Finite Element Methods for Shape Optimization and Applications

Gunay Doğan[a], Pedro Morin[b], Ricardo H. Nochetto[a] and Marco Verani[c]

[a] Department of Mathematics, University of Maryland, College Park, USA.
[b] Departamento de Matemática, Facultad de Ingeniería Química and Instituto de Matemática Aplicada del Litoral, Universidad Nacional del Litoral, CONICET. Santa Fe, ARGENTINA.
[c] MOX, Dipartimento di Matematica, Politecnico di Milano, Milano, ITALY.

Abstract

We present a variational framework for shape optimization problems that establishes and clarifies explicit connections among the continuous formulation, its full discretization and the resulting linear algebraic systems. Our approach hinges on the following essential features: shape differential calculus, a semi-implicit time discretization and a finite element method for space discretization. We use shape differential calculus to express variations of bulk and surface energies with respect to domain changes. The semi-implicit time discretization allows us to track the domain boundary without an explicit parametrization, and has the flexibility to choose different descent directions by varying the scalar product used for the computation of normal velocity. We propose a Schur complement approach to solve the resulting linear systems efficiently. We discuss applications of this framework to image segmentation, optimal shape design for PDE, and surface diffusion, along with the choice of suitable scalar products in each case. We illustrate the method with several numerical experiments, some developing pinch-off and topological changes in finite time.

Keywords: Shape optimization, scalar product, gradient flow, implicit discretization, finite elements, surface diffusion, image segmentation.

AMS subject classification: 49Q10, 65M60, 65K10

1 Shape Optimization and Gradient Flows

Shape optimization problems are ubiquitous in science, engineering and industrial applications. They can be formulated as minimization problems with respect to the shape of a domain Ω in $\mathbb{R}^d$. If $y(\Omega)$ is the solution of a boundary value problem in Ω

$$Ly(\Omega) = 0, \tag{1}$$

and $J(\Omega, y(\Omega))$ is a cost functional, then we consider the minimization problem

$$\Omega^* \in \mathcal{U}_{ad} : \qquad J(\Omega^*, y((\Omega^*)) = \inf_{\Omega \in \mathcal{U}_{ad}} J(\Omega, y(\Omega)), \tag{2}$$

where $\mathcal{U}_{ad}$ is the set of admissible domains in $\mathbb{R}^d$. If the problem is purely geometric, namely there is no state constraint (1), then we simply denote the functional $J(\Omega)$. We refer to the books [24, 19, 15, 11, 21] for a description of this problem and numerous applied examples (see e.g [20, 16]). In any case, we review some basic material in §2 and discuss three relevant examples throughout the paper.

Our main goal is to present a variational method which explicitly and clearly leads to first design a flow $\Omega(t)$, starting from an initial configuration $\Omega(0)$ to a relative minimum $\Omega(\infty)$, that decreases the function $t \mapsto J(\Omega(t), y(\Omega(t))$, and next to discretize in time and space, thereby obtaining a natural descent direction. Our approach hinges on three essential features:

- *Shape sensitivity analysis*: this allows us to express variations of bulk and surface energies with respect to domain changes and formalize the notion of shape derivative and thus shape gradient.

- *Semi-implicit time discretization*: this is crucial in order to maintain an implicit computation of geometric quantities such as mean curvature and normal velocity but not the entire geometry. This can be realized without explicit parametrization of the domain boundary, and is sufficiently flexible to accommodate several scalar products for the computation of normal velocity depending on the application.

- *Adaptive finite element method for space discretization*: this is important for the intrinsic computation of mean curvature as well as the control of local meshsize to increase resolution.

We discuss shape sensitivity in §2, with special emphasis on our three sample problems, and present the time and space discretization of the resulting gradient flows in §3. We finally conclude in §4 with several numerical experiments that illustrate performance of the method, choice of scalar products, and large domain deformations sometimes leading to pinch-off and topological change.

In the rest of the introduction we briefly describe our three basic model problems and the notion of gradient flow. We now introduce our examples, image segmentation, optimal shape design for PDE, and surface diffusion. They are simple models of shape optimization with quite distinct behavior and requirements, which can nonetheless be studied within a unified framework. We make also explicit the concept of shape derivative of $J(\Omega)$ in the direction of a normal velocity V, namely

$$dJ(\Omega; V) = \int_\Gamma GV dS, \tag{3}$$

but derive the expressions of G in §2 for each case. We then indicate how to exploit this information to design a gradient flow. Throughout the paper we will denote with Γ that part of the boundary of Ω which is free to deform, with κ the sum of the principal curvatures and with $\vec{\nu}$ the unit outer normal of Γ; thus $V := \vec{V} \cdot \vec{\nu}$. We assume that circle with outward normal has positive mean curvature.

1.1 Image Segmentation

Image segmentation has been one of the central problems of image processing ever since the inception of this discipline. Given an image the goal is to identify the "objects" or homogeneous regions with respect to some image features, such as image intensity, texture etc. The *geodesic active contour model* proposed in [9] addresses this problem in an energy minimization context and identifies object boundaries by a set of curves (or surfaces in 3D). In the following we cast this model within our framework.

Let $I(x) : \mathcal{D} \subset \mathbb{R}^d \to \mathbb{R}$ be a given smoothed-out image intensity function on an open and bounded image domain $\mathcal{D}$. Since values of $I(x)$ vary significantly at object boundaries, the image gradient $\nabla I(x)$ gets large at these locations. We can use this to define the edge detector function $H(x)$ as follows

$$H(x) = h(|\nabla I(x)|), \qquad h(s) = \frac{1}{1+s^2},$$

so that $H(x)$ is small on object boundaries and $H(x) \approx 1$ on smooth regions of the image. We now associate an energy $J(\Omega)$ to a given curve Γ and enclosed domain Ω so that object boundaries correspond to local minima of $J(\Omega)$. Such an energy is given by the geometric functional

$$J(\Omega) := \int_\Gamma H(x) dS + \lambda \int_\Omega H(x) dx, \qquad \lambda \geq 0. \tag{4}$$

Note that the first integral is minimized when Γ coincides with the object boundaries in the image. It is also common to include the domain integral in the optimization process because it speeds up the convergence of the curve to the object boundaries and helps detection of object concavities; see [5], [9] for more details. We will see in §2.2.1 that G in (3) has the explicit form

$$G = (\kappa + \lambda) H(x) + \partial_\nu H(x). \tag{5}$$

1.2 Optimal Shape Design for PDE

Motivated by the optimal shape design of a drug eluting stent, we consider an extremely simplified problem, still presenting some of the main mathematical difficulties one has to face in trying to solve a more realistic situation (for more details on the mathematical modelling see e.g. [27]). A drug eluting stent

is a normal metal stent that has been coated with a drug that is known to interfere with the process of restenosis of the artery. Roughly speaking in optimal shape design for drug eluting stents, one is interested in optimizing some of the geometric properties of the stent in order to control the distribution of the drug released in the arterial wall.

In this context Ω is the cross-section of part of the arterial wall, Γ is the boundary of the cross section of a stent wire, D is the control region for the drug distribution and z_g is some clinical data to match.

Let Ω_i, $i = 1, 2$ be sufficiently regular open bounded sets of $\mathbb{R}^d$, such that $\Omega_1 \subset \Omega_2$. We denote by $\Omega = \Omega_2 \setminus \overline{\Omega}_1$ and $\partial\Omega = \Gamma \cup \Sigma$, where $\Gamma = \partial\Omega_1$ and $\Sigma = \partial\Omega_2 \setminus \Gamma$. Finally, let D be an open bounded set of $\mathbb{R}^d$ such that $D \subset\subset \Omega$. Let us now define the set $\mathcal{U}_{ad}$ of admissible domains in $\mathbb{R}^d$: it contains all domains obtained through a deformation of Ω by keeping Σ fixed and by moving only Γ in such a way that $\Gamma \cap D = \emptyset$.

We are interested in the solution of a simple shape optimization problem of the form (2), associated to the energy functional

$$J(\Omega, y(\Omega)) := \frac{1}{2}\int_D \Big(y(\Omega) - z_g\Big)^2 dx + \gamma \int_\Gamma dS, \tag{6}$$

where $\gamma > 0$ is a penalization parameter for the length of the moving surface Γ, z_g is a given function on $\mathbb{R}^d$ and $y(\Omega)$ is the solution of the following elliptic problem on Ω

$$-\Delta y = 0 \quad \text{in } \Omega, \qquad y = 0 \quad \text{on } \Sigma, \qquad \partial_\nu y = 1 \quad \text{on} \quad \Gamma. \tag{7}$$

We will see in §2.2.2 that G in (3) has the explicit form

$$G := -\nabla_\Gamma y \nabla_\Gamma p + \kappa p + \kappa\gamma. \tag{8}$$

1.3 Surface Diffusion and Epitaxially Stressed Solids

A very simple model of epitaxially stressed thin films can be described as follows [4, 8, 25]. Consider an elastic solid with lattice spacing different from that of a substrate. This mismatch induces stresses in the solid. On the other hand, material particles on the free surface Γ in contact with gas are free to move and rearrange their position so as to minimize surface tension, thereby yielding a plastic deformation of the solid. Phenomenological arguments lead to the four order (highly nonlinear) PDE

$$V = -\Delta_\Gamma(\gamma\kappa + \epsilon) \tag{9}$$

where γ is the surface tension constant and ϵ is the elastic energy density on Γ. In this paper we consider a simplified situation in that elasticity is replaced by the Laplace operator in Ω and thus $\epsilon := |\nabla y(\Omega)|^2$, where $y(\Omega)$ solves the boundary value problem

$$-\Delta y(\Omega) = 0 \quad \text{in } \Omega, \qquad \partial_\nu y(\Omega) = 0 \quad \text{on } \Gamma, \qquad y(\Omega) = g \quad \text{on } \Sigma, \tag{10}$$

where Σ is that part of the boundary of Ω in contact with the substrate (to mimic a misfit). We will see in §3.3 that the physical law (9) is the H^{-1}-gradient flow for the energy functional

$$J(\Omega, y(\Omega)) := \int_\Omega |\nabla y(\Omega)|^2 + \gamma \int_\Gamma dS, \tag{11}$$

whereas we will show in §2.2.3 that G in (3) has the explicit form

$$G := |\nabla y(\Omega)|^2 + \gamma\kappa. \tag{12}$$

1.4 Shape Gradient Flows

We observe that in all the examples above, the function G in (3) has the form

$$G = g(x, \Omega)\kappa + f(x, \Omega). \tag{13}$$

This explicit expression can be exploited to deform Ω in the direction V of maximal decrease of the functional $J(\Omega, y(\Omega))$. To do this, we first introduce a bilinear form $b(\cdot, \cdot)$ on Γ which induces a scalar product, and next consider the gradient flow

$$b(V, W) = -\int_\Gamma GW, \qquad \forall W, \tag{14}$$

where Γ (and hence G) implicitly depend on V by means of a suitable system of ODE describing the deformation of Ω through V. If $\mathcal{B}$ is a (elliptic) operator such that $\langle \mathcal{B}V, W\rangle = b(V, W)$, then (14) is equivalent to solving the elliptic PDE on the surface Γ for the normal velocity V

$$\mathcal{B}V = -G. \tag{15}$$

We point out that so far we have not discretized the underlying problem but still have been able to find a descent direction for the domain shape. The next step is to discretize in time in such a manner that we retain the implicit computation of curvature in (13), for stability purposes, but not the full geometry. This linearization process is fully discussed in §3.3 and is followed by space discretization via finite element methods in §3.5. The ensuing variational approach is rather flexible to accommodate several scalar products $b(\cdot, \cdot)$ depending on the application, as discussed in §3.3 and §4. Roughly speaking we can distinguish between applications where the gradient flow has a physical meaning (e.g. surface diffusion), and where it does not (e.g. image segmentation or optimal shape design for PDE). In the first case the choice of the scalar product is dictated by physics, whereas in the latter case it can be driven by issues concerning the well-posedness of (15), as discussed in the example of optimal shape design for PDE, or by stability and rate of convergence of the resulting numerical scheme, as described in the example of image segmentation.

In designing a numerical scheme (e.g. gradient method) for the approximation of the continuous gradient flow (14), and hence the construction of a sequence of domains $\{\Omega_n\}_{n\geq 0}$ aiming at convergence to $\Omega^* = \text{argmin}_{\Omega\in\mathcal{U}_{ad}} J(\Omega)$, the following chief question arises:

$$\begin{array}{l} \textit{Given a domain } \Omega\textit{, is it possible to choose a vector field} \\ \vec{V} \textit{ deforming } \Omega \textit{ into } \tilde{\Omega} \textit{ such that } J(\tilde{\Omega}) < J(\Omega)\textit{?} \end{array} \tag{16}$$

A key step in answering this question is the shape sensitivity analysis of the mapping $\Omega \mapsto J(\Omega)$. This is briefly discussed in §2.

2 Shape Sensitivity Analysis

In §2.1 we introduce some elements of shape calculus, along with related references, necessary to properly carry out the shape sensitivity of the model problems in §2.2.

2.1 Shape Differential Calculus

We start by briefly recalling some useful notions of differential geometry. Let us be given $h \in C^2(\Gamma)$ and an extension $\tilde{h}$ of h, $\tilde{h} \in C^2(U)$ and $\tilde{h}|_\Gamma = h$ on Γ where U is a tubular neighborhood of Γ in $\mathbb{R}^d$. Then the *tangential gradient* $\nabla_\Gamma h$ of h is defined as follows:

$$\nabla_\Gamma h = (\nabla\tilde{h} \ - \ \frac{\partial\tilde{h}}{\partial\nu}\vec{\nu})|_\Gamma,$$

where $\vec{\nu}$ denotes the normal vector to Γ. For an open set of class C^2 with boundary Γ, we define the *tangential divergence* of $\vec{V}$ by

$$\text{div}_\Gamma\vec{V} = (\text{div}\vec{V} - < D\vec{V}\cdot\vec{\nu}, \vec{\nu} >)|_\Gamma, \tag{17}$$

where $D\vec{V}$ denotes the Jacobian matrix of $\vec{V}$.
Finally the *Laplace-Beltrami operator* Δ_Γ on Γ is defined as follows

$$\Delta_\Gamma h = \text{div}_\Gamma(\nabla_\Gamma h). \tag{18}$$

2.1.1 The Velocity Method

We consider now a hold-all $\mathcal{D}$ which contains Ω and a vector field $\vec{V}$ defined on $\mathcal{D}$ which is used to define the continuous sequence of perturbed sets $\{\Omega_t\}_{t\geq 0}$, with $\Omega_0 := \Omega$. Each point $x \in \Omega_0$ is continuously deformed by an ODE defined by the field $\vec{V}$. The parameter which controls the amplitude of the deformation is denoted by t (a fictitious time).

We now consider the system of ODEs

$$\left\{ \begin{array}{rcl} \frac{dx}{dt} & = & \vec{V}(x(t)), \quad \forall t \in I \\ \\ x(0) & = & X. \end{array} \right. \tag{19}$$

where $X \in \Omega_0 = \Omega$. This defines the mapping

$$x(\cdot, t) : X \in \Omega \to x(t, X) \in \mathbb{R}^d, \tag{20}$$

and also the perturbed sets

$$\Omega_t = \{x(t, X) : \quad X \in \Omega_0\}. \tag{21}$$

We recall [24] that the family of perturbed sets has its regularity preserved for $\vec{V}$ smooth enough: if Ω_0 is of class C^r, $r \leq k$, then for any $t \in I$, Ω_t is also of class C^r.

2.1.2 Derivative of Shape Functionals

Let $J(\Omega)$ be a shape functional; examples of such functionals have been given in Section 1. The Eulerian derivative (or Shape derivative) of the functional $J(\Omega)$ at Ω, in the direction of the vector field $\vec{V}$ is defined as the limit

$$dJ(\Omega; \vec{V}) = \lim_{t \to 0} \frac{1}{t}(J(\Omega_t) - J(\Omega)). \tag{22}$$

Let B be a Hilbert space of perturbating vector fields. The functional $J(\Omega)$ is said to be shape differentiable at Ω in B if the Eulerian derivative $dJ(\Omega; \vec{V})$ exists for all $\vec{V} \in B$ and the mapping $\vec{V} \to dJ(\Omega; \vec{V})$ is linear and continuous on B. An analogous definition can be introduced for functionals $J(\Gamma)$ depending on a $d-1$ manifold as an independent variable.

We now recall a series of results from shape calculus in $\mathbb{R}^d$. We start with the shape derivative of domain and boundary integrals of functions not depending on the geometry.

Lemma 2.1 ([24, Prop. 2.45]) *Suppose $\phi \in W^{1,1}(\mathbb{R}^d)$ and $\Omega \subset \mathbb{R}^d$ is open and bounded. Then the functional*

$$J(\Omega) = \int_\Omega \phi dx \tag{23}$$

is shape differentiable. The shape derivative of J is given by

$$dJ(\Omega; \vec{V}) = \int_\Omega \operatorname{div}(\phi \vec{V}) dx. \tag{24}$$

If $\Gamma = \partial\Omega$ is of class C^1, then

$$dJ(\Omega; \vec{V}) = \int_\Gamma \phi V ds, \tag{25}$$

where $V = < \vec{V}, \vec{\nu} >$, being $\vec{\nu}$ the outward unit normal vector to Ω and $< \cdot, \cdot >$ the inner product on $\mathbb{R}^d$.

Lemma 2.2 *Suppose $\psi \in W^{2,1}(\mathbb{R}^d)$ and Γ is of class C^1. Then the functional*

$$J(\Gamma) = \int_\Gamma \psi dS \tag{26}$$

is shape differentiable and [24, Prop. 2.50]

$$dJ(\Gamma; \vec{V}) = \int_\Gamma (< \nabla\psi, \vec{V} > + \psi \mathrm{div}_\Gamma \vec{V}) ds, \tag{27}$$

or equivalently [24, (2.145)]

$$dJ(\Gamma; \vec{V}) = \int_\Gamma (\frac{\partial \psi}{\partial \nu} + \psi\kappa) V dS, \tag{28}$$

where κ is the mean curvature of Γ.

Let us now consider more general functionals $J(\Omega)$, which are useful when we consider problems of optimal shape design for partial differential equations, like the one introduced in Section 1.2. In particular we are interested in computing sensitivities for functionals of the form

$$J(\Omega) = \int_\Omega \phi(x, \Omega) dx, \qquad \text{or} \qquad J(\Gamma) = \int_\Gamma \psi(x, \Gamma) ds, \tag{29}$$

where the functions $\phi(\Omega, \cdot) : \Omega \to \mathbb{R}$ and $\psi(\Gamma, \cdot) : \Gamma \to \mathbb{R}$ themselves depend on the geometric variables Ω and Γ, respectively. To handle the computation of the sensitivities of such functionals we need to take care of the derivatives of ϕ and ψ with respect to Ω and Γ.
First of all we recall the notion of **material derivative** $\dot{\phi}(\Omega; \vec{V})$ of ϕ at Ω in direction $\vec{V}$. It is defined [24, Def. 2.71] as follows:

$$\dot{\phi}(\Omega; \vec{V}) = \lim_{t \to 0} \frac{1}{t} (\phi(\Omega_t) \circ x(\cdot, t) - \phi(\Omega_0)) \tag{30}$$

where the mapping $x(\cdot, t)$ is defined as in (20).

The analogous definition holds for functions $\psi(\Gamma, \cdot)$ which are defined on boundaries and not on domains.
Let us now now recall the notion of **shape derivative** $\phi'(\Omega; \vec{V})$ of ϕ at Ω in direction $\vec{V}$. It is defined [24, Def. 2.85] as

$$\phi'(\Omega; \vec{V}) = \dot{\phi}(\Omega; \vec{V}) - < \nabla\phi, \vec{V} > . \tag{31}$$

Accordingly for boundary functions $\psi(\Gamma) : \Gamma \to \mathbb{R}$, the shape derivative is defined [24, Def. 2.88] as

$$\psi'(\Gamma; \vec{V}) = \dot{\psi}(\Gamma; \vec{V}) - < \nabla_\Gamma \psi, \vec{V} > |_\Gamma. \tag{32}$$

With these notions we are able to calculate the Eulerian derivatives for the above shape functionals.

Proposition 2.1 ([24, Sect. 2.31, 2.33]) *Suppose $\phi = \phi(\Omega, x)$ is given such that the material derivative $\dot{\phi}(\Omega; \vec{V})$ and the shape derivative $\phi'(\Omega; \vec{V})$ exist. Then, the cost functional in (29) is shape differentiable and we have*

$$dJ(\Omega; \vec{V}) = \int_\Omega \phi'(\Omega; \vec{V}) dx + \int_\Gamma \phi V dS. \tag{33}$$

For boundary functions $\phi(\Gamma)$ we get

$$dJ(\Gamma; \vec{V}) = \int_\Gamma \phi'(\Gamma; \vec{V}) dS + \int_\Gamma \kappa \phi V dS, \tag{34}$$

where κ is the mean curvature. If $\phi(\Gamma, \cdot) = \psi(\Omega, \cdot)|_\Gamma$, then we have

$$dJ(\Gamma; \vec{V}) = \int_\Gamma \psi'(\Omega; \vec{V})|_\Gamma dS + \int_\Gamma (\frac{\partial \psi}{\partial \nu} + \kappa \psi) V dS. \tag{35}$$

To use this Proposition we need to be able to compute the shape derivative of solutions $y(\Omega)$ to elliptic boundary value problems. We consider now a simple case, though sufficient for our later developments: let f, g, h be functions defined on $\mathbb{R}^d$, i.e. they do not depend on Ω

$$-\Delta y(\Omega) = f \quad \text{in } \Omega, \qquad y(\Omega) = g \quad \text{on } \Sigma, \qquad \partial_\nu y(\Omega) = h \quad \text{on } \Gamma. \tag{36}$$

Lemma 2.3 ([26],[24, Sect. 3.1 and 3.2]) *The shape derivative $y' := y'(\Omega, \vec{V})$ of $y(\Omega)$ in (36) satisfies the following boundary value problem*

$$\left\{ \begin{array}{rcll} -\Delta y' & = & 0, & \text{in } \Omega \\ y' & = & V \partial_\nu (g - y(\Omega)) & \text{on } \Sigma \\ \partial_\nu y' & = & \mathrm{div}_\Gamma (V \nabla_\Gamma y(\Omega)) + (\kappa h + \partial_\nu h + f) V & \text{on } \Gamma. \end{array} \right. \tag{37}$$

Let us conclude this part with a Riesz representation theorem that will play an important role in the sequel.

Lemma 2.4 (Hadamard-Zolésio Theorem [24, Sect 2.11 and Th. 2.27]) *The Eulerian derivative of a domain or boundary functional always has a representation of the form*

$$dJ(\Omega; \vec{V}) = \langle G, V \rangle_\Gamma, \tag{38}$$

where we denote by $\langle \cdot, \cdot \rangle_\Gamma$ a duality pair on Γ; that is, the Eulerian derivative is concentrated on Γ.

2.2 Shape Derivatives of the Model Problems

For each shape functional introduced in Section 1.2 we compute the shape derivative and derive the explicit expressions (5), (8), and (12) of the Riesz representative G.

2.2.1 Image Segmentation

Following Lemmas 2.1, 2.2, we can write the shape derivative of $J(\Omega)$ as follows:

$$dJ(\Omega;V) = \int_\Gamma \Big((\kappa+\lambda)H(x) + \frac{\partial H(x)}{\partial \nu}\Big) V dS. \tag{39}$$

Then the shape gradient is

$$G = (\kappa+\lambda)H(x) + \frac{\partial H(x)}{\partial \nu}, \tag{40}$$

and has the form (13) with $g = H(x)$ and $f = \lambda H(x) + \frac{\partial H(x)}{\partial \nu}$.

2.2.2 Optimal Shape Design for PDE

Let us now compute the shape derivative of the functional

$$J(\Omega, y(\Omega)) = J_1(\Omega, y(\Omega)) + J_2(\Omega) = \frac{1}{2}\int_D (y(\Omega) - z_g)^2 dx + \gamma \int_\Gamma d\Gamma, \tag{41}$$

where $y(\Omega)$ solves the elliptic problem (7).

Let us first consider the shape derivative $y' := y'(\Omega; \vec{V})$ at Ω in the direction $\vec{V}$, where we allow $\vec{V}$ to be non-zero only in a neighborhood of Γ (i.e. D and Σ are both assumed to be fixed). According to Lemma 2.3, y' is the solution of the following elliptic problem

$$\left\{ \begin{array}{rcll} -\Delta y' &=& 0 & \text{in } \Omega \\ y' &=& -V\partial_\nu y = 0 & \text{on } \Sigma \\ \partial_\nu y' &=& \mathrm{div}_\Gamma(V\nabla_\Gamma y) + \kappa V & \text{on } \Gamma. \end{array} \right. \tag{42}$$

In order to relate the L^2-norm in $J_1(\Omega, y(\Omega))$ it is natural to introduce the following adjoint problem

$$\left\{ \begin{array}{rcll} -\Delta p &=& \chi_D(y - z_g) & \text{in } \Omega \\ p &=& 0 & \text{on } \Sigma \\ \partial_\nu p &=& 0 & \text{in } \Gamma, \end{array} \right. \tag{43}$$

whence

$$\begin{aligned} dJ_1(\Omega, V) &= \int_D (y - z_g)y' = -\int_\Omega \Delta p y' \\ &= \int_\Omega \nabla p \nabla y' - \int_\Gamma \partial_\nu p y' - \int_\Sigma \partial_\nu p y' \\ (y' = 0 \text{ on } \Sigma, \partial_\nu p = 0 \text{ on } \Gamma) &= -\int_\Omega \Delta y' p + \int_\Gamma \partial_\nu y' p + \int_\Sigma \partial_\nu y' p \\ (p = 0 \text{ on } \Sigma) &= -\int_\Omega \Delta y' p + \int_\Gamma \partial_\nu y' p \\ &= \int_\Gamma (\mathrm{div}_\Gamma(V\nabla_\Gamma y) + \kappa V) p \\ &= \int_\Gamma (-\nabla_\Gamma y \nabla_\Gamma p + \kappa p) V. \end{aligned}$$

By using Lemma 2.2 we have

$$dJ_2(\Omega, V) = \gamma \int_\Gamma kV d\Gamma. \tag{44}$$

Hence the Structure Theorem (see Lemma 2.4) holds with the Riesz representation function

$$G = -\nabla_\Gamma y \nabla_\Gamma p + \kappa p + \kappa\gamma, \tag{45}$$

which has the form (13) with $g = p + \gamma$ and $f = -\nabla_\Gamma y \nabla_\Gamma p$.

2.2.3 Surface Diffusion and Epitaxially Stressed Solids

We now compute the shape derivative of the functional (11), namely,

$$J(\Omega, y(\Omega)) = J_1(\Omega, y(\Omega)) + J_2(\Omega) = \int_\Omega |\nabla y(\Omega)|^2 + \gamma \int_\Gamma dS, \tag{46}$$

with $y(\Omega)$ satisfying (10). It follows from (37) that the shape derivative $y' := y'(\Omega; \vec{V})$ of $y(\Omega)$ satisfies

$$-\Delta y' = 0 \quad \text{in } \Omega, \qquad y' = 0 \quad \text{on } \Sigma, \qquad \partial_\nu y' = \mathrm{div}_\Gamma(V \nabla_\Gamma y), \quad \text{on } \Gamma. \tag{47}$$

Consequently, using (33), we obtain

$$dJ_1(\Omega; \vec{V}) = 2 \int_\Omega \nabla y \nabla y' + \int_\Gamma |\nabla y|^2 V$$

and

$$\int_\Omega \nabla y \nabla y' = -\langle \Delta y, y' \rangle + \int_\Gamma y' \partial_\nu y = 0,$$

because of (10). Since the shape derivative for $J_2(\Omega)$ obeys (44), we have thus derived the expression

$$dJ(\Omega; \vec{V}) = \int_\Gamma \Big(|\nabla y|^2 + \gamma\kappa \Big) V ds.$$

This implies that the shape gradient G is

$$G = |\nabla y|^2 + \gamma\kappa,$$

and has the form (13) with $g = \gamma$ and $f = |\nabla y|^2$.

3 Discrete Gradient Flows

Now we are ready to answer the chief question (16) and provide a strategy to build a sequence $\{\Omega_n\}_{n \geq 0}$ such that $J(\Omega_{n+1}) \leq J(\Omega_n)$. We first consider in §3.1 an implicit time discretization, as proposed by Luckhaus [18] and Almgren,

Taylor and Wang [1] for the Stefan problem with Gibbs-Thomson law. This technique is also related to the idea of minimizing movements introduced by E. De Giorgi [2, 3].

Since the resulting scheme is not practical, we then study an explicit time linearization in §3.2, followed in §3.3 by a semi-implicit time discretization which keeps geometric quantities such as velocity and curvature implicit but the rest of the geometry explicit.

3.1 Implicit Time Discretization

Let τ_n be a variable time-step, $t_{n+1} = t_n + \tau_n$ and let Ω_{n+1} be the solution of the following penalized minimization problem:

$$\text{argmin}_{\Omega_{n+1}} \Big(J(\Omega_{n+1}) + \frac{1}{2\tau_n} d^2(\Omega_{n+1}, \Omega_n) \Big), \tag{48}$$

where the "distance term" $\frac{1}{2\tau_n} d^2(\Omega_{n+1}, \Omega_n)$ penalizes the distance between Ω_{n+1} and Ω_n. In order to specify the distance function $d(\cdot, \cdot)$, we consider $\vec{V}_{n+1} := \vec{V}(\vec{X}_{n+1})$ to be the *implicit* Euler approximation of (19):

$$\vec{X}_{n+1} = \vec{X}_n + \tau_n \vec{V}_{n+1}; \tag{49}$$

Note that Ω_{n+1} is described by the set of points $\vec{X}_{n+1}$ and that $\vec{V}_{n+1}$ is defined in Ω_{n+1}, so (49) does not specify $\vec{V}_{n+1}$ directly.

Let $V_{n+1} \in B(\Gamma_{n+1})$, where $(B(\Gamma_{n+1}), b_{n+1}(\cdot, \cdot), \| \cdot \|_{\Gamma_{n+1}})$ is a Hilbert space defined on the deformable part Γ_{n+1} of the boundary of Ω_{n+1}, thereby measuring the (boundary) smoothness of the vector fields. The natural choice

$$d(\Omega_{n+1}, \Omega_n) = \| \tau_n V_{n+1} \|_{B(\Gamma_{n+1})},$$

converts (48) into the following minimization problem for $\vec{V}_{n+1} = V_{n+1} \vec{\nu}_{n+1}$:

$$\text{argmin}_{\vec{V}_{n+1}} \Big(J(\Omega_n + \tau_n \vec{V}_{n+1}) + \frac{1}{2\tau_n} \| \tau_n V_{n+1} \|^2_{B(\Gamma_{n+1})} \Big). \tag{50}$$

The optimality condition reads as follows

$$dJ(\Omega_{n+1}; \tau_n \vec{W}) + \frac{1}{\tau_n} b_{n+1}(\tau_n V_{n+1}, \tau_n W) = 0, \quad \forall W \in B(\Gamma_{n+1}), \tag{51}$$

in terms of the variation $\vec{W} = W \vec{\nu}_{n+1}$ of $\vec{V}_{n+1}$. Such a condition, via Hadamard-Zolésio Lemma 3, is equivalent to

$$\langle G_{n+1}, \tau_n W \rangle_{\Gamma_{n+1}} = -\frac{1}{\tau_n} (\tau_n V_{n+1}, \tau_n W)_{B(\Gamma_{n+1})}, \quad \forall W \in B(\Gamma_{n+1}).$$

This yields the following ideal equation for V_{n+1}

$$b_{n+1}(V_{n+1}, W) = -\langle G_{n+1}, W \rangle_{\Gamma_{n+1}}, \quad \forall W \in B(\Gamma_{n+1}), \tag{52}$$

which is *implicit* in that the domain Ω_{n+1} is unknown and thus part of the problem (see also [7]). However, a crucial consequence of (50) important for theory is

$$J(\Omega_{n+1}) \leq J(\Omega_n + \tau_n \vec{V}_{n+1}) + \frac{\tau_n}{2}\|V_{n+1}\|^2_{B(\Gamma_{n+1})} \leq J(\Omega_n), \tag{53}$$

as results from taking $\vec{V}_{n+1} = 0$ in (50). Consequently,

$$J(\Omega_k) + \frac{1}{2}\sum_{i=1}^{k} \tau_i \|\vec{V}_i\|^2_{B(\Omega_i)} \leq J(\Omega_0), \qquad \forall k \geq 1.$$

Solving the nonlinear problem (52) is unaffordable directly and would require iteration. The following linearization technique may either replace the implicit solve or be used as one step in an iterative process.

3.2 Explicit Linearization

The key idea is to replace in (52) the new domain Ω_{n+1} and its boundary Γ_{n+1}, which are unknown, by the current ones Ω_n and Γ_n. This gives rise to the following elliptic PDE on Γ_n: find $V_{n+1} \in B(\Gamma_n)$ such that

$$b_n(V_{n+1}, W) = -\langle G_n, W\rangle_{\Gamma_n}, \qquad \forall W \in B(\Gamma_n). \tag{54}$$

Then Ω_{n+1} results from the explicit update $\vec{X}_{n+1} = \vec{X}_n + \tau \vec{V}_{n+1}$, but the energy decrease property (53) is no longer valid. Nevertheless equation (54) still provides a weaker energy decrease. In fact, if one chooses $\vec{V}_{n+1} = V_{n+1}\vec{\nu}_n$, being V_{n+1} the solution of

$$b_n(V_{n+1}, W) = -\langle G_n, W\rangle_{\Gamma_n}, \qquad \forall W \in B(\Gamma_n), \tag{55}$$

then there holds that

$$dJ(\Gamma_n; \vec{V}_{n+1}) = \langle G_n, V_{n+1}\rangle_{\Gamma_n} = -b_n(V_{n+1}, V_{n+1}) \leq -\|V_{n+1}\|^2_{B(\Gamma_n)}, \tag{56}$$

that is $\vec{V}_{n+1}$ provides a descent direction for the energy $J(\Omega_n)$ (see [10]). However, (53) may not be valid, thereby leading to the bisection of τ_n (backtracking) until (53) holds. This is guaranteed by (56) which expresses the instantaneous decrease of energy.

3.3 Semi-Implicit Time Discretization

To derive an effective algorithm, we still need to address two critical issues:

> *Computing geometric quantities such as curvature and normal velocity implicitly, but most of the geometry explicitly, thereby reaching a compromise between the schemes of §§3.1 and 3.2.* (57)

Providing a variational method to compute curvature that could be used in unstructured meshes. (58)

To deal with (57) we let $\mathcal{B}_n$ denote the linear operator defined by the scalar product $b_n(\cdot,\cdot)$ on Γ_n, namely,

$$\langle \mathcal{B}_n V, W\rangle = b_n(V,W) \qquad \forall V, W \in B(\Gamma_n).$$

Recalling the special form (13) of G, we thus propose the following *semi-implicit* computation of V_{n+1} and κ_{n+1}:

$$\mathcal{B}_n V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n). \tag{59}$$

This relation satisfies neither (53) nor (56) but tends to (15) for $\tau_n \to 0$. So backtracking must be employed to guarantee energy decrease.

To assess (58) we resort to basic differential geometry which asserts that the Laplace-Beltrami operator Δ_Γ of the position vector $\vec{X}$ of Γ is the vector mean curvature $\vec{\kappa}$ of Γ, namely,

$$-\Delta_\Gamma \vec{X} = \vec{\kappa} = \kappa\vec{\nu}. \tag{60}$$

Hereafter we use the minus sign to be consistent with the convention that a circle with outward unit normal $\vec{\nu}$ has positive curvature. The use of (60) for computation is due to Dziuk [13]. Since we need the scalar curvature κ instead of $\vec{\kappa}$, we proceed as Bänsch, Morin, and Nochetto [6] and consider the four unknowns $\vec{\kappa}, \kappa, V, \vec{V}$ along with their algebraic relations:

$$\kappa = \vec{\kappa}\cdot\vec{\nu}, \qquad \vec{V} = V\vec{\nu}. \tag{61}$$

If we take $\Gamma = \Gamma_{n+1}$, then for consistency with (59) we enforce (60) and (61) semi-implicitly

$$-\Delta_{\Gamma_n}\vec{X}_{n+1} = \vec{\kappa}_{n+1}, \qquad \kappa_{n+1} = \vec{\kappa}_{n+1}\cdot\vec{\nu}_n, \qquad \vec{V}_{n+1} = V_{n+1}\vec{\nu}_n.$$

Finally, to close the system we must relate position $\vec{X}_{n+1}$ of Γ_{n+1} and velocity $\vec{V}_{n+1}$. We impose

$$\vec{X}_{n+1} = \vec{X}_n + \tau_n \vec{V}_{n+1}, \tag{62}$$

and thereby get the *semi-implicit scheme*: find $(\vec{\kappa}_{n+1}, \kappa_{n+1}, V_{n+1}, \vec{V}_{n+1})$ s.t.

$$\vec{\kappa}_{n+1} + \tau_n \Delta_{\Gamma_n}\vec{V}_{n+1} = -\Delta_{\Gamma_n}\vec{X}_n \tag{63}$$

$$\kappa_{n+1} - \vec{\kappa}_{n+1}\cdot\vec{\nu}_n = 0 \tag{64}$$

$$\mathcal{B}_n V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n) \tag{65}$$

$$\vec{V}_{n+1} - V_{n+1}\vec{\nu}_n = 0. \tag{66}$$

3.4 Choosing the Scalar Product

Depending on the application one is interested in (e.g. image segmentation, optimal shape design for PDE or surface diffusion and epitaxially stressed solids), there are several possibilities in choosing the space $B(\Gamma)$ and the associated bilinear form $b(\cdot,\cdot)$.

A first possibility is to choose $b(\cdot,\cdot)$ to coincide with the $L^2(\Gamma)$ scalar product. Then equation (65) takes the form

$$V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{67}$$

which is a *backward-forward* parabolic-type equation, depending on the sign of the function g. In fact, (67) is locally backward (ill-posed) in regions where $g < 0$ and forward otherwise. This issue is of particular importance in the case of optimal shape design for PDE, where the sign of g is unknown (see §4).

A second possibility is to choose $b(\cdot,\cdot)$ to coincide with a weighted $H^1(\Gamma)$ scalar product. In this case equation (65) reads

$$-\mathrm{div}_\Gamma(\alpha\nabla_\Gamma V_{n+1}) + \beta V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{68}$$

where β and α are some positive functions. In §4 we will see that this choice (for suitable β's and α's) is well suited to stabilize the (ill-posed) L^2 gradient flow in the case of optimal shape design for PDE and it is helpful in increasing the rate of convergence of the numerical scheme in the case of image segmentation.

A third option is to choose the bilinear form $b(\cdot,\cdot)$ to coincide with the $H^{-1}(\Gamma)$ scalar product. If we assume $g(\Omega_n) = 1$ for simplicity, then equation (65) becomes

$$V_{n+1} - \Delta_\Gamma \kappa_{n+1} = \Delta_\Gamma f(\Omega_n). \tag{69}$$

This is the case for epitaxially stressed solids. Their dynamics described by the physical law (9) is given by the H^{-1} gradient flow of the energy functional (11). Hence the choice of the scalar product has to be driven by consistency with the physics of the underlying phenomena.

Now to derive a weak formulation, we can proceed as in [6], that is multiply by suitable test functions, integrate by parts the terms involving Δ_Γ, and ignore boundary terms either because Γ_n is closed or we assume Dirichlet boundary conditions. This is explained in detail for the $H^{-1}(\Gamma)$ case in [6]. In the following we describe the the finite element formulation for the weighted $H^1(\Gamma)$ case.

3.5 Finite Element Discretization

We now discuss the finite element discretization of (63)-(66). Let $\mathcal{T} = \mathcal{T}_n$ be a regular but possibly graded mesh of triangular finite elements over the surface $\Gamma = \Gamma_n$, which, from now on, is assumed to be polyhedral. To simplify the notations we hereafter drop the scripts n and $n+1$. Let $T \in \mathcal{T}$ be a typical triangle and let $\vec{\nu}_T = (\nu_T^i)_{i=1}^d$ be the unit normal to T pointing outwards. We

denote by $\vec{\nu}$ the outward unit normal to Γ, which satisfies $\vec{\nu}|_T = \vec{\nu}_T$ for all $T \in \mathcal{T}$, and is thus discontinuous across inter-element boundaries. Let $\{\phi_i\}_{i=1}^I$ be the set of canonical basis functions of the finite element space $\mathcal{V}(\Gamma)$ of continuous piecewise polynomials P^k of degree $\leq k$ over $\mathcal{T}$ for $k \geq 1$; we also set $\vec{\mathcal{V}}(\Gamma) := \mathcal{V}(\Gamma)^d$. We thus have a conforming approximation $\mathcal{V}(\Gamma)$ of $H^1(\Gamma)$.

We now multiply equations (63)-(66) by test functions $\phi \in \mathcal{V}(\Gamma)$ and $\vec{\phi} \in \vec{\mathcal{V}}(\Gamma)$ and integrate by parts terms involving the second order operator Δ_Γ. If $\langle \cdot, \cdot \rangle$ denotes the L^2 inner product over Γ, we arrive at the fully discrete problem: seek $\vec{V}, \vec{\kappa} \in \vec{\mathcal{V}}(\Gamma)$, $V, k \in \mathcal{V}(\Gamma)$, such that

$$\langle \vec{\kappa}, \vec{\phi} \rangle - \tau \langle \nabla_\Gamma \vec{V}, \nabla_\Gamma \vec{\phi} \rangle = \langle \nabla_\Gamma \vec{X}, \nabla_\Gamma \vec{\phi} \rangle \quad \forall \vec{\phi} \in \vec{\mathcal{V}}(\Gamma), \tag{70}$$

$$\langle \kappa, \phi \rangle - \langle \vec{\kappa} \cdot \vec{\nu}, \phi \rangle = 0 \quad \forall \phi \in \mathcal{V}(\Gamma), \tag{71}$$

$$\langle \alpha \nabla_\Gamma V, \nabla_\Gamma \phi \rangle + \langle \beta V, \phi \rangle + \langle g\kappa, \phi \rangle = -\langle f, \phi \rangle \quad \forall \phi \in \mathcal{V}(\Gamma), \tag{72}$$

$$\langle \vec{V}, \vec{\phi} \rangle - \langle V, \vec{\phi} \cdot \vec{\nu} \rangle = 0 \quad \forall \vec{\phi} \in \vec{\mathcal{V}}(\Gamma). \tag{73}$$

3.6 Matrix Formulation

We turn our attention to an equivalent matrix formulation to the fully discrete problem. Given the matrix entries

$$M_{g_{i,j}} := \langle g\phi_i, \phi_j \rangle, \quad M_{\beta_{i,j}} := \langle \beta\phi_i, \phi_j \rangle, \quad M_{i,j} := \langle \phi_i, \phi_j \rangle, \quad \vec{M}_{i,j} := M_{i,j}\vec{Id},$$
$$\vec{N}_{i,j} := (N^k_{i,j})^d_{k=1} := \langle \phi_i, \phi_j \nu^k \rangle^d_{k=1},$$
$$A_{i,j} := \langle \nabla_\Gamma \phi_i, \nabla_\Gamma \phi_j \rangle, \quad A_{\alpha i,j} := \langle \alpha \nabla_\Gamma \phi_i, \nabla_\Gamma \phi_j \rangle, \quad \vec{A}_{i,j} := A_{i,j}\vec{Id},$$

with $\vec{Id} \in \mathbb{R}^{d\times d}$ being the identity matrix and $(\vec{e}_k)^d_{k=1}$ the canonical basis of $\mathbb{R}^d$, the mass and stiffness matrices are

$$M_g := (M_{g_{i,j}})^I_{i,j=1}, \quad M_\beta := (M_{\beta_{i,j}})^I_{i,j=1}, \quad M := (M_{i,j})^I_{i,j=1}, \quad \vec{M} := (\vec{M}_{i,j})^I_{i,j=1},$$
$$\vec{N} := (\vec{N}_{i,j})^I_{i,j=1},$$
$$A_\alpha := (A_{\alpha i,j})^I_{i,j=1}, \quad A := (A_{i,j})^I_{i,j=1}, \quad \vec{A} := (\vec{A}_{i,j})^I_{i,j=1}.$$

We point out that $\vec{M}, \vec{A}$ and $\vec{N}$ possess matrix-valued entries and therefore the matrix-vector product is understood in the following sense

$$\vec{M}\vec{\mathbf{V}} = \big(\sum_{j=1}^I \vec{M}_{i,j}\vec{V}_j\big)^I_{i=1}, \tag{74}$$

each component $\vec{V}_i$ of $\vec{\mathbf{V}}$, as well as each of $\vec{M}\vec{\mathbf{V}}$, is itself a vector in $\mathbb{R}^d$.
We use the convention that a vector of nodal values of a finite element function is written in bold face: $\mathbf{V} = (V_i)^I_{i=1} \in \mathbb{R}^I$ is equivalent to $\mathbf{V} = \sum_{i=1}^I V_i\phi_i \in \mathcal{V}(\Gamma)$. We are now in a position to write the matrix formulation of (63)-(66). Upon expanding the unknown scalar function $V, k \in \mathcal{V}(\Gamma)$ and vector functions $\vec{V}, \vec{k} \in$

$\vec{\mathcal{V}}$ in terms of the basis functions and setting $\phi = \phi_i$ and $\vec{\phi}^k = \phi\vec{e}_k$, we easily arrive at

$$-\tau\vec{A}\vec{\mathbf{V}} + \vec{M}\vec{\mathbf{K}} = \vec{A}\vec{\mathbf{X}} \tag{75}$$

$$M\mathbf{K} - \vec{N}^T\vec{\mathbf{K}} = \mathbf{0} \tag{76}$$

$$(A_\alpha + M_\beta)\mathbf{V} + M_g\mathbf{K} = -\mathbf{f} \tag{77}$$

$$\vec{M}\vec{\mathbf{V}} - \vec{N}\mathbf{V} = \vec{0}. \tag{78}$$

$$\tag{79}$$

This system can be written equivalently in block-matrix form and 2D as follows:

$$C\mathbf{x} = \mathbf{g},$$

where $\mathbf{x} := (V_1, V_2, \kappa, K_1, K_2, V)^T$ and

$$C = \begin{pmatrix} M & 0 & 0 & 0 & 0 & -N_1 \\ 0 & M & 0 & 0 & 0 & -N_2 \\ 0 & 0 & M & -N_1 & -N_2 & 0 \\ -\tau A & 0 & 0 & M & 0 & 0 \\ 0 & -\tau A & 0 & 0 & M & 0 \\ 0 & 0 & M_g & 0 & 0 & A_\alpha + M_\beta \end{pmatrix}, \quad \mathbf{g} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ AX_1 \\ AX_2 \\ -f \end{pmatrix}. \tag{80}$$

3.7 Schur Complement Approach

Let us rewrite the system $C\mathbf{x} = \mathbf{g}$ in the following way

$$\begin{pmatrix} \vec{M} & 0 & 0 & -\vec{N} \\ 0 & M & -\vec{N}^T & 0 \\ -\tau\vec{A} & 0 & \vec{M} & 0 \\ 0 & M_g & 0 & A_\alpha + M_\beta \end{pmatrix} \begin{pmatrix} \vec{V} \\ \kappa \\ \vec{K} \\ V \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vec{A}\vec{X} \\ -f \end{pmatrix}, \tag{81}$$

or equivalently, with obvious meaning of the notations,

$$\begin{pmatrix} \Gamma & N \\ \Sigma & \tilde{A} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} 0 \\ h \end{pmatrix}. \tag{82}$$

As the following equalities hold

$$X_1 = -\Gamma^{-1}NX_2 \tag{83}$$

$$(-\Sigma\Gamma^{-1}N + \tilde{A})X_2 = h, \tag{84}$$

then equation (84) is equivalent to

$$\begin{pmatrix} \vec{M} & -\tau\vec{A}\vec{M}^{-1}\vec{N} \\ M_gM^{-1}\vec{N}^T & A_\alpha + M_\beta \end{pmatrix} \begin{pmatrix} \vec{K} \\ V \end{pmatrix} = \begin{pmatrix} \vec{A}\vec{X} \\ -f \end{pmatrix}. \tag{85}$$

Finally the Schur complement reads as follows

$$(\tau M_gM^{-1}\vec{N}^T\vec{M}^{-1}\vec{A}\vec{M}^{-1}\vec{N} + A_\alpha + M_\beta)V = -f - M_gM^{-1}\vec{N}^T\vec{M}^{-1}\vec{A}\vec{X}. \tag{86}$$

4 Numerical Experiments

The numerical experiments presented here were implemented and carried out within the finite element toolbox ALBERTA of Schmidt and Siebert [22]. Computations are driven by curve or surface evolution of the deformable part Γ of the domain Ω. To solve elliptic PDE in Ω, as in §§4.3 and 4.4.2, we invoked the 2d mesh generator TRIANGLE of Shewchuk [23], which partitions Ω into shape regular triangles. Finally, we used GEOMVIEW [14] for visualization.

4.1 Implementation

It is typical of surface evolution undergoing large deformations that triangles may tangle and cross, and that their angles may become large. These mesh distortions limit resolution and approximability, as well as impair computations thereby leading to numerical artifacts. We have resorted to a number of geometric enhancements as proposed by Bänsch, Morin and Nochetto for surface diffusion [6]. An additional feature for curves in 2D is the capability for topological changes. We list these features below and briefly comment on them.

- *Mesh Regularization*: This is a procedure to maintain mesh quality, namely to keep all angles on element stars approximately of the same size. Mesh regularization is a redistribution of nodes on the surface, which entails a tangential flow and does not affect the normal motion. We use the volume preserving Gauss-Seidel type iteration of [6].

- *Time Adaptivity*: This is a procedure to allow large timesteps when the normal velocity does not exhibit large variations, and to force small timesteps when the relative position change of the nodes of an element may exceed the element size. This accounts for very disparate time scales and prevent mesh distortion and even node crossing. We have two types of timestep control. The first one, following [6], is geometric and limits the tangential motion of nodes: if $\mathbf{z}_0, \mathbf{z}_1$ are nodes belonging to a triangle T on Γ_n, and $\vec{\tau}_T$ is any unit tangent vector to T, then

$$\tau \big| (\vec{V}_{n+1}(\mathbf{z}_0) - \vec{V}_{n+1}(\mathbf{z}_1)) \cdot \vec{\tau}_T \big| \leq C \tau h_T |\nabla_\Gamma \vec{V}_{n+1}|_T| \leq \epsilon \tau h_T,$$

  with $C, \epsilon > 0$ mesh independent constants.

- *Backtracking*: We monitor energy decrease $J(\Omega_{n+1}) < J(\Omega_n)$ in the following way: After the timestep has been accepted according to the previous criterion we check that the functional value of the new domain is smaller than that of the previous domain. If this is not the case, we divide the timestep by two and re-compute, repeating if necessary until the functional value is smaller than the previous one. The algorithm stops when the timestep necessary for functional decrease is smaller than a pre-specified minimum timestep.

- *Space Adaptivity*: This is about keeping, via refining/coarsening, an accurate representation of Γ_n in the sense that the node density correlates with the local variation (regularity) of Γ_n. We measure the latter intrinsically by monitoring the variation of normals $\vec{\nu}$. Since the pointwise accuracy of a mesh in representing a smooth surface is proportional to $h_S^2|\nabla_\Gamma \vec{\nu}|$, we impose [6]

$$h_S^2|\nabla_\Gamma \vec{\nu}| \approx h_S^2 \frac{|\vec{\nu}_1 - \vec{\nu}_2|}{h_S} \approx \beta_S h_S \leq \epsilon;$$

  here T_1, T_2 are two adjacent elements in Γ_n with unit normals $\vec{\nu}_1, \vec{\nu}_2$ and common side (node in 2d) S, β_S is the angle between $\vec{\nu}_1, \vec{\nu}_2$, and $\epsilon > 0$ is a given constant.

- *Angle Width Control*: This consists of a single splitting (one bisection) of those elements with angles wider than a certain threshold $\beta_{\max}$, followed by a few mesh regularization sweeps [6]. This procedure is important close to pinch-off where mesh distortion increases dramatically (see Figure 11).

- *Topological Changes in 2D*: This is given by a set of algorithms that allow us to perform topological changes in 2D, such as merging and splitting. At each iteration, we check for element intersections that signal topological changes. If there are intersections, we adjust the time step and local resolution. We reconnect the elements at intersection location and delete some elements if necessary. See [12] for details of the procedure.

4.2 Image Segmentation

In this section we perform numerical experiments with a number of synthetic images, by minimizing the geodesic active contour functional

$$J(\Omega) := \int_\Gamma H(x) dS + \lambda \int_\Omega H(x) dx.$$

The process starts with an initial curve (or surface in 3D), which is iteratively deformed to decrease its energy at each step, and should terminate at the boundaries of the objects in the image.

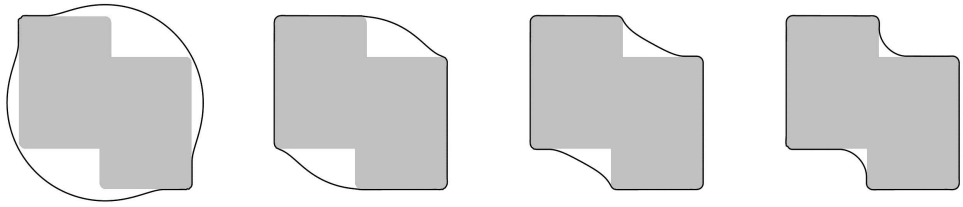
4.2.1 Two Gradient Flows: L^2 vs Weighted H^1

In our first experiment, we have a simple image with one (connected) object in it and we want to compare the L^2 gradient flow with the one resulting from the following weighted H^1 scalar product

$$b(V, W) = \int_\Gamma \alpha \nabla_\Gamma V \nabla_\Gamma W + \beta V W, \tag{87}$$

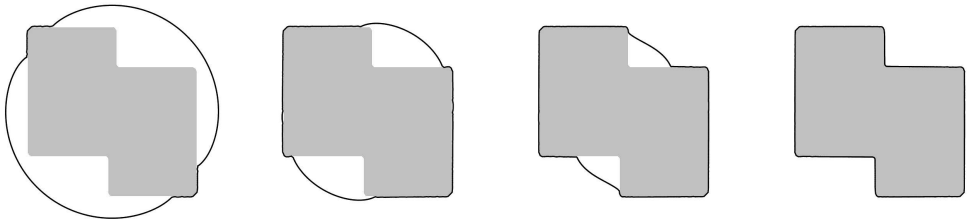
where

$$\alpha = H, \quad \beta = \left(\nu^T D^2 H \nu + (2\kappa + \lambda)\frac{\partial H}{\partial \nu} + \lambda \kappa H\right)_+$$



$k = 400$ $J = 11.176$ | $k = 600$ $J = 9.822$ | $k = 900$ $J = 9.240$ | $k = 1400$ $J = 8.950$

Figure 1: Detection of a simple object with L^2 gradient descent ($\lambda = 30$).



$k = 30$ $J = 12.146$ | $k = 80$ $J = 8.419$ | $k = 110$ $J = 7.545$ | $k = 160$ $J = 6.513$

Figure 2: Detection of a simple object with weighted H^1 gradient descent ($\lambda = 30$). This flow is faster and more accurate than the L^2 gradient flow of Fig. 1.

and $(\cdot)_+ = \max(\cdot, \epsilon), \epsilon > 0$. This scalar product has been obtained in [17] by taking the second order shape derivative into account, thereby resulting in a Newton-type flow.

Both the L^2 and weighted H^1 Newton-type flows successfully detect the object, but the latter exhibits a smaller number of iterations and thus a higher rate of convergence (see Figure 1 and Figure 2). In this case the choice of the "good" scalar product has been essentially driven by issues concerning stability and rate of convergence of the descent method.

In the rest of section 4.2 we perform the numerical experiments by using the weighted H^1 scalar product (87). We should remark that the choice of ϵ is a subtle issue here. Hintermüller and Ring report in [17] that a small value suffices in general. Our experiments also show that $\epsilon = 0.1$ is well-behaved for a 100×100 image if we take interpixel distance equal to 1 to obtain a computing domain of $[0, 100]^2$. However we prefer to take $\tilde{x} = 0.01x$ and rescale the computing domain to $[0, 1]^2$. This change of variables scales α by a factor 10^4 through D^2H, $\frac{\partial H}{\partial \nu}$ and κ. To preserve the time scale, ϵ must also be replaced with $\tilde{\epsilon} = (0.01)^{-2}\epsilon = 10^3$. This is the value we use in our simulations.

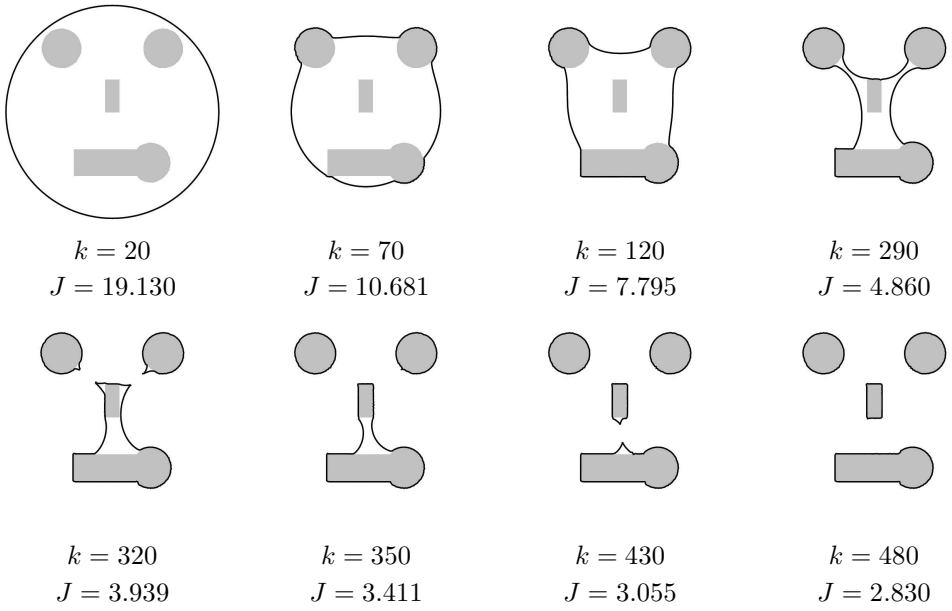


Figure 3: Detection of multiple objects with weighted H^1 gradient descent ($\lambda = 40$). The initial curve breaks into four curves and recovers all four objects in the image

4.2.2 Multiple Objects

In this example we have an image with multiple objects in it. We start with a single closed curve, that evolves and breaks into four smaller curves, which eventually converge to the boundaries of the objects (see Figure 3). The topological changes have been handled by using numerical tools developed in [12].

4.2.3 A 3D Image

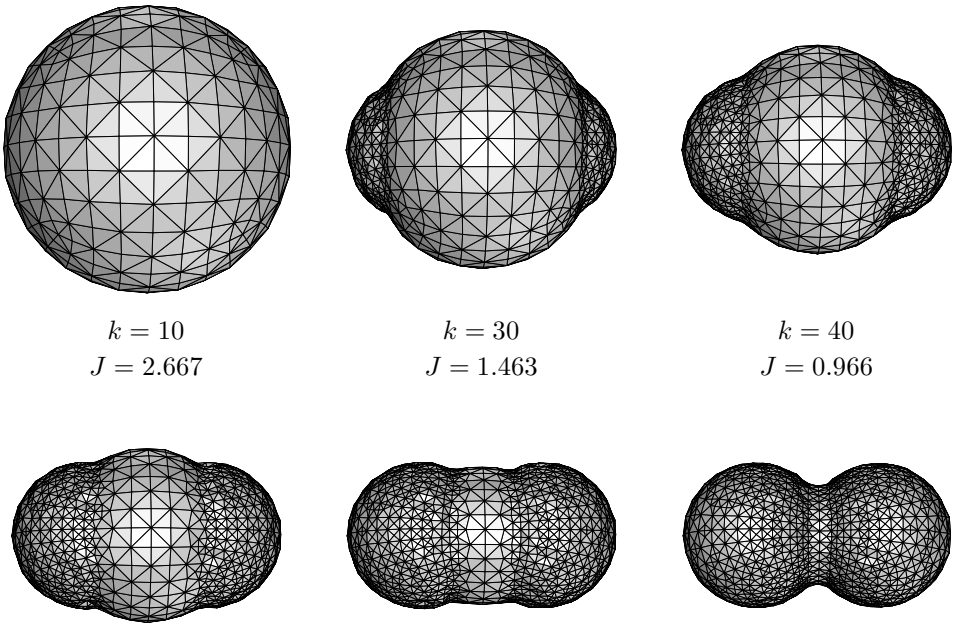
We finally consider a 3D image consisting of two spheres touching each other. The initial configuration of the iterative process is a spherical surface.

4.3 Optimal Shape Design for PDE

In this section we present some numerical experiments for the model problem of §1.2 and §2.2.2. Here, the energy functional reads

$$J(\Omega) := \frac{1}{2} \int_D \Big(y(\Omega) - z_g \Big)^2 dx$$

where z_g is a given *target* function on a subdomain D of Ω. The goal is to have the solution y of (7) closest to z_g in the least squares sense, inside D. We assume that both Ω and D are polygonal domains, and when we generate the meshes for solving the elliptic problems on Ω we enforce the mesh generator to match the



$k = 10$
$J = 2.667$

$k = 30$
$J = 1.463$

$k = 40$
$J = 0.966$

$k = 50$
$J = 0.536$

$k = 60$
$J = 0.230$

$k = 110$
$J = 0.036$

Figure 4: Detection of a 3D object with H^1 gradient descent ($\lambda = 0$).

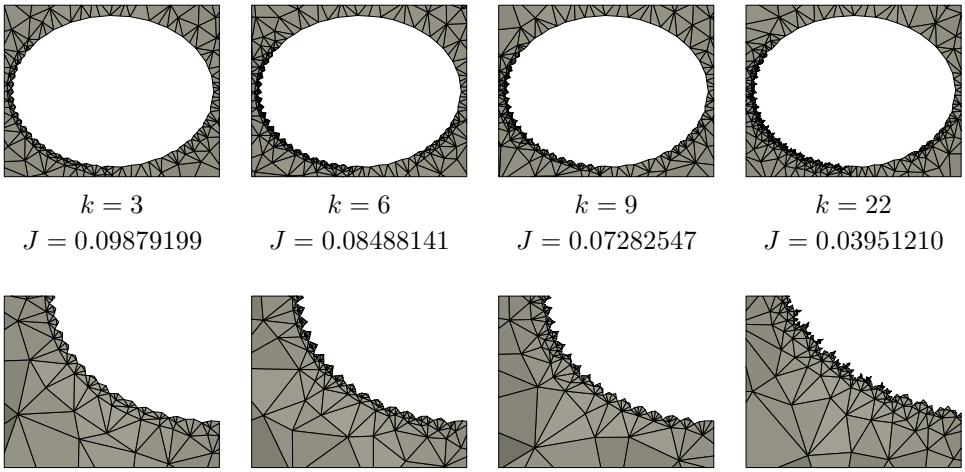


Figure 5: L^2 Evolution of a non-centered ellipse to a centered circle. The exact solution is a circle of radius one centered at the origin, and the initial configuration is a small ellipse centered at $(0.5, 0.7)$. See initial configuration and exact solution in figure 6 below. The L^2 scalar product produces a dynamics which is locally stable where $p > 0$ (upper right) and locally unstable where $p < 0$ (lower left).

boundary of D exactly. This minimizes the quadrature error for the right-hand side evaluation in the equations for both y and p.

4.3.1 The choice of Scalar Product

We performed the numerical experiments with a bilinear form $b(\cdot, \cdot)$ corresponding to a weighted H^1 scalar product, which gives rise to the elliptic PDE (68) on Γ_n:

$$-\mathrm{div}_\Gamma(\alpha \nabla_\Gamma V_{n+1}) + \beta V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{88}$$

where $g = p$, $f = -\nabla_\Gamma y \nabla_\Gamma p$, and the weight functions β and α were chosen appropriately.

The first attempt consisted in using just the $L^2(\Gamma)$ scalar product, which is obtained by taking $\alpha = 0$ and $\beta = 1$. In this case (88) reduces to

$$V_{n+1} + p\kappa_{n+1} = \nabla_\Gamma y \nabla_\Gamma p, \tag{89}$$

which is a *backward-forward* parabolic-type equation, depending on the sign of the adjoint solution p. More precisely, (89) is locally backward parabolic in regions where $p < 0$ and forward otherwise.

The illposedness of (89) in regions where $p < 0$ produces strong oscillations on the surface, which can be observed in Figure 5. This ruled out the simple minded option of using just the $L^2(\Gamma)$ scalar product.

The idea behind the choice of an adequate scalar product was to set $\beta = 1$ on the whole surface Γ and take α variable to smooth the evolution where it would produce oscillations.

After trying different possibilities we arrived at a compromise solution for the definition of α. We wanted the evolution to produce smooth surfaces to be able to compute, but at the same time we wanted to respect as much as possible the information given by the primal and dual solutions about the gradient of the shape functional.

The essential idea was to take $\alpha = 1$ where $p < 0$ and α small where $p > 0$. In order to avoid spurious singularities in the velocity we decided to make α vary smoothly from one element to the neighbor. In order to do that we defined α as constant on each edge e, but this constant was taken depending on $p_M := \max_e p$ and $p_m := \min_e p$ as follows: we defined the average $\bar{p} = \frac{1}{2}(p_M + p_m)$ and the oscillation $\text{osc} = p_M - p_m$ of p over e. If h_e denotes the length of e, we let $\alpha|_e$ be defined as follows:

$$\alpha|_e = \begin{cases} 1, & \text{if} \quad p_m < 0 \\ 1 - \frac{\bar{p}(1-h_e^2)}{\text{osc}}, & \text{if} \quad p_m \geq 0 \ \& \ \bar{p} \leq \text{osc} \\ h_e^2, & \text{if} \quad p_m \geq 0 \ \& \ \bar{p} > \text{osc}. \end{cases} \tag{90}$$

That is, in the transition region where p changes sign, $\alpha|_e$ is some kind of linear interpolation value between 1 and h_e^2.

4.3.2 Test 1: Exact solution

In this section we present an example with a known optimal shape. To build this example, we let $z_g = \log \frac{3}{|x|}$ be the exact solution of Laplace's equation on the ring $\{1 < |x| < 3\}$ with homogeneous Dirichlet data on $\{|x| = 3\}$, and outer normal derivative equal to 1 on $\{|x| = 1\}$. We let $D = \{2 \leq |x| \leq 2.5\}$, and we recall that the functional to minimize is

$$J(\Omega, y(\Omega)) := \frac{1}{2} \int_D \Big(y(\Omega) - z_g \Big)^2 dx.$$

Then $\Omega^* := \{1 < |x| < 3\}$ is the global minimum.

As we mentioned before, we tested several scalar products. The weighted H^1-scalar product with α as in (90) turned out to be a reasonable compromise between numerical stability and rate of convergence. We present a sequence of computations in Figure 6 starting from a noncentered ellipse. In contrast, the L^2-inner product yields unstable computations as documented in Figure 5 for the same initial configuration.

We also studied the evolution from different initial configurations and observed that the algorithm always reaches a local minimum, but not necessarily the global minimizer Ω^*.

In Figure 7 we show the evolution obtained with an initial configuration consisting of two squares. This evolution stops at a local minimum different from

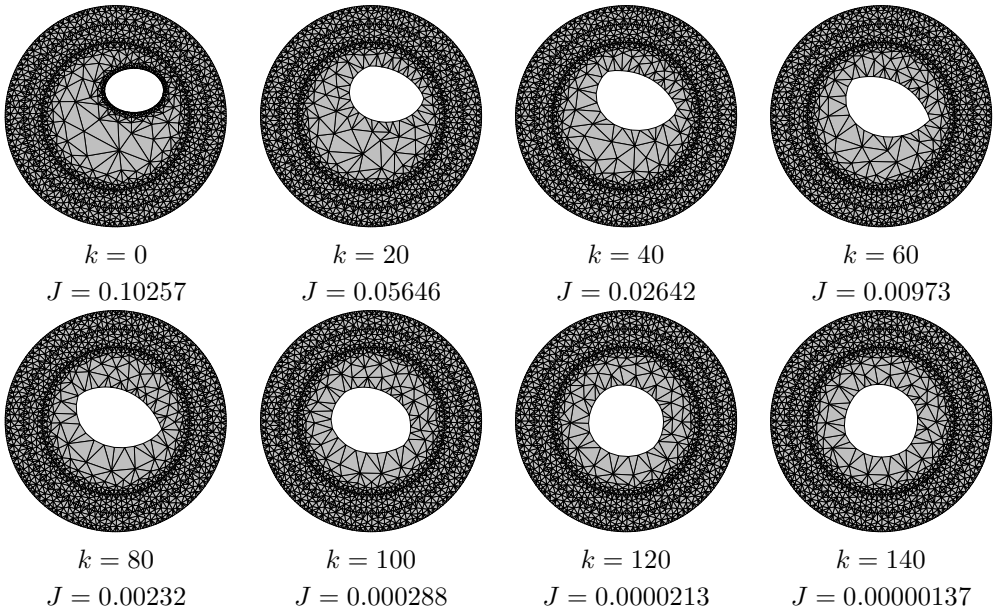


Figure 6: Weighted H^1-evolution of a non-centered ellipse to a centered circle. This dynamics is stable and efficient to detect a local minimizer.

the optimal configuration Ω^* but nevertheless the functional decreases several orders of magnitude.

An observation common to all simulations is that the reduction of $J(\Omega)$ and the change of shape of Γ are fast at the beginning but somewhat slow close to the optimal shape. We do not report these results here.

4.3.3 Test 2: Unknown solutions

We present here two examples where the exact minimizer is not known.

The first example consists of the same initial configuration and the same ring D as in the examples above, but the goal is to minimize

$$J(\Omega, y(\Omega)) := \frac{1}{2}\int_D \Big(y(\Omega) - z_g\Big)^2 dx.$$

with the goal function z_g given in polar coordinates by

$$z_g(r, \theta) = 0.45 + 0.4\cos(6\,\theta).$$

That is, the objective function z_g is oscillating with respect to the angle θ. The goal of this experiment is to check whether this gradient flows are able to capture more complicated geometries than circles.

In Figure 8 we present some snapshots of the approximating domains Ω_n together with the value of the functional $J(\Omega_n)$. It is interesting to notice that

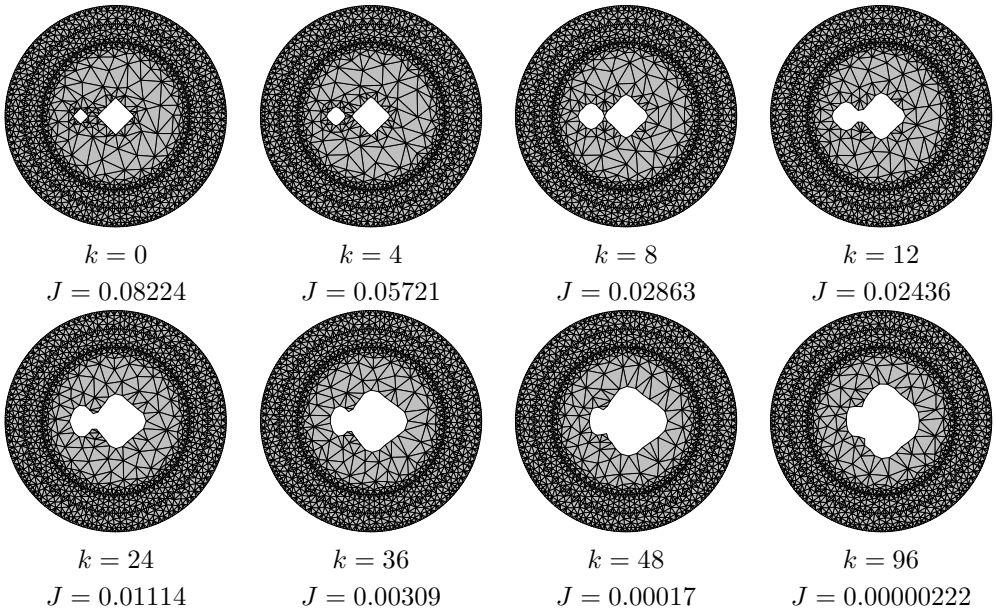


Figure 7: Evolution of two squares merging to a centered circle. The evolution stops at a local minimum before reaching the optimal configuration.

the functional value does not decrease as drastically as in the previous example. However, it is worth mentioning that the flow is able to capture the direction of decrease, even when the energy decrease is very small in relative terms. For example, between the fifth and sixth images, the energy decrease is just 0.03%, but the method is still able to detect how the curve should be modified to get an energy reduction.

Next, we consider a problem which resembles a lot a problem found in real applications. We take the starting domain to be

$$\Omega = \left\{(x_1, x_2) \in \mathbb{R}^2 : \max\{x_1, x_2\} < 3, \text{ and } |(x_1, x_2)| > 0.25\right\}.$$

We consider homogeneous Dirichlet boundary conditions on the outer boundary, which is kept fixed, and the outer normal derivative equal to 1 in the "inner boundary". The goal is to optimize the shape of the inner boundary in order to minimize the functional

$$J(\Omega, y(\Omega)) := \frac{1}{2}\int_D \Big(y(\Omega) - z_g\Big)^2 dx.$$

with $z_g \equiv 0.45$ on the domain $D := \left\{(x_1, x_2) \in \mathbb{R}^2 : 2.0 < \max\{x_1, x_2\} < 2.5\right\}$. That is, the ideal goal would be that the solution y is equal to 0.45 on the (square) ring D. We present a sequence of computations in Figure 9 starting from a small centered circle (of radius 0.5). The little circle evolves first into a big circle, and later into a smoothed square obtaining an energy reduction that

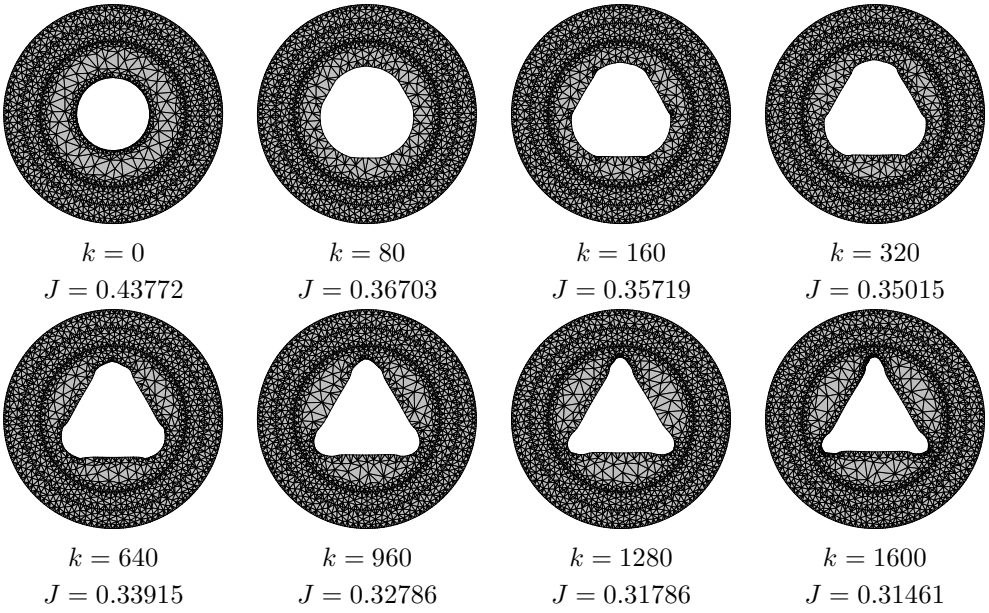


Figure 8: Evolution of a small circle towards a smoothed triangle. The exact solution is not known. The goal is to have the solution u as closest to an oscillating function z_g in the ring $D = \big\{(x, y) \in \mathbb{R}^2 : 2.0 < |(x, y)| < 2.5\big\}$.

goes from an initial value of 0.97481 to a final value of 0.19826. As mentioned before, there is a quick energy reduction at the beginning and a slower reduction in the end, which is typical of gradient flows.

4.4 Surface Diffusion and Epitaxially Stressed Solids

We present a couple of simulations exhibiting pinch-off in finite time, for the model problem of §1.3 and §2.2.3:

$$J(\Omega, y(\Omega)) = \int_\Omega |\nabla y(\Omega)|^2 + \int_\Gamma dS.$$

We first consider in §4.4.1 the pure geometric motion by surface diffusion, namely $y(\Omega) = 0$, and next the coupled problem in §4.4.2:

$$V = \Delta_\Gamma(\kappa + |\nabla_\Gamma y(\Omega)|^2).$$

We illustrate the use of space-time adaptivity as well as mesh smoothing and angle width control as explained in §4.1 to maintain mesh quality.

4.4.1 Test 1: Surface Diffusion with Pinch-off

Let the initial surface $\Gamma(0)$ be an $8 \times 1 \times 1$ prism. This surface evolution is geometric and, even though it is regularizing, it leads to pinch-off in finite time

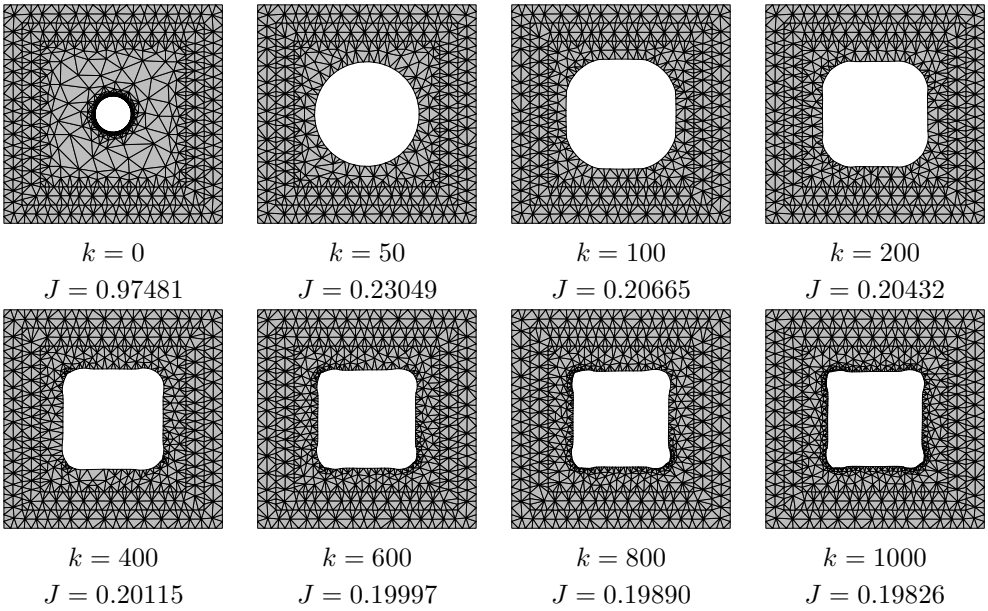


Figure 9: Evolution of a small circle towards a smoothed square. The exact solution is not known. The goal is to have the solution u as closest to 0.45 as possible in the region $D = \{(x_1, x_2) \in \mathbb{R}^2 : 2.0 < \max\{x_1, x_2\} < 2.5\}$.

as depicted in Figure 10. This is a key example for the use of mesh smoothing and space-time adaptivity to avoid mesh distortion. However, close to the pinch-off some elements would tend to degenerate if it were not for the angle width control. Their combined effect is displayed is Figure 11.

4.4.2 Test 2: Formation of an Inclusion

We now couple surface diffusion of the free surface of a film with the Laplace equation in the bulk, as explained in §§1.3 and 2.2.3. We imposed the Dirichlet boundary condition $y = x$ on both the bottom and lateral boundary. The initial free surface $\Gamma(0)$ is a small cosine perturbation of the flat case, and its evolution $\Gamma(t)$ is periodic in x. We observe that $\Gamma(t)$ retains the graph property for a while and eventually develops into a mushroom-like shape which closes up forming an insertion. See Figure 12 for details. This preliminary experiment, conducted in collaboration with Eberhard Bänsch during a *Research in Pairs* in Oberwolfach, reveals a superior performance of TRIANGLE [23] with respect to mesh deformation techniques in 2d. The situation in 3d is quite different and needs to be explored further.

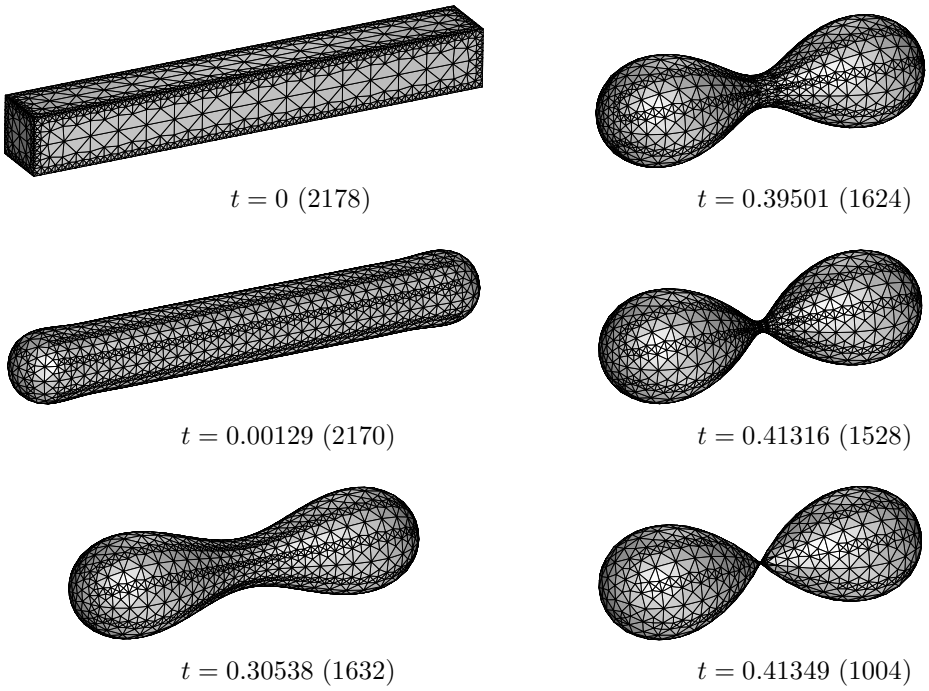


Figure 10: Pinch-off in finite time. Evolution of an $8 \times 1 \times 1$ prism at various time instants leading to a dumbbell and cusp formation (between parentheses we indicate the number of vertices used to represent the surface.) The evolution was computed using timestep control, mesh regularization, mesh refinement/coarsening, and a routine for controlling angle width (taken from [6]).

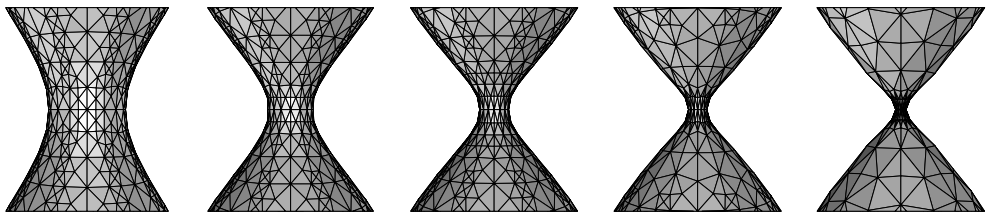


Figure 11: Detailed view of the pinch-off for the $8 \times 1 \times 1$ prism. The control of wide angles, coupled with mesh regularization, refinement and coarsening cure mesh distortion until the very moment of pinch-off, when the elements are rather elongated but not degenerate. An angle is considered to be wide when bigger than 120^{o} (taken from [6]).

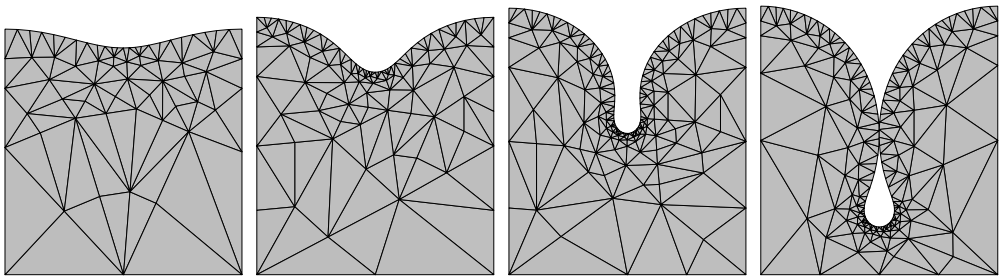


Figure 12: Coupling surface diffusion with the Laplace operator in the bulk leads to a mushroom-like free surface that gives rise to an inclusion.

Acknowledgments

The First Author and the Third Author are partially supported by NSF Grants DMS-0204670 and DMS-0505454. The Second Author is partially supported by CONICET through Grant PEI 6208, by SeCyT of Argentina through Grant PICT 03-13023 and by NSF Grant DMS-0204670. The Fourth Author is partially supported by the European IHP Network "Breaking Complexity", by NSF Grant DMS-0204670 and by M.U.R.S.T. Cofin 2003 Grant "Numerical Modelling for Scientific Computing and Advanced Applications".

References

[1] F. Almgren, J.E. Taylor, and L. Wang. Curvature-driven flows: a variational approach. *SIAM J. Control Optim.*, 31(2):387–438, 1993.

[2] L. Ambrosio. Minimizing movements. *Rend. Accad. Naz. Sci. XL Mem. Mat. Appl. (5)*, 19:191–246, 1995.

[3] L. Ambrosio, N. Gigli, and G. Savaré. *Gradient flows in metric spaces and in the space of probability measures*. Lectures in Mathematics ETH Zürich. Birkhäuser Verlag, Basel, 2005.

[4] R. J. Asaro and W. A. Tiller. Surface morphology development during stress corrosion cracking: Part i: via surface diffusion. *Metall. Trans.*, 3:1789–1796, 1972.

[5] G. Aubert and P. Kornprobst. *Mathematical problems in image processing*, volume 147 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2002. Partial differential equations and the calculus of variations, With a foreword by Olivier Faugeras.

[6] E. Bänsch, P. Morin, and R.H. Nochetto. A finite element method for surface diffusion: the parametric case. *J. Comput. Phys.*, 203(1):321–343, 2005.

[7] M. Burger. A framework for the construction of level set methods for shape optimization and reconstruction. *Interfaces Free Bound.*, 5(3):301–329, 2003.

[8] J. Cahn and J. Taylor. Surface motion by surface diffusion. *Acta Metall. Mater.*, 42:1045–1063, 1994.

[9] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *IJCV*, 22(1):61–79, 1997.

[10] J. Céa. Numerical methods for optimal shape design. In E. J. Haug and J. Céa, editors, *Proceedings of the NATO Advanced Study Institute on Optimization of Distributed Parameter Structural Systems*. Martinus Nijhoff Publishers, 1981.

[11] 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.

[12] G. Doğan. Topological changes and adaptivity for curve evolution. *in preparation*.

[13] G. Dziuk. An algorithm for evolutionary surfaces. *Numer. Math.*, 58(6):603–611, 1991.

[14] Geometry Center at the University of Minnesota. Geomview, Interactive 3D Viewing Program. `www.geomview.org`.

[15] A. Henrot and M. Pierre. *Variation et optimisation de formes*, volume 48 of *Mathématiques et Applications*. Springer, 2005.

[16] A. Henrot and G. Villemin. An optimum design problem in magnetostatics. *M2AN Math. Model. Numer. Anal.*, 36(2):223–239, 2002.

[17] M. Hintermüller and W. Ring. A second order shape optimization approach for image segmentation. *SIAM J. Appl. Math.*, 64(2):442–467 (electronic), 2003/04.

[18] S. Luckhaus. Solutions for the two-phase Stefan problem with the Gibbs-Thomson law for the melting temperature. *European J. Appl. Math.*, 1(2):101–111, 1990.

[19] B. Mohammadi and O. Pironneau. *Applied shape optimization for fluids*. Numerical Mathematics and Scientific Computation. The Clarendon Press Oxford University Press, New York, 2001. Oxford Science Publications.

[20] A. Novruzi and J. R. Roche. Newton's method in shape optimisation: a three-dimensional case. *BIT*, 40(1):102–120, 2000.

[21] O. Pironneau. *Optimal shape design for elliptic systems*. Springer Series in Computational Physics. Springer-Verlag, New York, 1984.

[22] A. Schmidt and K. Siebert. *Design of adaptive finite element software*, volume 42 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2005. The finite element toolbox ALBERTA, With 1 CD-ROM (Unix/Linux).

[23] J.R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[24] J. Sokołowski and J.-P. Zolésio. *Introduction to shape optimization*, volume 16 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1992. Shape sensitivity analysis.

[25] B. J. Spencer, S. H. Davis, and P. W. Voorhees. Morphological instability in epitaxially-strained dislocation-free solid films: nonlinear evolution. *Phys. Rev. B*, 47(2):9760–9777, 1993.

[26] J.-P. Zolésio. The material derivative (or speed method for shape optimization). In E. J. Haug and J. Céa, editors, *Proceedings of the NATO Advanced Study Institute on Optimization of Distributed Parameter Structural Systems*. Martinus Nijhoff Publishers, 1981.

[27] P. Zunino. Multidimensional pharmacokinetic models applied to the design of drug eluting stents. *Journal on Cardiovascular Engineering*, 4(2), 2004.