Project

In this project you will simulate the Swiss mountain glacier called the Arolla glacier. The ice is slowly deforming as the glacier is moving down the mountain. One of the most important benchmark experiments (i.e. experiments to test a code/model on) used in ice sheet and glacier modelling consists in modelling the ice velocity in a 2D cross section of this glacier. You will do so using a PDE called the First Order (FO) model. This is the mathematical model commonly used by NASA.

Figure 1: The Arolla glacier

In the paper attached at the end of this pdf, you find the FO model written down in equations 15-17. All notations are explained in the paper. The equations are non-linear, since the viscosity η depends on the horizontal velocity u (v_x in the paper). This means that glacial ice is a non-Newtonian fluid, just like tooth-paste - its viscosity depends the flow dynamics. Start by considering that the glacier is frozen to the underlying mountain, so you can set a homogeneous Dirichlet condition there. At the ice/atmosphere interface, the boundary condition is given by equation 19. The velocity field is dominated by the horizontal velocity u (v_x in the paper), and you will thus first focus on solving the model for u only (i.e. you will solve equation 15).

1. What is the order of equation 15?
2. Is equation (15) hyperbolic, parabolic or elliptic? Assume a constant viscosity to answer this question.
3. Write down a FEM formulation for solving equation 15 with boundary conditions. Write down the definition of the discrete space in which the test and trial functions live.
4. Implement a linear version of equation 15 in your software of choice. To achieve linearity, you can set the viscosity η to any scalar number for now e.g. 1. All other relevant parameter values are in theory given in Table 1 in the paper, but it's better to use the values in Table 1 of this other paper, as they are scaled to more favorable units (avoiding e.g. cancellation issues). If you want you can use the Poisson code that you worked with during the exercises as a starting point. See some tips and tricks for the coding at the end of the project description. Include a plot of the velocity field in the hand in.

5. For a constant η, prove the Galerkin orthogonality for equation (15)

$$\int_\Omega 2\eta\partial_x(u-u_h)\partial_x v_h + \int_\Omega \frac{1}{2}\eta\partial_y(u-u_h)\partial_y v_h = 0 \tag{1}$$

where u is the continous horizontal velocity, and u_h the FEM approximation of the horizontal velocity, and v_h is a test function. Note that the vertical coordinate y is denoted with z in the article. You can assume that ∂_h in the continous setting equals ∂_h in the discrete setting.

6. For a constant η, prove the Best Approximation Result for equation (15) in either the norm

$$2\eta\|\partial_x(u-u_h)\|_{L^2(\Omega)} + \frac{\eta}{2}\|\partial_y(u-u_h)\|_{L^2(\Omega)} \tag{2}$$

or the norm

$$\|\partial_x(u-u_h)\|_{L^2(\Omega)} + \|\partial_y(u-u_h)\|_{L^2(\Omega)} = \|\nabla(u-u_h)\|_{L^2(\Omega)} \tag{3}$$

Depending on what you find easier.

7. Make an a priori error estimate for equation (15) with constant viscosity. Estimate the error of $2\eta\|\partial_x(u-u_h)\|_{L^2(\Omega)} + \frac{\eta}{2}\|\partial_y(u-u_h)\|_{L^2(\Omega)}$.

   Hint: Depending on how you go about this, these inequalities might or not might not be useful:

   - $1 < \frac{\eta}{2}$.
   - For a positive a, b, c, d, it holds that $ad + be < (a+b)d + (a+b)e$
   - There exists a constant C so that $(a+b)^2 < C(a^2+b^2)$ or a positive a and b.
   - Young's inequality $ab < \frac{a^p}{p} + \frac{b^q}{q}$, with $\frac{1}{p} + \frac{1}{q} = 1$

8. Now you can compute the viscosity and solve the full non-linear problem (in the last lecture we will talk about non-linear problems). In the viscosity of the paper (equation 18) there is a typo, the correct formula is

$$\eta = A^{-1/3}((\partial_x u_x)^2 + \frac{1}{4}(\partial_z u_x)^2)^{-1/3}. \tag{4}$$

   The viscosity will be infinite if all derivatives of the velocity components are zero. Fix this problem by adding a small number ϵ (e.g. $\epsilon = 1e-15$). Use Picard/Fixed-point iterations, as Newton method has some convergence problems in glacier simulations.

$$\eta = A^{-1/3}((\partial_x u_x)^2 + \frac{1}{4}(\partial_z u_x)^2 + \epsilon)^{-1/3}. \tag{5}$$

   Include a plot of the velocity field in the hand in (with colorbars!) and compare it to the plots of Figure 5 of the paper.

9. How does the size of ϵ affect the convergence of the Picard iterations? Experiment and present your results in a table or a plot. Write a couple of sentences that explains your results.

0.1 Some FEniCSx tips:

- An x-derivative of a function f is obtained by taking `ufl.Dx(f, 0)` and a z-derivative by taking `ufl.Dx(f, 1)` (assuming the same coordinate system as in the paper).

- A mesh for Arolla is available on the course web page (two files, one .xdmf, one .h5, is needed). In the file `helpfunctions.py` on the course webpage you find a function which loads this mesh. The file loads the xdmf file, but the h5 file needs to be in the same folder as the xdmf-file.
- Getting access to the surface position h given the shape of the mesh is a little bit involved. Therefore there is a function for that included in `helpfunctions.py`. You can copy and paste the helpfunctions into the top of your script (or import them if you prefer).
- You will have different boundary conditions on different parts of the domain. Therefore, you need to assign a number to each part of the boundary, which can be done calling `mark_bed_surface` that you find amonst the helpfunctions `boundaries = mark_bed_surface(mesh)`. The function `mark_bed_surface` assigns number 2 to the bedrock under the glacier, and number 1 to the ice/atmosphere interface. You can use that number when you assign boundary conditions. E.g. like this:

```
boundaries = mark_bed_surface(mesh)
bed_facets = boundaries.find(2)
bed_dofs = fem.locate_dofs_topological(V, fdim, bed_facets)
bc_u = fem.dirichletbc(u_D, bed_dofs, V)
```

The deadline for this project is 16th of Janurary, 24.00. You can work individually or in groups of 2 or 3. Hand in a document with the required plots, answers to the questions, and analysis as well as your code. Put it all in a zip-file and hand in via the course webpage.

Journal of Computational Physics 330 (2017) 633–649

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A meshfree approach to non-Newtonian free surface ice flow: Application to the Haut Glacier d'Arolla

Josefin Ahlkrona, Victor Shcherbakov *,[1]

Division of Scientific Computing, Department of Information Technology, Uppsala University, Uppsala, Sweden

ARTICLE INFO

Article history:
Received 18 March 2016
Received in revised form 17 October 2016
Accepted 19 October 2016
Available online 24 October 2016

Keywords:
Ice sheet modeling
Non-Newtonian fluid
Free surface flow
Meshfree method
Radial basis function
Partition of unity

ABSTRACT

Numerical models of glacier and ice sheet dynamics traditionally employ finite difference or finite element methods. Although these are highly developed and mature methods, they suffer from some drawbacks, such as inability to handle complex geometries (finite differences) or a costly assembly procedure for nonlinear problems (finite elements). Additionally, they are mesh-based, and therefore moving domains become a challenge. In this paper, we introduce a novel meshfree approach based on a radial basis function (RBF) method. The meshfree nature of RBF methods enables efficient handling of moving margins and free ice surface. RBF methods are also accurate, easy to implement, and allow for reduction the computational cost associated with the linear system assembly, since stated in strong form. To demonstrate the global RBF method we model the velocity field of ice flow in the Haut Glacier d'Arolla, which is governed by the nonlinear Stokes equations. We test the method for different basal conditions and for a free moving surface. We also compare the global RBF method with its localized counterpart—the RBF partition of unity method (RBF-PUM)—that allows for a significant gain in the computational efficiency. Both RBF methods are compared with the classical finite element method in terms of accuracy and efficiency. We find that the RBF methods are more efficient than the finite element method and well suited for ice dynamics modeling, especially the partition of unity approach.

1. Introduction

Glaciers and ice sheets interact with the global climate system and are two of the main contributors to sea level rise [1]. The dynamics of past ice masses also shaped many of our landscapes. Numerical modeling of ice sheets and glaciers is a crucial tool both to predict future evolution of ice masses, and to understand past configurations.

Glacial ice slowly moves and deforms under its own weight, creeping down a valley or spreading over a continent. In this context, glacial ice can be described as a non-Newtonian, incompressible, very viscous fluid. The flow can be modeled by a set of nonlinear Stokes equations, and the movement of the ice–atmosphere interface constitutes a free surface problem. The nonlinearities, sensitivity of the free surface, large domains, and typically long time spans make numerical simulation computationally challenging.

Early ice sheet models were based on crude approximations of the governing equations, which where typically discretized using the finite difference (FD) method [2–4]. As computer power increased, more complex models gained popularity, such

* Corresponding author.
E-mail address: victor.shcherbakov@it.uu.se (V. Shcherbakov).
[1] The list of authors is organized in the alphabetical order. Both authors contributed equally to the study.

http://dx.doi.org/10.1016/j.jcp.2016.10.045
0021-9991/

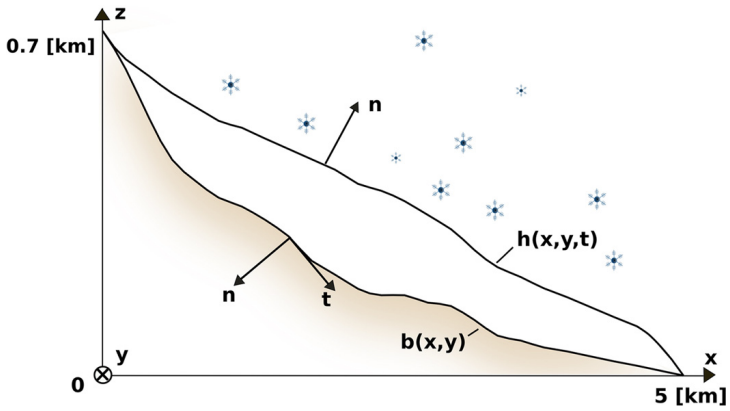

Fig. 1. The Haut Glacier d'Arolla in a Cartesian coordinate system. For aesthetic reasons, the coordinates are exaggerated in the vertical direction. The ice surface position is described by $h(x, y, t)$ and the bedrock underneath is given by $b(x, y)$. The origin of the coordinate system is 2500 m above sea level. The normal vector **n**, is pointing outwards from the ice. In two dimensions there is one tangential vector **t**, and in three dimensions two orthogonal vectors span a tangential plane.

as the first order (FO) model, also called the Blatter–Pattyn model [5,6], or even the exact model. Also the discretization schemes have changed, and an increasing number of models now use finite element (FE) methods [7–11], as these allow for complex geometries.

The finite element method, as it is implemented in ice sheet models today, does however suffer from some drawbacks. Since glaciers and ice sheets have a free surface that may change under climate conditions or due to motions of ice masses, the finite element discretization requires constant remeshing as the ice moves. This remeshing should account for both vertical movement of the surface, as well as changes in the marginal position of the ice sheet or glacier. Due to computational costs, the latter is often omitted. Another disadvantage of the finite element method is that the nonlinearity of the problem requires not only a repeated solution of a linear system (inside a nonlinear solver), but also a repeated assembly of a linear system. The cost of the assembly may severely dominate the simulation time [12].

We can avoid these issues by employing a meshfree numerical method, which does not require remeshing of the entire computational domain and avoids part of the repeated assembly phase. Rather than remeshing, a meshfree method requires placing (removing) additional computational nodes in the regions, which appear (disappear) due to displacement of the surface. For this purpose we exploit a radial basis function (RBF) method. An RBF approximation can be constructed on a set of scattered nodes and depends only upon the distances between the computational nodes. This makes the RBF approach very flexible with respect to the domain geometry and is suitable for problems defined on evolving domains.

Another valuable property of RBF methods, besides their meshfree nature, is a high convergence rate, even exponential if certain basis functions are used and the domain boundary is sufficiently regular [13,14]. This may be advantageous for large ice sheets simulations where the memory consumption starts to play an important role. However, the approximation by RBF methods, when global collocation is used, results in a system of equations with a, in contrast to FD and FE methods, dense coefficient matrix. In order to overcome this issue we implement a radial basis function partition of unity method (RBF-PUM). The partition of unity based approach allows for a substantial sparsification of the coefficient matrix. This reduces the computational effort, while maintaining a similarly high convergence rate.

The first use of RBF based methods was in cartography, geodesy and digital terrain models in order to reduce errors in data interpolation [15,16]. In glaciology radial basis functions have been used to interpolate e.g. radar data and surface elevation data [17,18]. Later RBF methods were used for solving partial differential equations [19], which now have applications in a wide range of areas, such as fluid dynamics [19–21], finance [22–24], quantum mechanics [25,26], etc. However, to the authors' knowledge, it has not been applied to ice sheet modeling so far. In this paper we aim to construct an RBF method to solve the FO model for the Haut Glacier d'Arolla, which is situated in Southern Switzerland. The method is implemented in MATLAB. Full scripts can be downloaded from http://www.it.uu.se/research/project/rbf/software/rbf_ice.

The paper is structured as follows. In Section 2, we present the equations governing glacial flow, both the exact form and the FO model, together with a discussion of computational aspects. We also present the Haut Glacier d'Arolla in this section. In Section 3, we introduce the RBF method. In Section 4, we describe the setup, discretization and results for four numerical experiments, three of which test different glaciological scenarios such as frozen basal conditions, partially sliding boundary conditions and a moving ice surface, and one which compares the global collocation approach with RBF-PUM. The RBF methods are also compared to a finite element solution. We draw conclusions in Section 5.

2. Glacier dynamics

2.1. Governing equations

We consider a glacier on a rigid bedrock topography $z = b(x, y)$, see Fig. 1. The ice surface position $z = h(x, y, t)$ is changing in time according to the glacier velocity $\mathbf{v} = (v_x, v_y, v_z)$, which is determined by the (nonlinear) Stokes equations,

$$-\nabla p + \nabla \cdot \left(\eta(\mathbf{v})(\nabla \mathbf{v} + (\nabla \mathbf{v})^T)\right) + \rho \mathbf{g} = \mathbf{0}, \tag{1}$$

$$\nabla \cdot \mathbf{v} = 0, \tag{2}$$

where p is the pressure, ρ is the density, $\rho\mathbf{g}$ is the force of gravity, and η the viscosity. The acceleration term that would have been included in the more general Navier–Stokes equations is neglected as the Reynolds number of ice is very low. For a Newtonian fluid the viscosity is constant and then (1) would be linear. For ice, the viscosity is determined by Glen's flow law,

$$\eta = \frac{1}{2}\mathcal{A}^{-1/n} d_{\text{eff}}^{-(1-1/n)}, \tag{3}$$

such that η depends on the velocity through the second invariant d_{eff} of the strain rate tensor $\mathbf{D}$

$$d_{\text{eff}} = \sqrt{\frac{1}{2}\text{tr}\mathbf{D}^2}, \quad \mathbf{D} = \frac{1}{2}\left(\nabla \mathbf{v} + (\nabla \mathbf{v})^T\right). \tag{4}$$

We assume the so-called Glen parameter, n, to take the standard value $n = 3$. The rate factor $\mathcal{A}$ depends on temperature via an exponential function. However for simplicity we consider isothermal conditions, i.e., $\mathcal{A}$ is constant. Even for a constant viscosity, the Stokes problem is computationally challenging. For a d-dimensional problem the equation system (1)–(2) has $d+1$ unknown variables, and as the pressure is absent from (2), it becomes a saddle point problem, requiring careful treatment. The nonlinear case is even more complicated, since the viscosity is dependent on the velocity field, and a nonlinear solver has to be employed. Typically the nonlinearity is resolved by fixed point or Newton iterations, repeatedly assembling and solving a linear Stokes system, and updating the viscosity in each iteration. As mentioned in the introduction, the time spent in the assembly phase in standard finite element models can dominate the solution phase [12,27]. This is because in the finite element method, the equations are stated in weak form such that the equations are multiplied by a test function $\mathbf{q}$, and integrated over the domain Ω. The non-linear term in (1) in weak form is

$$\int_\Omega \frac{1}{2}\eta(\mathbf{v})\left(\nabla \mathbf{v} + (\nabla \mathbf{v})^T\right)\left(\nabla \mathbf{q} + (\nabla \mathbf{q})^T\right) dV. \tag{5}$$

Since the viscosity is inside the integral, it is inseparable from the divergence operator and symmetric gradient, such that the entire term has to be assembled together at once in each nonlinear iteration. In the RBF method, on the other hand, the equations are stated in strong form, and thus the discrete divergence operator and the symmetric gradient operator in the nonlinear term can be represented as matrices, and the viscosity as a vector. The two matrices are independent of the viscosity, such that they can be assembled outside the nonlinear iterations and only multiplied with the viscosity vector inside the nonlinear iterations.

Another challenge is the singular nature of the constitutive law (3). For zero strain rate, d, the viscosity is infinite, which may lead to ill-conditioned matrices and slowly converging iterative solvers. The problem sizes are typically large, for example, a finite element discretization of Greenland requires a mesh of at least 300 000 elements [28]. Therefore, it is crucial to address these issues efficiently. We do this partly by employing RBF methods and partly by using a higher order approximation to the Stokes model.

After the velocity field is computed, the ice surface position is updated by solving the free surface equation

$$\partial_t h + v_x|_{z=h}\partial_x h + v_y|_{z=h}\partial_y h = v_z + a_s, \tag{6}$$

where a_s is the net accumulation/ablation at the ice surface, which depends on precipitation and surface air temperature. The solution of the free surface problem is not computationally demanding in itself, as it is only of dimensionality $d-1$. However, as the velocities enters as coefficients in (6) and are dependent on the geometry, and, thereby, h, the equation is nonlinear and sensitive to large time steps. Practical time steps span from weeks to a few years. Meanwhile, desired simulation time spans are large. For paleo-simulations, i.e., simulations performed in order to understand the past, the time span of interest is often a full glacial cycle ($\sim 100\,000$ years). In order to predict future sea level rise, simulations spanning a few centuries are sufficient, even though the initialization of the model often requires a prior paleo-simulation. Moreover, the mesh needs to be updated as the surface moves. Updating the vertical position of the ice surface can be done in a relatively efficient way in finite element models if extruded meshes are employed, while updating the geometry in the horizontal direction (the ice margins) is expensive. As a result, most of the ice sheet models keep margins fixed. However, by employing an RBF method we can avoid the constant remeshing of the domain, hence, significantly facilitating calculations.

2.2. Boundary conditions

At the ice surface, h, the atmospheric stresses are negligible, implying

$$(-p\mathbf{I} + 2\eta\mathbf{D}) \cdot \mathbf{n} = 0, \tag{7}$$

where $\mathbf{I}$ is the identity matrix. The conditions at the ice–bedrock interface, b, depend on geological, hydrological and thermal conditions which are often unknown or uncertain. If the ice base is frozen, a no slip condition

$$\mathbf{v}|_b = 0 \tag{8}$$

is appropriate. If the temperature at the ice base is above the pressure melting point and there are sediments or water present at the interface the glacier will slide. For such scenarios we employ a sliding law on the form,

$$(\mathbf{v}\cdot\mathbf{t}_i)|_b = -(\mathbf{t}_i\cdot(-p\mathbf{I}+2\eta\mathbf{D})\cdot\mathbf{n})|_b/\beta, \quad i=1,2, \tag{9}$$

$$(\mathbf{v}\cdot\mathbf{n})|_b = 0, \tag{10}$$

where β describes the friction at the base. As the conditions under an ice sheet or glacier are difficult to observe, β is often determined by inverse modeling.

2.3. The First Order Stokes model

Due to limited computer resources it has not been possible to perform simulations using the exact Stokes equations for a whole ice sheet until recently [7–10,28,29]. It is not yet possible to simulate very long time spans and large domains. Even on supercomputers we are limited to a couple of hundred years for Greenland, and even shorter times for Antarctica. To avoid large systems of equations and circumvent any issues related to the saddle point nature of the Stokes problem we employ a high order approximation to the Stokes equations—the First Order (FO) Stokes model. The FO model, also called the Blatter–Pattyn model, is a high order approximation that was originally derived by [5] and further refined by [6]. It exploits the fact that the ice body is thin, such that the ratio between the typical thickness of the ice, $[H]$, and the typical width, $[L]$, is small. Under this assumption, $\partial_x v_z$ and $\partial_y v_z$ are neglected, and $\partial_x(\eta(\partial_z v_x + \partial_x v_z))$ and $\partial_y(\eta(\partial_z v_y + \partial_y v_z))$ are considered negligible in comparison to $\partial_z(2\eta\partial_z v_z)$. These simplifications allow for decoupling the vertical component of the momentum equation in (1) from the horizontal components. Then the horizontal velocity components (v_x, v_y) can be obtained by solving the following system of equations in only $d-1$ unknowns

$$\partial_x\left(\eta(2\partial_x v_x + \partial_y v_y)\right) + \frac{1}{2}\partial_y\left(\eta(\partial_x v_y + \partial_y v_x)\right) + \frac{1}{2}\partial_z\left(\eta\partial_z v_x\right) = \rho g\partial_x h, \tag{11}$$

$$\frac{1}{2}\partial_x\left(\eta(\partial_x v_y + \partial_y v_x)\right) + \partial_y\left(\eta(2\partial_y v_y + \partial_x v_x)\right) + \frac{1}{2}\partial_z\left(\eta\partial_z v_y\right) = \rho g\partial_y h. \tag{12}$$

Once the horizontal velocity is obtained, the vertical velocity is given by the mass conservation equation (2) and the pressure is given by

$$p = -2\eta(\partial_x v_x + \partial_y v_y) + \rho g(h-z). \tag{13}$$

The FO model was shown theoretically to be second order accurate in the aspect ratio $\delta = [H]/[L]$ by [30]. It was also compared to the exact Stokes equations in numerical experiments on the Greenland Ice Sheet by [8], proving it to be highly accurate.

2.4. Haut Glacier d'Arolla

We apply the RBF method to a two dimensional simulation along the 5 km long central flow line of the Haut Glacier d'Arolla in the Swiss Alps, as it was during the Little Ice Age in 1930, see Fig. 1. This geometry was one of the test cases in a benchmark experiment designed for evaluating the accuracy of ice sheet models, the ISMIP–HOM benchmark [31]. The initial surface and bedrock topography is available on the ISMIP–HOM website [32], with a resolution of $\delta x = 100$ m. The benchmark case consists of two diagnostic simulations, one with no slip conditions at the base according to (8), and one with partial slip, such that β in (9)–(10) is

$$\begin{cases} \beta = 0, & \text{if } 2200\text{ m} \le x \le 2500\text{ m}, \\ \beta >> 1, & \text{elsewhere}, \end{cases} \tag{14}$$

that is, no slip condition is applied except for between $x = 2200$ m and $x = 2500$ m where free slip occurs. This is a challenging boundary condition, as the transition between no slip and free slip is discontinuous. In addition to these simulations, we will also let the free surface evolve in time by solving (6). The free surface will move purely due to transport, as we are not aware of any dataset describing the accumulation and ablation at Haut Glacier d'Arolla during the 1930's and therefore set $a_s = 0$ m/year, without loss of generality. The two dimensional flow line version of the FO equations determining the velocity field and ice surface position of the Haut Glacier d'Arolla, is

$$2\partial_x\left(\eta\partial_x v_x\right) + \frac{1}{2}\partial_z\left(\eta\partial_z v_x\right) = \rho g\partial_x h, \tag{15}$$

$$\partial_z v_z = -\partial_x v_x, \tag{16}$$

$$\partial_t h + v_x|_{z=s}\partial_x h = v_z, \tag{17}$$

Table 1
Parameter values for the Haut Glacier d'Arolla simulations.

Parameter	Value
Rate factor $\mathcal{A}$	10^{-16} Pa^{-n} yr^{-1}
Density ρ	910 kg m^{-3}
Gravitational constant g	9.81 m s^{-2}
Glen's parameter n	3

Table 2
Commonly used radial basis functions.

RBF	$\phi(r)$
Multiquadric (MQ)	$(1+(\varepsilon r)^2)^{1/2}$
Inverse Multiquadric (IMQ)	$(1+(\varepsilon r)^2)^{-1/2}$
Inverse Quadratic (IQ)	$(1+(\varepsilon r)^2)^{-1}$
Gaussian (GA)	$e^{-(\varepsilon r)^2}$

where the viscosity is

$$\eta = \frac{1}{4}\mathcal{A}^{-1/3}\left((\partial_x v_x)^2 + \frac{1}{4}(\partial_z v_z)^2\right)^{-1/3}. \tag{18}$$

Here the Glen parameter was set to the standard value $n = 3$. The values of all problem parameters for the Haut Glacier d'Arolla experiment follow ISMIP–HOM and are given in Table 1. The boundary conditions at the surface and the base are also simplified. By considering the FO approximation to the strain rate tensor **D** and expressing the normal and tangential vector as functions of the surface gradient $\partial_x h$ and bedrock gradient $\partial_x b$, the stress free boundary condition (7) in two dimensions reduces to

$$\eta\left(2\partial_x v_x \partial_x h - \frac{1}{2}\partial_z v_x\right) = 0. \tag{19}$$

Similarly, the slip boundary condition (9)–(10) reduces to,

$$2\eta\left(2\partial_x v_x \partial_x b - \frac{1}{2}\partial_z v_x\right) + \sqrt{(\partial_x b)^2 + 1}\beta v_x = 0. \tag{20}$$

The no slip condition is trivially $v_x = v_z = 0$.

3. Radial basis function methods

In this section we introduce the concept of radial basis function (RBF) methods. In RBF methods, the domain is discretized by scattering a set of nodes. Each node is associated with a radial basis function, whose value depends only on the distance from its center. This makes RBF methods very flexible with respect to the geometry of the domain. The solution is sought as a linear combination of the basis function. Below follows the mathematical description of the RBF method in terms of general PDE problems.

Given N distinct scattered nodes $\underline{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, $\mathbf{x}_i \in \Omega \subset \mathbf{R}^d$, the RBF interpolant, $\mathcal{J}_u$, of a function with values $\underline{u} = [u(\mathbf{x}_1), u(\mathbf{x}_2), \ldots, u(\mathbf{x}_N)]$ defined at those nodes takes the form [33,34]

$$\mathcal{J}_u(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \phi(\|\mathbf{x} - \mathbf{x}_j\|), \quad \mathbf{x} \in \Omega, \tag{21}$$

where α_j are unknown coefficients, $\|\cdot\|$ is the Euclidean norm and $\phi(r)$ is a real-valued radial basis function (RBF). A few commonly used RBFs are presented in Table 2. In all our experiments we use Multiquadric (MQ) basis functions.

In order to determine the coefficients α_j we enforce the interpolation conditions by global collocation at the node points

$$\mathcal{J}_u(\mathbf{x}_j) = u(\mathbf{x}_j), \quad j = 1, 2, \ldots N. \tag{22}$$

As a result we obtain a linear system

$$A\underline{\alpha} = \underline{u}, \tag{23}$$

where $A_{ij} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$. For the RBFs presented in Table 2 the matrix A is always invertible, that is, a solution to system (23) can always be found. Such an RBF approximation yields exponential convergence for the basis functions from Table 2 for smooth problems [13,14]. However, it is important to mention that approximation properties of RBF based methods strongly depend upon the shape parameter ε, that governs the width of the basis function. A better approximation is

usually observed when ε is a relatively small value. However, in this case the RBF matrix A becomes ill-conditioned, which leads to inaccurate calculations. In order to avoid this issue a stable evaluation technique, such as the Contour–Padé [35] or the RBF-QR method [36], can be applied.

3.1. Kansa's method

The global RBF method, also known as Kansa's method, discussed in the previous section can easily be extended to the solution of linear boundary value problems (BVPs) [19]. Consider an elliptic BVP

$$\begin{cases} \mathcal{L}u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \mathcal{F}u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases} \tag{24}$$

where $\mathcal{L}$ is the interior differential operator, $\mathcal{F}$ is the boundary differential operator, and f, g are some functions. To attack this problem numerically, we scatter N discretization nodes in Ω. Without loss of generality, we assume that the first N_I nodes belong to the interior of Ω and the last $N_B = N - N_I$ nodes belong to the boundary $\partial\Omega$. We seek a solution to system (24) in the form of the RBF interpolant (21). Collocating at the node points we obtain the following linear system

$$C\underline{\alpha} := \begin{bmatrix} L \\ F \end{bmatrix} \underline{\alpha} := \begin{bmatrix} L_{II} & L_{IB} \\ F_{BI} & F_{BB} \end{bmatrix} \begin{bmatrix} \underline{\alpha}_I \\ \underline{\alpha}_B \end{bmatrix} = \begin{bmatrix} \underline{f}_I \\ \underline{g}_B \end{bmatrix}, \tag{25}$$

where L, F, $\underline{f}$, and $\underline{g}$ are discrete representations of the continuous quantities, and the subscripts I and B denote that the quantities are evaluated in the interior and the boundary nodes, respectively. The matrices L and F consist of elements $L_{ij} = \mathcal{L}\phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$ and $F_{ij} = \mathcal{F}\phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$. For convenience, we switch from solving for $\underline{\alpha}$ to directly solving for the sought solution $\underline{u}$. By rewriting equation (23) we find

$$\underline{\alpha} = A^{-1}\underline{u}. \tag{26}$$

As it was already mentioned, A^{-1} exists for the choices of basis functions as in Table 2. Consequently, system (25) now takes the form

$$\begin{bmatrix} L_{II} & L_{IB} \\ F_{BI} & F_{BB} \end{bmatrix} \begin{bmatrix} \underline{\alpha}_I \\ \underline{\alpha}_B \end{bmatrix} = \begin{bmatrix} L_{II} & L_{IB} \\ F_{BI} & F_{BB} \end{bmatrix} A^{-1} \begin{bmatrix} \underline{u}_I \\ \underline{u}_B \end{bmatrix} = \begin{bmatrix} \underline{f}_I \\ \underline{g}_B \end{bmatrix}. \tag{27}$$

In the same manner solutions of nonlinear BVPs can be approximated by the global RBF method. Consider a nonlinear BVP

$$\mathcal{P}[\mathbf{x}, u(\mathbf{x}), \mathcal{D}u(\mathbf{x})] = 0 \quad \Rightarrow \quad \begin{cases} \mathcal{P}_1 = 0, & \mathbf{x} \in \Omega, \\ \mathcal{P}_2 = 0, & \mathbf{x} \in \partial\Omega, \end{cases} \tag{28}$$

where $\mathcal{P}_1$ is the interior nonlinear operator, $\mathcal{P}_2$ is the boundary nonlinear operator, and $\mathcal{D}$ is a shorthand notation for differential operators, such as ∂_x, ∂_y, ∇. Collocating (28) based on (21), we obtain a nonlinear system of equations

$$P(\underline{\alpha}) := \mathcal{P}[\underline{x}, \mathcal{J}_u(\underline{x}), \mathcal{D}\mathcal{J}_u(\underline{x})] = 0. \tag{29}$$

The RBF solution to the BVP (28) can then be found as $\mathcal{J}_u(\mathbf{x}; \underline{\alpha}^*)$, where $\underline{\alpha}^*$ is a root of the nonlinear system (29) that is sought by a nonlinear solver. In our particular case the nonlinear solver is a fixed point iteration method.

3.2. The radial basis function partition of unity method

The global RBF approximation results in a system of equations with a dense coefficient matrix. This implies that a significant computational effort is required to solve the system. In order to bypass this issue we employ a partition of unity technique, which was originally introduced for finite element methods in [37] and was later adopted to RBF methods [22, 24,38]. The partition of unity approach allows for significant sparsification of the coefficient matrix. Thereby, the high computational cost associated with the global method is overcome, while a similarly high accuracy is maintained. Furthermore, a partition based formulation is well suited for parallel implementations.

We construct an open cover $\{\Omega^i\}_{i=1}^M$ of Ω, where Ω^i are overlapping patches, such that

$$\Omega \subset \bigcup_{i=1}^{M} \Omega^i. \tag{30}$$

In each patch we define a local interpolant similarly to (21)

$$\mathcal{J}_u^i(\mathbf{x}) = \sum_{j=1}^{N^i} \alpha_j^i \phi(\|\mathbf{x} - \mathbf{x}_j^i\|), \quad \mathbf{x} \in \Omega, \tag{31}$$

where N^i is the number of node points, which fall inside the i-th patch. The local interpolants are combined into a global interpolant

$$\mathcal{J}_u(\mathbf{x}) = \sum_{i=1}^{M} w^i(\mathbf{x})\mathcal{J}_u^i(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{32}$$

where $\{w^i\}_{i=1}^M$ is a partition of unity subordinated to the open cover $\{\Omega^i\}_{i=1}^M$, that is, w^i is compactly supported on Ω^i and

$$\sum_{i=1}^{M} w^i(\mathbf{x}) = 1, \quad \mathbf{x} \in \Omega. \tag{33}$$

The partition of unity weight functions can be constructed using Shepard's method [39]

$$w^i(\mathbf{x}) = \frac{\varphi^i(\mathbf{x})}{\sum_{k=1}^{M} \varphi^k(\mathbf{x})}, \quad i = 1, 2, \ldots, M, \tag{34}$$

where $\varphi^i(\mathbf{x})$ is compactly supported on Ω^i. We choose $C^2(\Omega)$ compactly supported Wendland functions [40]

$$\varphi(r) = \begin{cases} (1-r)^4(4r+1), & \text{if } 0 \le r \le 1, \\ 0, & \text{if } r > 1. \end{cases} \tag{35}$$

We need at least C^2 continuity of the partition of unity weights, since the model for the glacier dynamics requires existence of the second derivative of the solution.

Collocation on (24) and (28) using (32) leads to a system of equations which, in contrast to the global method, has a sparse coefficient matrix. It is useful to bear in mind the following formulas for derivatives of the interpolant, while constructing discrete representations of differential operators.

$$\partial_x \mathcal{J}_u(\mathbf{x}) = \sum_{i=1}^{M} \left[\partial_x w^i(\mathbf{x})\mathcal{J}_u^i(\mathbf{x}) + w^i(\mathbf{x})\partial_x \mathcal{J}_u^i(\mathbf{x}) \right], \tag{36}$$

$$\begin{aligned} \partial_{xy}^2 \mathcal{J}_u(\mathbf{x}) = \sum_{i=1}^{M} \Big[& \partial_{xy}^2 w^i(\mathbf{x})\mathcal{J}_u^i(\mathbf{x}) + \partial_x w^i(\mathbf{x})\partial_y \mathcal{J}_u^i(\mathbf{x}) + \\ & \partial_y w^i(\mathbf{x})\partial_x \mathcal{J}_u^i(\mathbf{x}) + w^i(\mathbf{x})\partial_{xy}^2 \mathcal{J}_u^i(\mathbf{x}) \Big]. \end{aligned} \tag{37}$$

If (32), (36), and (37) are evaluated on a set of discrete nodes $\underline{x}$ we obtain

$$\mathcal{J}_u(\underline{x}) = \sum_{i=1}^{M} R^i W^i A^i \underline{\alpha}^i = \sum_{i=1}^{M} R^i W^i \underline{u}^i, \tag{38}$$

$$\begin{aligned} \partial_x \mathcal{J}_u(\underline{x}) &= \sum_{i=1}^{M} R^i \left[W_x^i A^i + W^i A_x^i \right] \underline{\alpha}^i = \\ &\sum_{i=1}^{M} R^i \left[W_x^i A^i + W^i A_x^i \right] (A^i)^{-1} \underline{u}^i, \end{aligned} \tag{39}$$

$$\begin{aligned} \partial_{xy}^2 \mathcal{J}_u(\underline{x}) &= \sum_{i=1}^{M} R^i \left[W_{xy}^i A^i + W_x^i A_y^i + W_y^i A_x^i + W^i A_{xy}^i \right] \underline{\alpha}^i = \\ &\sum_{i=1}^{M} R^i \left[W_{xy}^i A^i + W_x^i A_y^i + W_y^i A_x^i + W^i A_{xy}^i \right] (A^i)^{-1} \underline{u}^i. \end{aligned} \tag{40}$$

Here, R^i is a permutation operator that maps the local index set $\Xi^i = \{1, 2, \ldots, N^i\}$ corresponding to the nodes in the i-th partition into the global index set $\Xi = \{1, 2, \ldots, N\}$, and W^i, W_x^i, and W_{xy}^i are diagonal matrices with elements $w^i(\mathbf{x}_j)$, $\partial_x w^i(\mathbf{x}_j)$, and $\partial_{xy}^2 w^i(\mathbf{x}_j)$, respectively, on the diagonal. The local RBF matrix is denoted by A^i, and A_x^i and A_{xy}^i are local derivative RBF matrices with elements $\partial_x \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$ and $\partial_{xy}^2 \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$, respectively.

Table 3
Experimental setting.

	Experiment 1	Experiment 2	Experiment 3	Experiment 4
Surface BC	Fixed	Fixed	Moving	Fixed
Base BC	No slip	Partial slip	No slip	No slip
h_{fill}	Varies	0.0544	0.0544	Varies
Indist. threshold	$h_{\text{fill}}/4$	0.016	0.016	$h_{\text{fill}}/4$
Nodes	Varies	575	≈ 575	Varies
RBF	MQ	MQ	MQ	MQ
$C = \varepsilon h_{\text{fill}}$	0.24	0.25	0.22	0.24
Method	Global RBF	Global RBF	Global RBF	RBF-PUM
Partitions	–	–	–	20
Overlap	–	–	–	$0.26H$
Tolerance τ_{tol}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Time step	–	–	1 month	–

4. Numerical experiments

4.1. Setup

Four numerical experiments are conducted on the Haut Glacier d'Arolla in order to demonstrate the usefulness of RBF methods for glaciological problems, and to measure the accuracy and efficiency of RBF methods in comparison to the finite element method.

Experiment 1: The purpose of this experiment is to show that the global RBF method is suitable for computation of the velocity field on the d'Arolla glacier, and to compare its performance with the finite element method in terms of accuracy and efficiency. The setup is identical to the ISMIP–HOM E experiment [31] with no slip conditions at the base and a fixed ice surface.

Experiment 2: This experiment aims to show that RBF methods are applicable for more complex basal boundary conditions. The no slip boundary conditions are exchanged for the partial, discontinuous slip boundary condition (14) also used in he ISMIP–HOM E experiment [31]. The global RBF method is applied.

Experiment 3: This experiment demonstrates how to implement a mesh free approach to simulate a moving ice surface. The ice surface evolution is computed in a transient simulation over two years, from 1930 to 1932, with no slip conditions at the base. The global RBF method is applied.

Experiment 4: The goal of this experiment is to compare the accuracy and efficiency of the global RBF method to its local counterpart, RBF-PUM. The physical setup is identical to Experiment 1.

In order to make a fair comparison with the finite element method we implemented a finite element solver of the FO model in MATLAB. This implementation employs linear elements as this is a common choice in glaciology, and is inspired by the examples in [41].

We summarize the settings for Experiments 1–4, including the details about discretization and parameter values (further described in the following section) in Table 3. Full versions of the MATLAB code for each experiment can be downloaded from http://www.it.uu.se/research/project/rbf/software/rbf_ice.

4.2. Solution procedure and domain discretization

4.2.1. The global method on a fixed domain (Experiments 1–2)

The general procedure for obtaining the velocity profile of the Haut Glacier d'Arolla for a fixed ice surface with the global method is outlined in Algorithm 1. This procedure is used to perform Experiment 1 and Experiment 2 and is the starting point of the approach in Experiment 3 and Experiment 4.

First, the initial node set for the d'Arolla glacier is determined (see Fig. 2). To construct this set we start by defining a background grid (Step 1, Algorithm 1). In general, it can be a set of scattered nodes, but we use a quasi-uniform node set for practical reasons. The density of the background grid is defined by the so-called fill distance h_{fill}. In this context the fill distance is defined as the maximum distance between two neighboring points in the background grid. Since the domain is thin, the number of computational nodes required in the x-direction is larger than the number of nodes required in the z-dimension. Note that Fig. 2 is exaggerated in the vertical direction for aesthetic reasons. For our experiments we use N_x nodes in the x-dimension and $N_z = N_x/4$ nodes in the z-dimension. After the background nodes are defined, we add the boundary nodes, select the background points which fall within the domain and remove external points (Steps 2, 3). Also, we remove all indistinct nodes leaving just one copy (the internal nodes may coincide with the boundary nodes), as illustrated in Fig. 2 (Step 4). We identify the indistinct nodes as the nodes which lie in a small neighborhood of each other. In this study we define this small neighborhood as a disk with radius $\approx h_{\text{fill}}/4$. Finally, the nodes that are included in the computations are the remaining internal nodes combined with the boundary nodes (Step 5). The resulting set of

Algorithm 1 Experiments 1 & 2.

1: Define background node set
2: Select nodes which fall inside glacier domain
3: Remove external nodes
4: Find and erase indistinct nodes; leave only one
5: Add boundary nodes $\underline{x}_B$ to internal nodes $\underline{x}_I$
6: Compute distances between nodes
7: Assemble RBF derivative matrices
8: Identify initial guess $v_{x,\text{old}}$
9: **while** err $\geq \tau_{\text{tol}}$ **do** fixed point iterations
10: Evaluate η as in (3)
11: Define differential operator as in (15)
12: Incorporate boundary condition
13: Solve system of equations to obtain v_x
14: Compute error err $= \|v_x - v_{x,\text{old}}\|_2$
15: Update $v_{x,\text{old}} = v_x$
16: **end while**
17: Evaluate (16) to find vertical velocity v_z

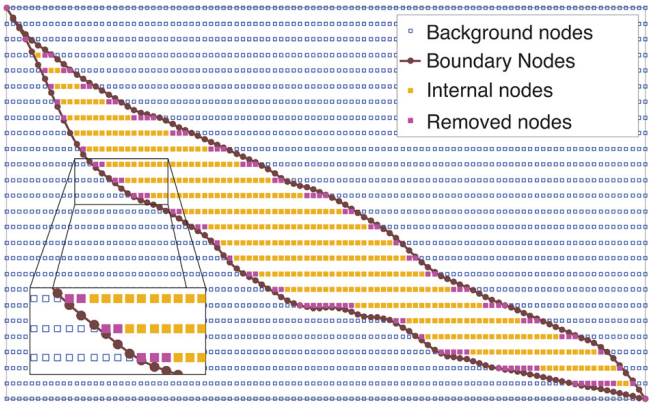

Fig. 2. The discretization of the Haut Glacier d'Arolla domain for the experiments with fixed boundaries.

computational nodes does not any more represent a uniform grid. This node set is used to conduct Experiment 1 and Experiment 2. For more details about the discretization parameters please see Table 3.

Once the computational node set has been determined, the RBF matrices are constructed given a suitable shape parameter ε (Steps 6, 7). Then, an initial guess for the velocity field and thereby viscosity is given (Step 8) and the nonlinear material law is resolved through a nonlinear iteration (Steps 9–16). Note that the main assembly is conducted outside the nonlinear solver (Steps 5, 6) in contrast to a finite element setting, where the entire assembly is executed within the nonlinear iteration since the equations are stated in weak form. Once the horizontal velocity is computed the vertical velocity (and also the pressure if needed) can be computed (Step 17).

4.2.2. Moving ice surface (Experiment 3)

For a moving surface, we need to solve the free surface equation (6) and update the computational node set as the surface moves. In each time step, the surface boundary nodes are displaced, while the background mesh remains fixed, so that some internal nodes fall out and get excluded from the computations (marked red in Fig. 3) while some external nodes get taken in (marked green in Fig. 3) and included in the computations. The differential operator also has to be updated every time step with respect to the current node set. However, we do not have to reassemble the RBF matrices every time step. The only supplementary calculations, which we have to execute in each time step, are to the find distances between the displaced boundary nodes and the current set of the internal nodes, and to assemble the parts of RBF matrices corresponding to those distances. This is a relatively small effort compared with a full matrix assembly. Thus, such an approach significantly facilitates computations. In Algorithm 2, a modified version of Algorithm 1 accommodating these changes is presented.

We begin the solving procedure by constructs a background node set in the same manner as before (Step 1, Algorithm 2). We assemble RBF matrices for the entire background grid and store them (Steps 2, 3). Then we set the free surface at its position as it was in 1930 and initialize the time integration loop (Steps 4–16). The time integration method is Euler backward. For each time step we select the internal points and dispose of the external and indistinct ones (Steps 6, 7). We pick out the parts of the "background" RBF matrices corresponding to the internal nodes (Step 8). Then we add the boundary points to the internal points to complete the computational node set (Step 9). Additionally we have to compute the distances from the internal nodes to the boundary nodes (since they were not in the background grid and are displaced every time step) and assemble the RBF matrices for them (Steps 10, 11). To construct the entire discrete differential operator we couple

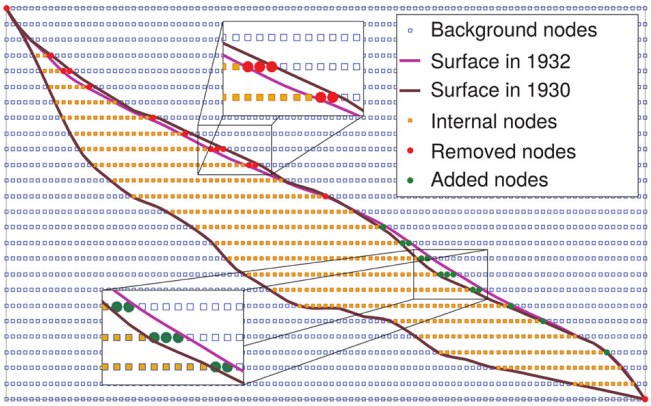

Fig. 3. Discretization of the Haut Glacier d'Arolla domain for a moving surface. Original (brown) and updated (magenta) free surface position. Green nodes are added and red removed. Note that indistinct nodes are not indicated in the figure for aesthetical reasons but they are treated as described in Section 4.2. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 2 Experiment 3.

1: Define background node set
2: Compute distances between nodes
3: Assemble RBF derivative matrices
4: Initialize surface position h
5: **for** each time step **do**
6: Select nodes which fall inside glacier domain
7: Find and erase indistinct nodes; leave only one
8: Pick part of matrices corresponding to internal nodes
9: Add boundary nodes to internal nodes $\underline{x} = [\underline{x}_I, \underline{x}_B]$
10: Compute distances between $\underline{x}_B$ and $\underline{x}$
11: Assemble RBF matrices for boundary nodes
12: Add RBF boundary matrices to matrices from step 8
13: Repeat steps 6–15 from Algorithm 1
14: Evaluate (16) to find vertical velocity v_z
15: Update surface position by solving (6)
16: **end for**

Algorithm 3 Experiment 4.

1: Repeat steps 1–4 from Algorithm 1
2: Identify partitioning
3: **for** 1 to number of partitions M **do**
4: Ascertain which nodes fall within i-th partition
5: Compute distances between nodes
6: Assemble local RBF derivative matrices
7: Compute partition of unity weight
8: Insert weighted local matrices into global matrices
9: **end for**
10: Repeat steps 6–15 from Algorithm 1

together the internal and the boundary RBF matrices (Step 12). To obtain the horizontal velocity field v_x we employ a nonlinear solver, that is the same as in the previous experiment (Step 13). The we need to solve for the horizontal velocity field v_z (Step 14) in order to be able to update the surface position (Step 15). We repeat all these calculations until we have not marched through the whole simulation time interval, that is in our case equal to two years.

4.2.3. Partition of unity (Experiment 4)

The procedure followed for the partition of unity approach is presented in Algorithm 3. We employ the same node set as in Section 4.2.1 (Step 1, Algorithm 3), but the domain is now subdivided into partitions, which we select to be of a disk shape, see Fig. 4 (Step 2). For each partition, local RBF matrices are assembled (Steps 5, 6) and combined into the global matrices according to (31) and (32) (Steps 7, 8). The remaining steps are identical to the global method.

There is a trade off between the number of partitions and the accuracy of the numerical method [22]. Typically, a low number of partitions leads to a higher accuracy, but also to more costly calculations because of a more dense structure of the system of equations. On the other hand, a large number of partitions leads to a worse approximation, but a higher computational efficiency. We chose 20 patches which gives sufficiently accurate results. The partitioning was chosen with respect to the domain geometry. The glacier is thin, and therefore it is logical to have only one layer of partitions in the vertical direction. The partition centers are quasi-uniformly distributed along the "mean" line (a line that is obtained my taking a mean value between the base and the surface positions). The overlap size is an extra degree of freedom, which can

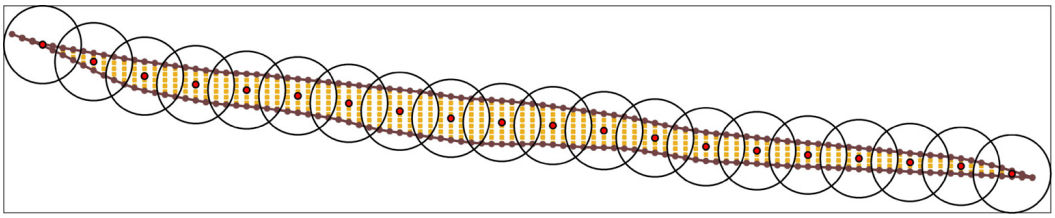

Fig. 4. The partitioning of the Haut Glacier d'Arolla domain. Unlike the other figures in this paper, the coordinates are not exaggerated in the vertical z direction in order to reflect that the patches have circular form.

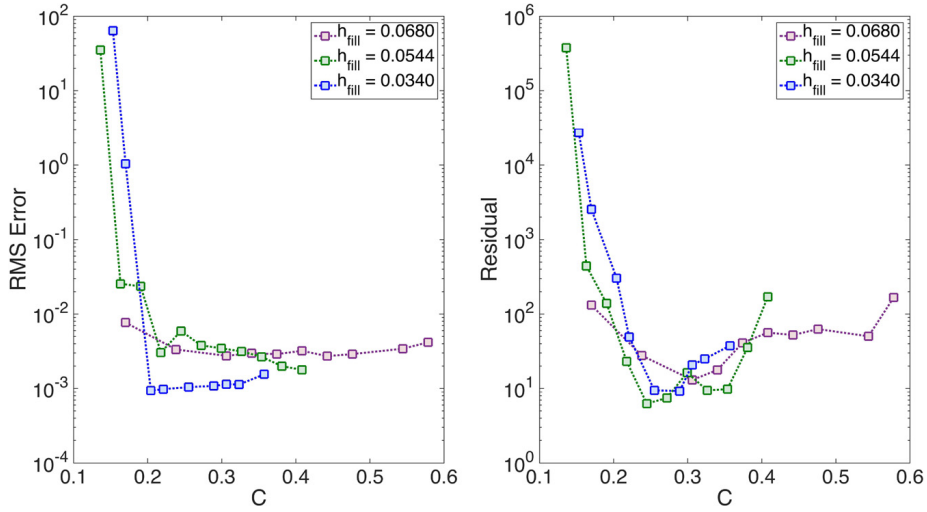

Fig. 5. Variation of the root mean square error (left panel) and the residual (right panel) with respect to $C = h/\varepsilon$ for different values of the fill distance.

be flexibly chosen. However, it should not be too large as well as not too small. We select 26% of the distance between the partition centers, H, as it is the minimal overlap size that allows to cover the domain with 20 patches uniformly distributed in x-direction.

4.3. Preparatory experiment: choice of shape parameter

The accuracy of an RBF approximation strongly depends upon the size of the shape parameter ε, that determines the width of the basis functions, as well as upon the fill distance h_{fill} [33]. Typically, the best approximation is achieved when the shape parameter is relatively small. However, in this case, the basis functions become flat, which leads to ill-conditioned systems of equations. In turn, ill-conditioning yields to inaccurate solutions. Therefore, the value of the shape parameter should be chosen with a special care. Unfortunately, the question of how to correctly select the shape parameter still remains unanswered. Commonly, the following relation is used to define the shape parameter value

$$\varepsilon = C/h_{\text{fill}}, \tag{41}$$

where C is some constant. The constant C is often chosen empirically. By assuming that ε obeys (41) we make a trade-off between conditioning and accuracy such that we can expect a robust performance (for the range of problems we consider). If there exists an analytical or an accurate reference solution, one can empirically investigate the variation of the error with respect to the shape parameter. If there is no reference solution, one can investigate the variation of the residual with respect to the shape parameter [42]. The value of the residual does not necessarily represent the qualitative characteristics of a method. However, it gives an insight into which range of values of the shape parameter is suitable.

Let us consider the problem setting from Experiment 1 with a fixed ice surface and no slip boundary condition at the base. In Fig. 5, we plot the dependence of the residual on the constant $C = \varepsilon/h_{\text{fill}}$ for three different values of the fill distance h_{fill}. In order to illustrate that this is a useful approach for ice sheet simulations we also plot the root mean square error in the solution against C for comparison. The error is computed by using the finite element reference solution obtained on a mesh with 12,167 nodes. The plots indicate that a reasonable value for the constant C is between 0.2 and 0.3. For the convergence experiments we choose the value $C = 0.24$ that gives $\varepsilon = 7$ on the finest grid. Thus, for Experiment 1 we proceed with the value of the shape parameter $\varepsilon = 0.24/h_{\text{fill}}$. For all other experiments we carry out similar tests to determine an appropriate shape parameter value. The values can be found in Table 3.

There exist stabilization methods, such as the Contour–Padé [35], the RBF-QR method [36,43], which allow for stable computations for small values of the shape parameter. However, these methods require an extra computational effort in order to find a stable basis, which becomes a significant part of the total effort for small problems ($\sim$ 1000 degrees of

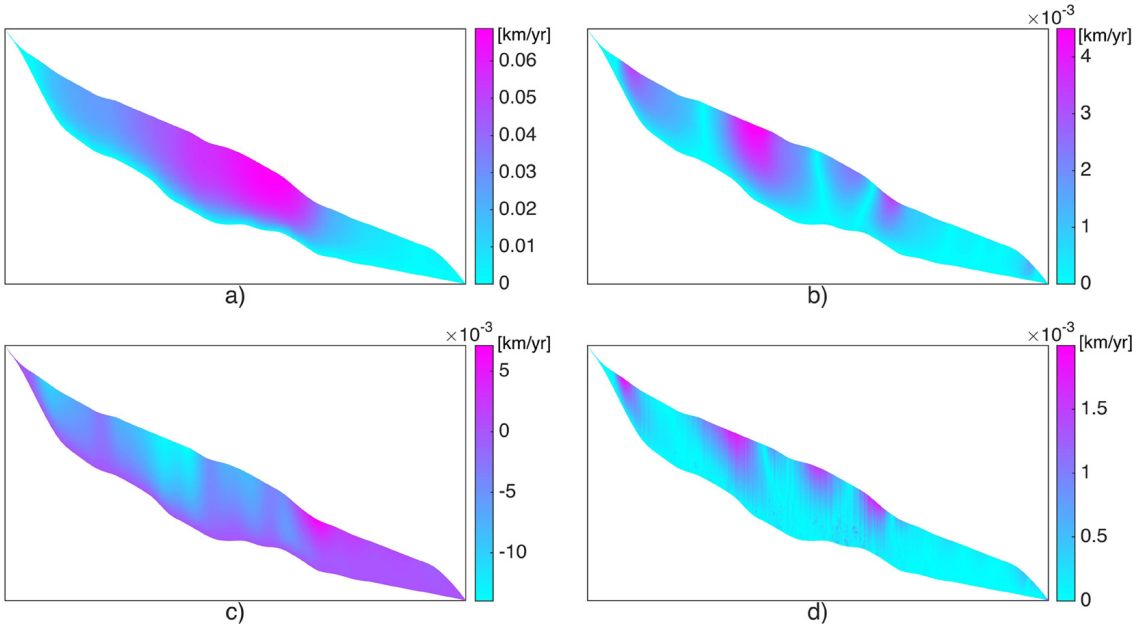

Fig. 6. Experiment 1: a) Horizontal velocity v_x obtained by the global RBF method. b) Error in the horizontal velocity. c) Vertical velocity v_z obtained by the global RBF method. d) Error in the vertical velocity. No slip conditions are applied at the base. The error was measured against a FEM reference solution computed on a mesh with 12,167 nodes.

freedom). Nevertheless, stable methods are an essential aid when problems with more than 5000 degrees of freedom are solved, since otherwise the RBF matrix becomes too ill-conditioned even for large values of the shape parameter. Because we deal with a relatively small problem, the application of stable methods goes beyond the scope of our study.

4.4. Results

4.4.1. Experiment 1: no slip boundary conditions

Fig. 6 shows the horizontal velocity v_x (panel a) and vertical velocity v_z (panel c) computed by the global RBF method, using a fill distance of $h_{\text{fill}} = 0.0544$. The error in both velocity components are shown in panels b) and d). The error was computed against the FEM reference solution defined on a mesh with 12,167 nodes.

In order to investigate the accuracy of the global RBF method we study the convergence of the method with respect to the fill distance. In addition, we study the convergence of the finite element method for varying cell size h_{cs} that is roughly equivalent to the fill distance and measured as the maximum distance between two neighboring nodes. Unfortunately, it was not possible to analytically estimate the convergence rate for the RBF method, because the theory for RBF methods has not been developed that far and, to our knowledge, there exist no available error estimates for an arbitrary geometry. However, since the domain is very irregular (the boundary is not Lipschitz continuous) and the nonlinear viscosity may have singularities we cannot expect a high order convergence. The results of the experiment are presented in Fig. 7 in the left panel. Both methods have a similar convergence rate. It is a second order convergence for the finite element method, that goes perfectly in line with the theory for linear elements, and a little higher than a second order convergence for the global RBF method.

Also we compare the global RBF method and FEM in terms of computational efficiency to achieve a certain error tolerance. In Fig. 7 in the right panel we can observe the CPU time that is required to get a certain error level. The CPU time is averaged over three runs (the program was run five times but the shortest and the longest times were disregarded, and the intermediate three were used for computations) in order to eliminate the uncertainty that is brought in by the state of the operating system. We see that the global RBF method is much faster than FEM. More details are presented in Table 4. There we can find the precise values of h_{fill} and h_{cs}, corresponding error values and CPU times as well as the number of iteration required for the nonlinear solver to converge. We see that even though FEM on average requires a similar number of nonlinear iteration the CPU time is larger than the one for the global RBF method, that is, each single iteration is cheaper for the RBF method. This is because for the finite element method a full matrix assembly has to be carried out in each iteration since it is stated in a weak form, while for the RBF method only re-evaluation of the nonlinear viscosity and a matrix–vector multiplication is required.

4.4.2. Experiment 2: partial slip boundary conditions

The partial slip condition (14) is challenging from the numerical approximation point of view due to a discontinuity in the basal boundary condition. Nevertheless, the RBF method allows for a sufficiently good resolution. The horizontal velocity field v_x is displayed in Fig. 8. The solution is computed for $h_{\text{fill}} = 0.0544$. This is a well-known benchmark test and

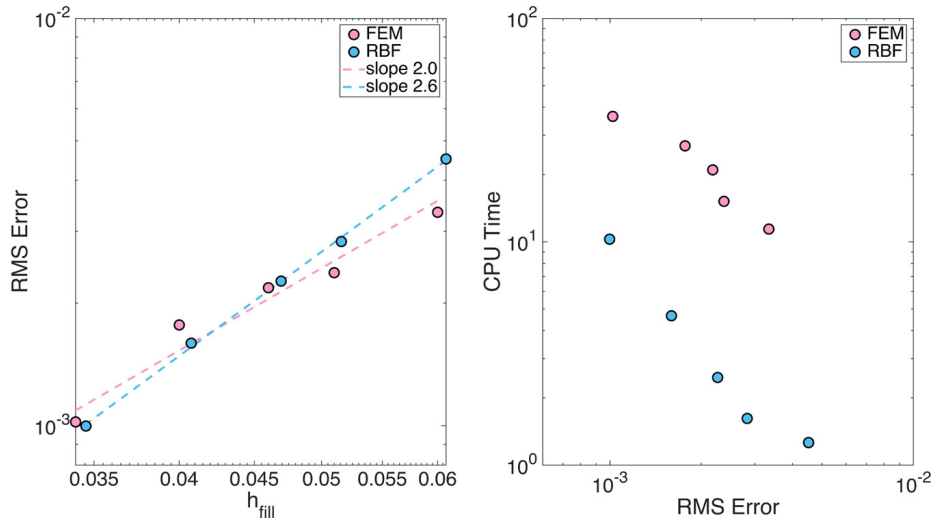

Fig. 7. Left: Convergence of FEM and the global RBF method with respect to the inter-nodal distance. Right: CPU time with respect to the error.

Table 4
Error and CPU time for several values of the inter-nodal distance for the global RBF method and FEM.

RBF					FEM				
Error	h_{fill}	N	iter	Time (s)	Error	h_{cs}	N	iter	Time (s)
0.0045	0.0608	524	29	1.2621	0.0033	0.0600	482	14	11.4055
0.0028	0.0516	711	14	1.6162	0.0024	0.0510	563	15	15.1375
0.0023	0.0469	855	13	2.4665	0.0022	0.0460	805	15	20.9487
0.0016	0.0408	1101	15	4.6602	0.0018	0.0400	972	15	26.8749
0.0010	0.0346	1505	15	10.2750	0.0010	0.0340	1251	15	36.4145

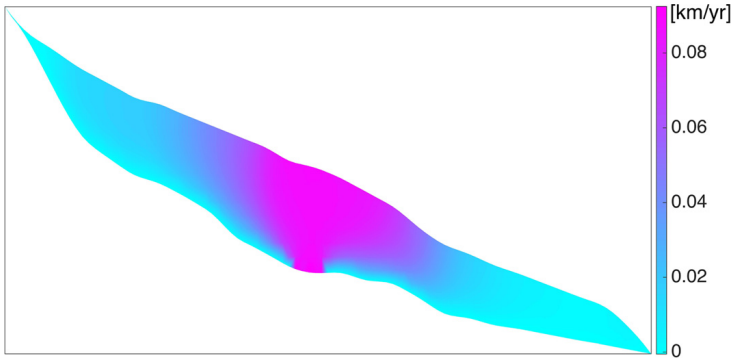

Fig. 8. Experiment 2: Horizontal velocity v_x computed with the global RBF method. Partial sliding is applied at the base.

the reader can justify that the RBF solution goes in line with the results of other studies by checking, for example, [44], where the solution is obtained by a finite element method. Other types of slip boundary conditions can as well be easily implemented in the RBF framework.

4.4.3. Experiment 3: moving ice surface

The horizontal velocity field v_x after two years (in 1932) is presented in the upper panel of Fig. 9. This test case was not a part of the original ISMIP–HOM E experiment [31]. As mentioned above, we benefit from using a meshfree approach, because the assembly of the RBF derivative matrices for the immobile background nodes can conveniently be placed outside of the time step iteration. Within the time step iteration only a small part, which corresponds to the displaced boundary nodes, has to be computed. In addition, we would like to stress that the RBF method allows for a stable moving surface evolution, which is a challenging problem in ice sheet modeling.

The lower left panel of Fig. 9 displays the surface elevation change after a two-year transient simulation. We notice that the ice thickness decreases in the upper part of the glacier and the ice accumulates in the lower part, that is, the ice masses slide down over time. Since the mathematical problem is set in the way that no ice can be released outside the domain, the total volume should be conserved. The lower right panel of Fig. 9 presents the relative volume change over the two-year

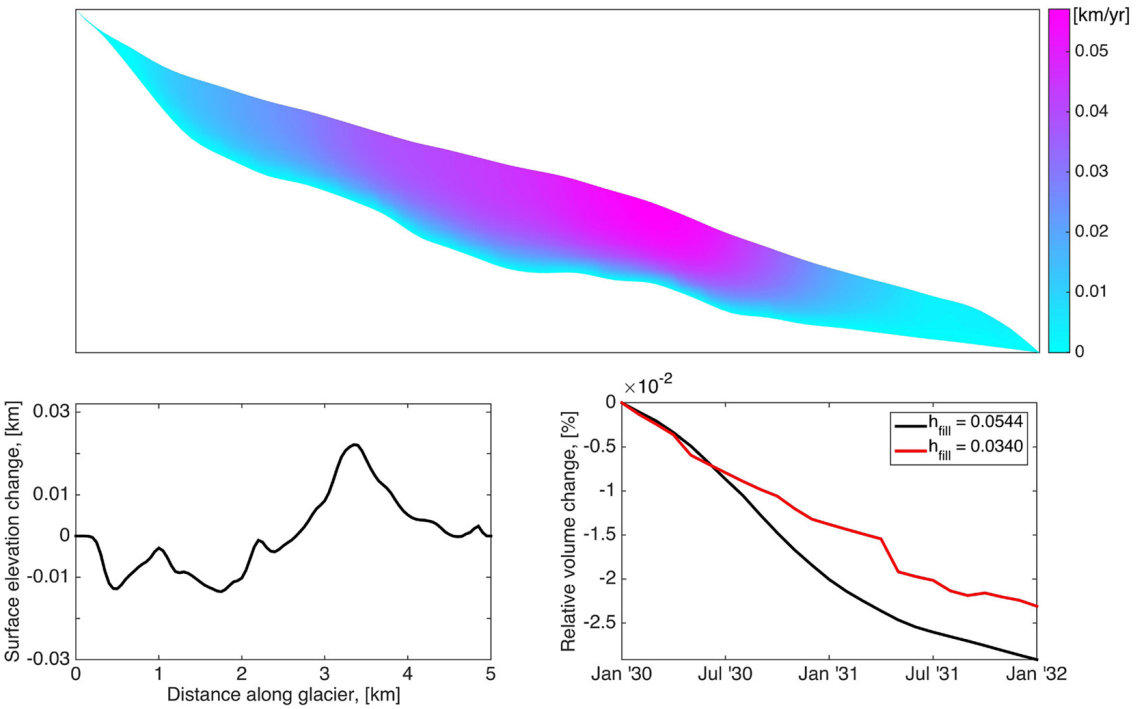

Fig. 9. Experiment 3: Glacier geometry and horizontal velocity v_x in 1932, after evolving the surface for two years (upper panel). The surface elevation change after a transient simulation over the two years (lower left panel). The volume of the glacier is preserved up to relative error of 0.03%. The relative volume change over the two-year period with respect to different values of the discretization parameter (lower right panel).

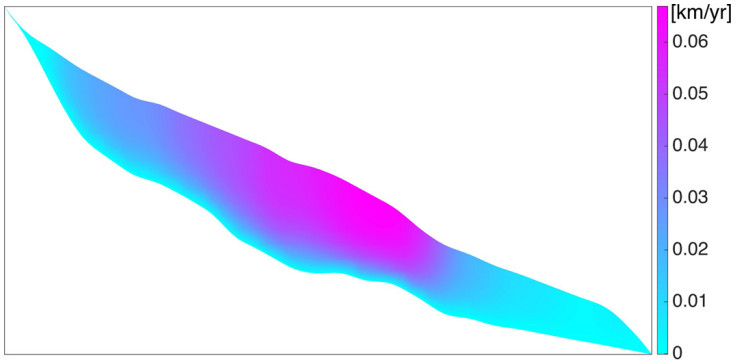

Fig. 10. Experiment 4: Horizontal velocity v_x computed with the RBF-PUM method. No slip conditions are applied at the base.

period. We observe a leakage of the volume, which happens due to numerical errors. Nevertheless, the leakage is negligible, and, furthermore, gradually diminishes under node set refinement.

4.4.4. Experiment 4: RBF partition of unity method

In Experiment 4, we consider the same settings with no slip boundary condition and fixed surface as in Experiment 1. Yet the solution method is now RBF-PUM. The horizontal velocity field v_x is computed with $h_{\text{fill}} = 0.0544$ and shown in Fig. 10. Also we compute the velocity field for varying fill distance and measure the root mean square error against the reference solution obtained on a mesh with 12,167 nodes by a finite element method. These results are used to investigate the convergence rate of RBF-PUM. In the left panel of Fig. 11 we can see a comparison between RBF-PUM and the global RBF method in terms of error versus fill distance. RBF-PUM exhibits nearly the same convergence rate that is slightly above second order. The number of partitions is kept fixed under node refinement.

The main purpose of this experiment is to illustrate the advantage of RBF-PUM over the global RBF method in terms of computational efficiency. Since RBF-PUM is a localized method, the computational effort spent on solving the system of equations is significantly less due to the sparse coefficient matrix. In the right panel of Fig. 11 as well as in Table 5 we present the execution times for the global RBF method and RBF-PUM. In order to eliminate the computer system bias we ran each code five times. From these five runs the shortest and the longest times were taken out of consideration and an average over the remaining three times is computed. Note that only the time of solving the nonlinear system is presented, while the time of the matrix assembly is disregarded for both methods. However, we emphasize that, in contrast to the global

Table 5
Error and CPU time for several values of the fill distance for the global RBF method and RBF-PUM.

RBF					RBF-PUM				
Error	h_{fill}	N	iter	Time (s)	Error	h_{fill}	N	iter	Time (s)
0.0045	0.0608	524	29	1.2621	0.0045	0.0619	500	22	0.3563
0.0028	0.0516	711	14	1.6162	0.0027	0.0523	669	17	0.4959
0.0023	0.0469	855	13	2.4665	0.0025	0.0463	814	18	0.8257
0.0016	0.0408	1101	15	4.6602	0.0015	0.0412	1002	28	2.0894
0.0010	0.0346	1505	15	10.2750	0.0013	0.0349	1348	25	3.7643

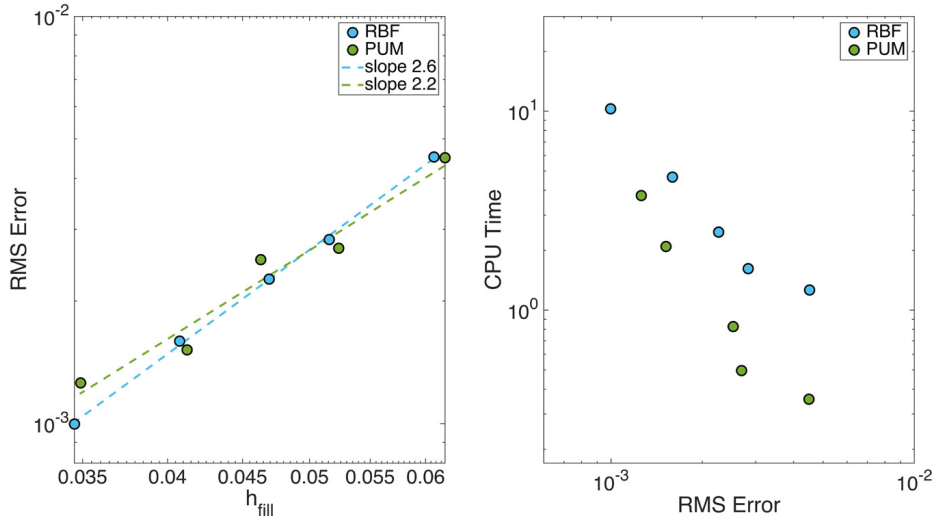

Fig. 11. Left: Convergence of the global RBF method and RBF-PUM with respect to the fill distance. Right: CPU time with respect to the error.

RBF method, the matrix assembly in the RBF-PUM case can be implemented in parallel, since all local RBF matrices can be computed independently. From the data presented in Table 5 we observe a substantial speedup for RBF-PUM compared to the global RBF method. Moreover, the advantage increases with the number degrees of freedom, if considered in terms of CPU time per one nonlinear iteration.

5. Conclusions

We have implemented the meshfree global RBF method and RBF-PUM for modeling the dynamics of the Haut Glacier d'Arolla. The underlying model was a higher order approximation to the governing nonlinear Stokes equations, the FO model. We considered several different glaciological scenarios: a transient simulation with a moving ice surface, frozen basal conditions and discontinuous partial slip basal conditions. We also compared the global RBF method and RBF-PUM with the finite element solver in terms of accuracy and computational efficiency.

We found that the RBF method is well suited for applications in glacier and ice sheet modeling. One advantage is its meshfree nature which allows for swift handling of the free surface. In a three dimensional simulation, the moving margin could be handled in the exact same way, which would be a major advantage as moving margins are one of the obstacles in modern ice sheet models. Another advantage usually associated with the RBF method is its higher accuracy properties. Due to the irregular domain of the Haut Glacier d'Arolla the order of accuracy is however only 2–3 and more research is needed to investigate if the convergence can be improved. Nevertheless, the RBF methods are significantly more efficient than a classical finite element method because the assembly phase—which dominates the simulation time for the finite element method—may be accelerated. This is thanks to the possibility to move parts of the matrix construction outside both the nonlinear iterations, resolving the non-Newtonian nature of ice, and outside the time step iteration in a transient simulation. The cost of the global RBF method associated with its dense coefficient matrix structure may be further reduced by applying the localized method RBF-PUM. Application of RBF-PUM leads to a sparse coefficient matrix resulting in more efficient computations, while maintaining a similar accuracy. Furthermore, the RBF-PUM allows for easy implementation of parallel algorithms. We also would like to stress that RBF methods can be favorable due to the ease of implementation, which may be of benefit to the glaciology community.

As a final remark the presented RBF models can be easily extended to three dimensional continental scale parallelized ice sheet simulations. However, the global RBF method may suffer from ill-conditioning when the problem size increases, and the use of stabilization techniques may increase the computational time significantly. Therefore, we recommend to adhere to RBF-PUM, where the problem with ill-conditioning will be partially resolved by localization and the cost to find a stable basis will be lower.

Acknowledgements

The authors thank Grady Wright for fruitful discussions, Elisabeth Larsson and Per Lötstedt for reading and commenting on this manuscript, and Slobodan Milovanović for help with visualizing the results. The authors would also like to thank the anonymous peer reviewers for valuable comments, which definitely improved this manuscript. Josefin Ahlkrona was supported by the Swedish strategic research program eSSENCE.

References

[1] J. Church, P. Clark, A. Cazenave, J. Gregory, S. Jevrejeva, A. Levermann, M. Merrifield, G. Milne, R. Nerem, P. Nunn, A. Payne, W. Pfeffer, D. Stammer, A. Unnikrishnan, Sea Level Change: in Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change, Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 2013, pp. 1137–1216, book section 13.
[2] P. Huybrechts, A 3-d model for the antarctic ice sheet: a sensitivity study on the glacial–interglacial contrast, Clim. Dyn. 5 (2) (1990) 79–92.
[3] R. Greve, A continuum-mechanical formulation for shallow polythermal ice sheets, Philos. Trans. R. Soc. Lond. A 355 (1997) 921–974.
[4] R. Calov, I. Marsiat, Simulations of the northern hemisphere through the last glacial–interglacial cycle with a vertically integrated and a three-dimensional thermomechanical ice-sheet model coupled to a climate model, Ann. Glaciol. 27 (1998) 169–176.
[5] H. Blatter, Velocity and stress fields in grounded glaciers: a simple algorithm for including deviatoric stress gradients, J. Glaciol. 41 (1995) 333–344.
[6] F. Pattyn, A new three-dimensional higher-order thermomechanical ice sheet model: Basic sensitivity, ice stream development, and ice flow across subglacial lakes, J. Geophys. Res. 108 (2003) 2382.
[7] H. Zhang, L. Ju, M. Gunzburger, T. Ringler, S. Price, Coupled models and parallel simulations for three-dimensional full-Stokes ice sheet modeling, Numer. Math. 4 (2011) 359–381.
[8] E. Larour, H. Seroussi, M. Morlighem, E. Rignot, Continental scale, high order, high spatial resolution, ice sheet modeling using the Ice Sheet System Model (ISSM), J. Geophys. Res. 117 (2012) F01022.
[9] N. Petra, H. Zhu, G. Stadler, T.J.R. Hughes, O. Ghattas, An inexact Gauss–Newton method for inversion of basal sliding and rheology parameters in a nonlinear Stokes ice sheet model, J. Glaciol. 58 (2012) 889–903.
[10] N. Petra, J. Martin, G. Stadler, O. Ghattas, A computational framework for infinite-dimensional Bayesian inverse problems, Part II: Newton MCMC with application to ice sheet flow inverse problems, SIAM J. Sci. Comput. 36 (2014) A1525–A1555.
[11] O. Gagliardini, T. Zwinger, F. Gillet-Chaulet, G. Durand, L. Favier, B. de Fleurian, R. Greve, M. Malinen, C. Martín, P. Råback, J. Ruokolainen, M. Sacchettini, M. Schäfer, H. Seddik, J. Thies, Capabilities and performance of Elmer/Ice, a new generation ice-sheet model, Geosci. Model Dev. 6 (2013) 1299–1318.
[12] J. Ahlkrona, P. Lötstedt, N. Kirchner, T. Zwinger, Dynamically coupling the non-linear stokes equations with the shallow ice approximation in glaciology: description and first applications of the ISCAL method, J. Comput. Phys. 308 (2016) 1–19.
[13] E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—I surface approximations and partial derivative estimates, Comput. Math. Appl. 19 (8–9) (1990) 127–145.
[14] C. Rieger, B. Zwicknagl, Sampling inequalities for infinitely smooth functions, with applications to interpolation and machine learning, Adv. Comput. Math. 32 (1) (2010) 103–129.
[15] R.L. Hardy, Multiquadric equations of topography and other irregular surfaces, J. Geophys. Res. 76 (8) (1971) 1905–1915.
[16] R.L. Hardy, Theory and applications of the multiquadric–biharmonic method 20 years of discovery 1968–1988, Comput. Math. Appl. 19 (8) (1990) 163–208.
[17] R.C.A. Hindmarsh, E.C. King, R. Mulvaney, H.F.J. Corr, G. Hiess, F. Gillet-Chaulet, Flow at ice-divide triple junctions: 2. Three-dimensional views of isochrone architecture from ice-penetrating radar surveys, J. Geophys. Res. 116 (F2) (2011) 1–14.
[18] T.C. Sutterley, I. Velicogna, E. Rignot, J. Mouginot, T. Flament, M.R. van den Broeke, J.M. van Wessem, C.H. Reijmer, Mass loss of the Amundsen Sea Embayment of West Antarctica from four independent techniques, Geophys. Res. Lett. 41 (23) (2014) 8421–8428, 2014GL061940.
[19] E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations, Comput. Math. Appl. 19 (8–9) (1990) 147–161.
[20] E.J. Fuselier, V. Shankar, G.B. Wright, A high-order radial basis function (RBF) Leray projection method for the solution of the incompressible unsteady Stokes equations, Comput. Fluids 128 (2016) 41–52.
[21] H. Wendland, Divergence-free kernel methods for approximating the Stokes problem, SIAM J. Numer. Anal. 47 (4) (2009) 3158–3179.
[22] V. Shcherbakov, E. Larsson, Radial basis function partition of unity methods for pricing vanilla basket options, Comput. Math. Appl. 71 (1) (2016) 185–200.
[23] G.E. Fasshauer, A.Q.M. Khaliq, D.A. Voss, Using mesh free approximation for multi-asset American option problems, J. Chin. Inst. Chem. Eng. 27 (4) (2004) 563–571.
[24] A. Safdari-Vaighani, A. Heryudono, E. Larsson, A radial basis function partition of unity collocation method for convection–diffusion equations, J. Sci. Comput. 64 (2) (2015) 341–367.
[25] M. Dehghan, A. Shokri, A numerical method for two-dimensional Schrödinger equation using collocation and radial basis functions, Comput. Math. Appl. 54 (1) (2007) 136–146.
[26] K. Kormann, E. Larsson, An RBF-Galerkin Approach to the Time-Dependent Schrödinger Equation, Tech. Rep. 2012-024, Department of Information Technology, Uppsala University, 2012.
[27] K. Valen-Sendstad, A. Logg, K.-A. Mardal, H. Narayanan, M. Mortensen, A comparison of finite element schemes for the incompressible Navier–Stokes equations, in: Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 399–420.
[28] H. Seddik, R. Greve, T. Zwinger, F. Gillet-Chaulet, O. Gagliardini, Simulations of the Greenland ice sheet 100 years into the future with the full Stokes model Elmer/Ice, J. Glaciol. 58 (2012) 427–440.
[29] F. Gillet-Chaulet, O. Gagliardini, H. Seddik, M. Nodet, G. Durand, C. Ritz, T. Zwinger, R. Greve, D.G. Vaughan, Greenland ice sheet contribution to sea-level rise from a new-generation ice-sheet model, Cryosphere 6 (2012) 1561–1576.
[30] C. Schoof, R. Hindmarsh, Thin-film flows with wall slip: an asymptotic analysis of higher order glacier flow models, Q. J. Mech. Appl. Math. 63 (2010) 73–114.
[31] F. Pattyn, L. Perichon, A. Aschwanden, B. Breuer, B. de Smedt, O. Gagliardini, G.H. Gudmundsson, R. Hindmarsh, A. Hubbard, J.V. Johnson, T. Kleiner, Y. Konovalov, C. Martin, A.J. Payne, D. Pollard, S. Price, M. Rückamp, F. Saito, O. Souček, S. Sugiyama, T. Zwinger, Benchmark experiments for higher-order and full-Stokes ice sheet models (ISMIP-HOM), Cryosphere 2 (2008) 95–108.
[32] F. Pattyn, Website for the Ice Sheet Model Intercomparison Project for Higher-Order ice sheet Models (ISMIP-HOM), http://homepages.ulb.ac.be/fpattyn/ismip/, January 2016.

[33] G.E. Fasshauer, Meshfree Approximation Methods with MATLAB, Interdiscipl. Math. Sci., vol. 6, World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2007.
[34] B. Fornberg, N. Flyer, A Primer on Radial Basis Functions with Applications to the Geosciences, CBMS-NSF Reg. Conf. Ser. Appl. Math., vol. 87, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2015.
[35] B. Fornberg, G. Wright, Stable computation of multiquadric interpolants for all values of the shape parameter, Comput. Math. Appl. 48 (5–6) (2004) 853–867.
[36] B. Fornberg, E. Larsson, N. Flyer, Stable computations with Gaussian radial basis functions, SIAM J. Sci. Comput. 33 (2) (2011) 869–892.
[37] I. Babuska, J.M. Melenk, The partition of unity method, Int. J. Numer. Methods Eng. 40 (4) (1997) 727–758.
[38] R. Cavoretto, A. De Rossi, Spherical interpolation using the partition of unity method: an efficient and flexible algorithm, Appl. Math. Lett. 25 (10) (2012) 1251–1256.
[39] D. Shepard, A two-dimensional interpolation function for irregularly-spaced data, in: Proceedings of the 1968 23rd ACM National Conference, ACM '68, ACM, New York, NY, USA, 1968, pp. 517–524.
[40] H. Wendland, Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree, Adv. Comput. Math. 4 (4) (1995) 389–396.
[41] M. Larson, F. Bengzon, The Finite Element Method: Theory, Implementation, and Application, Texts Comput. Sci. Eng., vol. 10, Springer-Verlag, Berlin, Heidelberg, 2013.
[42] A. Cheng, M. Golberg, E. Kansa, G. Zammito, Exponential convergence and Hc multiquadric collocation method for partial differential equations, Numer. Methods Partial Differ. Equ. 19 (5) (2003) 571–594.
[43] A. Heryudono, E. Larsson, A. Ramage, L. von Sydow, Preconditioning for radial basis function partition of unity methods, J. Sci. Comput. 67 (3) (2016) 1089–1109.
[44] M. Perego, M. Gunzburger, J. Burkardt, Parallel finite-element implementation for higher-order ice-sheet models, J. Glaciol. 58 (207) (2012) 76–88.