vapaan reunan tehtävät ovat luonteeltaan vahvasti epälineaarisia, joten niitä ei voi ratkoa suorin menetelmin. Vapaan reunan ratkaisumenetelmät ovat yleisesti iteratiivisia ratkaisumenetelmiä, joissa tuntematon geometria ratkaistaan laskemalla sarja tehtäviä kiinnitetyissä alueissa.

Työssä esitetty ratkaisumenetelmä pohjautuu jatkuvan tason herkkyysanalyysiin sekä diskretisoitujen yhtälöiden automaattiseen derivointiin. Jotta diskretoidun yhtälön automaattinen derivointi vastaisi jatkuvalle yhtälölle tehtyä herkkyysanalyysiä, on elementtimenetelmää modifioitava.

Esitetyn menetelmän tehokkuus testataan ja havainnollistetaan numeeristen esimerkkien avulla.