NORGES TEKNISK-NATURVITENSKAPELIGE
UNIVERSITET

High order numerical approximation of minimal surfaces

by

Øystein Tråsdahl, Einar M. Rønquist

PREPRINT
NUMERICS NO. 7/2010

NORWEGIAN UNIVERSITY OF
SCIENCE AND TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL
http://www.math.ntnu.no/preprint/numerics/2010/N7-2010.pdf
Address: Department of Mathematical Sciences, Norwegian University of Science and
Technology, N-7491 Trondheim, Norway.

High order numerical approximation of minimal surfaces

Øystein Tråsdahl, Einar M. Rønquist

September 14, 2010

We present an algorithm for finding high order numerical approximations of minimal surfaces with a fixed boundary. The algorithm employs parametrization by high order polynomials and a linearization of the weak formulation of the Laplace-Beltrami operator to arrive at an iterative procedure to evolve from a given initial surface to the final minimal surface. For the steady state solution we measure the approximation error in a case where the exact solution is known (the catenoid). In the framework of parametric interpolation, the choice of interpolation points (mesh nodes) is directly affecting the approximation error, and we discuss how to best update the mesh on the evolutionary surface such that the parametrization remains smooth. In our test case we achieve exponential convergence in the approximation of the minimal surface as the polynomial degree increases, but the rate of convergence greatly differs with different choices of mesh update algorithms.

Keywords: Minimal surfaces, mean curvature, free surface flow, evolutionary surfaces, mesh update techniques

1 Introduction

Surfaces of least area, called minimal surfaces, is a field of study that has intrigued scientists for many years and has been studied extensively [1, 2, 3]. Part of the interest stems from the fact that they are so easily realizable physically in the form of soap films, and for this reason they have been studied not only mathematically, but also physically for many years. An important early contribution came from the physicist J. A. F. Plateau, who studied them experimentally and determined some interesting geometric properties [4]. A breakthrough in the mathematical study of minimal surfaces came around 1930 with the works of J. Douglas [5] and T. Radó [6], who established some important theory around the existence of minimal surfaces.

The problem of finding exact minimal surfaces is very hard and in general unsolved. Only a few minimal surfaces have been found in closed form, and numerical methods are therefore an important tool. For non-parametric surfaces, methods have been proposed by Concus [7], Greenspan [8], Elcrat and Lancaster [9], Hoppe [10].

Corresponding author: Einar M. Rønquist. Tel.: +47 73 59 35 47. Fax: +47 73 59 35 24.
Email address: ronquist@math.ntnu.no

For parametric surfaces, the minimal surface problem has been solved with finite element methods by Brakke [11], Hinata *et al.* [12] and Wagner [13], whereas Coppin and Greenspan [14] use direct simulation of surface tension forces on a grid of marker particles. Chopp [15] has proposed a level set method which allows for natural handling of topological changes, but gives only linear convergence. It also employs a three-dimensional volume mesh, which is expensive and undesirable in our case, since we are only interested in a three-dimensional surface.

The goal with this work is to find a high order numerical approximation of a minimal surface with a given boundary. We start off with an introduction to minimal surface conditions, and from there we derive a weak form of the problem. The problem is discretized using high order polynomials, and we show how it results in a nonlinear system that can be solved with an iterative method. The iterations make our solution an evolutionary surface, and it leads to the question of mesh update techniques, which will be discussed in some detail. These techniques are also needed in moving boundary problems with arbitrary Lagrangian-Eulerian (ALE) formulations [16], and the algorithms presented in this paper are also relevant in an ALE setting.

We conclude with some numerical results showing the convergence properties of our method, as well as an example of a case where the exact solution is unknown. For two coaxial rings, a stable solution called the catenoid is known analytically and can be used to measure accuracy.

2 Problem formulation

Consider a two-dimensional surface Ω in $\mathbb{R}^3$ with a fixed boundary $\partial\Omega$, represented locally by a diffeomorphism $\boldsymbol{\varphi} : \hat{\Omega} \subset \mathbb{R}^2 \to \mathbb{R}^3$. At every point on Ω we can define the *mean curvature* by

$$\kappa = \frac{1}{2}(\kappa_1 + \kappa_2),$$

where κ_1 and κ_2 are the principal curvatures. The mapping $\boldsymbol{\varphi}$ and the mean curvature are linked through the relation

$$\Delta_\Omega \boldsymbol{\varphi} = -\kappa\, \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{n}$ is a unit surface normal vector, and Δ_Ω is the Laplace-Beltrami operator, a generalization of the Laplace operator to Riemannian manifolds. Minimal surfaces are characterized by the property that the mean curvature is everywhere zero. From (1) we conclude that minimal surfaces are solutions to the set of three partial differential equations

$$\begin{aligned} \Delta_\Omega \boldsymbol{\varphi} &= \mathbf{0} \quad \text{in } \hat{\Omega}, \\ \boldsymbol{\varphi} &= \boldsymbol{\varphi}_0 \quad \text{on } \partial\hat{\Omega}, \end{aligned} \tag{2}$$

where $\boldsymbol{\varphi}_0$ is simply a parametrization of the boundary $\partial\Omega$. Note that $\boldsymbol{\varphi}$ has three components, one for each coordinate direction. Apart from the trivial case where $\partial\Omega$ lies in a plane, and the minimal surface is obviously a plane surface, these partial differential equations are nonlinear.

As an example, consider the simpler case where the surface can be described by a function $z(x, y)$. Then (2) reduces to the (scalar) Plateau problem [17, 18]

$$\nabla \cdot \left(\frac{\nabla z}{\sqrt{1 + |\nabla z|^2}} \right) = 0,$$

with prescribed boundary conditions.

2.1 Weak formulation

A peculiar aspect of the minimal surface problem (2) is that the differential operator Δ_Ω is inextricably linked to the solution itself. This makes it very hard to solve the problem analytically except for in a few special cases.

We therefore start by considering the simpler, but related problem

$$\begin{aligned} \Delta_\Omega \hat{u} &= 0 \quad \text{in } \hat{\Omega}, \\ \hat{u} &= \hat{u}_0 \quad \text{on } \partial\hat{\Omega}, \end{aligned} \tag{3}$$

where $\hat{u}$ is a *scalar* function defined on $\hat{\Omega}$, and $\hat{u}_0$ is some given boundary condition. Here we assume the mapping $\boldsymbol{\varphi}$ to be known *a priori*. The problem can then be transformed to a problem defined on Ω by means of $\boldsymbol{\varphi}$. Assume that this mapping describes the surface in a Cartesian coordinate system,

$$\boldsymbol{\varphi}(\xi,\eta) = \begin{pmatrix} \varphi_1(\xi,\eta) \\ \varphi_2(\xi,\eta) \\ \varphi_3(\xi,\eta) \end{pmatrix} = \begin{pmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{pmatrix}.$$

The Jacobian associated with this mapping is given as

$$J = \begin{pmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \\ z_\xi & z_\eta \end{pmatrix}.$$

Let $u = \hat{u} \circ \boldsymbol{\varphi}^{-1}$; the inverse $\boldsymbol{\varphi}^{-1}$ exists since $\boldsymbol{\varphi}$ is a diffeomorphism. In the Cartesian coordinate system the Laplace-Beltrami operator simplifies to the well-known Laplace operator in $\mathbb{R}^3$,

$$\Delta_\Omega \hat{u} = \Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}.$$

Hence, solving (3) is equivalent to solving the problem

$$\begin{aligned} \Delta u &= 0 \quad \text{in } \Omega, \\ u &= u_0 \quad \text{on } \partial\Omega, \end{aligned} \tag{4}$$

where $u_0 = \hat{u}_0 \circ \boldsymbol{\varphi}^{-1}$, and $\partial\Omega$ is the image of $\partial\hat{\Omega}$ under the mapping $\boldsymbol{\varphi}$. The change of variables has moved the complexity from the operator to the domain itself. However, one advantage of this transformation is that the derivation of a weak formulation of (4) becomes easy. The Galerkin problem is given as: find $u \in Y^D \equiv \{v \in H^1(\Omega) \mid v|_{\partial\Omega} = u_0\}$ such that

$$\int_\Omega (\nabla v)^T \nabla u \,\mathrm{d}\Omega = 0 \qquad \forall v \in Y \equiv H_0^1(\Omega), \tag{5}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})^T$ is the standard gradient operator in $\mathbb{R}^3$. The integral (5) is not readily evaluated since Ω is a curved surface. We therefore apply a change of variables to transform it back to the reference domain $\hat{\Omega}$. An infinitesimal surface area $\mathrm{d}\Omega$ on the curved surface can be expressed in the reference variables as

$$\mathrm{d}\Omega = g \,\mathrm{d}\hat{\Omega},$$

where the metric g is defined in terms of the Jacobian J of $\boldsymbol{\varphi}$ by

$$g = \sqrt{\det(J^T J)}.$$

Gradients on Ω are related to gradients in the two-dimensional reference domain $\hat{\Omega}$ through the Jacobian $\tilde{J}$ of $\boldsymbol{\varphi}^{-1}$. Written out, we have

$$\nabla u = \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} = \begin{pmatrix} \hat{u}_\xi \xi_x + \hat{u}_\eta \eta_x \\ \hat{u}_\xi \xi_y + \hat{u}_\eta \eta_y \\ \hat{u}_\xi \xi_z + \hat{u}_\eta \eta_z \end{pmatrix} = \begin{pmatrix} \xi_x & \eta_x \\ \xi_y & \eta_y \\ \xi_z & \eta_z \end{pmatrix} \begin{pmatrix} \hat{u}_\xi \\ \hat{u}_\eta \end{pmatrix} = \tilde{J}^T \hat{\nabla} \hat{u},$$

where $\hat{\nabla} = (\frac{\partial}{\partial \xi}, \frac{\partial}{\partial \eta})^T$ is the two-dimensional gradient on the reference domain. Note the reappearance of the function $\hat{u}$; it is the same as in (3) since we use the particular inverse mapping $\boldsymbol{\varphi}^{-1}$.

The integral in (5) can now be expressed in reference variables as

$$\begin{aligned} \int_\Omega (\nabla v)^T \nabla u \, \mathrm{d}\Omega &= \int_{\hat{\Omega}} (\tilde{J}^T \hat{\nabla} \hat{v})^T \tilde{J}^T \hat{\nabla} \hat{u} \, g \, \mathrm{d}\hat{\Omega} \\ &= \int_{\hat{\Omega}} (\hat{\nabla} \hat{v})^T \tilde{J} \tilde{J}^T \hat{\nabla} \hat{u} \, g \, \mathrm{d}\hat{\Omega}. \end{aligned}$$

We can eliminate the dependence on the inverse mapping $\boldsymbol{\varphi}^{-1}$ by using the fact that the two Jacobian matrices J and $\tilde{J}$ are related as

$$\tilde{J}\tilde{J}^T = (J^T J)^{-1}.$$

The resulting integral can then be expressed as the bilinear form

$$a(\hat{v}, \hat{u}) = \int_{\hat{\Omega}} \kappa \, (\hat{\nabla} \hat{v})^T G \, \hat{\nabla} \hat{u} \, g \, \mathrm{d}\hat{\Omega}, \tag{6}$$

where $G = (J^T J)^{-1}$ and $\kappa = 1$ (the reason for introducing the parameter κ will be explained below). The matrix G is obviously symmetric, and it is also positive definite, since, for all $\boldsymbol{q} \in \mathbf{R}^2$, $\boldsymbol{q} \neq \mathbf{0}$,

$$\boldsymbol{q}^T G^{-1} \boldsymbol{q} = \boldsymbol{q}^T J^T J \boldsymbol{q} = (J\boldsymbol{q})^T (J\boldsymbol{q}) > 0.$$

Hence, $a(\hat{v}, \hat{v}) > 0$ for all $\hat{v} \in H_0^1(\hat{\Omega})$, $\hat{v} \neq 0$, and thus the bilinear form $a(\cdot, \cdot)$ is symmetric and positive definite (SPD).

We now introduce the space $\hat{Y}^D = \{\hat{w} \in H^1(\hat{\Omega}) \mid \hat{w}|_{\partial\hat{\Omega}} = \hat{u}_0\}$. The weak formulation of (3) is then given as: find $\hat{u} \in \hat{Y}^D$ such that

$$a(\hat{v}, \hat{u}) = 0 \qquad \forall \hat{v} \in H_0^1(\hat{\Omega}). \tag{7}$$

By comparing the strong formulations (2) and (3), we see that a weak formulation of the former is just a vector version of the latter: find $\boldsymbol{\varphi} = (\varphi_1, \varphi_2, \varphi_3)^T = (x, y, z)^T \in \hat{X}^D \equiv \{\hat{\boldsymbol{w}} \in (H^1(\hat{\Omega}))^3 \mid \hat{\boldsymbol{w}}|_{\partial\hat{\Omega}} = \boldsymbol{\varphi}_0\}$ such that

$$a(\hat{v}_i, \varphi_i) = 0 \qquad \forall \hat{v}_i \in H_0^1(\hat{\Omega}), \quad i = 1, 2, 3. \tag{8}$$

The notation in (8) hides an important fact: $a(\cdot, \cdot)$ is *not* a bilinear form for this particular argument because of the hidden dependence of $\boldsymbol{\varphi}$ in G and g. We will get back to this problem shortly.

2.2 Relation to free surface flow

There is a close link between minimal surfaces and free surface flows that deserves some attention here, particularly because the mesh update techniques described later in this paper have relevance to the numerical treatment of such flows. Furthermore, the insight obtained by viewing the different perspectives on surface minimization and the different derivations of the weak formulations are themselves sufficient reasons for this exposition.

For free surface flow, the surface tension represents a molecular force that acts to minimize the free surface at all time. Consider a three-dimensional unsteady flow with a free surface Ω. The total stress force acting on the free surface is the sum of a normal component $\boldsymbol{F}_n$ and a tangential component $\boldsymbol{F}_t$ and is given by [19]

$$\boldsymbol{F} = \boldsymbol{F}_n + \boldsymbol{F}_t = \gamma\kappa\boldsymbol{n} + \nabla_\Omega \gamma,$$

where γ is the surface tension, $\boldsymbol{n}$ is the outward unit normal vector and ∇_Ω is the surface gradient. The free surface flow is described by the Navier-Stokes equations, for which surface tension forces are represented by the boundary conditions

$$\begin{aligned} n_i \sigma_{ij} n_j &= \gamma\kappa, \\ t_i \sigma_{ij} n_j &= t_i (\nabla_\Omega \gamma)_i, \end{aligned}$$

where n_i and n_j are components of the unit normal vector $\boldsymbol{n}$, t_i is a component of a unit tangent vector $\boldsymbol{t}$, and σ_{ij} is a component of the stress tensor. Summation over repeated indices is assumed. A natural imposition of the free surface boundary conditions in a variational formulation of the Navier-Stokes equations yields the integral

$$\int_\Omega v_i \sigma_{ij} n_j \, \mathrm{d}\Omega, \qquad i = 1, 2, 3,$$

where v_i is a test function. This term includes both normal and tangential contributions. In [20] it was shown that this integral can be expressed as

$$\int_\Omega v_i \sigma_{ij} n_j \, \mathrm{d}\Omega = -\int_{\hat\Omega} \gamma \, \hat v_{i,\alpha} \, g_i^\alpha \, g \, \mathrm{d}\hat\Omega, \qquad i = 1, 2, 3,$$

where $\hat v_{i,\alpha}$ denotes the partial derivative of $\hat v_i = v_i \circ \boldsymbol{\varphi}$ with respect to the reference variable r^α (here, $r^1 = \xi$ and $r^2 = \eta$), and g_i^α is the i'th component of the *contravariant base-vector* $\boldsymbol{g}^\alpha$. From differential geometry we have [21]

$$\boldsymbol{g}^\alpha = g^{\alpha\beta} \boldsymbol{g}_\beta$$

where $g^{\alpha\beta}$ is the *contravariant metric tensor*, which in matrix notation is nothing but our matrix $G = (J^T J)^{-1}$. The vector $\boldsymbol{g}_\beta$ is the *covariant base-vector* and is defined as the partial derivative of the mapping with respect to the reference variable r^β, i,e., $\boldsymbol{g}_\beta = \boldsymbol{\varphi}_{,\beta}$. Thus, $\boldsymbol{g}_\beta$, $\beta = 1, 2$, represent two vectors spanning the tangent plane at a particular point on the surface. Inserting this into the integral yields

$$\begin{aligned} \int_{\hat\Omega} \gamma \, \hat v_{i,\alpha} \, g_i^\alpha \, g \, \mathrm{d}\hat\Omega &= \int_{\hat\Omega} \gamma \, \hat v_{i,\alpha} \, g^{\alpha\beta} \varphi_{i,\beta} \, g \, \mathrm{d}\hat\Omega \\ &= \int_{\hat\Omega} \gamma \, (\hat\nabla \hat v_i)^T G \, \hat\nabla \varphi_i \, g \, \mathrm{d}\hat\Omega \\ &= a(\hat v_i, \varphi_i), \qquad i = 1, 2, 3. \end{aligned}$$

Hence, it is interesting to observe that the contributions from the free surface boundary conditions (both normal and tangential) can be expressed by the form (6) in the particular case with $\hat{u} = \varphi_i$, $i = 1, 2, 3$, and with $\kappa = 1$ replaced by $\kappa = \gamma$. Note that for surface-tension-driven flows (Marangoni-type problems), γ is not a constant, but is still a positive quantity over the free surface.

2.3 Linearization and iterative scheme

The system (7) is linear in the unknown $\hat{u}$ and is readily solved with a finite or spectral (element) method. It also has the advantage that the bilinear form is SPD, so that the corresponding algebraic system can easily be solved using the Conjugate Gradients (CG) method.

The problem (8) is nonlinear, but can be solved by introducing an iterative scheme. At each iteration we start with a *known* surface Ω^n, parametrized by $\boldsymbol{\varphi}^n$, and we move to the next iteration by letting

$$\boldsymbol{\varphi}^{n+1} = \boldsymbol{\varphi}^n + \Delta\boldsymbol{\varphi}^{n+1},$$

where $\Delta\boldsymbol{\varphi}^{n+1}$ is a vector field with components $\Delta\varphi_i^{n+1}$, $i = 1, 2, 3$, that are the solutions of

$$a(\hat{v}_i, \varphi_i^n + \Delta\varphi_i^{n+1}) = 0, \qquad i = 1, 2, 3. \tag{9}$$

Here, $a(\cdot,\cdot)$ represents an integral over the unknown surface Ω^{n+1}, and the unknown $\Delta\boldsymbol{\varphi}^{n+1}$ enters into the nonlinear terms G and g and makes the entire system nonlinear. However, assuming that the update $\Delta\boldsymbol{\varphi}^{n+1}$ is relatively small, we can approximate $a(\cdot,\cdot)$ by an integral over the *known* surface Ω^n. We do this by “freezing” G and g at the values G^n and g^n that are computed from the current mapping $\boldsymbol{\varphi}^n$. This approximation yields a bilinear form

$$a^n(\hat{v}, \hat{w}) = \int_{\hat{\Omega}} (\hat{\nabla}\hat{v})^T \, G^n \, \hat{\nabla}\hat{w} \, g^n \, \mathrm{d}\hat{\Omega} \tag{10}$$

which is also SPD, since G^n is an SPD matrix and we consider $\hat{v} \in H_0^1(\hat{\Omega})$. Note that we have omitted κ in (10) since $\kappa = 1$. The linearized version of (9) is then

$$a^n(\hat{v}_i, \Delta\varphi_i^{n+1}) = -a^n(\hat{v}_i, \varphi_i^n), \qquad i = 1, 2, 3, \tag{11}$$

which is suitable for a numerical discretization.

3 Discretization

For the numerical solution of the Galerkin problem (8) we apply a spectral discretization based on high order polynomials. The relevant discrete function spaces are

$$\begin{aligned} \hat{X}_N &= \{\hat{\boldsymbol{w}} \in H_0^1(\hat{\Omega})^3 \mid \hat{\boldsymbol{w}} \in \mathbb{P}_N(\hat{\Omega})^3\}, \\ \hat{X}_N^D &= \{\hat{\boldsymbol{w}} \in H^1(\hat{\Omega})^3 \mid \hat{\boldsymbol{w}} \in \mathbb{P}_N(\hat{\Omega})^3 \text{ and } \hat{\boldsymbol{w}} = \boldsymbol{\varphi}_0 \text{ on } \partial\hat{\Omega}\}. \end{aligned}$$

As a basis for these spaces we choose the tensor-product Lagrangian interpolants through the Gauss-Lobatto-Legendre (GLL) points $\xi_0, \ldots, \xi_N$. If ψ_N represents a component of an element in $\hat{X}_N$ or $\hat{X}_N^D$, this component is expressed as

$$\psi_N(\xi, \eta) = \sum_{i=0}^{N} \sum_{j=0}^{N} \psi_{ij} \ell_i(\xi) \ell_j(\eta), \tag{12}$$

where some of the basis coefficients are given by the prescribed boundary values. This enables us to compute partial derivatives easily via differentiation matrices, and we can evaluate all integrals with sufficient accuracy with GLL quadrature. Applying quadrature leads to the definition of the discrete version of $a(\cdot,\cdot)$,

$$a_N(\hat{v},\hat{w}) = \sum_{\alpha=0}^{N}\sum_{\beta=0}^{N} \rho_\alpha \rho_\beta \left((\hat{\nabla}\hat{v})^T \, G \, \hat{\nabla}\hat{w} \, g \right)\Big|_{\alpha\beta},$$

where $\rho_\alpha, \alpha = 0,\ldots,N$, are the GLL quadrature weights and the subscript $\alpha\beta$ means that we evaluate the integrand in the tensor-product GLL point (ξ_α, ξ_β). The discrete problem, in vector notation, is then: find $\boldsymbol{\varphi}_N \in \hat{X}_N^D$ such that

$$\boldsymbol{a}_N(\hat{\boldsymbol{v}}_N, \boldsymbol{\varphi}_N) = \mathbf{0} \qquad \forall \hat{\boldsymbol{v}}_N \in \hat{X}_N. \tag{13}$$

The boundary conditions are met by choosing the nodal values of $\boldsymbol{\varphi}_N$ (corresponding to the basis coefficients in (12)) to be interpolation points on the boundary $\partial\Omega$.

By applying discretization to the iterative scheme (11) we arrive at an algebraic system

$$A^n \Delta\boldsymbol{\phi}^{n+1} = -A^n \boldsymbol{\phi}^n \tag{14}$$

where A^n is the discrete, linearized Laplace-Beltrami operator, $\boldsymbol{\phi}^n$ is a vector containing the nodal values of $\boldsymbol{\varphi}_N$ at iteration level n, and $\Delta\boldsymbol{\phi}^{n+1}$ is a vector of the change in the nodal values of $\boldsymbol{\varphi}_N$. Since the bilinear form (10) is SPD, the matrix A^n is SPD, and the system is readily solved with CG iterations.

3.1 Mesh construction

Since the Lagrangian interpolants satisfy $\ell_j(\xi_i) = \delta_i^j$ at the GLL points, the basis coefficients in (12) represent the nodes on a curvilinear mesh on the numerical surface. In the context of polynomial interpolation, i.e., if $\psi_N = I_N\psi$ for a given function ψ, then the mesh nodes are defined by evaluating ψ in a predefined set of interpolation points,

$$\psi_{ij} = \psi(\xi_i, \xi_j),$$

in our case the tensor-product GLL points. In interpolation of parametric surfaces, each parametric function is interpolated separately. From basic interpolation theory we know that for a scalar function $\hat{u} \in H^\sigma(\hat{\Omega})$ the interpolation error is bounded by [22]

$$||\hat{u} - I_N\hat{u}||_{L^2(\hat{\Omega})} \leq CN^{-\sigma}||\hat{u}||_{H^\sigma(\hat{\Omega})}, \tag{15}$$

where N is the polynomial degree and C is a constant. In our case this holds for each of the parametric functions. Hence, the accuracy of the interpolation of the surface depends on the regularity of the parametric functions. If the mesh points stem from interpolation of parametric functions of low regularity, a poor approximation results, even if the surface is smooth. However, a parametric surface can always be *reparametrized*, and some parametrizations may be better suited for polynomial approximation than others [23]. A mesh constructed from interpolation of analytic parametric functions gives a very good approximation of the surface (with exponential convergence rate).

As an example, consider the catenoid, which is a minimal surface and hence smooth. A natural parametrization of a catenoid of height H and “waist” radius R_m (radius at the midpoint between the two boundary circles) is

$$\begin{aligned} \varphi_1(\xi,\eta) &= R_m \cosh\left(\frac{H\eta}{2R_m}\right)\cos(\pi\xi), \\ \varphi_2(\xi,\eta) &= R_m \cosh\left(\frac{H\eta}{2R_m}\right)\sin(\pi\xi), \\ \varphi_3(\xi,\eta) &= \frac{H}{2}\eta, \end{aligned} \tag{16}$$

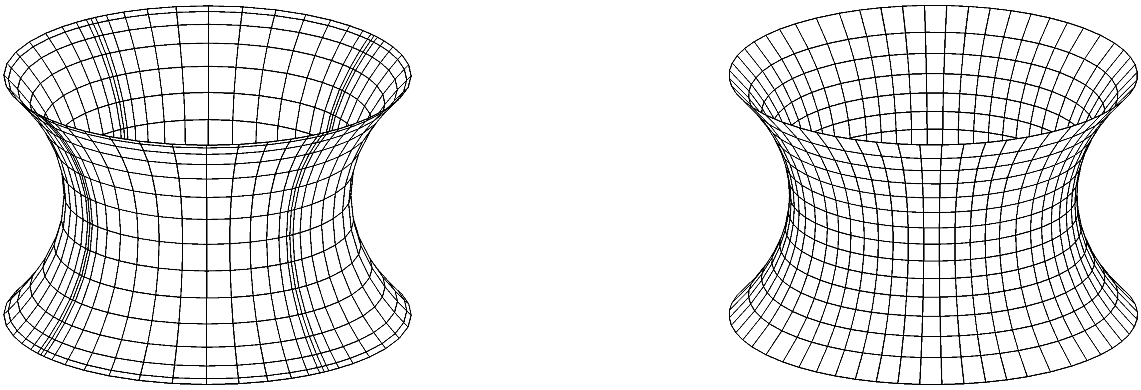
where $-1 \leq \xi, \eta \leq 1$. Here, all functions are smooth and convergence will be exponential as a function of the polynomial degree N. On the other hand, consider a reparametrization

$$\tilde{\boldsymbol{\varphi}}(\xi,\eta) = \boldsymbol{\varphi}(\theta(\xi),\theta(\eta)) \tag{17}$$

where

$$\theta(t) = \frac{\arcsin t}{\arcsin 1}.$$

Obviously, $\tilde{\boldsymbol{\varphi}}$ represents the same surface, since θ is a bijective mapping from $[-1,1]$ onto itself. However, the low regularity of $\arcsin(t)$ ruins the regularity of the parametric functions and causes low order algebraic convergence in the polynomial interpolation according to (15). Figure 1 shows the difference between the meshes generated by interpolating the two parametrizations when we use four spectral elements and a polynomial degree $N = 15$. We remark that the extension to consider spectral elements instead of a pure spectral method is standard and straightforward.



(a) Interpolation of (16)

(b) Interpolation of (17)

Figure 1: The two meshes are generated from interpolating (16) and (17) in the tensor-product GLL points using four spectral elements.

In order to measure the interpolation error we need a metric for computing the distance between two surfaces. The Hausdorff metric [24] applies to general surfaces in $\mathbb{R}^3$, but is not very well suited for numerical computation. We define a similar metric, customized for rotationally symmetric surfaces. At each point, we measure the distance between the exact and the numerical surface in the radial direction relative to the symmetry axis. The radius of the catenoid is given by the catenary function

$$R(z) = R_m \cosh\left(\frac{z}{R_m}\right), \tag{18}$$

assuming the z-axis is the symmetry axis. The error is then defined as the supremum of all the radial distances,

$$||\boldsymbol{\varphi} - \boldsymbol{\varphi}_N|| = \sup_{\xi,\eta\in[-1,1]} \left|\sqrt{\varphi_{1,N}(\xi,\eta)^2 + \varphi_{2,N}(\xi,\eta)^2} - R(\varphi_{3,N}(\xi,\eta))\right|, \tag{19}$$

where $\varphi_{i,N}(\xi,\eta)$, $i = 1,2,3$, are the components of $\boldsymbol{\varphi}_N(\xi,\eta)$. In practice (in the pure spectral case), the supremum is taken over the tensor-product GLL points on a very fine grid; the extension to the spectral element case also includes a supremum over all the elements.

The two different parametrizations (16) and (17) are compared by interpolating them with polynomials in the tensor-product GLL points with four spectral elements. We define the height of the catenoid to be $H = 2$, such that the radius of the boundary circles are $R = R_m \cosh \frac{1}{R_m}$. The catenoid is only stable if $R/H > 0.755$ [1]; we safely choose $R = 1.6$.

As predicted, (16) yields exponential convergence, whereas (17) yields slow algebraic convergence; see Figure 2.

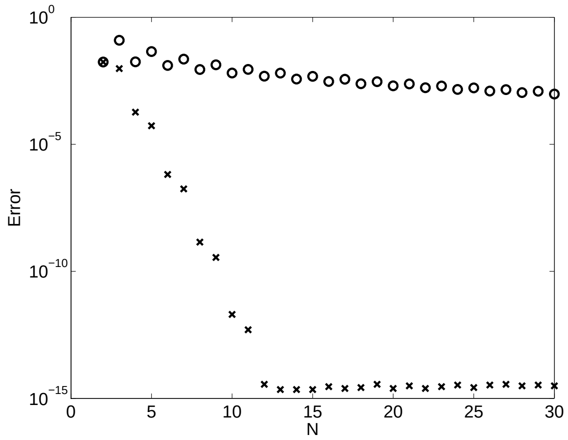


Figure 2: The two different parametrizations (16) and (17) are interpolated in the tensor-product GLL points using four spectral elements. The interpolation error is measured in a discrete version of the norm (19). The analytic parametrization (16) yields exponential convergence (×), while the parametrization of low regularity (17) yields low order algebraic convergence (∘).

Our main objective is not to interpolate a given surface since we do not assume a priori knowledge of the exact solution. However, the solution of (13) will be a polynomial approximation of the catenoid (although not necessarily an interpolation), and then the results from this section will serve as a reference.

3.2 Mesh update algorithms

In our iterative scheme the mesh changes at each iteration level as we move from one surface to the next. The iterations can be stated as a two-step algorithm:

1. Solve (14) for $\Delta\boldsymbol{\phi}^{n+1}$.

2. Update the geometry accordingly.

The most straightforward implementation of the second step is

$$\boldsymbol{\phi}^{n+1} = \boldsymbol{\phi}^n + \Delta\boldsymbol{\phi}^{n+1}. \tag{20}$$

However, this is not the only option, as we can choose to add small tangential components to $\Delta\boldsymbol{\phi}^{n+1}$ to obtain a different mesh in the next configuration $\boldsymbol{\varphi}_N^{n+1}$. This can be used to control the distribution of the mesh nodes during the iterations and retain a "good" mesh.

Our main problem is that we do not know the exact surface we are approximating until the iterations have converged, so we do not know which mesh gives us the best representation of the next state of the surface. Retaining an optimal mesh in an evolutionary geometry is a very complicated and generally unsolved problem [25]. It makes it even more difficult that the problem of optimal representation of a given stationary parametric surface remains unsolved. Numerical investigation of the problem is made difficult by the dearth of geometrically interesting evolutionary surface problems for which the exact solutions are known at all times.

We do not claim to solve any of these problems in this paper, but we will compare three different mesh update algorithms which highlight some of the important aspects of evolutionary surfaces. One algorithm is the straightforward one (20), which we will refer to as the *Lagrangian update*. A heuristic argument for choosing this algorithm is as follows: minimal surfaces are smooth, and it seems reasonable that the transition from a smooth initial surface to a minimal surface is smooth (imagine a deforming soap film with no topological changes). If the representation of the initial surface is good, then hopefully the representation of the final stationary surface will also be good. The algorithm is simple since it does not require any post-processing after solving (14).

A second algorithm is defined by removing all tangential components from $\Delta\boldsymbol{\phi}^{n+1}$ (where the tangents are computed numerically based on $\boldsymbol{\varphi}_N^n$) and moving the mesh nodes in a direction normal to the current surface Ω^n. This makes sense if one assumes not too large distortions of the surface during the iterations; if the next surface is very similar to the current one, then the optimal representations of the two are most likely very similar too. It is implemented by finding the unit normal vector at each mesh point (numerically) and then project the update $\Delta\boldsymbol{\phi}^{n+1}$ onto these vectors. The algorithm is also discussed in [25].

The third algorithm is specifically designed for our test case. It is based on the observation that for a non-minimal surface with the same boundary as the catenoid (e.g., a cylinder), the evolution of the surface will mostly consist of a motion in a direction radially to the symmetry axis. In addition, a natural representation of rotationally symmetric surfaces is using an affine mapping along the rotation axis, as in (16). To retain this from the initial surface, we restrict the mesh update to the radial direction by projecting $\Delta\boldsymbol{\phi}^{n+1}$ onto the unit radial vector. If the initial mesh corresponds to an affine mapping along the z-axis, so will the final configuration.

3.3 Comparison with mean curvature flow

Finding minimal surfaces can also be done by solving the *time-dependent* PDE

$$\frac{\partial\boldsymbol{\varphi}}{\partial t} = \Delta_\Omega\boldsymbol{\varphi}, \tag{21}$$

over a large time interval $[T_0, T]$. If the solution reaches a steady state within $t = T$, then that is necessarily also a solution to (2) and hence a minimal surface. This problem is called *mean curvature flow*, since the time-derivative of the solution points in the direction of the mean curvature. It has been studied numerically with a finite element method in [26].

A numerical treatment of (21) with a spectral element method will involve much the same ingredients as we have seen in the previous sections. The starting point is a weak formulation of the PDE, and spatial discretization is applied based on high order polynomial representations. This results in the semi-discrete system

$$\frac{\partial}{\partial t} B\boldsymbol{\phi} = -A\boldsymbol{\phi},$$

where B is the mass matrix and A is the discrete Laplace-Beltrami operator. We would prefer to treat this problem with an implicit time integration method, due to the step restrictions induced by A. However, we have the same problem with the nonlinear factors G and g as before, so in order for the system to be solvable with CG iterations, these terms must be treated explicitly. Hence the system will never be fully implicit. Actually, this imposes a relatively severe step restriction which makes the method inefficient if we are only interested in steady state solutions.

There is also another drawback with the time-dependent problem compared to our iterative scheme, namely the lack of control over the numerical mesh. In mean curvature flow $\boldsymbol{\phi}^{n+1}$ is fully determined by the algebraic system we solve at each time-step, and we may have to re-mesh in order to avoid severely distorted meshes and possible breakdowns.

4 Numerical Results

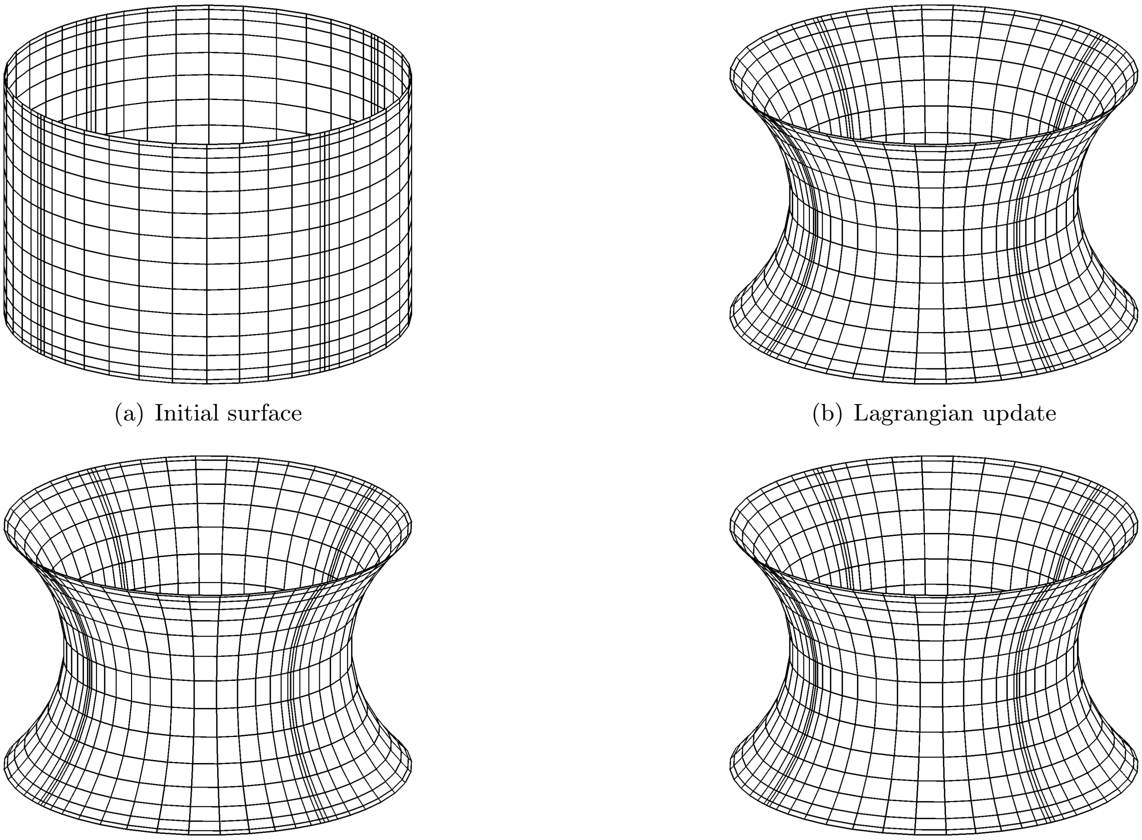
The catenoid represents a good test case for an iterative minimal surface algorithm because it is one of the few geometrically non-trivial problems for which an exact solution is known. The radius is given as a function of the spatial variable on the rotational axis in (18). In order to make the results comparable to those in Figure 2, we use the same height $H = 2$ and radius $R = 1.6$.

The iterative scheme requires the definition of an initial surface. A natural starting point is the cylinder with radius R. This surface is most naturally parametrized by trigonometric functions for x and y and an affine mapping for z. This is equivalent to “patching” the reference domain on the cylinder wall, which retains something very similar to the tensor-product GLL mesh from the reference domain. We use four spectral elements, which can be recognized in the mesh-structure in Figure 3(a).

The chosen parametrization of the initial surface consists of analytic functions and is very suitable for polynomial interpolation. It is also relatively similar to a good parametrization of the catenoid (16), so if the chosen mesh update algorithm yields small distortions of the mesh, then we can expect something close to an optimal polynomial representation of the catenoid at steady state.

The difference between the different mesh update algorithms is hardly visible in the steady state solutions shown in Figure 3. Still, there is a significant difference in the convergence rate for the different algorithms; see Figure 4. The radial update algorithm will by construction end up in almost exactly the mesh defined by interpolating (16), and therefore converges with approximately the same speed. The plain Lagrangian update algorithm needs twice the polynomial degree to reach the same level of accuracy, while the normal vector update algorithm needs three times the polynomial degree. The relatively poor performance of the latter is caused by a slight movement of mesh points towards the boundary circles, distorting the affine mapping $\varphi_{3,N}(\xi, \eta)$.

However, there is a problem with the Lagrangian update algorithm that is not shown in Figure 4: the algorithm is unstable. After the steady state is reached, small numerical errors in the solution of (10) keep causing small perturbations in the mesh, and after a while the mesh becomes so distorted that the surface evolves *away* from the steady state. This is illustrated in Figure 5, which displays the error as a function of the iteration number at a fixed polynomial degree $N = 15$. We see that all three update algorithms converge at


(a) Initial surface

(b) Lagrangian update

(c) Update in normal direction

(d) Update in radial direction

Figure 3: The initial surface (a) is a cylinder, represented using four elements and a polynomial degree $N = 15$. The steady state numerical solutions are displayed for the three different mesh update algorithms: (b) Lagrangian updates (20); (c) update in the direction of the normal vector; and (d) update in the radial direction towards the rotation axis.

approximately the same rate, reaching the steady state within 100 iterations; the number of iterations needed depends on the level of accuracy reached. The solutions corresponding to the normal and radial update algorithms stay at the steady state, and the size of the updates $\Delta\varphi^{n+1}$ remain at machine precision level. The Lagrangian update algorithm, on the other hand, sees an *increase* in the error from around $n = 200$, and from there it continues to increase until the surface collapses. This behavior is also seen for other values of N.

The first example had a clear symmetry in the mapping of the initial surface. To show that the results do not depend on this symmetry, we repeat the numerical experiment, but with element boundaries spiraling around the cylinder; see Figure 6. It is important to note that this “twisting” of the cylinder must be done in a smooth fashion; if the element boundaries cannot be represented by parametric functions of high regularity, then the representation of the entire surface will suffer. By twisting the cylinder like a spiral with a constant “angle of rotation”, we retain a smooth parametrization.

The similarity between the cylinder and the catenoid again makes the radial update algorithm the best alternative. Figure 7 shows the same relation between the mesh update algorithms as we saw with the plain parametrization of the cylinder in Figure 4. Note that the Lagrangian update algorithm is still unstable with this new mesh configuration.

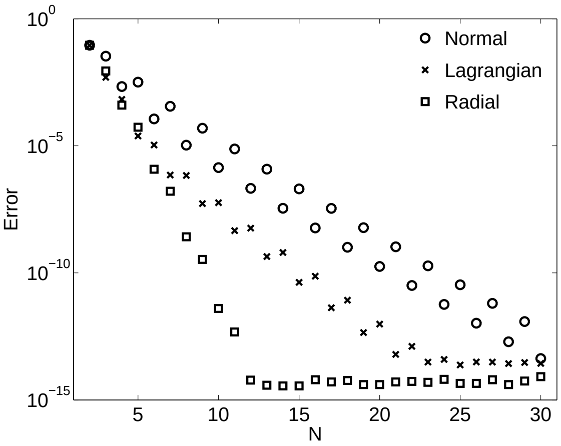


Figure 4: Error in the stationary solution, measured in a discrete version of the norm (19), as a function of the polynomial degree, N. The initial state is a cylinder. Different mesh update algorithms yield different levels of accuracy for a given N.

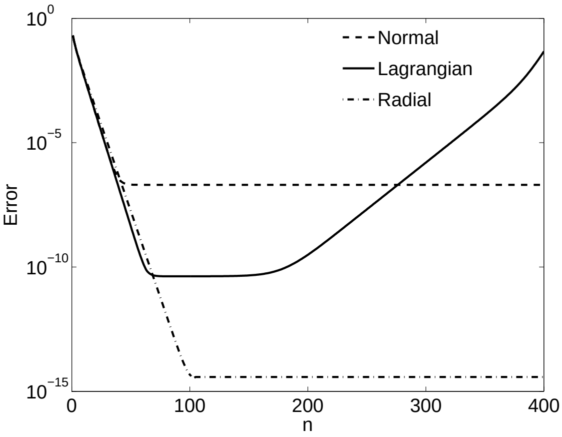
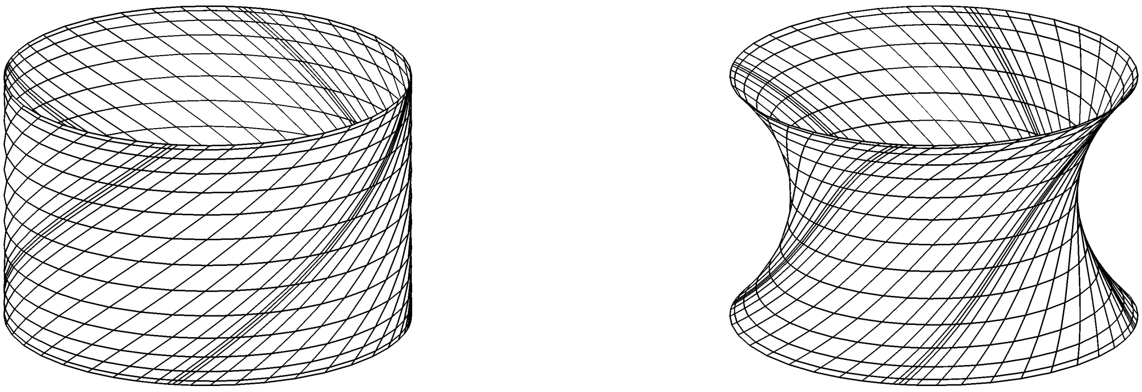


Figure 5: Error in the stationary solution, measured in the same norm, but now as a function of the iteration level n at a fixed polynomial degree $N = 15$. The normal and radial mesh update algorithms are stable at steady state, whereas the plain Lagrangian update algorithm yields small perturbations of the mesh at steady state, which after many iterations cause large mesh distortions.

We now investigate the impact of starting "further away" from the minimal surface (in terms of the norm (19)). Let the initial surface be the rotational surface with radius $R(z) = 1.6 + \frac{1}{2}(1+z)(1-z)$. This yields a surface that resembles a sphere with parts of the upper and lower hemispheres cut off; see Figure 8(a). The parametrization includes an affine mapping $\varphi_{3,N}(\xi, \eta)$, meaning that the radial update scheme should converge at exactly the same speed as before. On the other hand, the normal update scheme will probably be affected, since the normal vectors on the initial surface are no longer horizontal. Figure 8(b) confirms this, showing that the mesh nodes have been displaced vertically during the iterations. This also affects the error in the steady state solution; see Figure 9. The normal update algorithm seems to stop converging when the error caused by the mesh distortion becomes dominant. The Lagrangian update algorithm converges at the same rate as before, but is still unstable.

Finally, we show an example of a case where the minimal surface is not known analytically. The boundary curves are now distorted circles, with different radii, not concentric



(a) Initial surface

(b) Minimal surface

Figure 6: The initial surface mesh is "twisted" such that each element boundary spirals along the cylinder wall. The final state is a catenoid with a "twisted" mesh.

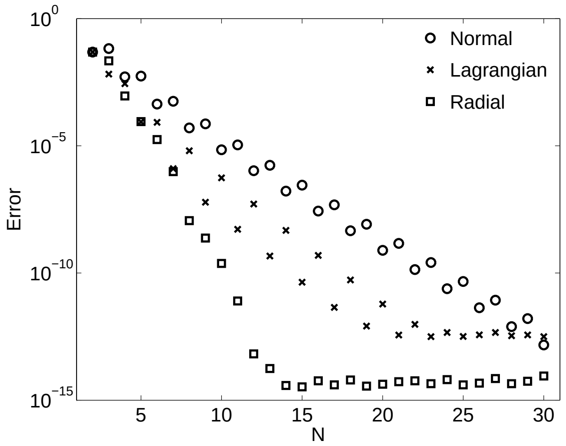


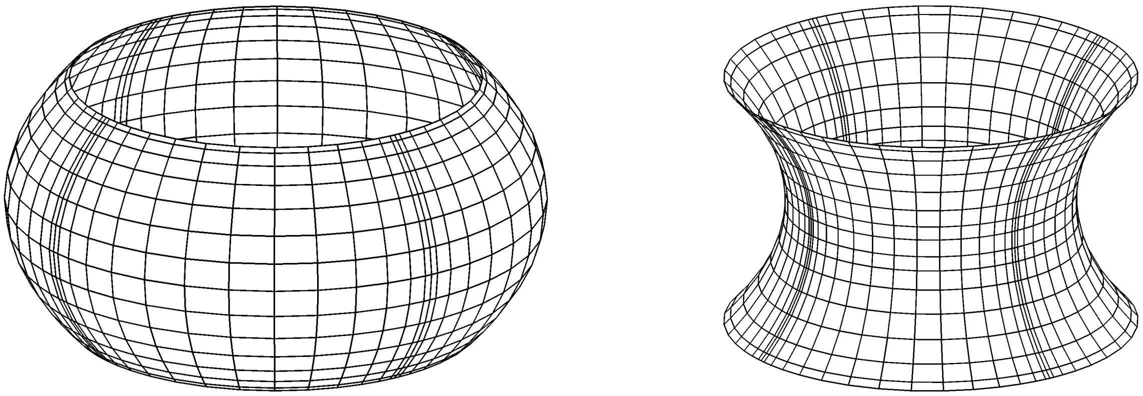
Figure 7: Error in the steady state solution for the catenoid with a "twisted" mesh. The performance of the different mesh update algorithms is almost exactly the same as for the plain parametrization; see Figure 4.

and oriented in different directions. The initial surface is an interpolation with straight lines between the boundary curves, making it somewhat like a distorted cylinder; see Figure 10. The minimal surface looks like a distorted catenoid.

5 Conclusions

An algorithm for finding a high order polynomial approximation of a minimal surface with a given boundary has been introduced. It is based on a weak formulation of the Laplace-Beltrami problem. The problem is nonlinear, so a linearization and iterative scheme is applied to solve the discrete problem. The algorithm does not handle topological changes.

As all minimal surfaces are smooth, we are able to achieve exponential convergence as the polynomial degree increases. The convergence rate, however, depends on the structure of the mesh points in the solution. Our algorithm allows freedom in how the mesh is moved during the iterations, and a good mesh update algorithm is needed to retain a smooth mapping between the two-dimensional reference domain and the three-dimensional surface; this is essential in order to obtain rapid convergence. We present three different mesh



(a) Initial surface

(b) Steady state solution with the normal update algorithm

Figure 8: The initial surface resembles part of a sphere; the radius is a parabola as a function of z. The surface normals on this surface have a non-zero component in the z direction, as opposed to the surface normals on the cylinder. When running the iterative scheme with the normal update algorithm, this changes the vertical distribution of mesh nodes, distorting the affine mapping $\varphi_{3,N}(\xi, \eta)$ quite severely.

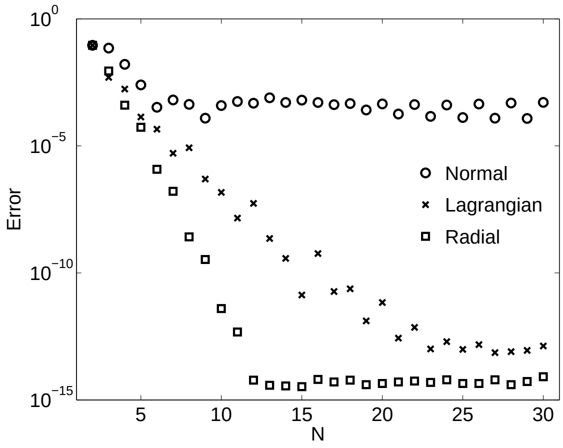
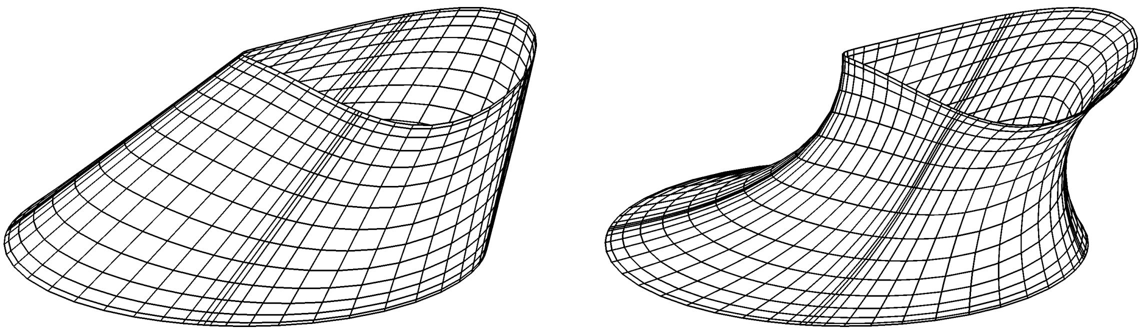


Figure 9: Error in the steady state solution when the initial condition is the surface displayed in Figure 8(a). The Lagrangian and radial update algorithms yield the same performance as before, but the normal update algorithm now seems to stabilize on an error of magnitude 10^{-4}.

update algorithms, which can all be applied to problems in an ALE setting. Numerical results show that naive Lagrangian updates are not always optimal, and neither are updates in the direction of the surface normal. The Lagrangian updates also have stability problems. In the case of the catenoid, a tailored algorithm with mesh motion normal to the rotational axis gives the fastest convergence in all three test cases.

Finding a general interface tracking algorithm that works better than the (relatively naive) existing ones is a subject of further work. One such algorithm has been suggested in [25]. In order to be able to even evaluate the quality of such algorithms, work must be done in finding the optimal polynomial representation of a given parametric surface or parametric curve. An effort to find good high order polynomial interpolations of parametric curves can be found in [23], but this effort still represents work in progress.



(a) Initial surface

(b) Minimal surface

Figure 10: Example of a minimal surface problem with analytically unknown exact solution. The normal mesh update algorithm is applied in the computation of the numerical approximation of the minimal surface.

Acknowledgment

The work has been supported by the Research Council of Norway under contract 185336/V30. The authors would like to thank Alf Emil Løvgren for helpful discussions on the derivation of the weak form of the Laplace-Beltrami operator.

References

[1] P.-G. De Gennes, F. Brochard-Wyart, D. Quere, Capillarity and wetting phenomena: drops, bubbles, pearls, waves, Springer Verlag, 2004.

[2] C. Isenberg, The science of soap films and soap bubbles, Dover Publications, Inc., 1992.

[3] R. Osserman, A survey of minimal surfaces, Dover Publications, Inc., 2002.

[4] J. A. F. Plateau, Statique expérimentale et théorique des liquides soumis aux seules forces moléculaires, Gauthier-Villars, 1873.

[5] J. Douglas, Solution of the problem of Plateau, Proceedings of the National Academy of Sciences 16 (1930) 242–248.

[6] T. Radó, On Plateau's problem, Annals of Mathematics 31 (3) (1930) 457–469.

[7] P. Concus, Numerical solution of the minimal surface equation, Mathematics of Computation 21 (99) (1967) 340–350.

[8] D. Greenspan, On approximating extremals of functionals–II theory and generalizations related to boundary value problems for nonlinear differential equations, International Journal of Engineering Science 5 (7) (1967) 571–588.

[9] A. R. Elcrat, K. E. Lancaster, On the behavior of a non-parametric minimal surface in a non-convex quadrilateral, Archive for Rational Mechanics and Analysis 94 (3) (1986) 209–226.

[10] R. H. W. Hoppe, Multigrid algorithms for variational inequalities, SIAM Journal on Numerical Analysis 24 (5) (1987) 1046–1065.

[11] K. A. Brakke, The surface evolver, Experimental Mathematics 1 (2) (1992) 141–165.

[12] M. Hinata, M. Shimasaki, T. Kiyono, Numerical solution of Plateau's problem by a finite element method, Mathematics of Computation 28 (125) (1974) 45–60.

[13] H. J. Wagner, A contribution to the numerical approximation of minimal surfaces, Computing 19 (1) (1977) 35–58.

[14] C. Coppin, D. Greenspan, A contribution to the particle modeling of soap films, Applied Mathematics and Computation 26 (4) (1988) 315–331.

[15] D. L. Chopp, Computing minimal surfaces via level set curvature flow, Journal of Computational Physics 106 (1993) 77–77.

[16] A. Huerta, A. Rodríguez-Ferran (eds.), The Arbitrary Lagrangian-Eulerian Formulation, Computer Methods in Applied Mechanics and Engineering 193 (39-41) (2004) 4073–4456.

[17] A. T. Fomenko, The Plateau Problem, Gordon and Breach Science Publishers, New York, 1990.

[18] M. Struwe, Plateau's Problem and the Calculus of Variations, Princeton University Press, Princeton, 1988.

[19] L. D. Landau, E. M. Lifshitz, Fluid Mechanics, Course of Theoretical Physics, Volume 6, Butterworth-Heinemann, 1987.

[20] L. W. Ho, A. T. Patera, Variational formulation of three-dimensional viscous free-surface flows: Natural imposition of surface tension boundary conditions, International Journal for Numerical Methods in Fluids 13 (1991) 691–698.

[21] E. Kreyszig, Differential Geometry, Dover Publications, Inc., 1991.

[22] C. Canuto, M. Y. Hussaini, A. Quarteroni, T. A. Zang, Spectral Methods, Fundamentals in Single Domains, Springer, 2006.

[23] T. Bjøntegaard, E. M. Rønquist, Ø. Tråsdahl, High order interpolation of curves in the plane, Tech. rep., Norwegian University of Science and Technology, http://www.math.ntnu.no/preprint/numerics/2009/N11-2009.pdf (2009).

[24] M. Ó. Searcóid, Metric spaces, Springer Verlag, 2006.

[25] T. Bjøntegaard, E. M. Rønquist, Accurate interface-tracking for arbitrary Lagrangian-Eulerian schemes, Journal of Computational Physics 228 (12) (2009) 4379–4399.

[26] G. Dziuk, An algorithm for evolutionary surfaces, Numerische Mathematik 58 (1) (1991) 603–611.