ANNALES MATHÉMATIQUES

BLAISE PASCAL

Marco Picasso, Jacques Rappaz and Adrian Reist

Numerical simulation of the motion of a three-dimensional glacier

Volume 15, n° 1 (2008), p. 1-28.

<http://ambp.cedram.org/item?id=AMBP_2008__15_1_1_0>

Publication éditée par le laboratoire de mathématiques de l'université Blaise-Pascal, UMR 6620 du CNRS Clermont-Ferrand — France

cedram

Article mis en ligne dans le cadre du Centre de diffusion des revues académiques de mathématiques
http://www.cedram.org/

Annales mathématiques Blaise Pascal **15**, 1-28 (2008)

Numerical simulation of the motion of a three-dimensional glacier

Marco Picasso
Jacques Rappaz
Adrian Reist

Abstract

The motion of a three-dimensional glacier is considered. Ice is modeled as an incompressible non-Newtonian fluid. At each time step, given the shape of the glacier, a nonlinear elliptic system has to be solved in order to obtain the two components of the horizontal velocity field. Then, the shape of the glacier is updated by solving a transport equation. Finite element techniques are used to compute the velocity field and to solve the transport equation. Numerical results are compared to experiments on Storglaciaren (Sweden) between 1959 and 1990.

1. Introduction

Over the last 150 years, the Swiss Alps lost 50 % of their glaciers. Today, most of alpine glaciers are retreating, leaving visible marks on the landscape. The future evolution of glaciers is not only of concern for the tourist industry, but also for agriculture and hydro-power production. The past evolution of glaciers is an archive of climatic changes [2].

Numerical glacier modelling has become an important tool for glaciologists in order to understand the interactions between glaciers and climate. Most of the numerical methods use finite differences with fixed cartesian grids [18].

Recently, finite element techniques have been introduced in this area, see for instance [20, 15], and several mathematical papers have been published in two space dimensions. The well posedness of a shallow ice model for the computation of the velocity is discussed in [10]. A priori and a posteriori error estimates are derived for the corresponding finite element

Keywords: glacier, ice, non-Newtonian fluid, finite elements.
Math. classification: 65N30,76M10.

method in [14, 8]. The question of modelling errors for this model is discussed in [6]. A finite element method to compute the ice velocity and the motion of the glacier is presented in [21].

The goal of this paper is to present a three-dimensional finite element model in order to compute the ice velocity and the motion of the glacier, thus extending the two-dimensional model presented in [21] to three space dimensions [23]. The model is presented in the next section. Given precipitations estimates, the size and motion of a glacier is computed. Energy balance is disregarded. The velocity field is modelled as an incompressible Non-Newtonian fluid. Assuming that the horizontal extent of the ice sheet is much larger than the ice thickness, a first order analysis of the momentum equations yields to a strongly non-linear elliptic system for the two horizontal components of the velocity field. Then, mass conservation yields to a transport equation for the glacier height, with a given right-hand side corresponding to the climatic input due to snow falls or ice melting. At each time step, given the shape of the glacier, the strongly non-linear elliptic system studied in [22] has to be solved in order to obtain the two components of the horizontal velocity field. Then, the shape of the glacier is updated by solving the transport equation for the glacier height. Finite element techniques are used to compute the velocity field and to solve the transport equation. The transport equation is discretized using an Arbitrary Lagrangian Eulerian (ALE) formulation. In section 4, numerical results are compared to experiments on Storglaciaren (Sweden) between 1959 and 1990.

When comparing the present paper to the two-dimensional model presented in [21], the interested reader will note that the overall methodology is the same. However, the extension from 2D to 3D computations is not obvious, specially from the point of view of mesh generation and deformation. This issue is explained in details in Section 3.

2. **The model**

Our goal is to present a model predicting the evolution of a glacier during decades, possibly centuries given the amount of ice accumulating or disappearing on the top surface of the glacier. At time t, the unknown ice domain $\Omega(t)$ is parametrized as follows :

$$\Omega(t) = \{(x, y, z) \in \mathbb{R}^3; (x, y) \in \Lambda(t); B(x, y) \leq z \leq S(x, y, t)\},$$

where $\Lambda(t)$ is the unknown glacier support, B is the prescribed mountain bedrock, $S = B + H$ the unknown top surface of the glacier, see Fig. 1 for the notations. Given the snow falls or ice melting, our goal is to compute when time t is going on, the horizontal components u and v of the ice velocity and the shape of the glacier, that is the glacier support Λ and the glacier height H.

Coupling the size and motion of a glacier with the energy balance would be a significant accomplishment in the field but is beyond the scope of the present paper. We refer to [13] for a full description of models involved in glacier dynamics.

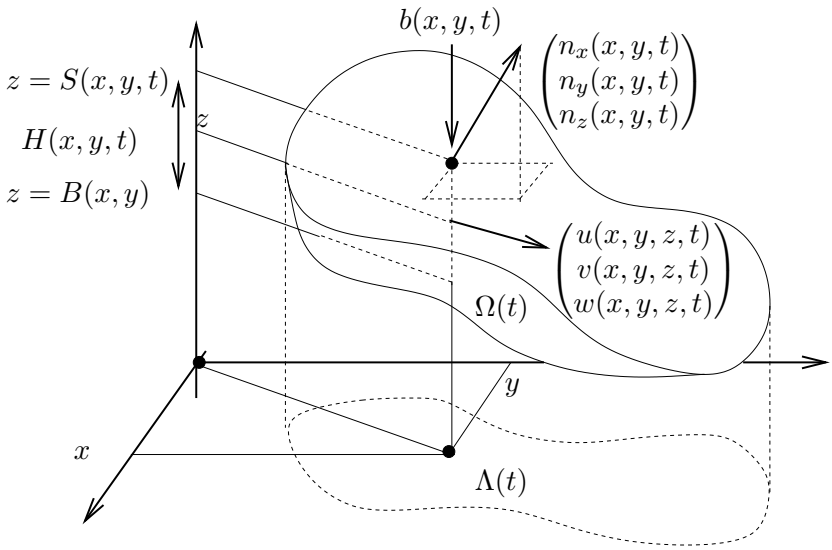

Figure 1. The notations. The ice domain at time t is $\Omega(t)$, its projection onto the xy plane is the glacier support $\Lambda(t)$. The mountain bedrock is parametrized by B, the top surface by S, the glacier height is H. The function b is a given quantity corresponding to mass accumulation or ablation (snow falls or ice melting). The velocity field of ice has components u, v and w. The normal at the top surface has components n_x, n_y and n_z.

2.1. **The velocity field**

For the time scales involved (from several years to a century), glacier ice is behaving like a fluid, not like a solid. Ice can be modeled as an incompressible viscous fluid and time-dependent effects can be neglected. Thus, given the ice domain $\Omega(t)$, a stationary problem has to be solved in order to obtain the ice velocity. Several models are available and the case of a shallow glacier is considered here. More precisely, it is assumed that the horizontal extent of the ice sheet is much larger than the ice thickness. In [3], a scale analysis is performed for such a glacier, the scaling parameter being the ratio between the ice thickness and the length (or width) of the glacier. For instance in small valley glaciers this ratio ranges between 0.01 and 0.1 (eventually more in steep regions). Below, we briefly present the corresponding first order model in three space dimensions, see [3] for details. A mathematical and numerical analysis of this model is provided in [22].

It should be noted that this first order model requires only the horizontal components u and v of the velocity field to be computed. Indeed, the horizontal component w is not contained in the set of equations anymore, although it can be recovered from the mass conservation equation. The fact that w is absent from the set of equations is due to the fact that w occurs only in terms which are of second order magnitude with respect to the small parameter (the ratio between the ice thickness and the length or width of the glacier). Thus, the feedback of the horizontal component w to the horizontal components u and v is small, or negligible in first order accuracy.

At time t, the equations governing the horizontal components $u(t), v(t) : \Omega(t) \to \mathbb{R}$ of the velocity field are as follows:

$$-\frac{\partial}{\partial x}\left(\mu\left(2\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)-\frac{\partial}{\partial y}\left(\frac{\mu}{2}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right) -\frac{\partial}{\partial z}\left(\frac{\mu}{2}\frac{\partial u}{\partial z}\right)=\rho g\frac{\partial S}{\partial x}, \quad (2.1)$$

$$-\frac{\partial}{\partial x}\left(\frac{\mu}{2}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right)-\frac{\partial}{\partial y}\left(\mu\left(\frac{\partial u}{\partial x}+2\frac{\partial v}{\partial y}\right)\right) -\frac{\partial}{\partial z}\left(\frac{\mu}{2}\frac{\partial v}{\partial z}\right)=\rho g\frac{\partial S}{\partial y}. \quad (2.2)$$

Here μ is the ice viscosity, which is implicitly defined by the relation

$$\frac{1}{\mu} = A\left(T_0^{n-1} + (s\mu)^{n-1}\right), \tag{2.3}$$

where A, T_0 and n are positive coefficients and s is given by

$$s = \frac{1}{\sqrt{2}}\left[\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 + \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)^2 + \right.$$
$$\left. + \frac{1}{2}\left(\frac{\partial u}{\partial z}\right)^2 + \frac{1}{2}\left(\frac{\partial v}{\partial z}\right)^2 + \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2\right]^{\frac{1}{2}}. \tag{2.4}$$

Let $\vec{n} = (n_x, n_y, n_z)^T$ denote the outward normal vector on the boundary of $\Omega(t)$. The boundary conditions on the top surface $z = S(x, y, t)$, $(x, y) \in \Lambda(t)$ correspond to zero force condition and are

$$\mu\left(2\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)n_x + \frac{\mu}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)n_y + \frac{\mu}{2}\frac{\partial u}{\partial z}n_z = 0, \tag{2.5}$$
$$\frac{\mu}{2}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right)n_x + \mu\left(\frac{\partial u}{\partial x} + 2\frac{\partial v}{\partial y}\right)n_y + \frac{\mu}{2}\frac{\partial v}{\partial z}n_z = 0. \tag{2.6}$$

On the mountain bedrock $z = B(x, y)$, $(x, y) \in \Lambda(t)$, zero Dirichlet conditions are imposed:

$$u = 0, \qquad v = 0. \tag{2.7}$$

It should be noted that the above no-slip boundary condition is too simple to reproduce complex phenomena such as glacier surges [12, 13]. Indeed, in temperate glaciers, due to water flowing between the bedrock and the ice sheet, ice may slip along the bedrock. In the 2D simulation of Gries glacier [21], the velocity on the bedrock was tuned so that it was fitting the measured velocity on the top surface. Here, the measured velocity on the top surface is available only in a small portion of the glacier (Storglaciaren), thus a different strategy is adopted. No-slip boundary conditions are imposed on the mountain bedrock and the parameter A involved in (2.3) is tuned so that the computed velocity best fits the measured one.

2.2. **A model for the glacier height**

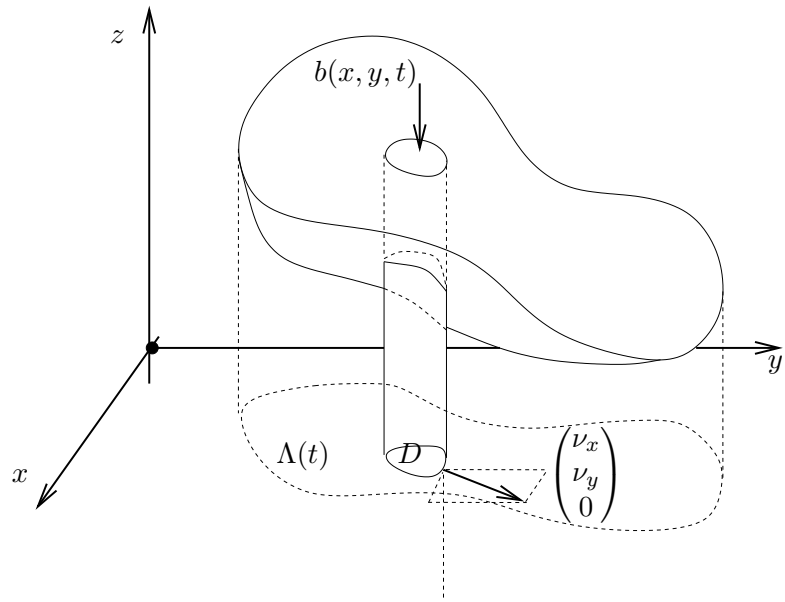

FIGURE 2. Mass conservation into a vertical cylinder with base D in the xy plane. The normal to ∂D has component ν_x and ν_y in the xy plane. The amount of mass due to snow accumulation or melting on the top surface is b.

In this subsection, the two components u and v of the horizontal velocity are assumed to be known in the ice domain $\Omega(t)$ and an additional equation governing the evolution of the glacier height H is derived. Consider, as in Fig. 2, a vertical cylinder with base D. Mass conservation into this cylinder writes

$$\frac{d}{dt}\left(\int_{(x,y)\in D}\int_{z=B(x,y)}^{z=S(x,y,t)} dzdxdy\right)$$
$$+\int_{(x,y)\in\partial D}\int_{z=B(x,y)}^{z=S(x,y,t)}(u\nu_x+v\nu_y)dzds(x,y)=\int_{(x,y)\in D} b(x,y,t)dxdy,$$

where b is a given quantity corresponding to mass accumulation or ablation (snow falls or ice melting) that can be measured by glaciologists. Since $S = B + H$, the divergence theorem applied to domain D yields

$$\int_{(x,y)\in D} \left(\frac{\partial H}{\partial t} + \frac{\partial}{\partial x}\Big(\int_B^S u \; dz\Big) + \frac{\partial}{\partial y}\Big(\int_B^S v \; dz\Big) \right) dxdy = \int_{(x,y)\in D} b dxdy. \quad (2.8)$$

Let $\bar{u}$, $\bar{v}$ be the vertical average of the horizontal velocities u, v defined by

$$\bar{u}(x,y,t) = \frac{1}{H(x,y,t)} \int_{B(x,y)}^{S(x,y,t)} u(x,y,z,t) dz, \quad (2.9)$$

$$\bar{v}(x,y,t) = \frac{1}{H(x,y,t)} \int_{B(x,y)}^{S(x,y,t)} v(x,y,z,t) dz. \quad (2.10)$$

Then, since the domain D is an arbitrary subset of the glacier support $\Lambda(t)$, (2.8) leads to

$$\frac{\partial H}{\partial t} + \frac{\partial}{\partial x}(\bar{u}H) + \frac{\partial}{\partial y}(\bar{v}H) = b \qquad \text{in } \Lambda(t). \quad (2.11)$$

The evolution of the ice domain $\Omega(t)$ is known whenever the glacier support $\Lambda(t)$ and the glacier thickness $H(t)$ are known. The glacier thickness is governed by the transport equation (2.11). In order to determine $\Lambda(t)$ we need to provide the additional equation

$$H(x,y,t) = 0 \qquad (x,y) \in \Lambda(t), \quad t > 0. \quad (2.12)$$

2.3. **Model recapitulation**

Our mathematical model is then the following. Given the glacier bedrock B, the initial glacier support $\Lambda(0)$, the initial glacier height $H(0)$, given positive parameters A, T_0, n, given the function b, find the horizontal velocities u, v satisfying (2.1) (2.2), the glacier height $H(t)$ satisfying (2.11) and the glacier support $\Lambda(t)$ satisfying (2.12).

3. **Algorithm**

The numerical procedure used to solve the above mathematical problem is based on our physical understanding of the problem and is the following.

At each time step, given an approximation of the ice domain $\Omega(t)$ we proceed as follows.

- Compute the horizontal velocity by solving (2.1) (2.2) with the boundary conditions (2.5)–(2.7) using continuous, piecewise linear finite elements on a tetrahedral mesh of $\Omega(t)$.
- Update the glacier support $\Lambda(t)$ using (2.12).
- Compute the new glacier height H by solving (2.11) with an Arbitrary Lagrangian Eulerian (ALE) finite element method.

The three steps of this decoupling algorithm are presented in details hereafter. At initial time $\Lambda(0)$ and

$$\begin{aligned} H(0) : \Lambda(0) &\to \mathbb{R} \\ (x, y) &\to H(x, y, 0) \end{aligned}$$

are known so that the initial ice domain

$$\Omega(0) = \{(x, y, z) \in \mathbb{R}^3; (x, y) \in \Lambda(0); B(x, y) \leq z \leq B(x, y) + H(x, y, 0)\}$$

is known. Let Δt be the time step, $t_m = m\Delta t$, $m = 0, 1, 2, \ldots$. Consider the time iteration number m and let Λ^m, $H^m : \Lambda^m \to \mathbb{R}$ and Ω^m be approximations of $\Lambda(t_m)$, $H(t_m) : \Lambda(t_m) \to \mathbb{R}$ and $\Omega(t_m)$, respectively. The goal is :

- to compute approximations $u^m, v^m : \Omega^m \to \mathbb{R}$ of the horizontal velocity components $u(t_m), v(t_m) : \Omega(t_m) \to \mathbb{R}$,
- to compute a new approximation Λ^{m+1} of $\Lambda(t_{m+1})$,
- to compute a new approximation $H^{m+1} : \Lambda^{m+1} \to \mathbb{R}$ of the glacier height $H(t_m) : \Lambda(t_{m+1}) \to \mathbb{R}$,
- to compute an approximation Ω^{m+1} of the ice domain $\Omega(t_{m+1})$.

3.1. **Computation of the velocity field**

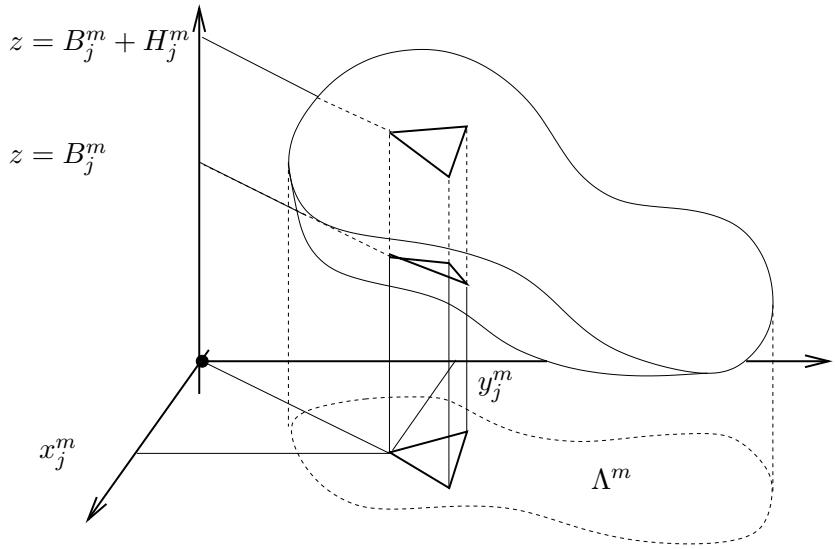

Figure 3. Notations for the finite element tetrahedral mesh of Ω^m. The glacier support Λ^m is meshed into triangles with vertices (x_j^m, y_j^m). Then, the vertices of the 3D mesh are aligned on top of those of the glacier support Λ^m.

In this subsection, the domain Λ^m and the function $H^m : \Lambda^m \to \mathbb{R}$ are known and we want to compute an approximation u^m, v^m of the horizontal velocity components $u(t_m), v(t_m) : \Omega(t_m) \to \mathbb{R}$. Let $\mathcal{T}_h(\Lambda^m)$ be a regular triangulation of Λ^m into triangles with size less than h. We assume that $H^m : \Lambda^m \to \mathbb{R}$ is a continuous piecewise linear function on each triangle of $\mathcal{T}_h(\Lambda^m)$. Let further B^m be a piecewise linear interpolation of B on this same triangulation and let $S^m = B^m + H^m$. The ice domain Ω^m is then defined by

$$\Omega^m = \{(x, y, z) : B^m(x, y) \leq z \leq S^m(x, y),\ (x, y) \in \Lambda^m\}, \tag{3.1}$$

and is meshed into tetrahedrons. For stability purposes, all the triangulations $\mathcal{T}_h(\Lambda^m)$ and all the tetrahedral meshes $\mathcal{T}_h(\Omega^m)$, $m = 0, 1, 2, ...$ have

the same topology. Moreover, the vertices of $\mathcal{T}_h(\Omega^m)$ are aligned along vertical segments with endpoints (x_j^m, y_j^m, B_j^m) and $(x_j^m, y_j^m, B_j^m + H_j^m)$, where (x_j^m, y_j^m) are the vertices of triangulation $\mathcal{T}_h(\Lambda^m)$ and $B_j^m = B^m(x_j^m, y_j^m)$, $H_j^m = H^m(x_j^m, y_j^m)$, see Fig. 3 and 4.

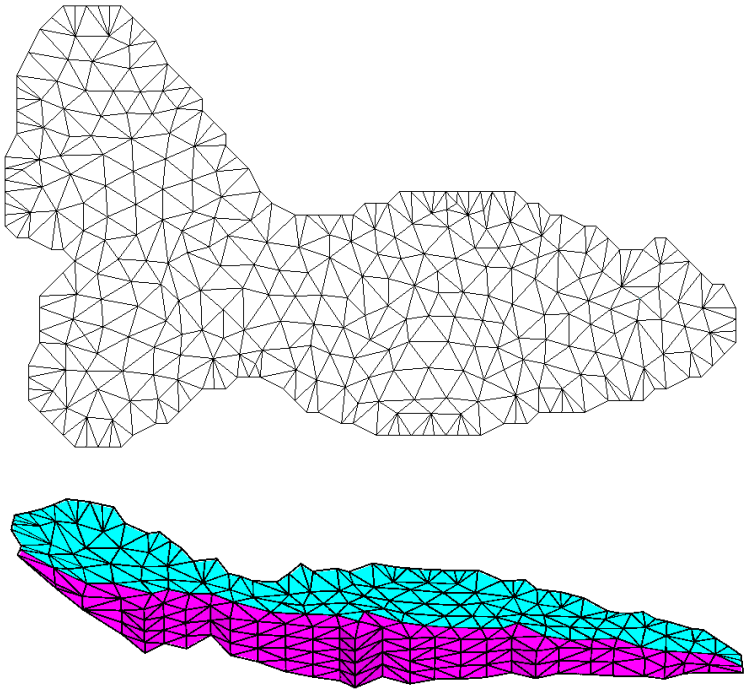

FIGURE 4. An example of finite element meshes. Top : mesh of the glacier support Λ^m into triangles. Bottom : mesh of the ice glacier domain Ω^m into tetrahedrons.

Then, equations (2.1) (2.2), with boundary conditions (2.5)-(2.7) are solved numerically using continuous, piecewise linear finite elements on the tetrahedral mesh $\mathcal{T}_h(\Omega^m)$. The finite element formulation then consists in finding (u^m, v^m) in the space of continuous, piecewise linear functions on the tetrahedrons of $\mathcal{T}_h(\Omega^m)$, vanishing on the bedrock of the glacier

and such that

$$\int_{\Omega^m}\Big\{\mu^m\left(2\frac{\partial u^m}{\partial x}+\frac{\partial v^m}{\partial y}\right)\frac{\partial\varphi}{\partial x}+\frac{\mu^m}{2}\left(\frac{\partial u^m}{\partial y}+\frac{\partial v^m}{\partial x}\right)\frac{\partial\varphi}{\partial y}$$
$$+\frac{\mu^m}{2}\frac{\partial u^m}{\partial z}\frac{\partial\varphi}{\partial z}++\frac{\mu^m}{2}\left(\frac{\partial u^m}{\partial y}+\frac{\partial v^m}{\partial x}\right)\frac{\partial\psi}{\partial x}$$
$$+\mu^m\left(\frac{\partial u^m}{\partial x}+2\frac{\partial v^m}{\partial y}\right)\frac{\partial\psi}{\partial y}+\frac{\mu^m}{2}\frac{\partial v^m}{\partial z}\frac{\partial\psi}{\partial z}\Big\}dxdydz$$
$$=\rho g\int_{\Omega^m}\left(\frac{\partial S^m}{\partial x}\varphi+\frac{\partial S^m}{\partial y}\psi\right)dxdydz, \quad (3.2)$$

for all test functions (φ,ψ), continuous, piecewise linear on the tetrahedrons of $\mathcal{T}_h(\Omega^m)$, vanishing on the bedrock of the glacier. Here μ^m is the ice viscosity which is implicitly defined from the derivatives of u^m, v^m as in (2.3) (2.4). It should be noted that existence of a solution to (3.2) and convergence when the mesh size h goes to zero has been addressed in [22]. The difficulty is due to the fact that (3.2) is a nonlinear problem since μ^m depends implicitly on the derivatives of the unknown velocities u^m, v^m. Picard's iterative method is used to solve the nonlinear system, no under-relaxation being necessary to reach convergence, see the proof in [23].

Since the vertices of $\mathcal{T}_h(\Omega^m)$ are vertically aligned, their projection onto the xy plane coincide with the vertices of $\mathcal{T}_h(\Lambda^m)$, which facilitates the computation of the vertically averaged horizontal velocity components $\bar{u}$, $\bar{v}$ defined by (2.9) (2.10). Using a trapezoidal quadrature formula we can compute approximations $\bar{u}_j^m$ and $\bar{v}_j^m$ of $\bar{u}(x_j^m,y_j^m,t_m)$ and $\bar{v}(x_j^m,y_j^m,t_m)$ as follows:

$$\begin{aligned}\bar{u}_j^m&=\frac{1}{N_j}\left(\frac{u^m(x_j^m,y_j^m,B_j^m)}{2}+\sum_{i=1}^{N_j-1}u^m(x_j^m,y_j^m,B_j^m+i\Delta z_j^m)\right.\\&\left.+\frac{u^m(x_j^m,y_j^m,B_j^m+H_j^m)}{2}\right),\\\bar{v}_j^m&=\frac{1}{N_j}\left(\frac{v^m(x_j^m,y_j^m,B_j^m)}{2}+\sum_{i=1}^{N_j-1}v^m(x_j^m,y_j^m,B_j^m+i\Delta z_j^m)\right.\\&\left.+\frac{v^m(x_j^m,y_j^m,B_j^m+H_j^m)}{2}\right).\end{aligned} \quad (3.3)$$

Here N_j is the number of vertices in the mesh $\mathcal{T}_h(\Omega^m)$ aligned on top of a vertex (x_j^m, y_j^m) in the triangulation $\mathcal{T}_h(\Lambda^m)$ (see Fig. 3) and $\Delta z_j^m = H^m(x_j^m, y_j^m)/N_j$ is the vertical mesh spacing. The functions $\bar{u}^m, \bar{v}^m : \Lambda^m \to \mathbb{R}$ are then defined as

$$\bar{u}^m(x,y) = \sum_{j=1}^{N} \bar{u}_j^m \varphi_j^m(x,y),$$
$$\bar{v}^m(x,y) = \sum_{j=1}^{N} \bar{v}_j^m \varphi_j^m(x,y),$$

where $\varphi_j^m(x,y)$ are the continuous, piecewise linear shape functions associated with the triangulation $\mathcal{T}_h(\Lambda^m)$.

3.2. **Computation of the new glacier support** Λ^{m+1}

In order to determine the new glacier support Λ^{m+1} we proceed in two steps: first, we move the vertices lying on the boundary of Λ^m, then we move the internal vertices of Λ^m.

3.2.1. **Moving the vertices lying on the boundary of** Λ^m**.**

At time t, let $(x(t), y(t))$ be a point lying on the boundary of glacier support $\Lambda(t)$. We compute an approximation of $x(t+\Delta t), y(t+\Delta t)$ using the first order expansion

$$x(t+\Delta t) \approx x(t) + \Delta t\ \dot{x}(t), \qquad y(t+\Delta t) \approx y(t) + \Delta t\ \dot{y}(t). \tag{3.4}$$

In order to compute the $\dot{x}(t), \dot{y}(t)$, we take the derivative of (2.12) with respect to time and obtain

$$\frac{\partial H}{\partial x}(x(t),y(t),t)\ \dot{x}(t) + \frac{\partial H}{\partial y}(x(t),y(t),t)\ \dot{y}(t) + \frac{\partial H}{\partial t}(x(t),y(t),t) = 0.$$

Using equation (2.11) we obtain

$$\begin{aligned}&\frac{\partial H}{\partial x}(x(t),y(t),t)\ \dot{x}(t) + \frac{\partial H}{\partial y}(x(t),y(t),t)\ \dot{y}(t)\\ &= \frac{\partial}{\partial x}(\bar{u}H)(x(t),y(t),t) + \frac{\partial}{\partial y}(\bar{v}H)(x(t),y(t),t) - b(x(t),y(t),t).\end{aligned} \tag{3.5}$$

Let $\vec{d} = (d_x, d_y)^T$, $d_x^2 + d_y^2 = 1$, be the direction in which the vertices located on the boundary of $\Lambda(t)$ will be moved. We have: $\dot{x}(t) = \delta d_x$ and $\dot{y}(t) = \delta d_y$, where δ is the velocity at $(x(t), y(t))$ in direction $\vec{d}$. From (3.5) we obtain

$$\delta = \delta(x(t), y(t), t) = \frac{\left(\frac{\partial}{\partial x}(\bar{u}H) + \frac{\partial}{\partial y}(\bar{v}H) - b\right)(x(t), y(t), t)}{\left(\frac{\partial H}{\partial x} d_x + \frac{\partial H}{\partial y} d_y\right)(x(t), y(t), t)}. \tag{3.6}$$

Now, let (x_j^m, y_j^m) be a vertex of the triangulation $\mathcal{T}_h(\Lambda^m)$ lying on the boundary of Λ^m. We compute the position of this vertex at time t_{m+1} as follows:

$$x_j^{m+1} = x_j^m + \Delta t \ \delta_j^m \ (d_x)_j^m, \qquad y_j^{m+1} = y_j^m + \Delta t \ \delta_j^m \ (d_y)_j^m, \tag{3.7}$$

where δ_j^m is an approximation of $\delta(x_j^m, y_j^m, t_m)$ defined by (3.6). Since $\bar{u}$, $\bar{v}$ and H are continuous, piecewise linear on each triangle K of $\mathcal{T}_h(\Lambda^m)$, their space derivatives are piecewise constant, so that a reconstructed gradient has to be computed at the vertices (x_j^m, y_j^m) in order to compute δ_j^m. This reconstructed gradient is obtained by averaging the gradient over all triangles containing (x_j^m, y_j^m).

It now remains to choose the direction $\vec{d}_j^m = ((d_x)_j^m, (d_y)_j^m)^T$ along which the boundary vertex (x_j^m, y_j^m) is moved. The most obvious choice is to move the points towards the direction normal to the boundary, that is $\vec{d}_j^m$ proportional to $-\nabla H^m(x_j^m, y_j^m)$. In practice, this choice is not the best one. Indeed, when the boundary of Λ^m has a strong curvature, then two neighboring vertices may intersect, which leads to a non conforming mesh, see Fig. 5. In order to remedy this problem, we modify the direction in which we move the boundary vertices. If the curvature at a given vertex (x_j^m, y_j^m) is larger than the curvature at its neighbors, then the direction $\vec{d}_j^m$ is expected to be aligned with the normal at (x_j^m, y_j^m). On the other side, if the curvature at a given vertex (x_j^m, y_j^m) is smaller than the curvature at its neighbors, then the direction $\vec{d}_j^m$ is expected to be aligned with the normal at its neighbors.

FIGURE 5. Moving the vertices lying on the boundary of Λ^m. Left : moving vertices in the direction normal to the boundary may result in crossover in the case of large curvature. Right : changing the direction prevents crossover.

The following algorithm is used to obtain the direction $\vec{d}_j^m$ at each vertex (x_j^m, y_j^m) lying on the boundary of Λ^m. We start with a direction normal to the boundary.

$$(\vec{d}_j^m)^0 = -\frac{\vec{\nabla} H_j^m}{|\vec{\nabla} H_j^m|}$$

where $\vec{\nabla} H_j^m$ is the reconstructed gradient of H^m at vertex (x_j^m, y_j^m). Then, the new direction is computed as a linear combination of its own direction vector $\vec{d}_j^m$ and the directions of the two neighbors, $\vec{d}_{j-1}^m$, $\vec{d}_{j+1}^m$, weighted by a coefficient c_j^m depending on the curvature. This first step propagates the direction of a node with large curvature only to its two immediate neighbors, which might not be sufficient to avoid crossover. We apply the above step several times to achieve the desired result :

$$(\vec{d}_j^m)^{k+1} = c_{j-1}^m (\vec{d}_{j-1}^m)^k + c_j^m (\vec{d}_j^m)^k + c_{j+1}^m (\vec{d}_{j+1}^m)^k \qquad k = 0, 1, 2, ..,$$

and then normalize the vector to one. The coefficient c_j^m depends on the angle θ_j^m reported in Fig. 6 and we set $c_j = \pi/\theta_j^m$.

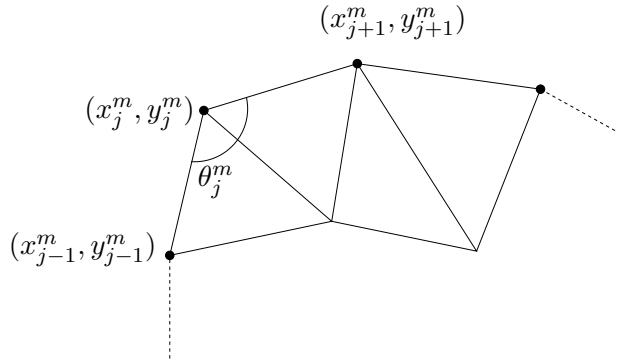

Figure 6. Definition of the angle θ_j^m.

3.2.2. **Moving the internal vertices of Λ^m.**

Once we have moved the boundary nodes of $\mathcal{T}_h(\Lambda^m)$, we move the interior nodes without changing mesh topology and obtain the new mesh $\mathcal{T}_h(\Lambda^{m+1})$. To this end, the Laplacian method (see for instance [19, 24]) is used. The method consists in solving the following system :

$$\Delta_m X^{m+1} = 0, \qquad \Delta_m Y^{m+1} = 0,$$

where X^{m+1} and Y^{m+1} are the new mesh coordinates and Δ_m is the Laplacian operator defined on the mesh prior to deformation. We use Dirichlet boundary conditions in order to impose the coordinates of the nodes on the new border:

$$X_j^{m+1} = x_j^{m+1}, \qquad Y_j^{m+1} = y_j^{m+1},$$

for all vertices (x_j^m, y_j^m) lying on the boundary of $\mathcal{T}_h(\Lambda^m)$. An example of mesh deformation is reported in Fig. 7 for Storglaciären, Sweden, the mesh deformation being exaggerated.

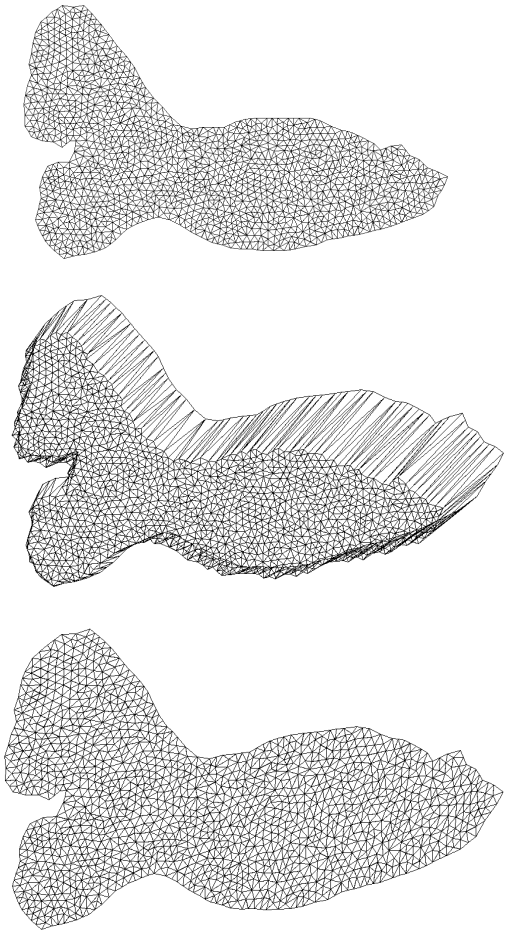

FIGURE 7. Mesh deformation of the support of Storglaciären, Sweden. Top : The mesh $\mathcal{T}_h(\Lambda^m)$ of the glacier support Λ^m. Middle : the vertices of the mesh $\mathcal{T}_h(\Lambda^m)$ lying on the boundary of Λ^m are moved (the deformation is exaggerated). Bottom : the internal vertices of $\mathcal{T}_h(\Lambda^m)$ are moved according to the Laplacian method, which yields the new mesh $\mathcal{T}_h(\Lambda^{m+1})$.

It should be noted that the Laplacian method for moving the grid points exhibits poor behavior when non-convex geometries are encountered. More elaborated strategies are available for avoiding this problem, see [7]. The use of such techniques has not been necessary in the framework of our model.

3.3. **Computation of the new glacier height** H^{m+1}

Our goal is now to compute an approximation $H^{m+1} : \Lambda^{m+1} \to \mathbb{R}$ of $H(t^{m+1})$, the solution of (2.11), given an approximation $H^m : \Lambda^m \to \mathbb{R}$. Since both H^m and H^{m+1} are computed on different domains, an Arbitrary Lagrangian Eulerian (ALE) formulation (see for instance [4]) will be used.

Equation (2.11) has to be solved numerically for all $(x, y) \in \Lambda(t)$, $t_m \leq t \leq t_{m+1}$. Then we introduce the change of variable :

$$\begin{aligned} \hat{x}(x, y, t) =& x + \int_t^{t_{m+1}} U(x, y, s)\, ds, \\ \hat{y}(x, y, t) =& y + \int_t^{t_{m+1}} V(x, y, s)\, ds, \end{aligned} \tag{3.8}$$

where $U(t)$, $V(t)$ is the mesh velocity in $\Lambda(t)$. This change of variable maps $\Lambda(t)$ on $\hat{\Lambda} = \Lambda(t_{m+1})$. Further we define the functions $\hat{H}$, $\hat{b}$, $\hat{U}$, $\hat{V}$, $\hat{\bar{u}}$, $\hat{\bar{v}}$ by

$$\begin{aligned} \hat{H}(\hat{x}, \hat{y}, t) =& H(x, y, t), \qquad & \hat{b}(\hat{x}, \hat{y}, t) = b(x, y, t), \\ \hat{U}(\hat{x}, \hat{y}, t) =& U(x, y, t), \qquad & \hat{V}(\hat{x}, \hat{y}, t) = V(x, y, t), \\ \hat{\bar{u}}(\hat{x}, \hat{y}, t) =& \bar{u}(x, y, t), \qquad & \hat{\bar{v}}(\hat{x}, \hat{y}, t) = \bar{v}(x, y, t). \end{aligned} \tag{3.9}$$

Equation (2.11) then writes

$$\begin{aligned} &\frac{\partial \hat{H}}{\partial t} - \hat{U}\frac{\partial \hat{H}}{\partial \hat{x}} - \hat{V}\frac{\partial \hat{H}}{\partial \hat{y}} \\ &\qquad + \left(\frac{\partial \hat{x}}{\partial x} + \frac{\partial \hat{x}}{\partial y}\right)\frac{\partial}{\partial \hat{x}}(\hat{\bar{u}}\hat{H}) + \left(\frac{\partial \hat{y}}{\partial x} + \frac{\partial \hat{y}}{\partial y}\right)\frac{\partial}{\partial \hat{y}}(\hat{\bar{v}}\hat{H}) = \hat{b}, \end{aligned} \tag{3.10}$$

in the cylinder $\Lambda(t_{m+1}) \times [t_m, t_{m+1}]$. The above equation is discretized in time using the following order one, implicit time discretization of (3.10). Given $\hat{H}^m : \hat{\Lambda} = \Lambda(t_{m+1}) \to \mathbb{R}$, the goal is to find $\hat{H}^{m+1} : \hat{\Lambda} = \Lambda(t_{m+1}) \to$

$\mathbb{R}$ such that $\hat{H}^{m+1} = 0$ on the boundary of $\Lambda(t_{m+1})$ and such that

$$\frac{\hat{H}^{m+1} - \hat{H}^m}{\Delta t} - \hat{U}(t_{m+1})\frac{\partial \hat{H}^{m+1}}{\partial x} - \hat{V}(t_{m+1})\frac{\partial \hat{H}^{m+1}}{\partial y} + \frac{\partial}{\partial x}\Big(\hat{\bar{u}}(t_m)\hat{H}^{m+1}\Big) + \frac{\partial}{\partial y}\Big(\hat{\bar{v}}(t_m)\hat{H}^{m+1}\Big) = \hat{b}(t_{m+1}), \quad (3.11)$$

in $\Lambda(t_{m+1})$. Here we used the fact that

$$\frac{\partial \hat{x}}{\partial x}(x, y, t_{m+1}) = \frac{\partial \hat{y}}{\partial y}(x, y, t_{m+1}) = 1,$$
$$\frac{\partial \hat{x}}{\partial y}(x, y, t_{m+1}) = \frac{\partial \hat{y}}{\partial x}(x, y, t_{m+1}) = 0.$$

Given an approximation $\hat{\Lambda} = \Lambda^{m+1}$ of the glacier support $\Lambda(t_{m+1})$, equation (3.11) is now solved using continuous, piecewise linear finite elements on the mesh $\mathcal{T}_h(\Lambda^{m+1})$, together with an artificial diffusion method. Let (x_j^m, y_j^m) and (x_j^{m+1}, y_j^{m+1}) be the vertices of $\mathcal{T}_h(\Lambda^m)$ and $\mathcal{T}_h(\Lambda^{m+1})$, respectively. Then, the mesh velocities $\hat{U}^{m+1}, \hat{V}^{m+1} : \hat{\Lambda} = \Lambda^{m+1} \to \mathbb{R}$ are defined by

$$\hat{U}^{m+1}(\hat{x}, \hat{y}) = \sum_{j=1}^{N} \frac{x_j^{m+1} - x_j^m}{\Delta t}\hat{\varphi}_j(\hat{x}, \hat{y}),$$
$$\hat{V}^{m+1}(\hat{x}, \hat{y}) = \sum_{j=1}^{N} \frac{y_j^{m+1} - y_j^m}{\Delta t}\hat{\varphi}_j(\hat{x}, \hat{y}),$$

where $\hat{\varphi}_j$, $j = 1, ..., N$, are the usual hat functions on the vertices of $\mathcal{T}_h(\hat{\Lambda}) = \mathcal{T}_h(\Lambda^{m+1})$. Therefore, the change of variable (3.8) becomes

$$\begin{aligned} \hat{x} &= x + \Delta t \hat{U}^{m+1}(\hat{x}, \hat{y}), \\ \hat{y} &= y + \Delta t \hat{V}^{m+1}(\hat{x}, \hat{y}), \end{aligned} \quad (3.12)$$

where $(\hat{x}, \hat{y}) \in \hat{\Lambda} = \Lambda^{m+1}$ and $(x, y) \in \Lambda^m$. Let $H^m : \Lambda^m \to \mathbb{R}$ be the computed glacier height at time t^m. We have

$$H^m(x, y) = \sum_{i=1}^{N} H_j^m \varphi_j(x, y) \qquad (x, y) \in \Lambda^m,$$

where φ_j, $j = 1, ..., N$, are the usual hat functions on the vertices of $\mathcal{T}_h(\Lambda^m)$. Since the transformation $\Lambda^m \to \hat{\Lambda} = \Lambda^{m+1}$ is piecewise linear, we

also have

$$\hat{H}^m(\hat{x},\hat{y}) = H^m(x,y) = \sum_{i=1}^{N} H_j^m \varphi_j(x,y) = \sum_{i=1}^{N} H_j^m \hat{\varphi}_j(\hat{x},\hat{y}).$$

Our finite element discretization of (3.11) then consists in finding $\hat{H}^{m+1} : \hat{\Lambda} = \Lambda^{m+1} \to \mathbb{R}$, continuous, piecewise linear, that is

$$\hat{H}^{m+1}(\hat{x},\hat{y}) = \sum_{i=1}^{N} \hat{H}_j^{m+1} \hat{\varphi}_j(\hat{x},\hat{y}),$$

such that $\hat{H}^{m+1} = 0$ on the boundary of $\hat{\Lambda} = \Lambda^{m+1}$ and such that

$$\begin{aligned}\int_{\hat{\Lambda}} \frac{\hat{H}^{m+1} - \hat{H}^m}{\Delta t} \hat{\varphi}\, d\hat{x}d\hat{y} + \int_{\hat{\Lambda}} \epsilon \vec{\nabla} \hat{H}^{m+1} \cdot \vec{\nabla} \hat{\varphi}\, d\hat{x}d\hat{y} \\ - \int_{\hat{\Lambda}} \left(\hat{U}^{m+1} \frac{\partial \hat{H}^{m+1}}{\partial \hat{x}} + \hat{V}^{m+1} \frac{\partial \hat{H}^{m+1}}{\partial \hat{y}} \right) \hat{\varphi}\, d\hat{x}d\hat{y} \\ - \int_{\hat{\Lambda}} \left(\hat{\bar{u}}^m \hat{H}^{m+1} \frac{\partial \hat{\varphi}}{\partial \hat{x}} + \hat{\bar{v}}^m \hat{H}^{m+1} \frac{\partial \hat{\varphi}}{\partial \hat{y}} \right) d\hat{x}d\hat{y} = \int_{\hat{\Lambda}} \hat{b}^{m+1} \hat{\varphi}\, d\hat{x}d\hat{y}, \quad (3.13)\end{aligned}$$

for all continuous, piecewise linear functions $\hat{\varphi}$ vanishing on the boundary of $\hat{\Lambda} = \Lambda^{m+1}$. The additional diffusion term has been added here for stability purposes, the classical SUPG or GLS method being not diffusive enough in our context. Following [11], the parameter ϵ is equal to the local mesh size times the mean local velocity. Also, the vertically averaged velocities $\hat{\bar{u}}^m, \hat{\bar{v}}^m : \hat{\Lambda} = \Lambda^{m+1} \to \mathbb{R}$ are defined by

$$\hat{\bar{u}}^m(\hat{x},\hat{y}) = \sum_{i=1}^{N} \bar{u}_j^m \hat{\varphi}_j(\hat{x},\hat{y}), \qquad \hat{\bar{v}}^m(\hat{x},\hat{y}) = \sum_{i=1}^{N} \bar{v}_j^m \hat{\varphi}_j(\hat{x},\hat{y}),$$

where $\bar{u}_j^m$ and $\bar{v}_j^m$ are defined in (3.3).

3.4. **Computation of the new ice domain Ω^{m+1}**

Given the new glacier support Λ^{m+1}, given the new glacier height $H^{m+1} : \Lambda^{m+1} \to \mathbb{R}$, the new ice domain Ω^{m+1} is defined by

$$\begin{aligned}\Omega^{m+1} = \{(x,y,z) \in \mathbb{R}^3; (x,y) \in \Lambda^{m+1}; \\ B^{m+1}(x,y) \le z \le B^{m+1}(x,y) + H^{m+1}(x,y)\},\end{aligned}$$

where B^{m+1} is a piecewise linear interpolation of the glacier bedrock B on the triangulation $\mathcal{T}_h(\Lambda^{m+1})$. For stability purposes, the tetrahedral mesh $\mathcal{T}_h(\Omega^{m+1})$ has the same topology than the previous one, $\mathcal{T}_h(\Omega^m)$. Moreover, the vertices of the tetrahedral mesh $\mathcal{T}_h(\Omega^{m+1})$ are aligned on top of those of the triangulation $\mathcal{T}_h(\Lambda^{m+1})$, as in Fig. 3 and 4.

4. **Numerical results**

4.1. **Validation of implementation**

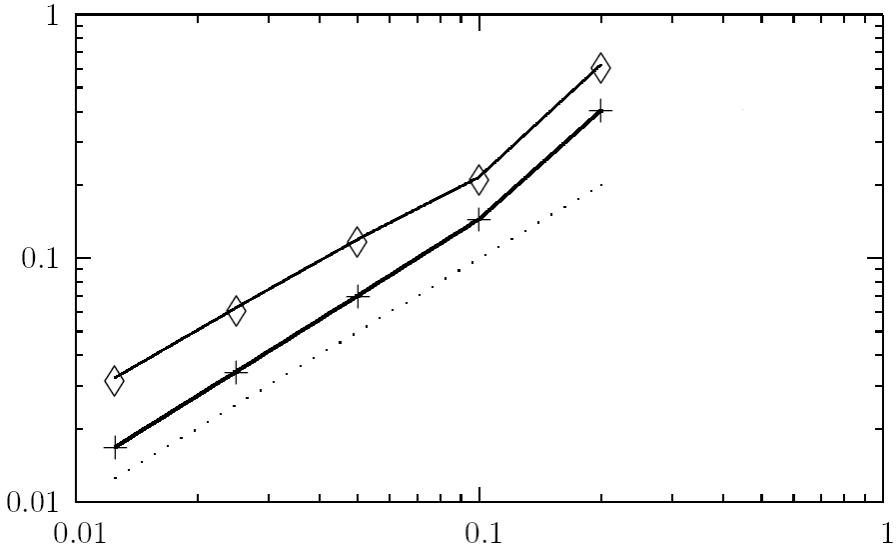

Figure 8. Validation of implementation for a simple test case. L^2 error for the glacier height H with respect to the mesh size h when setting $\Delta t = h$. Diamonds : the glacier height is computed using eq. (25); crosses : the glacier height is computed using using SUPG; dots : slope proportional to h.

The analysis and simulation of the horizontal velocities u, v has already been validated in [22]. Our goal is to validate the coupling between the velocity and the shape of the glacier. The numerical procedure is tested with an exact solution. We choose the horizontal velocity components $u(x, y, z) = yz$ and $v(x, y, z) = xz$ and the glacier height

$$H(x, y, t) = 1 - \frac{x^2}{(1+t)^2} - y^2.$$

The glacier support $\Lambda(t)$ is then the elliptic domain centered at the origin and with large and small axis equal to $1 + t$ and 1, respectively. Then we set the corresponding right-hand sides in (2.1) (2.2) and (2.11). We run the simulation from time $t = 0$ to $t = t^M = 0.5$. Let e be the L^2 error for the glacier height H at final time :

$$e = \left(\int_{\Lambda^M} (H(t^M) - H^M)^2 dxdy \right)^{1/2}.$$

In Figure 8 we have reported the error e with respect to the mesh size h when setting the time step $\Delta t = h$. Numerical results show that the L^2 error for the glacier height H at final time is of order h. Also, we have compared the formulation (3.13) to the classical SUPG formulation of [5]. The SUPG scheme is more accurate but both methods seem to display the same convergence order. However, the SUPG method has not been used for the numerical simulation of Storglaciaren hereafter since it produces oscillations of the free surface which prevents robust simulations to be performed.

4.2. **Numerical simulation of Storglaciaren between 1959 and 1990**

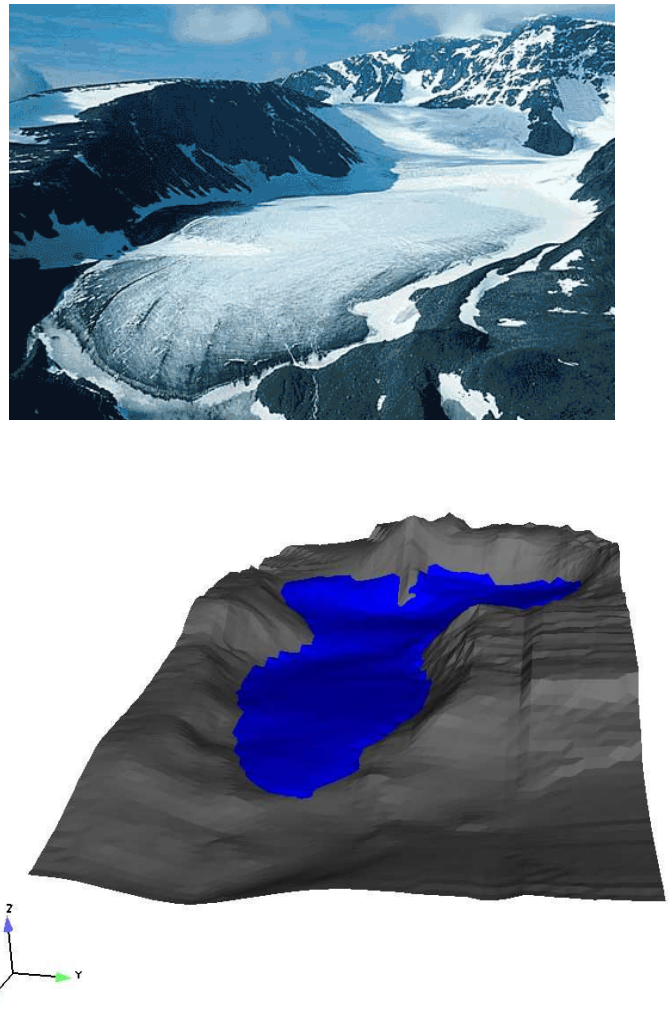

Figure 9. Top : Picture of Storglaciären (Tarfala, Sweden) http://www.natvet.su.se/aktuellt/tarfala.html. Bottom : initial shape of the glacier in 1959.

We test our numerical procedure for a three dimensional glacier corresponding to Storglaciären (Sweden), see Fig. 9. Experimental data is available for the surface velocity in a small patch located at the middle of the glacier [16, 17]. We set $n = 3$ and $T_0 = \sqrt{0.1}$ as in [21] and tune the rate factor A so that the computed velocity best fits the measured one. We obtain $A = 0.08$, which is close to the value found in [1].

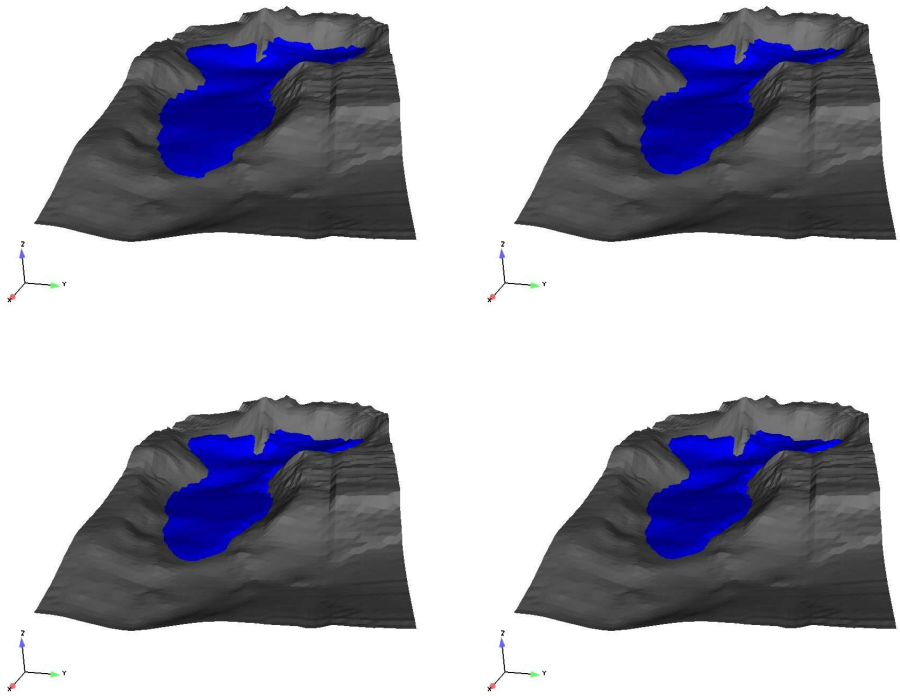

Figure 10. Numerical simulation of the retreat of Storglaciären. From left to right, top to bottom : shape of the glacier at year 1970, 1980, 1990, 2000.

We have available summer and winter mass balance b for each year, from winter 1960/61 to summer 1994. In order to obtain a constant mass

balance for each year, the summer value (negative) is simply removed to the winter value (positive). The glacier geometry was measured for the years 1959, 1969, 1980 and 1990 [16, 17]. We run our simulation starting with the glacier geometry of 1959, see Fig. 9, the mesh having 1200 vertices. For the missing mass balance b of 1959 and 1960 we use the ones of 1961. The time step is half a year, we stop the simulation in 1990 (the CPU time is less than one hour on a Pentium M processor 2.13GHz) and compare the obtained glacier shape to the measured one. The shape of the glacier during the simulation is shown in Fig. 10, the projection of the meshes onto the xy plane are reported in Fig. 11. The difference between computed and measured glacier thicknesses of years 1969, 1980 and 1990 are reported in Fig. 12 and is less than 10 meters (less than 5%), except for a small region located close to the highest point of the glacier. These results are very similar those obtained in [1], where a finite difference method based on the algorithm due to [9] was used.

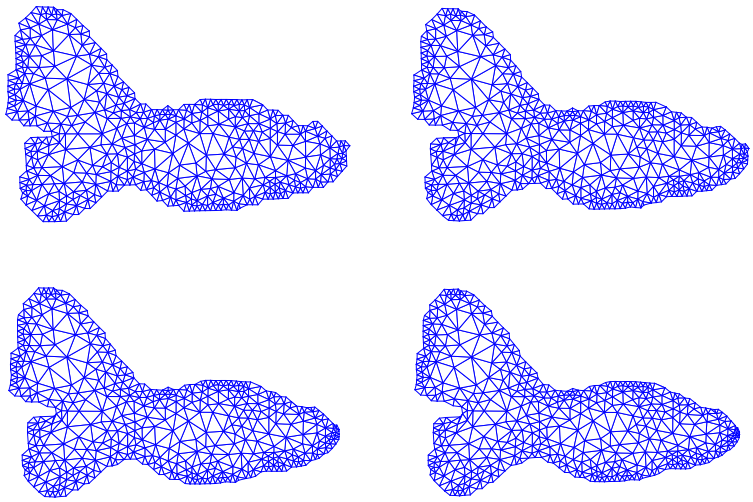

Figure 11. Meshes of Storglaciären, Sweden, view from above. The triangulation $\mathcal{T}_h(\Lambda^m)$ of the glacier support Λ^m is reported. Top left : year 1960, top right : 1970, bottom left : 1980, bottom right 1990.

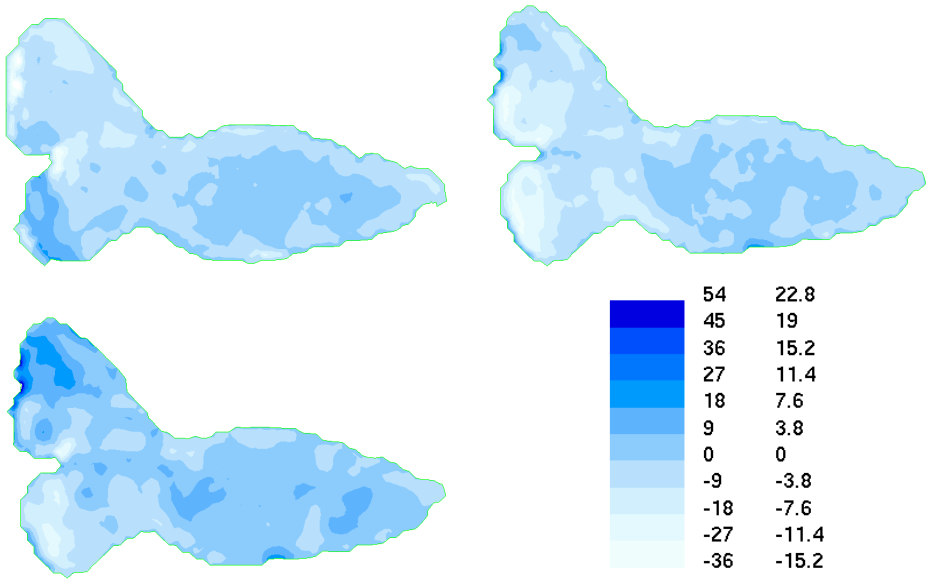

Figure 12. Differences between measured and computed ice thickness. for years 1969 (top left), 1980 (top right) and 1990 (bottom left). Positive values indicate that the measured ice thickness is larger that the computed one. The caption is in meters and %.

Acknowledgements

Heinz Blatter, Institute for Atmospheric and Climate Science, ETH-Zürich, is acknowledged for providing the model and for valuable discussions. The measurements corresponding to Storglaciären [16, 17] have been digitized by Peter Jansson, Department of Physical Geography and Quaternary Geology, Stockholm University.

References

[1] Olaf Albrecht, Peter Jansson, and Heinz Blatter, *Modelling Glacier Response to Measured Mass-Balance Forcing*, Annals of Glaciology **31** (2000), 91–96.

[2] R. Alley, P.U. Clark, P. Huybrechts, and I. Joughin, *Ice sheets and sea-level change*, Science **310** (2005), 456–460.

[3] Heinz Blatter, *Velocity and Stress Fields in Grounded Glaciers: A Simple Algorithm for Including Deviatoric Stress Gradients*, Journal of Glaciology **41** (1995), no. 138, 333–344.

[4] D. Boffi and L. Gastaldi, *Stability and geometric conservation laws for ALE formulations*, Comput. Methods Appl. Mech. Engrg. **193** (2004), no. 42-44, 4717–4739.

[5] Alexander N. Brooks and Thomas J. R. Hughes, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg. **32** (1982), no. 1-3, 199–259, FENOMECH '81, Part I (Stuttgart, 1981). MR MR679322 (83k:76005)

[6] G.F. Carey, W. Barth, J.A. Woods, B.S. Kirk, M.L. Anderson, S. Chow, and W. Bangerth, *Modelling error and constitutive relations in simulation of flow and transport*, Int. J. Numer. Meth. Fluids **46** (2004), 1211–1236.

[7] Graham F. Carey, *Computational grids*, Series in Computational and Physical Processes in Mechanics and Thermal Sciences, Taylor & Francis, Washington, DC, 1997, Generation, adaptation, and solution strategies. MR MR1483891 (99b:65161)

[8] S. Chow, G.F. Carey, and M.L. Anderson, *Finite element approximations of a glaciology problem*, Math. Model. Numer. Anal. **38** (741–756), no. 5, 2004.

[9] J. Colinge and H. Blatter, *Stress and velocity fields in glaciers: Part I. finite difference schemes for higher-order glacier models*, Journal of Glaciology **44** (1998), no. 149, 457–466.

[10] J. Colinge and J. Rappaz, *A strongly nonlinear problem arising in glaciology*, Math. Model. Numer. Anal. **33** (1999), no. 2, 395–406.

[11] K. Erikson, D. Estep, P. Hansbo, and C. Johnson, *Computational Differential Equations*, Cambridge University Press, 1996.

[12] A. C. Fowler, *A mathematical analysis of glacier surges*, SIAM J. Appl. Math. **49** (1989), no. 1, 246–263. MR MR978837 (89m:86025)

[13] ———, *Glaciers and ice sheets*, The mathematics of models for climatology and environment (Puerto de la Cruz, 1995), NATO ASI Ser. Ser. I Glob. Environ. Change, vol. 48, Springer, Berlin, 1997, pp. 301–336. MR MR1635292

[14] R. Glowinski and J. Rappaz, *Approximation of a nonlinear elliptic problem arising in a non-newtonian fluid flow model in glaciology*, Math. Model. Numer. Anal. **37** (175–186), no. 1, 2003.

[15] G.H. Gudmundsson, *A three-dimensional numerical model of the confluence area of unteraargletscher, bernese alps, Switzerland*, J. Glaciol **45** (1999), no. 150, 219–230.

[16] U. C. Herzfeld, M. G. Eriksson, and P. Holmlund, *On the Influence of Kriging Parameters on the Cartographic Output - A study in Mapping Subglacial Topography*, Mathematical Geol. **27** (1993), no. 7, 881–900.

[17] P. Holmlund, *Maps of Storglaciären and their use in glacier monitoring studies. (incl. 2 maps of the glaciers in the Tarfala valley in the scale 1:10 000)*, Geogr. Ann. **78 A** (1996), no. 2–3, 193–196.

[18] P. Huybrechts, T. Payne, and the EISMINT intercomparison group, *The eismint benchmark for testing ice-sheet models*, Annals of Glaciology **23** (1996).

[19] R. Löner and C. Yang, *Improved ALE mesh velocities for moving boundaries*, Comm. Num. Meth. Eng. **12** (1996), 599–608.

[20] C. Martin, F. Navarro, J. Otero, M.L. Cuadrado, and M.L. Corcuera, *Three-dimensional modelling of the dynamics of johnsons glacier, livingston island, antarctica*, Annals of Glaciology **39** (2004), 1–8.

[21] Marco Picasso, Jacques Rappaz, Adrian Reist, Heinz Blatter, and Martin Funk, *Numerical simulation of the motion of a two-dimensional glacier*, Int. J. Numer. Meth. Eng. **60** (2004), 995–1009.

[22] J. Rappaz and A. Reist, *Mathematical and Numerical Analysis of a Three Dimensional Fluid Flow Model in Glaciology*, M3AS **15** (2005), no. 1, 37–52.

[23] A. Reist, *Mathematical analysis and numerical simulation of the motion of a glacier*, Ph.D. thesis, Ecole Polytechnique Fédérale de Lausanne, 1015 Lausanne, 2005.

[24] I. Robertson and S. Sherwin, *Free-surface flow simulation using hp/spectral elements*, J. Comp. Phys. **155** (1999), 26–53.

Marco Picasso
Institut d'analyse et calcul scientifique,
Ecole Polytechnique Fédérale de
Lausanne, 1015 Lausanne, Switzerland
marco.picasso@epfl.ch

Jacques Rappaz
Institut d'analyse et calcul scientifique,
Ecole Polytechnique Fédérale de
Lausanne, 1015 Lausanne, Switzerland
jacques.rappaz@epfl.ch

Adrian Reist
Institut d'analyse et calcul scientifique,
Ecole Polytechnique Fédérale de
Lausanne, 1015 Lausanne, Switzerland