Reports of the Department of Mathematical Information Technology
Series B. Scientific Computing
No. B. 12/2007

Shape optimization of systems governed by Bernoulli free boundary problems

Jukka I. Toivanen Raino A. E. Mäkinen
Jaroslav Haslinger

University of Jyväskylä
Department of Mathematical Information Technology
P.O. Box 35 (Agora)
FI–40014 University of Jyväskylä
FINLAND
fax +358 14 260 2731
http://www.mit.jyu.fi/

ISBN 978-951-39-2860-5
ISSN 1456-436X

Shape optimization of systems governed by Bernoulli free boundary problems*

Jukka I. Toivanen† Raino A. E. Mäkinen† Jaroslav Haslinger‡

Abstract

In this work we consider shape optimization of systems, which are governed by external Bernoulli free boundary problems. A pseudo-solid approach for solving discrete free boundary problems is introduced. The solution strategy readily allows us to obtain geometrical sensitivities of the system, which can then be used to solve e.g. inverse design problems. Numerical examples show that the location of the free boundary can, to some extent, be controlled by changing the shape of the other component of the boundary.

1 Introduction

Free boundary problems of Bernoulli type arise in mathematical modeling of the ideal fluid flow and the electro-chemical machining, for example. The exterior Bernoulli problem can be formally stated as follows: Given domain ω and a constant $\gamma < 0$, find a domain $\Omega \supset \omega$ and a potential u such that

$$\begin{cases} \Delta u = 0 & \text{in } \Omega \setminus \overline{\omega} \\ u = 1 & \text{on } \partial\omega \\ u = 0 \quad \text{and} \quad \dfrac{\partial u}{\partial \boldsymbol{n}} = \gamma & \text{on } \partial\Omega \end{cases} \tag{1}$$

(see Figure 1).

Free boundary problems have in common the difficulty that the geometry (here the domain Ω) has to be determined simultaneously with the solution of the state problem, which implies that a numerical solution has to be done iteratively [11].

*This research was supported by the Academy of Finland, grants #112415, #112445, the Finnish Cultural Foundation, and Tekniikan Edistämissäätiö. The third author acknowledges MSM0021620839.

†Department of Mathematical Information Technology, University of Jyväskylä, PO Box 35 (Agora), FI-40014 University of Jyväskylä, Finland, jukka.toivanen@jyu.fi, raino.makinen@jyu.fi

‡Department of Numerical Mathematics, Faculty of Mathematics and Physics, Charles University, Sokolovska 83, 186 75 Praha 8, Czech Republic, hasling@karlin.mff.cuni.cz

Possible solution strategies include trial methods, linearization methods (continuous or discrete) [4], and shape optimization methods [9].

In the so-called trial methods one solves a relaxed problem (i.e. one of the boundary conditions on $\partial\Omega$ is discarded) on a fixed computational grid and then updates the location of the free boundary based on the violation of the discarded boundary condition. The grid is then deformed or regenerated to correspond to the new iterate. This process is continued until the previously discarded boundary condition is approximately satisfied. In Newton's method the positions of the grid points on the free boundary are introduced directly as unknowns. The resulting coupled system is then solved using the Newton iteration, thus computing the field variables and the position of the free boundary simultaneously.

In this paper we are interested in the analysis and approximation of the following identification/control problem. We are looking for a domain $\omega \subset \Omega$ such that the free boundary $\partial\Omega$ and the corresponding potential u minimize a suitable least squares cost functional. Thus we solve a "double free boundary problem", i.e. we have to identify a design and a physical moving boundary.

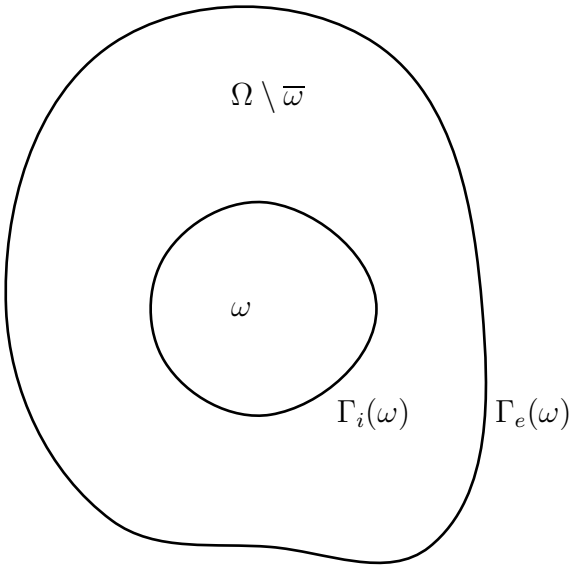

Figure 1: Geometry of the exterior Bernoulli problem

2 Setting of the problem

We start with the definition of the state problem represented by an exterior Bernoulli free boundary problem.

Let $\gamma < 0$ and an open set $\omega \subset \mathbb{R}^2$ with a sufficiently regular boundary $\partial\omega$ be

given. Our aim is to find a set $\Omega \supset \overline{\omega}$ and a function $u : \Omega \setminus \overline{\omega} \to \mathbb{R}$ satisfying

$$\left\{ \begin{array}{rl} \Delta u = 0 & \text{in } \Omega \setminus \overline{\omega} \\ u = 1 & \text{on } \partial\omega \\ \left.\begin{array}{l} u = 0 \\ \dfrac{\partial u}{\partial \boldsymbol{n}} = \gamma \end{array}\right\} & \text{on } \partial\Omega. \end{array} \right. \tag{$\mathcal{P}(\omega)$}$$

Next we shall consider ω to be a *control variable* by means of which the shape of Ω will be governed. To this end we introduce a system $\widetilde{\mathcal{O}}$ of all admissible ω. To emphasize that solutions of $(\mathcal{P}(\omega))$, $\omega \in \widetilde{\mathcal{O}}$ depend on a particular choice of ω we shall write $\Omega(\omega)$, $u(\omega)$ in what follows. The outer, inner component of the boundary of $\Omega(\omega) \setminus \overline{\omega}$ will be denoted by $\Gamma_e(\omega)$ and $\Gamma_i(\omega)$, respectively (see Figure 1).

Our goal will be to find "an optimal" $\Gamma_e(\omega)$ by minimizing an appropriate cost functional J which depends on $(\Omega(\omega), u(\omega))$ – a solution to $(\mathcal{P}(\omega))$. We shall suppose that $\widetilde{\mathcal{O}}$ is chosen in such a way that $(\mathcal{P}(\omega))$ has *at least* one solution for any $\omega \in \widetilde{\mathcal{O}}$. The set of all solutions of $(\mathcal{P}(\omega))$ for a given $\omega \in \widetilde{\mathcal{O}}$ will be denoted by $X(\omega)$.

An optimization problem reads as follows:

$$\left\{ \begin{array}{l} \text{Find } \omega^* \in \widetilde{\mathcal{O}} \text{ such that} \\ J(\Omega(\omega^*), u(\omega^*)) \le J(\Omega(\omega), u(\omega)) \end{array} \right. \tag{$\widetilde{\mathbb{P}}$}$$

holds for any $(\Omega(\omega), u(\omega)) \in X(\omega)$ and every $\omega \in \widetilde{\mathcal{O}}$.

To simplify our presentation and numerical realization we choose $\widetilde{\mathcal{O}}$ in such a way that $(\mathcal{P}(\omega))$ has a *unique* solution for any $\omega \in \widetilde{\mathcal{O}}$. If it is so, problem $(\widetilde{\mathbb{P}})$ can be written in the following form:

$$\left\{ \begin{array}{l} \text{Find } \omega^* \in \widetilde{\mathcal{O}} \text{ such that} \\ J(\Omega(\omega^*), u(\omega^*)) \le J(\Omega(\omega), u(\omega)) \qquad \forall \omega \in \widetilde{\mathcal{O}}. \end{array} \right. \tag{$\mathbb{P}$}$$

A possible choice of $\widetilde{\mathcal{O}}$ which guarantees the uniqueness of the solution to $(\mathcal{P}(\omega))$ as well as its stability with respect to $\omega \in \widetilde{\mathcal{O}}$ is given by

$$\begin{aligned} \widetilde{\mathcal{O}} = \{\omega \subset \mathbb{R}^2 \mid \omega_0 \subset \omega \subset \omega_1, \omega \text{ is star-like with respect to} \\ \text{all points in the ball } B_\delta(0) \text{ and } \partial\omega \text{ is of the class } C^2\}, \end{aligned} \tag{2}$$

where ω_0, ω_1 are given non-empty open sets, ω_0 contains the origin, and the radius $\delta > 0$ is the same for all $\omega \in \widetilde{\mathcal{O}}$.

Indeed, in [1] the following results have been proven:

Theorem 1 *Let $\widetilde{\mathcal{O}}$ be defined by* (2)*. Then for every $\omega \in \widetilde{\mathcal{O}}$ problem $(\mathcal{P}(\omega))$ has a unique solution $(\Omega(\omega), u(\omega))$. The outer component $\Gamma_e(\omega)$ is of the class C^∞ and is star-like with respect to all points in $B_\delta(0)$. In addition, from $\partial\omega_n \rightrightarrows \partial\omega$ (uniformly) $(\omega_n, \omega \in \widetilde{\mathcal{O}})$ it follows that $\Gamma_e(\omega_n) \rightrightarrows \Gamma_e(\omega)$.*

In the next sections we shall use the following cost functionals:

$$J(\Omega(\omega), u(\omega)) = \|\Gamma_e(\omega) - \hat{\Gamma}\|^2 \tag{3}$$

$$J(\Omega(\omega), u(\omega)) = \|u(\omega) - z_d\|^2_{0,\Omega(\omega)\setminus\overline{\omega}}, \tag{4}$$

where $\hat{\Gamma}$ is the target boundary and $z_d \in L^2_{\text{loc}}(\mathbb{R}^2)$ is a given function. The norm $\|\cdot\|$ in (3) is chosen to be continuous with respect to uniform convergence of free boundaries:

$$\Gamma_e(\omega_n) \rightrightarrows \Gamma_e(\omega) \implies \|\Gamma_e(\omega_n) - \Gamma_e(\omega)\| \to 0, \quad n \to \infty. \tag{5}$$

To ensure the existence of a solution to $(\mathbb{P})$ one needs also a *compactness* property of $\widetilde{\mathcal{O}}$. This can be obtained by restricting ourselves to an appropriate subset of $\widetilde{\mathcal{O}}$, defined by (2). Let $\mathcal{O} \subseteq \widetilde{\mathcal{O}}$ be compact in the following sense:

$$\begin{cases} \text{for any sequence } \{\omega_n\},\ \omega_n \in \mathcal{O} \\ \text{there exists a subsequence } \{\omega_{n_j}\} \subset \{\omega_n\} \text{ and } \omega \in \mathcal{O} \text{ such that } \partial\omega_{n_j} \rightrightarrows \partial\omega. \end{cases} \tag{6}$$

Let us comment on the assumption (6). In addition to (2), suppose that the system $\mathcal{O}$ consists of domains possessing the *uniform cone property*. Then from any sequence $\{\omega_n\},\ \omega_n \in \mathcal{O}$ one can choose a subsequence $\{\omega_{n_j}\} \subset \{\omega_n\}$ converging in the Hausdorff metric to a domain ω which possesses the same cone property as elements of $\{\omega_n\}$ ([10]). Moreover, it holds that $\partial\omega_{n_j} \rightrightarrows \partial\omega$ implying that ω is star-like with respect to all points in $B_\delta(0)$. To get the C^2-regularity of $\partial\omega$ it is sufficient to suppose that radial functions describing the boundaries of domains from $\mathcal{O}$ are uniformly bounded in the $C^{2,1}$-norm. Thus the compactness property (6) is satisfied.

We now are able to prove the main result of this section.

Theorem 2 *Let J be defined by* (3) *or* (4) *and let $\mathcal{O} \subseteq \widetilde{\mathcal{O}}$ satisfy* (6)*. Then $(\mathbb{P})$ has at least one solution with $\widetilde{\mathcal{O}} := \mathcal{O}$.*

Proof: Denote by $\{\omega_n\},\ \omega_n \in \mathcal{O}$ a minimizing sequence of $(\mathbb{P})$:

$$\inf_{\omega\in\mathcal{O}} J(\Omega(\omega), u(\omega)) = \lim_{n\to\infty} J(\Omega(\omega_n), u(\omega_n)).$$

From (6) we know that there exists a subsequence $\{\omega_{n_j}\} \subset \{\omega_n\}$ and $\omega^* \in \mathcal{O}$ such that

$$\partial\omega_{n_j} \rightrightarrows \partial\omega^*, \quad j \to \infty$$

and consequently

$$\Gamma_e(\omega_{n_j}) \rightrightarrows \Gamma_e(\omega^*)$$

as follows from Theorem 1. For J defined by (3) we immediately see that $(\Omega(\omega^*), u(\omega^*))$ is a solution of $(\mathbb{P})$ making use of (5).

Let D be a rectangle containing $\Omega(\omega^*)$ and $\Omega(\omega_n)$ $\forall n \in \mathbb{N}$ in its interior. Then it is easy to show that

$$\tilde{u}(\omega_n) \to \tilde{u}(\omega^*) \quad \text{in } H^1_0(D), \tag{7}$$

where "˜" stands for the zero extension of functions from the domain of their definition on D (see [9]). From this it easily follows that

$$\|u(\omega_n) - z_d\|_{0,\Omega(\omega_n)\setminus\overline{\omega}_n} \to \|u(\omega^*) - z_d\|_{0,\Omega(\omega^*)\setminus\overline{\omega}^*}.$$

Thus $(\mathbb{P})$ has a solution if J is defined by (4), too. □

3 Pseudo-solid approach for the free boundary problem

Positions of the grid points on the free boundary are introduced directly as unknowns in Newton's method. Usually this means that the dependence of the location of the internal grid points on the locations of the boundary ones must be known. In [12], for example, this is done by constructing a conformal mapping between the computational domain and a simple reference domain. Instead, our solution strategy for the free boundary problem is as follows. Let $\widehat{\Xi} := \hat{\Omega} \setminus \overline{\hat{\omega}} \subset \mathbb{R}^2$ be a *fixed* double connected *reference* domain. The outer, inner component of its boundary will be denoted by $\hat{\Gamma}_e$, $\hat{\Gamma}_i$, respectively. Our aim will be to find a mapping $F : \mathbb{R}^2 \to \mathbb{R}^2$, $\hat{\Omega} \mapsto F(\hat{\Omega}) := \Omega$, $\hat{\omega} \mapsto F(\hat{\omega}) := \omega$ such that the outer component $\Gamma_e(\Xi)$ of the double connected domain $\Xi = \Omega \setminus \overline{\omega}$ is the free boundary in (1).

To construct a mapping F we use the so-called *pseudo-solid approach* [13]. We treat $\widehat{\Xi}$ as an elastic solid that undergoes a deformation such that the deformed solid defines the domain Ξ (see Figure 2). Thus, problem (1) is strongly coupled with the linear elasticity system. An external loading applied to $\widehat{\Xi}$ then has the role of the control variable in the free boundary problem. This approach has been used to solve free surface flow problems (see e.g. [3, 15]), but to our knowledge it has not yet been applied to Bernoulli free boundary problems.

Next we present the weak formulation of the pseudo-solid approach for the Bernoulli problem. For any $\boldsymbol{w} \in W^{ad}$:={"sufficiently" small and regular deformations} we define a domain

$$\Xi_{\boldsymbol{w}} = \{\boldsymbol{x} \in \mathbb{R}^2 \mid \boldsymbol{x} = \hat{\boldsymbol{x}} + \boldsymbol{w}(\hat{\boldsymbol{x}}), \quad \hat{\boldsymbol{x}} \in \widehat{\Xi}\}.$$

Let c be a constant and $g : [0, 2\pi] \to \mathbb{R}$ be a sufficiently smooth function, $g(0) = g(2\pi)$. We define the following function spaces:

$$W_g = \{\boldsymbol{w} \in [H^1(\widehat{\Xi})]^2 \mid \boldsymbol{w}_{|\hat{\Gamma}_i} = g(\theta)(\cos\theta, \sin\theta),\ \theta \in [0, 2\pi[\,\} \tag{8}$$

$$V_c(\Xi) = \{\varphi \in H^1(\Xi) \mid \varphi = c \quad \text{on } \Gamma_i(\Xi)\} \tag{9}$$

where $\Gamma_i(\Xi)$ is the inner component of the boundary of Ξ.

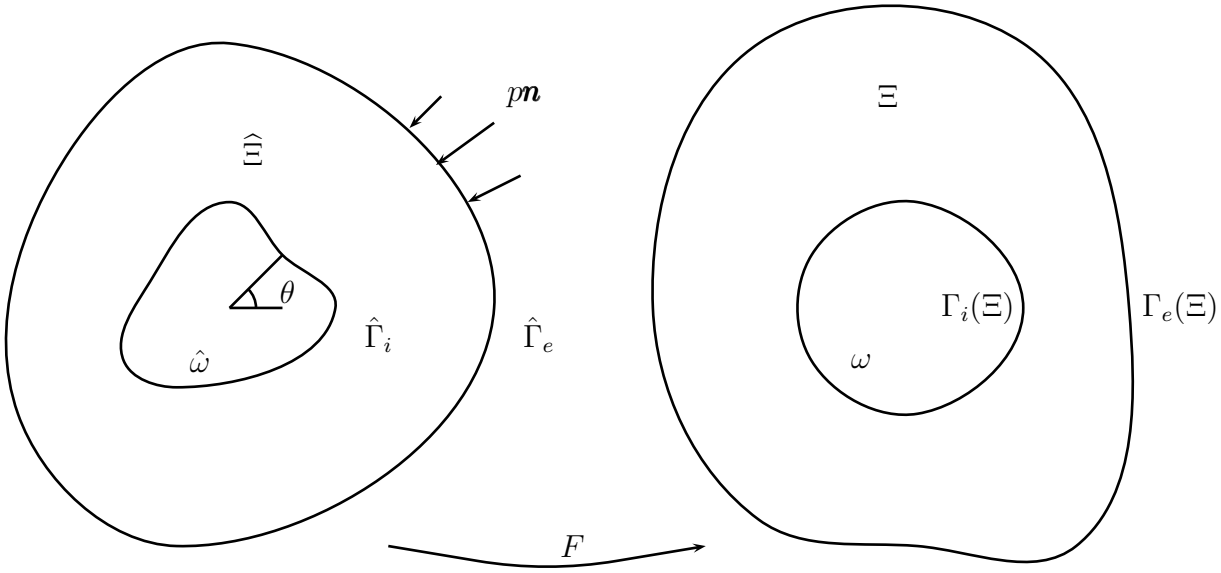

Figure 2: Reference domain (left) and the real geometry of the Bernoulli problem (right).

The weak pseudo-solid formulation then reads: Given g, find $(\boldsymbol{v}, u, p) \in W_g \times V_1(\Xi_{\boldsymbol{v}}) \times L^2(\hat{\Gamma}_e)$ such that

$$\int_{\Xi_{\boldsymbol{v}}} \nabla u \cdot \nabla \varphi \, dx = \gamma \int_{\Gamma_e(\Xi_{\boldsymbol{v}})} \varphi \, ds \quad \forall \varphi \in V_0(\Xi_{\boldsymbol{v}}) \tag{10}$$

$$\int_{\Gamma_e(\Xi_{\boldsymbol{v}})} u\psi \, ds = 0 \quad \forall \psi \in L^2(\Gamma_e(\Xi_{\boldsymbol{v}})) \tag{11}$$

$$\int_{\widehat{\Xi}} \sigma(\boldsymbol{v}) : \varepsilon(\boldsymbol{w}) \, dx = \int_{\hat{\Gamma}_e} p\boldsymbol{n} \cdot \boldsymbol{w} \, ds \quad \forall \boldsymbol{w} \in W_0 \tag{12}$$

Equations (10) and (11) constitute the weak form of (1) while (12) is the weak form of the linear elasticity problem in $\hat{\Xi}$. Here $p\boldsymbol{n}$ is the (unknown) external force. The components of the strain and stress tensors $\varepsilon = \{\varepsilon_{ij}\}$and $\sigma = \{\sigma_{ij}\}$ are given by

$$\varepsilon_{ij}(\boldsymbol{v}) = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right), \quad \sigma_{ij}(\boldsymbol{v}) = 2\mu\epsilon_{ij}(\boldsymbol{v}) + \lambda\delta_{ij}\nabla \cdot \boldsymbol{v}, \quad i, j = 1, 2,$$

where $\boldsymbol{v}$ is a displacement field and μ and λ are Lamé's coefficients. Since in our case the linear elasticity system does not have any physical meaning, Lamé's coefficients can be chosen quite freely. In this paper the choice $\mu = 0.5$ and $\lambda = 0$ was used. Prescribed displacements are specified on $\hat{\Gamma}_i$ by the radial function g.

4 Finite element discretization of the direct problem

In the pseudo-solid approach we simultaneously seek the scalar function u, the pressure p, and the deformation field $\boldsymbol{v}$ which deforms the reference domain $\widehat{\Xi}$ into the

one that solves (1). The elasticity system (12) is thus solved in the undeformed configuration $\hat{\Xi}$ of the pseudo-solid, whereas equations (10) and (11) are solved in the deformed one. Therefore, they have to be discretized by different meshes, too. Let us denote the nodal co-ordinates of a triangulation of $\widehat{\Xi}$ by $\hat{\boldsymbol{X}}$. We simply transform this triangulation into the one of Ξ by $\boldsymbol{X} = F(\hat{\boldsymbol{X}})$, where F is defined by the discrete displacement field $\boldsymbol{v}_h$ being the approximation of $\boldsymbol{v}$, i.e.

$$\boldsymbol{X} = \hat{\boldsymbol{X}} + \boldsymbol{v}_h. \tag{13}$$

The algebraic form of (10), (11) resulting from an appropriate discretization can be written as $\boldsymbol{r}_1(\boldsymbol{q}_u, \boldsymbol{q}_v) = \boldsymbol{0}$, $\boldsymbol{r}_2(\boldsymbol{q}_u, \boldsymbol{q}_v) = \boldsymbol{0}$, respectively and the linear elasticity system (12) as $\boldsymbol{r}_3(\boldsymbol{q}_v, \boldsymbol{q}_p) = \boldsymbol{0}$. Here the dependence of $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ on $\boldsymbol{q}_v$ is through the mesh nodal co-ordinates, as specified by (13). Dimensions of the vectors $\boldsymbol{q}_u$, $\boldsymbol{q}_v$ and $\boldsymbol{q}_p$ are n, $2n$ and n_e respectively, where n is the number of the nodes in the mesh and n_e is the number of the nodes on Γ_e.

Let us introduce notation

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q}_u \\ \boldsymbol{q}_v \\ \boldsymbol{q}_p \end{bmatrix} \quad \text{and} \quad \boldsymbol{r} = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \end{bmatrix}. \tag{14}$$

Then the algebraic form of the discretized coupled system (10) – (12) can be written in short as $\boldsymbol{r}(\boldsymbol{q}) = \boldsymbol{0}$. This system will be solved using Newton's method:

$$\boldsymbol{q}^{(k+1)} = \boldsymbol{q}^{(k)} - \left(\frac{\partial \boldsymbol{r}(\boldsymbol{q}^{(k)})}{\partial \boldsymbol{q}} \right)^{-1} \boldsymbol{r}(\boldsymbol{q}^{(k)}) \tag{15}$$

with the Jacobian matrix

$$\left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} \right) = \begin{bmatrix} \frac{\partial \boldsymbol{r}_1}{\partial \boldsymbol{q}_u} & \frac{\partial \boldsymbol{r}_1}{\partial \boldsymbol{q}_v} & \boldsymbol{0} \\ \frac{\partial \boldsymbol{r}_2}{\partial \boldsymbol{q}_u} & \frac{\partial \boldsymbol{r}_2}{\partial \boldsymbol{q}_v} & \boldsymbol{0} \\ \boldsymbol{0} & \frac{\partial \boldsymbol{r}_3}{\partial \boldsymbol{q}_v} & \frac{\partial \boldsymbol{r}_3}{\partial \boldsymbol{q}_p} \end{bmatrix} \tag{16}$$

The overall solution process proceeds as follows:

1. Get a good initial guess for Newton's method:
   (a) Solve the linear elasticity system with the function g giving the prescribed displacement on the inner boundary $\hat{\Gamma}_i$ while the outer boundary $\hat{\Gamma}_e$ is taken to be pressure free ($p_h^{(0)} = 0$). An initial guess $\boldsymbol{v}_h^{(0)}$ is obtained.
   (b) Set $\boldsymbol{X} = \hat{\boldsymbol{X}} + \boldsymbol{v}_h^{(0)}$.
   (c) Solve the Laplace equation (10) on this new mesh to obtain an initial guess $u_h^{(0)}$.
   (d) Set $k := 0$.

2. Assemble the Jacobian (16) at the current iterate $\boldsymbol{q}^{(k)}$. The residuals $\boldsymbol{r}_1(\boldsymbol{q}^{(k)})$ and $\boldsymbol{r}_2(\boldsymbol{q}^{(k)})$ are thus evaluated using the mesh with the nodal co-ordinates $\boldsymbol{X} = \hat{\boldsymbol{X}} + \boldsymbol{v}_h^{(k)}$.

3. If the norm $\|\boldsymbol{r}(\boldsymbol{q}^{(k+1)})\| \leq toler$ STOP, else perform the Newton update (15) and go to step 2.

Remark 1 As a result of the Newton iteration, we get an approximation for the potential u_h and the corresponding domain Ξ through $\boldsymbol{v}_h$. As a side product, we also obtain the discrete pressure p_h, which is needed to deform the mesh of the reference domain into the mesh of the final domain.

Remark 2 Despite the slightly nonstandard coupling between the equations, the Jacobian matrix (16) is easy to compute using the automatic differentiation. Introduction to the principles of automatic differentiation can be found for example in [8].

Remark 3 Since the coupled system is non-linear, it is possible that the Newton iteration will not converge. This usually happens when the deformation of the original mesh is so large that the resulting mesh would be highly distorted. This problem is addressed using the mesh regeneration strategy, as will be explained later.

5 Discrete optimization problem

The cost functionals (3) and (4) in the discrete setting are expressed as

$$\mathcal{J}_1(\boldsymbol{v}_h(\boldsymbol{\alpha})) = \int_{\Gamma_e(\Xi_{\boldsymbol{v}_h})} (R(\theta) - \widehat{R}(\theta))^2 \, ds, \tag{17}$$

and

$$\mathcal{J}_2(u_h(\boldsymbol{\alpha}), \boldsymbol{v}_h(\boldsymbol{\alpha})) = \int_{\Xi_{\boldsymbol{v}_h}} (u_h - z_d)^2 \, dx \tag{18}$$

where $\boldsymbol{\alpha}$ is the vector of discrete design variables defining the shape of $\Gamma_i(\Xi)$ and u_h and $\boldsymbol{v}_h$ satisfy the discretized coupled system (10) – (12). Here $R(\theta)$ is the radius of the free boundary corresponding to $\boldsymbol{\alpha}$ (a suitable discretization of g) and $\widehat{R}(\theta)$ is the radius of the target free boundary at the angle θ.

The integration in (18) takes place in the domain which solves the discrete free boundary problem, and the integration in (17) is carried out along the corresponding free boundary. That is, for a given design $\boldsymbol{\alpha}$ we solve the free boundary problem using the approach presented in Section 3 to obtain the corresponding domain $\Xi_{\boldsymbol{v}_h}$ and the outer boundary $\Gamma_e(\Xi_{\boldsymbol{v}_h})$. The cost related to $\boldsymbol{\alpha}$ is then evaluated using either (17) or (18).

Notice that dependencies of $\mathcal{J}_1$ and $\mathcal{J}_2$ on $\boldsymbol{v}_h$ are not trivial at all. The domain $\Xi_{\boldsymbol{v}_h}$ and in particular, the free boundary $\Gamma_e(\Xi_{\boldsymbol{v}_h})$ obviously depend on $\boldsymbol{v}_h$. In addition, so do all entities in $\mathcal{J}_1$ and $\mathcal{J}_2$ that are of the geometric nature, such as the radius R, angle θ and the function z_d, which in general depends on the location.

5.1 Shape parametrization

In this work we restrict ourselves to star shaped geometries. The inner boundary Γ_i is defined using the polar co-ordinates and the radius $g(\theta)$ is parametrized using uniform B-splines of degree 3 in what follows.

Let $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_{N-1})$ be the vector of control variables and θ be a given angle in the interval $[0, 2\pi[$. The B-spline of degree 0 is defined as

$$B_i^0(\theta) = \begin{cases} 1, & \text{if } t_i \leq \theta < t_{i+1} \\ 0, & \text{otherwise.} \end{cases}$$

Here t_i are the knots, i.e. locations of the control points. For uniform B-splines the distance between successive knots is constant, in this case $t_{i+1} - t_i = 2\pi/N$.

The B-splines of a higher degree are defined recursively:

$$B_i^k(\theta) = \frac{\theta - t_i}{t_{i+k} - t_i} B_i^{k-1}(\theta) + \frac{t_{i+k+1} - \theta}{t_{i+k+1} - t_{i+1}} B_{i+1}^{k-1}(\theta), \quad \theta \in [0, 2\pi[.$$

Finally, the radius function is given by

$$g(\theta) = \sum_{i=0}^{N-1} \alpha_i B_i^k(\theta) \tag{19}$$

Notice, that at most $k+1$ basis functions B_i^k are non-zero at each θ. Since the indices of the control variables are taken modulo N, no end conditions are needed for the spline parameterization.

5.2 Sensitivity analysis

In this section we explain how the cost functionals $\mathcal{J}_1$ and $\mathcal{J}_2$ can be differentiated with respect to $\boldsymbol{\alpha}$, which defines the shape of the inner boundary Γ_i. The principles remain the same for both cost functionals, the only difference is the way in which the cost functional depends on the solution $\boldsymbol{q}$. Therefore, in the following section we denote by $\mathcal{J}$ either $\mathcal{J}_1$ or $\mathcal{J}_2$.

The sensitivity analysis can be done using either the *direct* or the *adjoint* approach. In both cases, the sensitivity analysis is performed on the discretized coupled system. This approach very naturally gives us the sensitivity of the location of the free boundary on $\boldsymbol{\alpha}$. This results in exact sensitivity analysis for the discretized problem.

5.2.1 Discrete direct differentiation

The solution vector $\boldsymbol{q}$ obviously depends on the design $\boldsymbol{\alpha}$ through the shape of the domain, but this dependence is implicit. Using notation (14), the discretized coupled system can be written as

$$\boldsymbol{r}(\boldsymbol{q}(\boldsymbol{\alpha}), \boldsymbol{\alpha}) = \boldsymbol{0}. \tag{20}$$

The implicit function theorem says that

$$\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\alpha}} = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\alpha}}, \tag{21}$$

from which the matrix of partial derivatives $\partial \boldsymbol{q}/\partial \boldsymbol{\alpha}$ can be computed.

Since the mesh displacements are also included in $\boldsymbol{q}$ through the linear elasticity system, this analysis also provides the sensitivity of the geometry on $\boldsymbol{\alpha}$. In particular, this analysis gives us the sensitivity of the location of the free boundary with respect to $\boldsymbol{\alpha}$.

The cost functional $\mathcal{J}$ can now be differentiated and its gradient is given by

$$\nabla_{\boldsymbol{\alpha}} \mathcal{J} = \left(\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\alpha}} \right)^T \nabla_{\boldsymbol{q}} \mathcal{J}. \tag{22}$$

5.2.2 Discrete adjoint formulation

Another way of performing the sensitivity analysis is based on the discrete adjoint approach [7]. We introduce the Lagrangian

$$\mathcal{L}(\boldsymbol{\alpha}, \boldsymbol{q}) = \mathcal{J}(\boldsymbol{q}) + \boldsymbol{\nu}^T \boldsymbol{r}(\boldsymbol{\alpha}, \boldsymbol{q}),$$

where $\boldsymbol{\nu}$ is the vector of Lagrange multipliers. Now

$$\nabla_{\boldsymbol{\alpha}} \mathcal{L} = \left(\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\alpha}} \right)^T \nabla_{\boldsymbol{q}} \mathcal{J} + \left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\alpha}} + \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\alpha}} \right)^T \boldsymbol{\nu} \tag{23}$$

which, by rearranging, yields

$$\nabla_{\boldsymbol{\alpha}} \mathcal{L} = \left(\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\alpha}} \right)^T \left\{ \nabla_{\boldsymbol{q}} \mathcal{J} + \left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} \right)^T \boldsymbol{\nu} \right\} + \left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\alpha}} \right)^T \boldsymbol{\nu}. \tag{24}$$

We choose the Lagrangian multipliers satisfying the adjoint equation

$$\left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} \right)^T \boldsymbol{\nu} = -\nabla_{\boldsymbol{q}} \mathcal{J}, \tag{25}$$

so that the gradient $\nabla_{\boldsymbol{\alpha}} \mathcal{L}$ can be expressed by

$$\nabla_{\boldsymbol{\alpha}} \mathcal{L} = \left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\alpha}} \right)^T \boldsymbol{\nu}. \tag{26}$$

The advantage of this approach over the direct discrete approach consists in solving only one set of linear equations (25), whereas in (21) there are N different right hand sides. Both approaches, however, lead to the same discrete gradient (up to numerical precision), so nothing is lost by using the adjoint approach.

Using a direct solver, the system (21) can be solved efficiently also for multiple right hand sides. On the other hand, if the problem was more complicated (3D or matrix factorization is not easy, e.g.), the adjoint approach would significantly increase the computational efficiency of the sensitivity analysis.

5.3 Mesh regeneration

When deformed meshes are used, robustness is always an important issue. If the mesh gets too distorted, significant errors in the numerical solution and gradient information may appear.

On the other hand, a regeneration of the mesh always takes some time, and it can also introduce noise in the cost functional and gradient values. This is due to the fact that even if two meshes represent exactly the same domain, the obtained solutions using these meshes are usually a little different. Therefore, also the cost functional values may be different. As a result, the cost functional value may increase due to the regeneration of the mesh. This can confuse the optimizer, because the cost functional value may suddenly increase in a direction that is supposed to be descent. To avoid this kind of problems we restarted the optimizer using the previous design as an initial guess every time when the mesh was regenerated.

As mentioned earlier, the Newton iteration used to solve the free boundary problem can fail. This happens more frequently at the early stage of the optimization process when the domain changes significantly. One of factors that affect the probability of failure is the initial guess for the location of the free boundary, i.e. how much does the mesh have to be deformed to solve the free boundary problem.

We also reject the design candidate in the case when the Newton iteration converges but the quality of some element in the resulting mesh is poor. For evaluating the element quality, we use the triangle quality metric presented e.g. in [2]. Quality of a triangle E is given by

$$Q(E) = \frac{4\sqrt{3}|E|}{l_1^2 + l_2^2 + l_3^2}, \tag{27}$$

where $|E|$ is the signed surface area of E and l_i is the length of the i:th edge of E. This metric equals 1 if the triangle is equilateral, 0 if it is degenerate, and a negative value if the element is inverted. The bound indicating a poor element was chosen to be 0.1.

We use the following strategy to obtain a reasonable compromise between the mesh quality and the computational overhead caused by the regeneration of the mesh: If the solver fails or the point gets rejected twice during one line search of an optimizer, we regenerate the mesh and restart the optimization. Although this condition seems to be quite limiting, the overall number of mesh regenerations was still quite modest. Besides, the progress of the optimization without the mesh regeneration would be quite slow anyway, because we would be forced to reject a lot of points, and the optimizer would have to take very short steps in the descent direction.

The new mesh is generated as follows. One fits by least squares a parameterized curve (similar to the inner boundary) to the outer boundary of the previous feasible mesh (that corresponds to a solution of a direct Bernoulli problem). The inner boundary is given by the boundary parameterization, corresponding to the latest feasible mesh. Having these inner and outer boundaries one generates a new mesh. Doing this, one never needs to excessively deform the mesh for the Laplace

equation.

6 Numerical examples

The problem (10) – (12) was discretized by standard triangular finite elements. All unknowns were approximated by piecewise linear continuous functions. Both inner and outer boundaries were discretized using 400 straight line segments in all examples. The mesh generator Netgen [14] was used to generate meshes. Systems of linear equations were solved by a direct linear solver called SuperLU [5].

The coupled system is non-linear, but behaves quite well. Usually 5-6 Newton iterations are sufficient to solve the free boundary problem up to a high precision. During the optimization a good initial guess from the previous evaluation can be used, in which case even less iterations are usually needed.

The exact sensitivity analysis is then performed to obtain the gradient of the cost functional in question. Finally, the ANSI-C version of a gradient based optimizer called Donlp2 [16] is used to solve the optimization problem.

6.1 Example 1

We start with an example whose solution is known. Let $\omega = B_1(0)$ and $\Omega = B_C(0)$, where $C \approx 1.76322$ is such that $C \ln C = 1$. The function

$$f = -C \ln(r) + 1,$$

satisfies $\Delta f = 0$ in $\Omega \setminus \overline{\omega}$, $f = 1$ on $\partial\omega$ and $f = 0$, $\nabla f \cdot \boldsymbol{n} = -1$ on $\partial\Omega$.

Therefore, setting $\gamma = -1$ in (10) and the target boundary in (17) to be $\widehat{R}(\theta) \equiv C$, we see that the circle of radius 1 is the global minimizer of the cost functional $\mathcal{J}_1$. The same holds true for $\mathcal{J}_2$ if we set $z_d = f$ in (18).

The inner boundary was parameterized using 12 control points with pseudo-randomly generated initial values between 0.6 and 1.4. The resulting initial inner boundary $\hat{\Gamma}_i$ was unsymmetric and not very close to the optimum as it can be seen from Figure 3. Further $\hat{\Gamma}_e$ was set to be a circle of radius C. Notice that with these boundaries the initial domain does not satisfy the Bernoulli problem. The outer boundary $\Gamma_e(\Xi_{\boldsymbol{v}_h})$ representing the free boundary and corresponding to this initial inner boundary is found using the pseudo-solid approach, and can be seen in Figure 4.

Optimization was performed using both cost functionals $\mathcal{J}_1$ and $\mathcal{J}_2$ starting from the same initial guess. A very low value 8.79×10^{-9} was reached after 34 optimization steps for $\mathcal{J}_1$. The resulting outer boundary and the target boundary practically coincide (see Figure 3). The final values of the design variables were as expected: indeed, $|\alpha_i - 1.0| < 2 \times 10^{-3}\ \forall i$.

The value 3.17×10^{-7} was obtained for the cost functional $\mathcal{J}_2$ after 24 iterations. Values of the design variables were again as expected: $|\alpha_i - 1.0| < 7 \times 10^{-4}\ \forall i$. The final inner and outer boundaries can be seen in Figure 4.

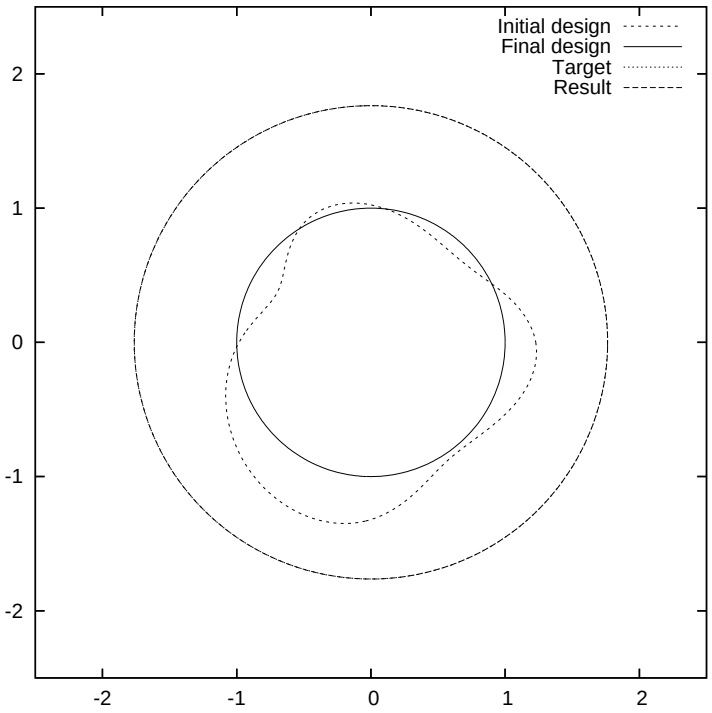

Figure 3: Initial and final designs, final free boundary and the target boundary in Example 1 using the cost functional $\mathcal{J}_1$.

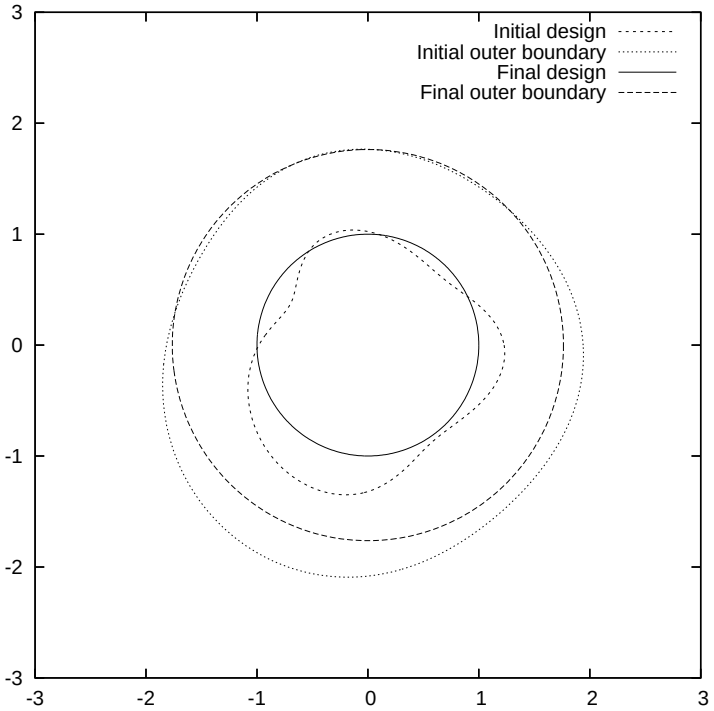

Figure 4: Initial and final inner and outer boundaries in Example 1 using the cost functional $\mathcal{J}_2$.

6.2 Example 2

In this example we minimize the cost functional $\mathcal{J}_1$ with the target boundary $\widehat{R}$ being a "square" with rounded corners (see Figure 5). Width and height of the square are 4. Each corner is rounded using a quarter of a circle of radius 1. For the magnitude of the normal derivative a value $\gamma = -1$ was used.

The optimization was performed using different numbers of design variables. The initial design for $\hat{\Gamma}_i$ was $\alpha_i = 1 \; \forall \; i$, and the initial outer boundary $\hat{\Gamma}_e$ was a circle of radius C centered at the origin in all cases. We specified box constraints: $0.2 \leq \alpha_i \leq 2.8 \; \forall \; i$.

As expected, the increasing number of design variables enabled us to obtain a better approximation of the free boundary $\Gamma_e(\Xi_{v_h})$ (see Table 1). However, the target boundary $\widehat{R}$ was not reached exactly in any of the optimizations. The optimum designs are shown in Figures 5, 6, and 7.

N	cost	iterations	mesh regenerations
16	3.67×10^{-3}	19	1
24	9.07×10^{-4}	178	7
32	5.18×10^{-4}	182	7
40	3.89×10^{-4}	312	9

Table 1: Cost functional value, number of optimization iterations, and number of mesh regenerations as a function of the number of design variables in Example 2.

In all cases some of the design variables attained the lower bound. Also the fact that $\widehat{R}$ is not of the class C^∞ indicates that the target boundary may not be reachable using star shaped inner boundaries (see Theorem 1). Indeed, no clear convergence to any specific shape can be seen for the increasing number of the design variables. On the contrary, the optimal boundaries more and more oscillate.

In this example the optimization problem seems to be ill-conditioned. One reason for the ill-conditioning can be seen in Figure 7, showing the inner boundary at different optimization iterations. Comparing with the convergence history (Figure 8) we see that quite significant changes of the inner boundary may have a very little effect on the value of the cost functional. The resulting free boundaries are not shown in Figure 7, because they practically coincide.

In practical applications one often has some restrictions on the admissible designs. To study the effect of geometrical bounds on the optimal geometry we specified the bounds on the second derivative of the radius R:

$$-c \leq \frac{(\alpha_{i+2} - 2\alpha_{i+1} + \alpha_i)}{\delta^2} \leq c, \tag{28}$$

where the indices are taken modulo N. Here δ is the distance between two consecutive design variables, namely $\delta = 2\pi/N$, and the parameter c equals 5. We used 40 and 80 design variables. With 40 variables the optimizer needed 15 iterations to

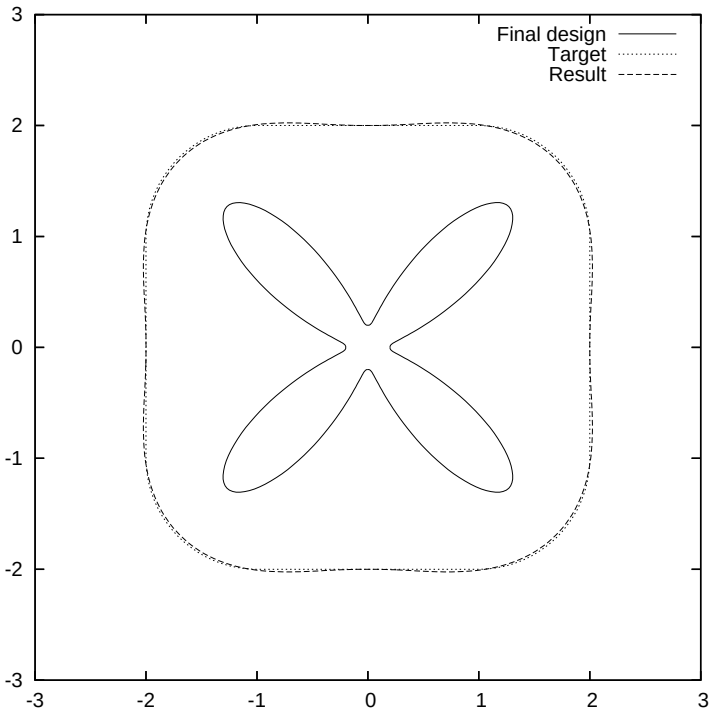

Figure 5: Example 2 with 16 design variables.

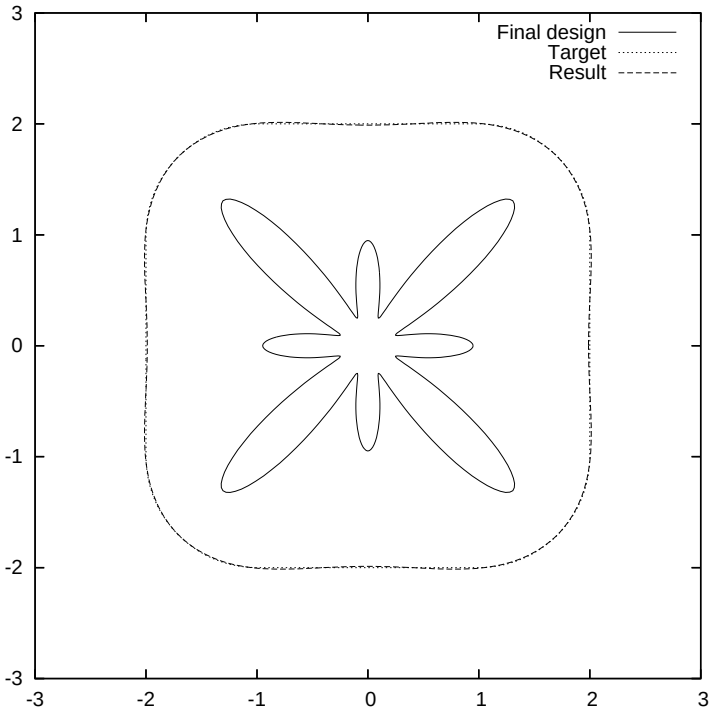

Figure 6: Example 2 with 24 design variables.

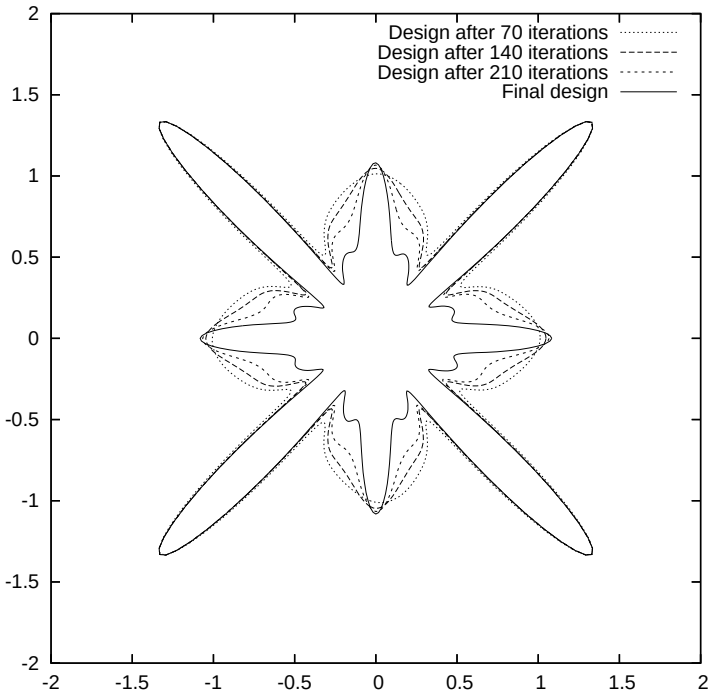

Figure 7: Example 2 with 40 design variables. Inner boundary at different stages of optimization.

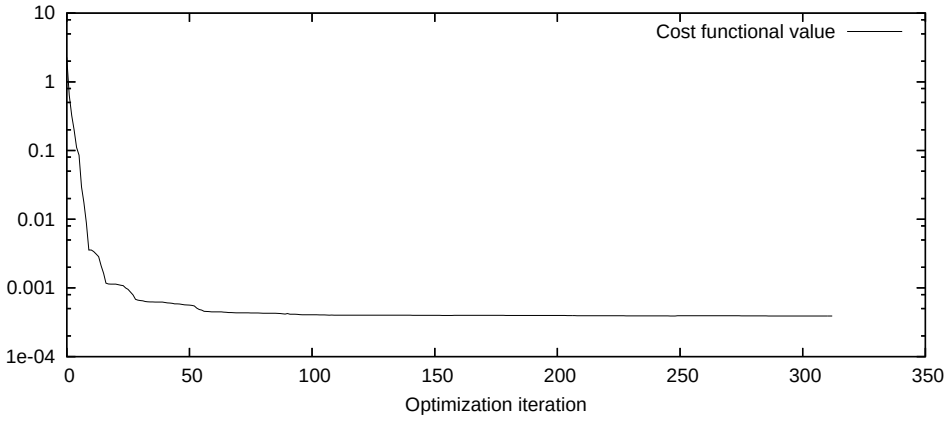

Figure 8: Example 2 with 40 design variables. Convergence history.

reduce the value of $\mathcal{J}_1$ to 6.92×10^{-2}. No mesh regenerations were needed. When 80 variables were used, the optimization was done in 18 steps, and up to three digits the same value of the cost functional was obtained. In this case, one mesh regeneration was performed. As seen from Figure 9, the designs are so alike that the curves representing the inner component of the boundary practically coincide. The effect of (28) on the final result is seen from the Figure 9: the inner boundary is indeed quite smooth, but the resulting free boundary is only a rough match of the target.

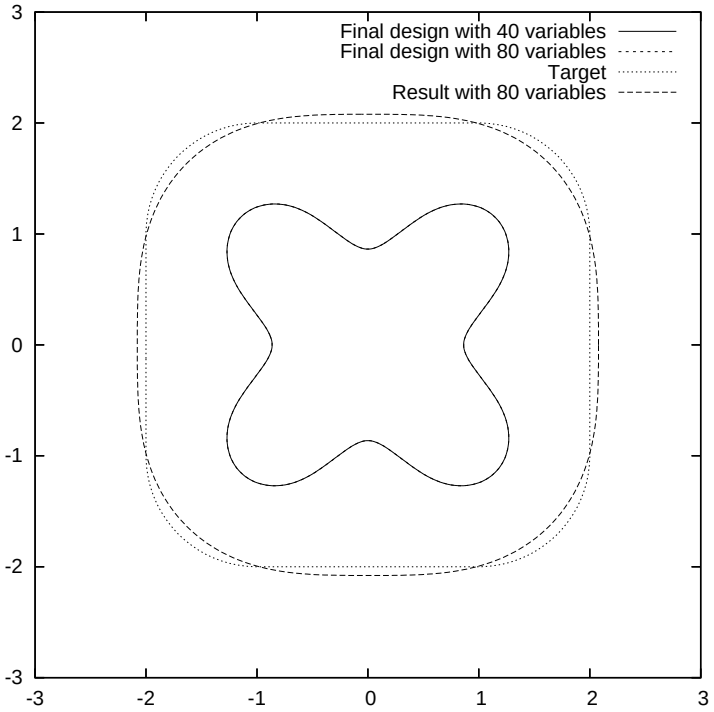

Figure 9: Results of Example 2 with bounds on the second derivative.

In the next two examples we analyze the influence of γ on the final design. It is known that for exterior Bernoulli free boundary problems with Γ_i fixed the respective free boundaries are asymptotic to a family of concentric circles with radii tending to infinity ([6]).

Let the target boundary $\widehat{R}$ in $\mathcal{J}_1$ be chosen as follows:

$$\widehat{R}(\theta) = 0.5\cos(\theta) + 0.8\cos(2\theta) + 2, \quad \theta \in [0, 2\pi[.$$

6.3 Example 3

We choose $\gamma = -3$ and parametrize the inner boundary by 40 design variables. The initial guess was the unit circle for the inner boundary $\hat{\Gamma}_i$ and the circle of radius C for the outer boundary $\hat{\Gamma}_e$. The final cost after 121 optimization iterations and 4 mesh regenerations was 2.90×10^{-7}. The optimum domain is shown in Figure 10.

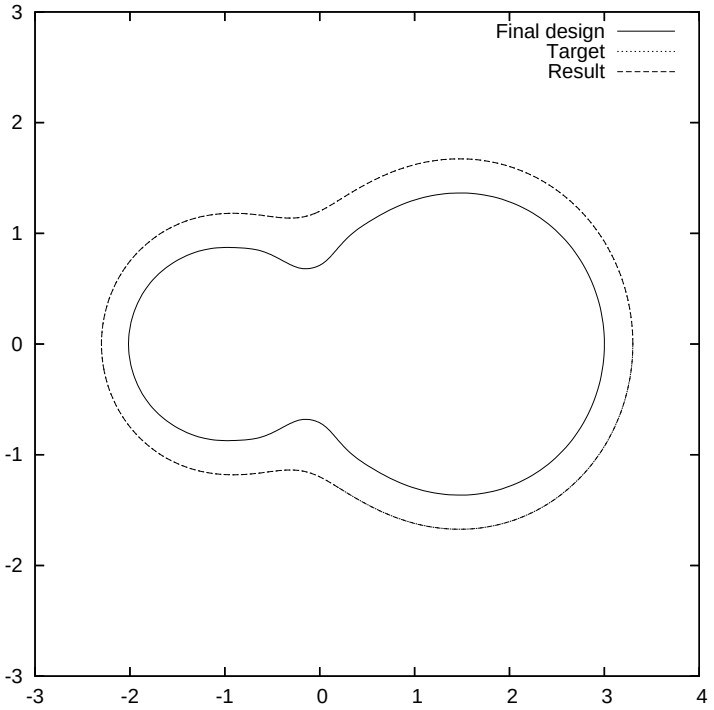

Figure 10: Results of Example 3.

6.4 Example 4

Now we set $\gamma = -2$. The inner boundary was now parametrized by 20 design variables. Our aim was to check whether or not the inner boundary intersects itself for the higher value of γ. Thus no lower bound was imposed on the radial coordinates α_i describing the inner component of the boundary.

The inner boundary really intersects itself. After 61 optimization steps and two mesh regenerations the value of the cost was reduced to 2.01×10^{-4}. At this stage the topology of the domain became "defective" and the mesh regeneration failed, so that the optimization could not continue. The final domain is shown in Figure 11.

This led us to a conclusion that the inner boundary consists of more than one component. For this reason we used two holes as an initial approximation of the inner boundary, each parameterized by the radial co-ordinates as previously. We started with 20 design variables (10 for each hole). The initial guess for the inner boundary $\hat{\Gamma}_i$ was constructed manually using two curves which are close to the inner boundary found in the previous example, as seen in Figure 12. The initial guess for the outer boundary $\hat{\Gamma}_e$ was also taken to be the one found in the previous example. After 41 additional optimization iterations the cost was reduced to 2.27×10^{-5}. The final domain is shown in Figure 12.

Finally, we repeated the previous computation using 40 design variables. Now the problem turns out to be ill-conditioned. After 425 optimization steps and 8 mesh regenerations the cost was reduced to 2.46×10^{-6}. Oscillations are visible and the final domain is shown in Figure 13.

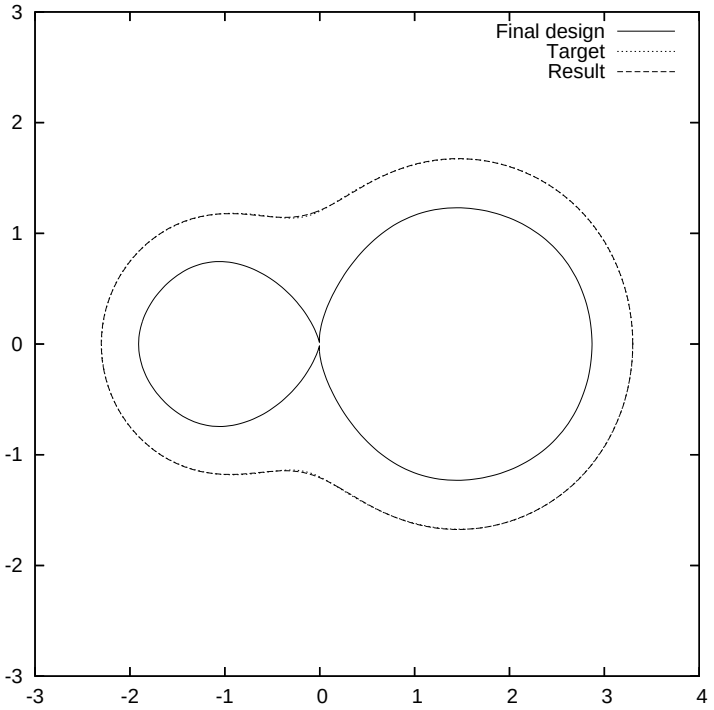

Figure 11: The obtained defective geometry in Example 4.

The disconnected character of the inner boundary can be explained by the fact that the value of γ is high whereas the target $\widehat{R}$ is "tight". When instead of $\widehat{R}$ we take $2\widehat{R}$ the optimal shape of ω is simply connected again. We started with 20 design variables and the initial guess $\alpha_i = 2\ \forall i$. After 28 optimization iterations and one mesh regeneration the cost was reduced to 1.19×10^{-5}. The final domain is shown in Figure 14.

6.5 Example 5

Finally we run the optimization using the cost functional $\mathcal{J}_2$ with

$$z_d = \min\left\{10, \left(x^4 + y^4\right)^{-\frac{1}{4}}\right\},$$

$\gamma = -3$ and 40 design variables. The initial guess for the inner boundary $\hat{\Gamma}_i$ was again a circle of radius 1. The optimizer needed 20 iterations and ended up in a design shown in Figure 15. One mesh regeneration was needed. The final value of $\mathcal{J}_2$ was 0.289.

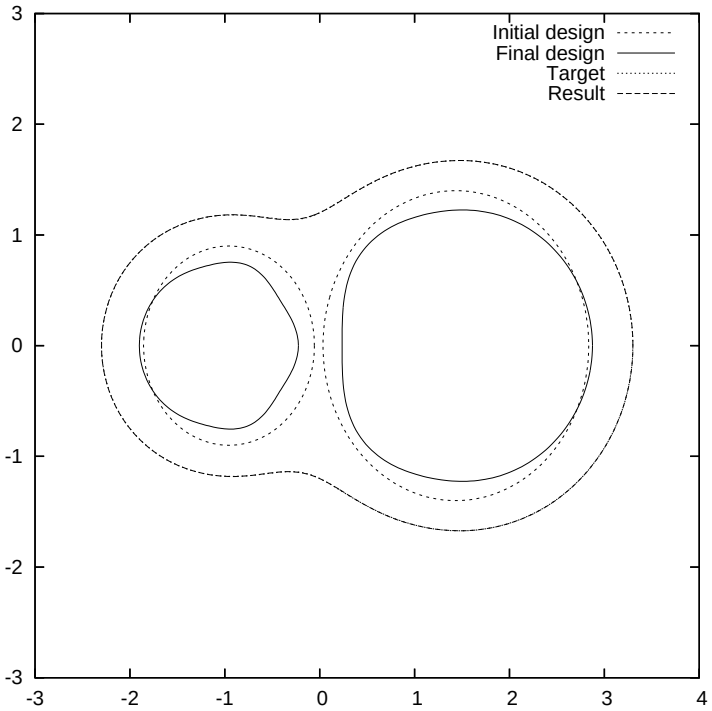

Figure 12: Example 4, two holes parameterized by 20 design variables.

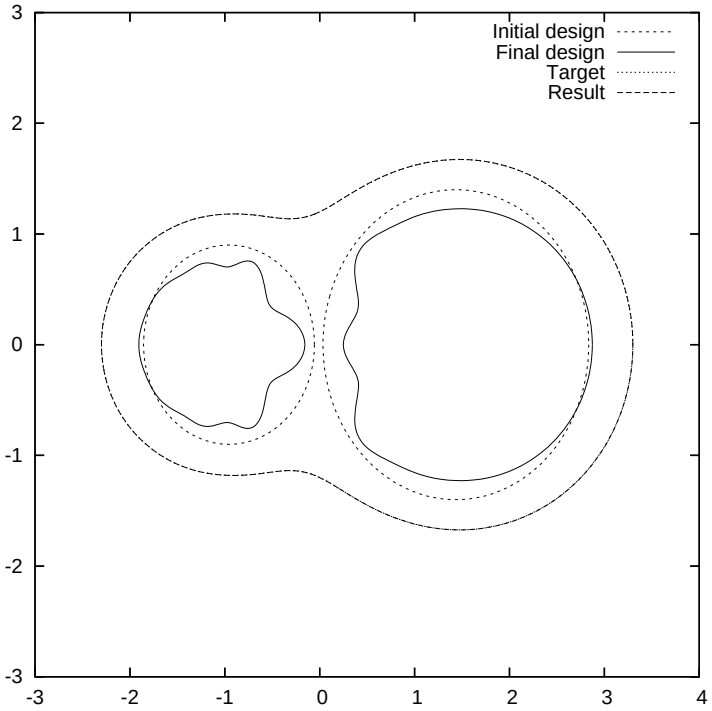

Figure 13: Example 4, two holes parameterized by 40 design variables

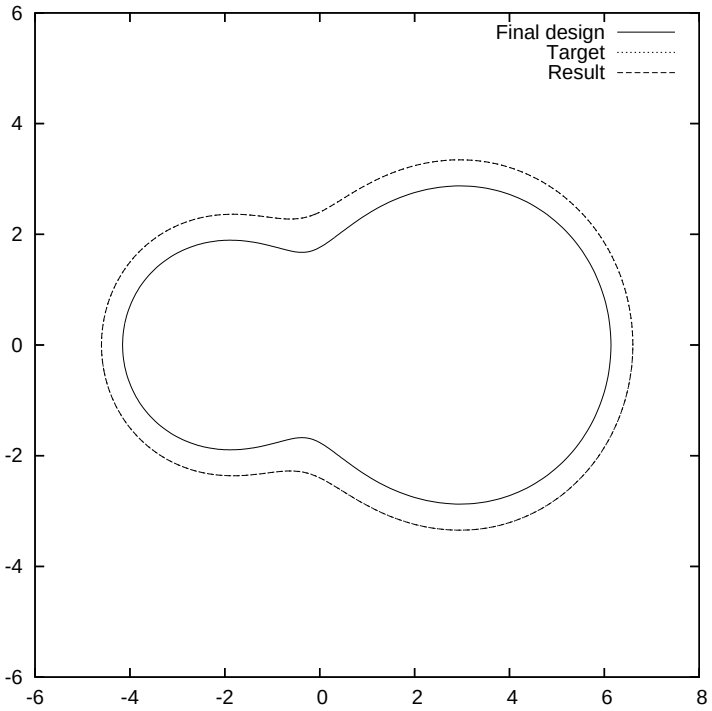

Figure 14: Example 4, $\gamma = -2$, one hole parameterized by 20 design variables

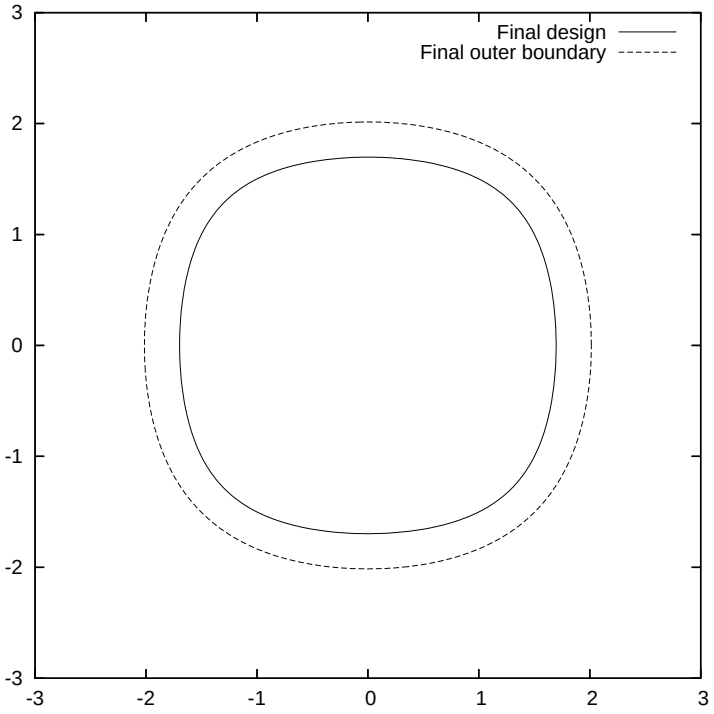

Figure 15: The final domain in Example 5.

7 Conclusions

A new numerical method for shape optimization of systems governed by external Bernoulli free boundary problems is introduced. Numerical results show that the method is efficient and reliable.

The discrete free boundary problems are solved by the so-called pseudo-solid approach. This solution strategy readily allows us to obtain geometrical sensitivities of the system, which can then be used to solve our inverse design problems. The method relies heavily on automatic differentiation, which is used to compute the Jacobian matrix of the coupled non-linear system and also other required partial derivatives.

Numerical examples show that the location of the external free boundary can be controlled by changing the shape of the inner component of the boundary. However, the optimization problem is often ill-conditioned in the sens that relative large changes of the inner boundary have only a little effect on the location of the free boundary. Without additional constraints the inner boundaries often tend to become oscillatory during the optimization.

References

[1] A. Acker and R. Mayer. A free boundary problem for the p-laplacian. *Electronic Journal of Differential Equations*, 1995(08):1–20, 1995.

[2] R.E. Bank and J. Xu. An algorithm for coarsening unstructured meshes. *Numerische Mathematik*, 73(1):1–36, 1996.

[3] R. A. Cairncross, P. R. Schunk, T. A. Baer, R. R. Rao, and P. A. Sackinger. A finite element method for free surface flows of incompressible fluids in three dimensions. Part I. Boundary fitted mesh motion. *International Journal for Numerical Methods in Fluids*, 33:375–403, 2000.

[4] 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(3):355–423, 1990.

[5] James W. Demmel, Stanley C. Eisenstat, John R. Gilbert, Xiaoye S. Li, and Joseph W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Analysis and Applications*, 20(3):720–755, 1999.

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

[7] Michael B. Giles and Niles A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65:393–415, 2000.

[8] A. Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, Philadelphia, 2000.

[9] J. Haslinger and R. A. E. Mäkinen. *Introduction to Shape Optimization: Theory, Approximation, and Computation*. SIAM, Philadelphia, 2003.

[10] L. Holzleitner. Hausdorff convergence of domains and their boundaries for shape optimal design. *Control and Cybernetics*, 30(1):23–44, 2001.

[11] K. Kärkkäinen and T. Tiihonen. Free surfaces: shape sensitivity analysis and numerical methods. *International Journal for Numerical Methods in Engineering*, 44(8):1079–1098, 1999.

[12] G. Mejak. Numerical solution of Bernoulli-type free boundary value problems by variable domain method. *International Journal for Numerical Methods in Engineering*, 37:4219–4245, 1994.

[13] P. A. Sackinger, P. R. Schunk, and R. R. Rao. A Newton-Raphson pseudo-solid domain mapping technique for free and moving boundary problems: A finite element implementation. *Journal of Computational Physics*, 125(1):83–103, 1996.

[14] J. Schöberl. Netgen. Sotfware available at http://www.hpfem.jku.at/netgen/.

[15] M. Souli and J. P. Zolesio. Arbitrary Lagrangian-Eulerian and free surface methods in fluid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 191:451–466, 2001.

[16] P. Spellucci. An SQP method for general nonlinear programs using only equality constrained subproblems. *Mathematical Programming*, 82:413–448, 1998. Software available at http://plato.la.asu.edu/donlp2.html.