Journal of Computational Physics 230 (2011) 4795–4810

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

High order numerical approximation of minimal surfaces

Øystein Tråsdahl, Einar M. Rønquist *

Norwegian University of Science and Technology, Department of Mathematical Sciences, NO-7491 Trondheim, Norway

ARTICLE INFO

Article history:
Received 14 September 2010
Received in revised form 28 February 2011
Accepted 1 March 2011
Available online 9 March 2011

Keywords:
Minimal surfaces
Mean curvature
Free surface flow
Evolutionary surfaces
Mesh update techniques

ABSTRACT

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 few cases where the exact solution is known. 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 cases we may 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. The present work is also of relevance to high order numerical approximation of fluid flow problems involving free surfaces.

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 [1–3]. 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 Douglas [5] and 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].

For parametric surfaces, the minimal surface problem has been solved with finite element methods by Dziuk and Hutchinson [11,12], Brakke [13], Hinata et al. [14] and Wagner [15], whereas Coppin and Greenspan [16] use direct simulation of surface tension forces on a grid of marker particles. Chopp [17] 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.

Minimal surfaces are smooth provided that the boundary curve is smooth. Spectral (or spectral element) methods based on high order polynomials should therefore, in principle, be very suitable numerical methods for such problems. However, better algorithms are still needed in order for high order methods to reach their full potential for computing minimal

* Corresponding author. Tel.: +47 73 59 35 47; fax: +47 73 59 35 24.
E-mail address: ronquist@math.ntnu.no (E.M. Rønquist).

0021-9991/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2011.03.003

surfaces or for tracking time-dependent interfaces. We feel that the sensitivity to the choice of interpolation points have not been properly addressed in the literature before, and these challenges are particularly acute for high order methods.

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 [18], and the algorithms presented in this paper are also relevant in an ALE setting.

We conclude with numerical results showing the convergence properties of our method; these results are based on considering surfaces with analytically known solutions (the catenoid, the Scherk surface, and the Enneper surface). We also show examples of cases where the exact solution is unknown.

2. Problem formulation

Consider a two-dimensional surface Ω in $\mathbb{R}^3$ with a fixed boundary $\partial\Omega$, represented locally by a diffeomorphism $\boldsymbol{\varphi} : \widehat{\Omega} \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$. The mapping $\boldsymbol{\varphi}$ satisfies the following set of three partial differential equations [19]

$$\Delta_\Omega \boldsymbol{\varphi} = 2\kappa \boldsymbol{n}, \tag{1}$$

where κ is the mean curvature, $\boldsymbol{n}$ 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 following system of equations,

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

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, 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 [20,21]

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

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

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

where $\hat{u}$ is a *scalar* function defined on $\widehat{\Omega}$, 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 $\partial\widehat{\Omega}$ 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 $u \in Y^D \equiv \{v \in H^1(\Omega) | v|_{\partial\Omega} = u_0\}$ such that

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

where $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)^T$ 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 $\widehat{\Omega}$. An infinitesimal surface area $\mathrm{d}\Omega$ on the curved surface can be expressed in the reference variables as

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

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 $\widehat{\Omega}$ through the Jacobian $\widetilde{J}$ of $\boldsymbol{\varphi}^{-1}$. Written out, we have

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

where $\widehat{\nabla} = \left(\frac{\partial}{\partial \xi}, \frac{\partial}{\partial \eta}\right)^T$ 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

$$\int_\Omega (\nabla v)^T \nabla u \, \mathrm{d}\Omega = \int_{\widehat{\Omega}} (\widetilde{J}^T \widehat{\nabla} \hat{v})^T \widetilde{J}^T \widehat{\nabla} \hat{u} \; g \; \mathrm{d}\widehat{\Omega} = \int_{\widehat{\Omega}} (\widehat{\nabla} \hat{v})^T \widetilde{J} \, \widetilde{J}^T \widehat{\nabla} \hat{u} \; g \; \mathrm{d}\widehat{\Omega}.$$

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

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

The resulting integral can then be expressed as the bilinear form

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

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

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

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

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

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

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 $\boldsymbol{\varphi} = (\varphi_1, \varphi_2, \varphi_3)^T = (x, y, z)^T \in \widehat{X}^D \equiv \left\{\hat{\boldsymbol{w}} \in (H^1(\widehat{\Omega}))^3 | \hat{\boldsymbol{w}}|_{\partial\widehat{\Omega}} = \boldsymbol{\varphi}_0\right\}$ such that

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

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 also have relevance to the numerical treatment of such flows.

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 [22]

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

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

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

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

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

where v_i is a test function. This term includes both normal and tangential contributions. In [23] it was shown that this integral can be expressed as

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

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 [24]

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

where $g^{\alpha\beta}$ is the *contravariant metric tensor*, which in matrix notation is nothing but our matrix $G = (J^TJ)^{-1}$. 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

$$\int_{\widehat{\Omega}}\gamma\hat{v}_{i,\alpha}g_i^\alpha g\,\mathrm{d}\widehat{\Omega} = \int_{\widehat{\Omega}}\gamma\hat{v}_{i,\alpha}g^{\alpha\beta}\varphi_{i,\beta}g\,\mathrm{d}\widehat{\Omega} = \int_{\widehat{\Omega}}\gamma(\widehat{\nabla}\hat{v}_i)^TG\widehat{\nabla}\varphi_i g\,\mathrm{d}\widehat{\Omega} = a(\hat{v}_i,\varphi_i), \quad i = 1,2,3.$$

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 $k = 1$ replaced by $k = \gamma$. 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

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

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

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

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

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

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 [25]. For simplicity, we consider a pure spectral method here, i.e., $\widehat{\Omega} = (-1,1)^2$; the extension to spectral elements is straight-forward and standard. The relevant discrete function spaces are

$$\widehat{X}_N = \left\{ \hat{\boldsymbol{w}} \in H_0^1(\widehat{\Omega})^3 | \hat{\boldsymbol{w}} \in \mathbb{P}_N(\widehat{\Omega})^3 \right\},$$
$$\widehat{X}_N^D = \left\{ \hat{\boldsymbol{w}} \in H^1(\widehat{\Omega})^3 | \hat{\boldsymbol{w}} \in \mathbb{P}_N(\widehat{\Omega})^3 \quad \text{and } \hat{\boldsymbol{w}} = \boldsymbol{\varphi}_0 \quad \text{on } \partial\widehat{\Omega} \right\}.$$

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 $\widehat{X}_N$ or $\widehat{X}_N^D$, 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 $a(\cdot,\cdot)$,

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

where ρ_α, $\alpha = 0, \ldots, N$, 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 $\boldsymbol{\varphi}_N \in \widehat{X}_N^D$ such that

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

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 $\hat{u} \in H^\sigma(\widehat{\Omega})$ the interpolation error is bounded by [25]

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

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.

As an example, consider the catenoid which is a minimal surface. A natural parametrization of a catenoid of height H and "waist" radius R_m (radius at the midpoint between the two boundary circles) is [26]

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

where $-1 \leqslant \xi, \eta \leqslant 1$. However, consider also the alternative parametrization

$$\begin{aligned}\tilde{\varphi}_1(\xi,\eta) &= R_m \cosh\left(\frac{H\eta}{2R_m}\right)\xi,\\ \tilde{\varphi}_2(\xi,\eta) &= \pm R_m \cosh\left(\frac{H\eta}{2R_m}\right)\sqrt{1-\xi^2},\\ \tilde{\varphi}_3(\xi,\eta) &= \frac{H}{2}\eta,\end{aligned} \tag{17}$$

again with $-1 \leqslant \xi, \eta \leqslant 1$. It is easy to see that $\boldsymbol{\varphi}$ and $\tilde{\boldsymbol{\varphi}}$ represent the same surface. However, when we approximate them with polynomial interpolation, $\boldsymbol{\varphi}$ yields a GLL distribution in arc length of interpolation points along the two boundary circles, whereas $\tilde{\boldsymbol{\varphi}}$ yields a chord distribution [27]. Fig. 1 shows the two meshes generated by interpolating the two parametrizations in the case of using four spectral elements and a polynomial degree $N = 15$. In this case the surface is first decomposed into four deformed quadrilateral elements and the reference domain $(-1,1)^2$ is mapped to each of these four spectral elements.

In order to measure the interpolation error we consider the distance between two surfaces measured along the surface normal to one of the surfaces, in our case the interpolant. To find this distance requires an iterative procedure, but in cases where a non-parametric representation of the exact surface is known this is straightforward and can be accomplished with Newton's method. The interpolation error is then defined as

$$\|\boldsymbol{\varphi} - \boldsymbol{\varphi}_N\| = \left(\frac{\int_{\Omega_N} e_N^2 \mathrm{d}\Omega_N}{\int_{\Omega_N} \mathrm{d}\Omega_N}\right)^{1/2}, \tag{18}$$

where e_N is the distance along the surface normal from the interpolant to the exact surface. The integrals over the numerical surface Ω_N are evaluated using Gauss quadrature of high degree.

The two different parametrizations (16) and (17) are now compared by considering a multi-domain version based on 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 expected, both mappings yield exponential convergence; see Fig. 2. However, this example illustrates the sensitivity to the particular mapping used: despite the fact that the difference between the two grids is not noticeable, the convergence rate is quite different.

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

3.2. Mesh update algorithms

Our iterative scheme 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

$$\boldsymbol{\phi}^{n+1} = \boldsymbol{\phi}^n + \Delta\boldsymbol{\phi}^{n+1}. \tag{19}$$

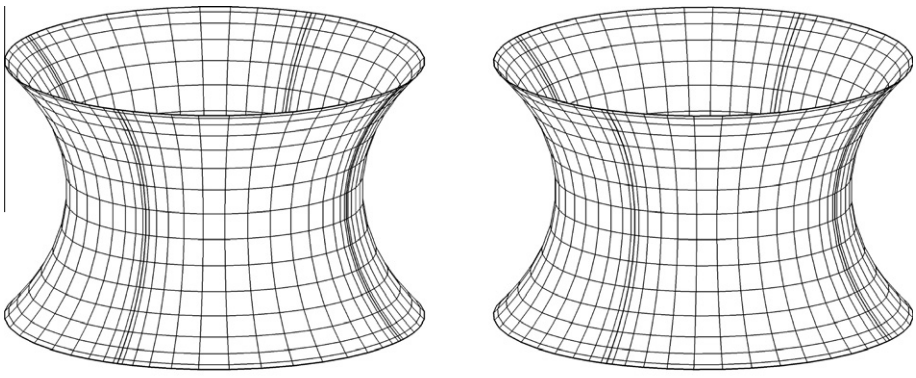

Fig. 1. The two meshes are generated by interpolating a multi-domain version of (16) and (17); here, four spectral elements are used. The meshes look almost the same.

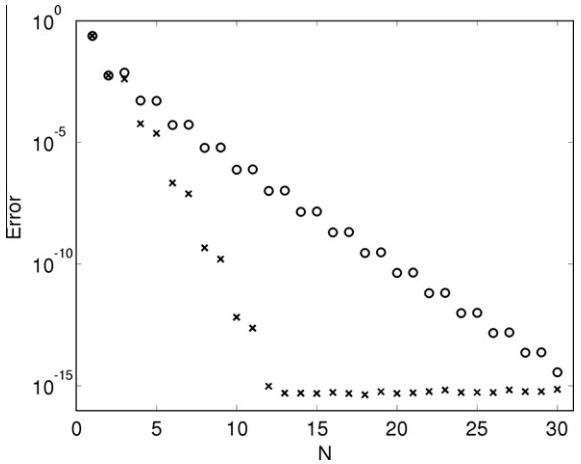

Fig. 2. A multi-domain version of the two different parametrizations (16) (×) and (17) (○) are interpolated in the tensor-product GLL points; here, four spectral elements are used. The interpolation error is measured in a discrete version of the norm (18). Both parametrizations yield exponential convergence, but there is a significant difference in the convergence rate.

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, i.e., a mesh that corresponds to a smooth mapping $\boldsymbol{\varphi}_N^{n+1}$.

Our main problem is that we do not know the surface we are approximating, 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 [28]. 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.

For the numerical results we present later, we will compare three different mesh update algorithms which highlight some of the important aspects of evolutionary surfaces and with particular relevance to high order discretization methods. One algorithm is the straightforward one (19), which we will refer to as the *Lagrangian update*. The method is the simplest of all since it does not require any post-processing after solving (14), but we have little control over the regularity of the resulting mapping.

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^n. This *normal update* approach can be motivated from the fact that it is the displacement in the normal direction which changes the shape of the surface (similar to the kinematic condition for free surface flows). It is implemented by finding the unit surface normal at each mesh point (numerically) and then projecting the update $\Delta\boldsymbol{\phi}^{n+1}$ onto these vectors. The algorithm is also discussed in [28].

The third algorithm can be viewed as a compromise between the two previous ones. If the normal component of $\Delta\boldsymbol{\phi}^{n+1}$ is larger than the tangential component in all the nodes on the computational surface, we do a Lagrangian update. Otherwise, we scale the tangential component everywhere by the largest factor such that the tangential component is never larger than the normal component. We denote this as a *restricted Lagrangian update*.

Besides these three, we will also consider a few special mesh update algorithms customized for the particular test case at hand.

3.3. Comparison with mean curvature flow

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

$$\frac{\partial\boldsymbol{\varphi}}{\partial t} = \Delta_\Omega\boldsymbol{\varphi}, \tag{20}$$

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 [29].

A numerical treatment of (20) 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 time 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 mesh. In mean curvature flow ϕ^{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

4.1. The catenoid

We first revisit the surface from Section 3.1, and use the same parameters H and R in order to make the results comparable to those in Fig. 2.

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, which yields a conformal mapping from the reference domain. We use four spectral elements, which can be recognized in the mesh-structure in Fig. 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 (see (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.

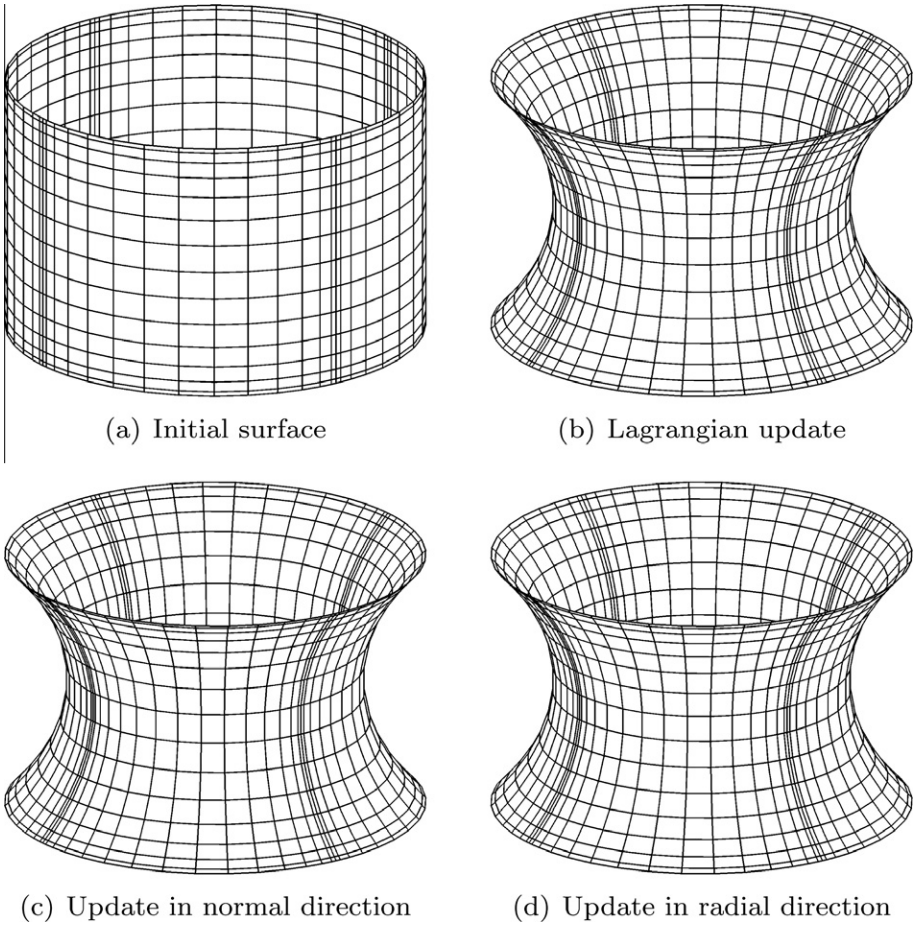

Fig. 3. The initial surface (a) is a cylinder represented using four spectral elements and a polynomial degree $N = 15$. The steady state solutions are displayed for the three different mesh update algorithms: (b) Lagrangian update (19); (c) update in the direction of the surface normal; and (d) update in the radial direction. The solution obtained using the restricted Lagrangian update algorithm is essentially the same as shown in (d).

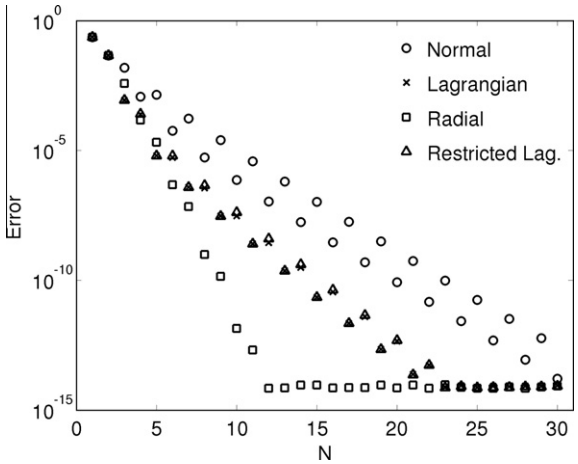

Fig. 4. Error in the steady state solution for the catenoid problem, measured in a discrete version of the norm (18), as a function of the polynomial degree, N. The initial state is a cylinder. Different mesh update algorithms yield different levels of accuracy for a given N.

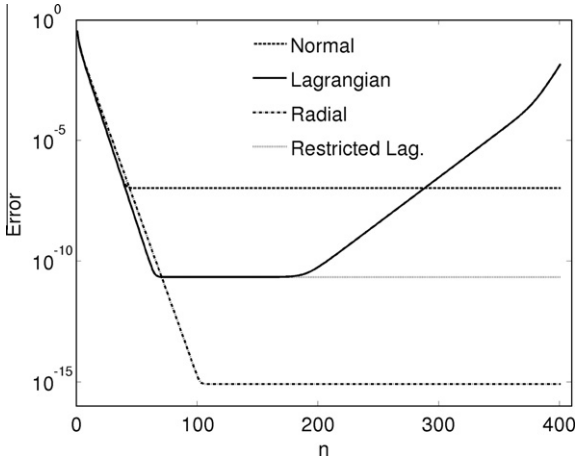

Fig. 5. Error in the stationary solution, measured in the same norm, but now as a function of the iteration level n at a fixed polynomial degree $N = 15$. The restricted Lagrangian, the normal, and the radial mesh update algorithms are all stable at steady state, whereas the pure Lagrangian update algorithm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

For the catenoid we also consider a customized mesh update algorithm: we restrict the mesh update to the radial direction by projecting $\Delta\boldsymbol{\phi}^{n+1}$ onto the unit radial vector towards the z-axis. Hence, the affine mapping along the z-axis of the initial surface will be retained during the iterations.

The difference between the different mesh update algorithms is hardly visible in the steady state solutions shown in Fig. 3. Still, there is a significant difference in the convergence rate for the different algorithms; see Fig. 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 rate. Both the pure and the restricted Lagrangian update algorithms need about twice the polynomial degree to reach machine precision, while the normal vector update algorithm needs about three times the polynomial degree. The relatively poor performance of the latter is caused by a slight movement of the mesh points towards the boundary circles, thus resulting in a nonaffine mapping $\varphi_{3,N}(\xi,\eta)$.

However, there is a problem with the pure Lagrangian update algorithm that is not shown in Fig. 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 the surface evolves *away* from the steady state. This is illustrated in Fig. 5, which displays the error as a function of the iteration number at a fixed polynomial degree $N = 15$. We see that all four update algorithms converge at 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 restricted Lagrangian, the normal, and the radial update algorithms remain at steady state, and the size of the updates $\Delta\boldsymbol{\varphi}^{n+1}$ remain at machine precision level. The pure 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.

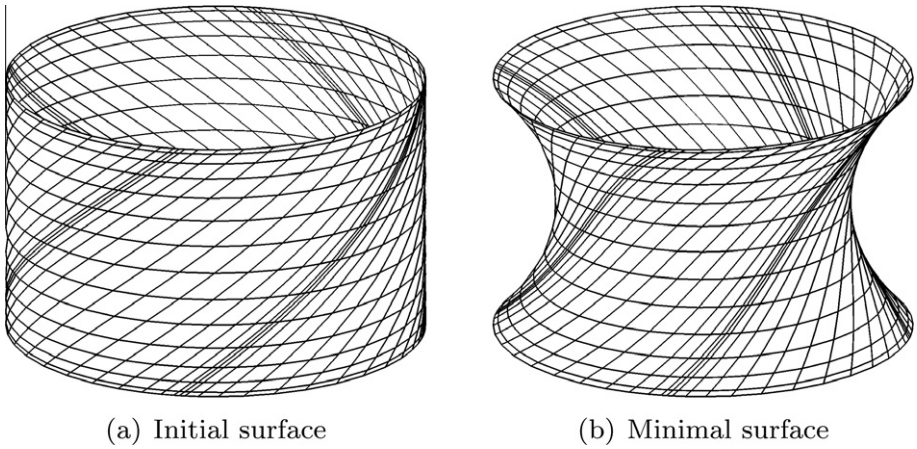

Fig. 6. The initial surface mesh is "twisted" such that each element boundary spirals along the cylinder wall. The steady state is a catenoid with a "twisted" mesh. The solution shown is obtained using the restricted Lagrangian mesh updates.

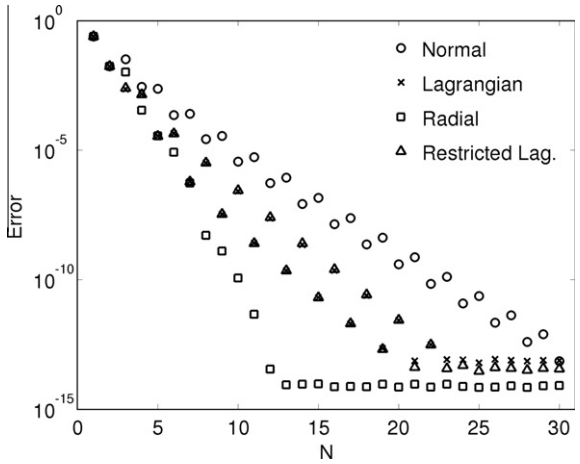

Fig. 7. Error in the steady state solution for the catenoid with a "twisted" mesh. The performance of the different mesh update algorithms is almost exactly the same as for the plain parametrization; see Fig. 4.

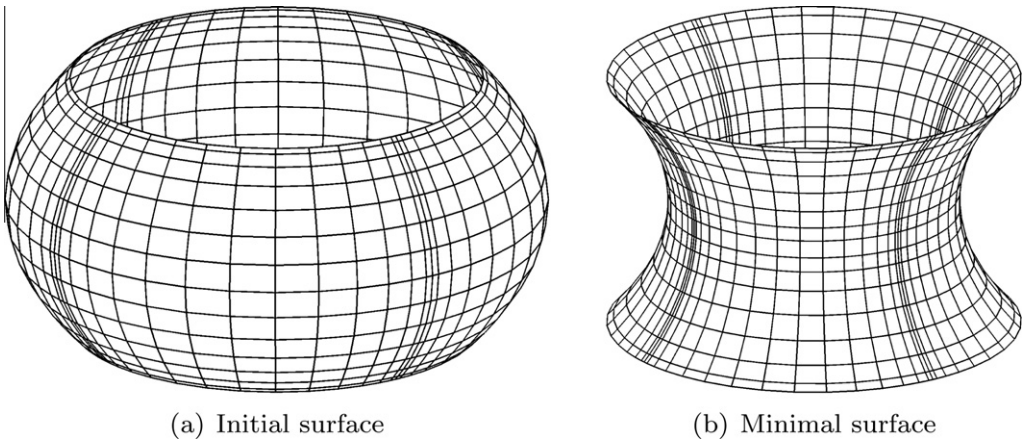

Fig. 8. (a) The initial surface resembles part of a sphere; the radius is a parabola as a function of z. The surface normals on this initial surface have a non-zero component in the z direction. (b) The minimal surface obtained with the normal update algorithm. The vertical component of the updates has changed the vertical distribution of the mesh nodes, resulting in a nonaffine mapping $\varphi_{3,N}(\xi,\eta)$.

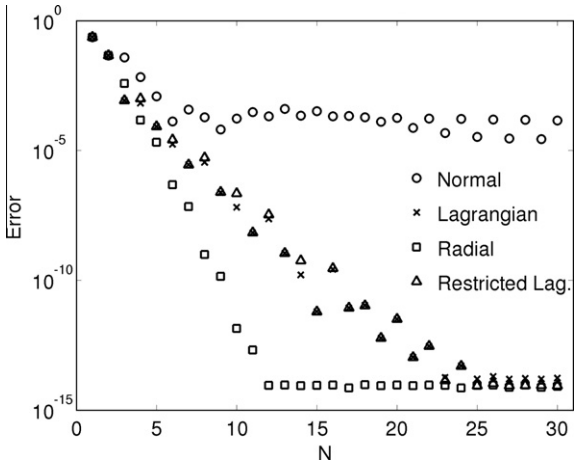

Fig. 9. Error in the steady state solution for the catenoid with an initial surface as displayed in Fig. 8(a). The Lagrangian and radial update algorithms yield the same performance as before, but the normal update algorithm now seems to stabilize on an error of magnitude 10^{-4}.

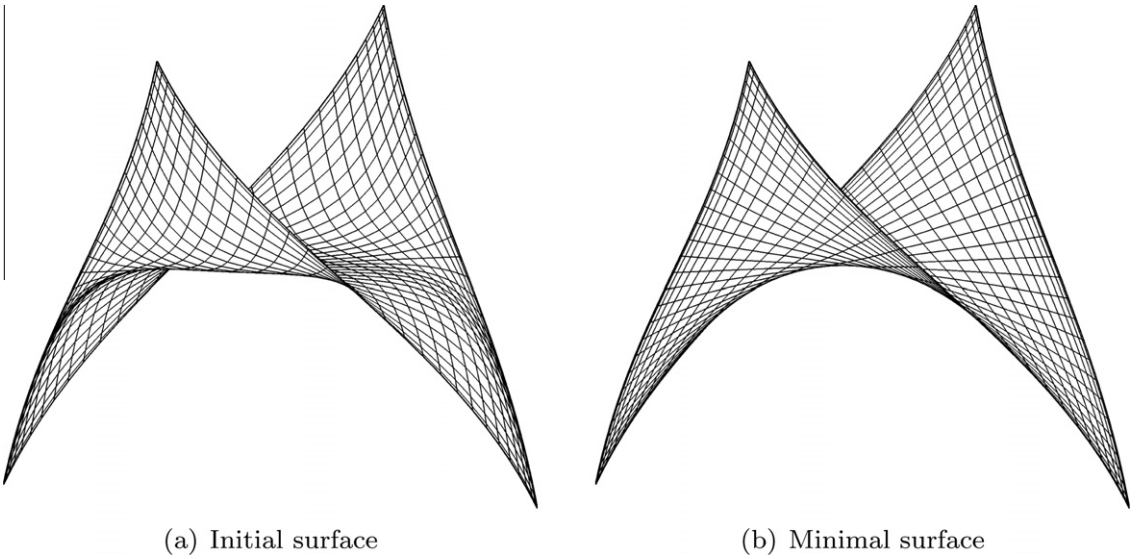

Fig. 10. Scherk's fifth surface, represented using a pure spectral method and a polynomial degree $N = 25$. The minimal surface is obtained using strictly vertical mesh updates, preventing severe mesh distortions.

This 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 Fig. 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. Fig. 7 shows the same relation between the mesh update algorithms as we saw with the plain parametrization of the cylinder in Fig. 4. Note that the Lagrangian update algorithm is still unstable with this new mesh configuration.

We now investigate the impact of starting "further away" from the minimal surface (in terms of the norm (18)). 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 Fig. 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, we expect the normal update scheme to be affected, since the normal vectors on the initial surface are no longer horizontal. Fig. 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 Fig. 9. The normal update algorithm seems to stop converging when the error caused

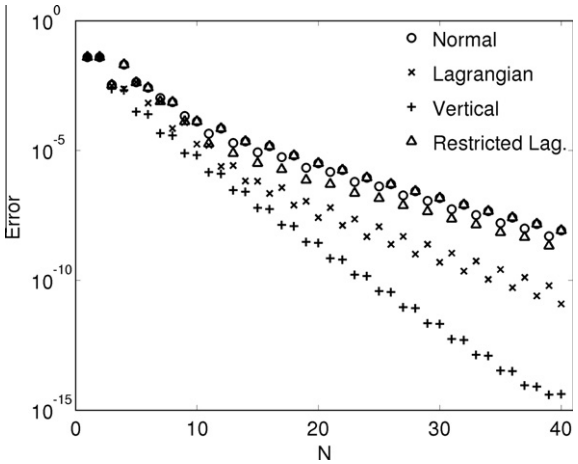

Fig. 11. Error in the steady state solution for Scherk's fifth surface measured by (18) and plotted as function of the polynomial degree N.

by the mesh distortion becomes dominant. The Lagrangian update algorithm converges at the same rate as before, but is still unstable.

4.2. Scherk's fifth surface

This surface can be represented non-parametrically by [26]

$$\sin(z) = \sinh(x)\sinh(y). \tag{21}$$

In the following, we consider this surface for x and y in the range -0.8 to 0.8. This allows us to represent the surface as a function $z(x,y)$ while at the same time avoiding $\frac{\partial z}{\partial x}$ and $\frac{\partial z}{\partial y}$ becoming unbounded and the corner angles going to zero. We represent this surface using a pure spectral method.

We start the iteration using the initial surface shown in Fig. 10. Note that the projection of the initial mesh to the xy-plane is still a (affinely mapped) tensor-product GLL mesh.

The Lagrangian, the restricted Lagrangian, and the normal mesh update algorithms are "general purpose" algorithms which can be used on any surface in our current framework. As for the catenoid, we also now construct a customized mesh update algorithm by restricting the updates to the z-direction. This ensures that the projected mesh will remain a tensor-product GLL mesh during the iterations. Since the minimal surface can be represented by an analytic function $z(x,y)$, it is well represented on such a mesh, and we can expect rapid exponential convergence for the customized mesh update scheme.

Fig. 11 shows the error in the steady state solution as measured by (18). As expected, the customized, strictly vertical mesh update algorithm yields exponential convergence, but we need a relatively high polynomial degree to reach machine precision; this is due to the fact that we use a pure spectral method for the entire surface. The multi-purpose mesh update techniques also yield exponential convergence, but the convergence rate is slower than the customized scheme. The Lagrangian update method is somewhat better than the normal update scheme. The reduction in convergence rate for the "general-purpose" methods is mainly due to a non-optimal mesh at steady state, which again is related to the particular initial surface. This example indicates the importance, as well as the sensitivity, of a good mapping between the reference domain $\widehat{\Omega}$ and the deformed surface Ω.

4.3. Enneper's surface

This surface can be parametrized by [26]

$$\begin{aligned}\varphi_1(u,v) &= u\left(1-\tfrac{1}{3}u^2+v^2\right),\\ \varphi_2(u,v) &= v\left(1-\tfrac{1}{3}v^2+u^2\right),\\ \varphi_3(u,v) &= u^2-v^2,\end{aligned} \tag{22}$$

where u and v are coordinates on a circular domain of radius R. For $R \leqslant 1$ the surface is stable and a global area minimizer, whereas for $1 < R < \sqrt{3}$ the given parametrization gives an unstable minimal surface. In the latter case there also exist two (symmetrically similar) stable minimal surfaces which are also global area minimizers. For $R \geqslant \sqrt{3}$ the boundary curve intersects itself.

We first consider the case $R = 0.8$, for which the solution is unique. The surface is represented numerically using 12 spectral elements; see Fig. 12. The initial surface is created by a cylindrical extension of the boundary curve.

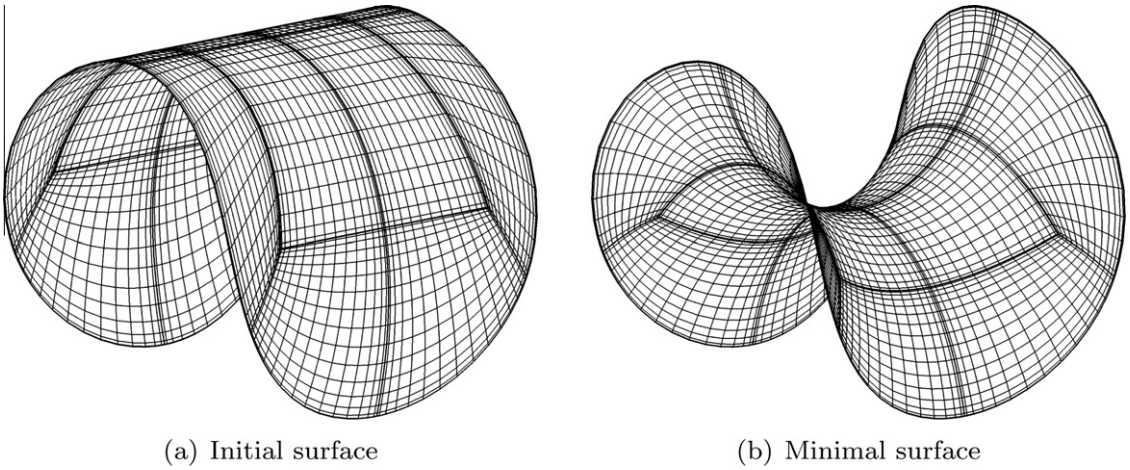

Fig. 12. Enneper's surface, represented using 12 spectral elements and a polynomial degree $N = 15$. The minimal surface is obtained using the restricted Lagrangian mesh updates.

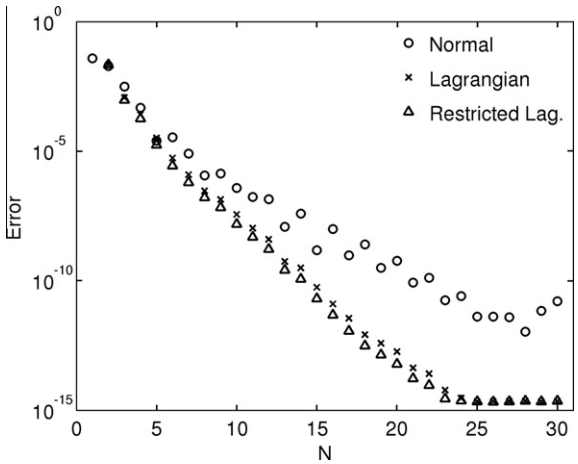

Fig. 13. Error in the steady state solution for the Enneper surface with $R = 0.8$ measured by the norm (18) and plotted as a function of the polynomial degree N.

The normal, the Lagrangian, and the restricted Lagrangian mesh update algorithms are again applied. The error, as measured by (18), is shown in Fig. 13. All mesh update algorithms yield exponential convergence, but the normal algorithm displays a slower and more uneven convergence rate.

For the Enneper surface, there is no particular symmetry that enables us to *a priori* determine a customized mesh update algorithm that will lead to a good mesh at steady state and hence an even more rapid convergence rate than observed with the "general-purpose" algorithms.

For $1 < R < \sqrt{3}$, we can obtain the two stable solutions and at the same time verify that the known solution (22) is unstable. We first use an interpolation of (22) to construct an initial surface. We then add random perturbations of order 10^{-10} to all the internal points of this surface and start the iterative algorithm. At steady state we end up in one of the two stable solutions. Fig. 14 shows all the three surfaces.

4.4. The trinoid

We conclude with two examples of geometrically more challenging minimal surfaces which we are able to obtain. The first is the Jorge–Meeks trinoid [26], which has three circles as boundary curves and looks like three catenoids attached at the end. Similarly to the catenoid, it collapses when the radius of the boundary circles is too small compared to the distance between them. The initial surface is *not* three cylinders, but rather three conical cylinders that interpolate linearly between the boundary curves. Fig. 15 shows the minimal surface obtained using the restricted Lagrangian update algorithm.

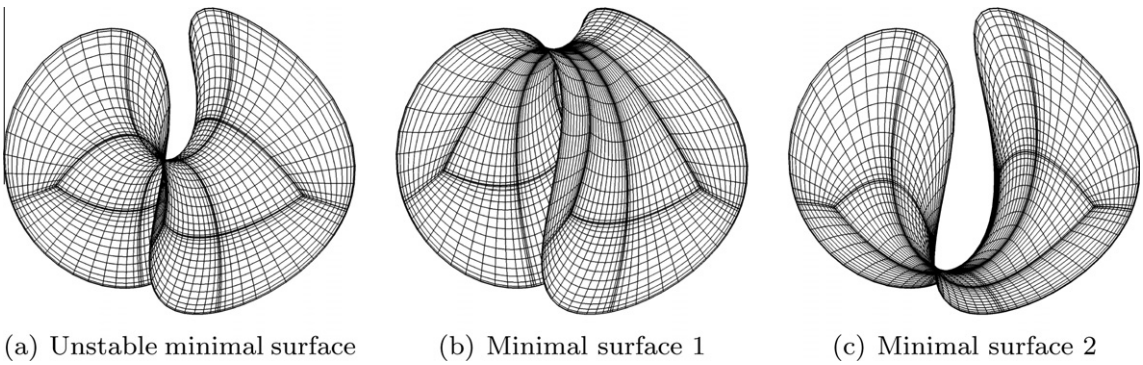

Fig. 14. The three minimal surfaces in the Enneper case for $R = 1.2$. The unstable solution (a) is known analytically (see (22)), and is depicted by interpolating this known parametrization. The two other solutions are stable and global-area minimizers. The surfaces (b) and (c) are here obtained using the restricted Lagrangian mesh update algorithm, starting from slightly perturbed versions of (a) (by adding random perturbations on the order of 10^{-10}).

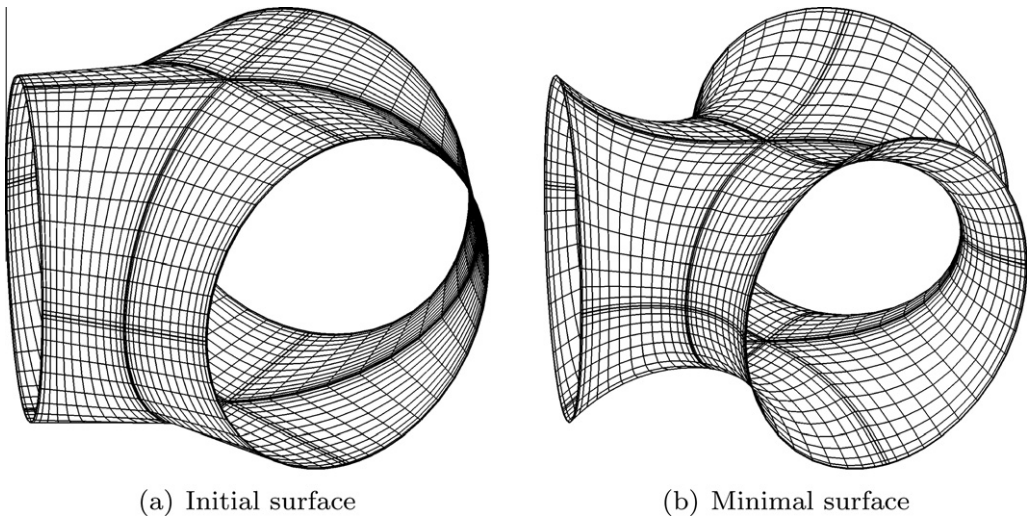

Fig. 15. The trinoid represented using 12 spectral elements.

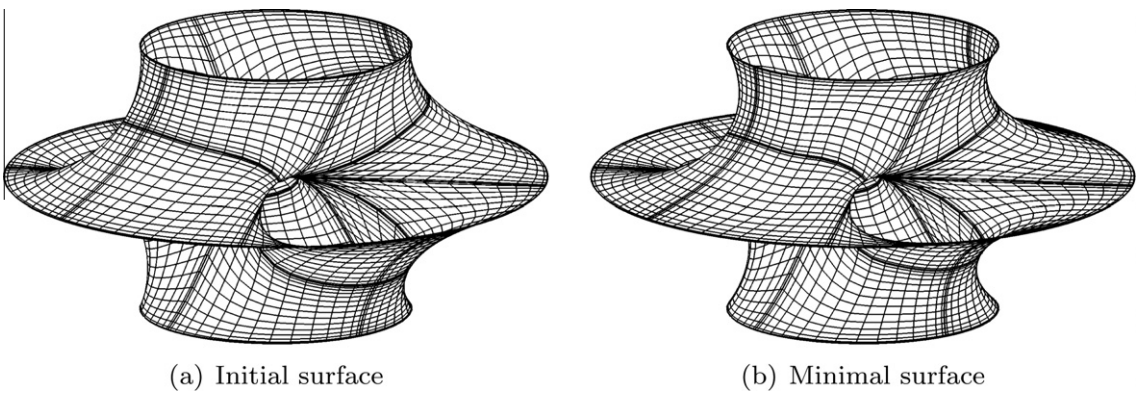

Fig. 16. The modified Costa surface represented using 16 spectral elements.

4.5. The Costa surface

The last surface is a variation (or a simplification) of the Costa surface [26], which is one of the more exotic minimal surfaces and discovered as late as 1982. Here, the radius of the two small boundary circles is $R = 1$, while the radius of the large

circle in the middle is $R = 2$. The vertical distance between the two small boundary circles is $H = 2$. We remark that this is not the exact Costa surface, since the large circle is a plane curve in our case, while the exact Costa surface stretches to infinity in the horizontal plane. Fig. 16 shows the minimal surface obtained using the restricted Lagrangian update algorithm.

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.

Minimal surfaces are smooth provided that the boundary curve is smooth. Spectral (or spectral element) methods based on high order polynomials should therefore be very suitable numerical methods for such problems. Indeed, our numerical results show that we are able to achieve exponential convergence as the polynomial degree increases. The convergence rate, however, depends strongly on the structure of the mesh points on the surface. Our iterative algorithm for computing minimal surfaces allows freedom in how the mesh points are moved during the iterations. 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 have compared three different mesh update algorithms for computing minimal surfaces. Our focus on minimal surfaces is partially motivated by the fact that there exist nontrivial minimal surfaces with analytical solutions which we can use for evaluation purposes. Numerical results show that purely Lagrangian updates are not always optimal, and neither are updates purely in the direction of the surface normal. For the Lagrangian update algorithm we always observe stability problems for sufficiently many iterations. A restricted Lagrangian update algorithm (where the displacement in the normal direction dominates the tangential displacement) seems to combine the good properties from a purely Lagrangian update and a purely normal update approach. The numerical results also show that a customized mesh update algorithm for the particular problem at hand often yields the best result.

The work presented here points to several important aspects related to using high order discretization methods for problems where the geometry is an unknown. The present work focuses on computing minimal surfaces, but the work is also highly relevant for fluid flow problems with free surfaces, in particular, when formulated in an Arbitrary Lagrangian Eulerian setting. For example, the normal update scheme is commonly used for such problems. Our numerical results demonstrate that high order (exponential) convergence rates can be realized, but great care has to be shown when selecting a mesh update algorithm. The results demonstrate the sensitivity in the error to the choice of interpolation points along the curved three-dimensional surface. The results also suggest that the restricted Lagrangian update scheme is a better choice than the normal update scheme.

Finding a general interface tracking algorithm that works better than the existing ones is a very challenging task. One such algorithm has been suggested in [28], but much work remains to be done in this direction.

Better algorithms are needed in order for high order methods to reach their full potential for computing minimal surfaces or for tracking time-dependent interfaces. We feel that the sensitivity to the choice of interpolation points has not been properly addressed in the literature before, and we stress that these challenges are particular for high order methods. More work is required in order to make high order methods robust and optimal for this class of problems.

Acknowledgments

The work has been supported by the Research Council of Norway under contract 185336/V30. The authors 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] G. Dziuk, J.E. Hutchinson, The discrete Plateau problem: algorithm and numerics, Mathematics of Computation 68 (225) (1999) 1–23.
[12] G. Dziuk, J.E. Hutchinson, The discrete Plateau problem: convergence results, Mathematics of Computation 68 (226) (1999) 519–546.
[13] K.A. Brakke, The surface evolver, Experimental Mathematics 1 (2) (1992) 141–165.
[14] 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.
[15] H.J. Wagner, A contribution to the numerical approximation of minimal surfaces, Computing 19 (1) (1977) 35–58.
[16] C. Coppin, D. Greenspan, A contribution to the particle modeling of soap films, Applied Mathematics and Computation 26 (4) (1988) 315–331.

[17] D.L. Chopp, Computing minimal surfaces via level set curvature flow, Journal of Computational Physics 106 (1993) 77–91.
[18] A. Huerta, A. Rodríguez-Ferran (Eds.), The arbitrary Lagrangian–Eulerian formulation, Computer Methods in Applied Mechanics and Engineering 193 (39–41) (2004) 4073–4456.
[19] T.J. Willmore, Riemannian Geometry, Clarendon Press, New York, 1993.
[20] A.T. Fomenko, The Plateau Problem, Gordon and Breach Science Publishers, New York, 1990.
[21] M. Struwe, Plateau's Problem and the Calculus of Variations, Princeton University Press, Princeton, 1988.
[22] L.D. Landau, E.M. Lifshitz, Fluid Mechanics, Course of Theoretical Physics, Vol. 6, Butterworth-Heinemann, 1987.
[23] 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.
[24] E. Kreyszig, Differential Geometry, Dover Publications, Inc., 1991.
[25] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang, Spectral Methods, in: Fundamentals in Single Domains, Springer, 2006.
[26] U. Dierkes, S. Hildebrandt, A. Küster, O. Wohlrab, Minimal surfaces. I, Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 295, Springer-Verlag, Berlin, 1992.
[27] T. Bjøntegaard, E.M. Rønquist, Ø. Tråsdahl, High order polynomial interpolation of parameterized curves, in: Spectral and High Order Methods for Partial Differential Equations, Lecture Notes in Computational Science and Engineering, vol. 76, Springer, 2011, pp. 365–372.
[28] T. Bjøntegaard, E.M. Rønquist, Accurate interface-tracking for arbitrary Lagrangian–Eulerian schemes, Journal of Computational Physics 228 (12) (2009) 4379–4399.
[29] G. Dziuk, An algorithm for evolutionary surfaces, Numerische Mathematik 58 (1) (1991) 603–611.