Geosci. Model Dev., 16, 5305–5322, 2023
https://doi.org/10.5194/gmd-16-5305-2023

Geoscientific Model Development
Open Access

A parallel implementation of the confined–unconfined aquifer system model for subglacial hydrology: design, verification, and performance analysis (CUAS-MPI v0.1.0)

Yannic Fischler[1], **Thomas Kleiner**[2], **Christian Bischof**[1], **Jeremie Schmiedel**[4], **Roiy Sayag**[4], **Raban Emunds**[1,2], **Lennart Frederik Oestreich**[1,2], **and Angelika Humbert**[2,3]

[1]Department of Computer Science, Technical University Darmstadt, Darmstadt, Hesse, Germany
[2]Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung, Bremerhaven, Bremen, Germany
[3]Faculty of Geosciences, University of Bremen, Bremen, Germany
[4]Department of Environmental Physics, BIDR, Ben-Gurion University of the Negev, Sde Boker, Israel

Correspondence: Yannic Fischler (yannic.fischler@tu-darmstadt.de)

Received: 22 December 2022 – Discussion started: 23 January 2023
Revised: 27 July 2023 – Accepted: 27 July 2023 – Published: 15 September 2023

Abstract. The subglacial hydrological system affects (i) the motion of ice sheets through sliding, (ii) the location of lakes at the ice margin, and (iii) the ocean circulation by freshwater discharge directly at the grounding line or (iv) via rivers flowing over land. For modeling this hydrology system, a previously developed porous-media concept called the confined–unconfined aquifer system (CUAS) is used. To allow for realistic simulations at the ice sheet scale, we developed CUAS-MPI, an MPI-parallel C/C++ implementation of CUAS (MPI: Message Passing Interface), which employs the Portable, Extensible Toolkit for Scientific Computation (PETSc) infrastructure for handling grids and equation systems. We validate the accuracy of the numerical results by comparing them with a set of analytical solutions to the model equations, which involve two types of boundary conditions. We then investigate the scaling behavior of CUAS-MPI and show that CUAS-MPI scales up to 3840 MPI processes running a realistic Greenland setup on the Lichtenberg HPC system. Our measurements also show that CUAS-MPI reaches a throughput comparable to that of ice sheet simulations, e.g., the Ice-sheet and Sea-level System Model (ISSM). Lastly, we discuss opportunities for ice sheet modeling, explore future coupling possibilities of CUAS-MPI with other simulations, and consider throughput bottlenecks and limits of further scaling.

1 Introduction

The dynamics of ice sheets in Greenland and Antarctica are highly related to the conditions at the ice base (e.g., Engelhardt and Kamb, 1997; Bindschadler et al., 2003; Hoffman et al., 2011; Siegert, 2000). At the base of Greenland, the ice is at least 33 % thawed (MacGregor et al., 2022) at the pressure melting point, with melt rates reaching as much as $0.19\,\mathrm{m\,a^{-1}}$ in the inland (Zeising and Humbert, 2021) and up to $15\,\mathrm{m\,a^{-1}}$ in outlet glaciers (Young et al., 2022). Hence, an extensive subglacial hydrological system is expected to exist. In addition, the margins of the Greenland Ice Sheet are experiencing massive surface melt in summer (Colosio et al., 2021), which is partially stored in supraglacial lakes (Schröder et al., 2020). Most of them drain eventually and deliver the water to the subglacial hydrological system rapidly (Neckel et al., 2020). As the subglacial system is hidden beneath ice hundreds to thousands of meters thick, observations are extremely sparse, and establishing a representative mathematical model is challenging.

Understanding of glacier hydrology has been developed in the past century mainly by investigating mountain glaciers. Comprehensive overviews are given by Fountain and Walder (1998) and Flowers (2015) including the involved processes, observational evidence, and numerical modeling. The need to incorporate the effect of subglacial hydrology on the lubrication of ice masses inspired so-called flux-routing (or balance-flux) schemes, which model the hydraulic potential assuming time-invariant steady-state water pressures derived from the ice overburden pressures (e.g., Budd and Jenssen, 1987; Le Brocq et al., 2009). Due to their simplicity, such thin water sheet models are still in use (e.g., Franke et al., 2021), despite their limitations in representing the system, most notably their inability to switch between distributed (inefficient) and channelized (efficient) water transport (Fig. 1). As they also do not include the water pressure, workarounds are needed to use them in sliding laws of ice sheet models (Kleiner and Humbert, 2014). A more advanced approach to simulate the inefficient system was made by Bueler and van Pelt (2015), which simulates the subglacial hydrology based on Darcy-type flow reformulated into an advection–diffusion–production equation for the evolution of the conserved water amount. An early attempt to include the effective pressure from both types, channels and cavities, in large-scale ice sheet modeling was presented by Arnold and Sharp (2002) based on the work of Fowler (1987) and Lliboutry (1978).

A major step in the formulation of subglacial hydrology models was the generalization of the system into a porous-medium sheet (e.g., Flowers and Clarke, 2002; Hewitt, 2011) resembling a multi-component hydrological system. This development resulted in more advanced mathematical models, including partial differential equations, with new numerical and computational demands. The models have in common that they solve a diffusion equation (either for the head or water layer thickness) and that they represent the components of the hydraulic system by evolving pore space: thus, the geometry of the drainage space (typically opening by melt and/or ice flow and closure by ice creep). Werder et al. (2013) developed a model that uses unstructured meshes with channels at element edges and distributed flow within the element with water exchange between the two components. De Fleurian et al. (2014) use a double-continuum approach with two different porous layers, one for the distributed system and one for the efficient system, with the second being important for the seasonal evolution of the hydrological system (de Fleurian et al., 2016). While in these approaches different sets of governing equations are given for the different systems, other approaches rely on one set of governing equations for both systems. Sommers et al. (2018) evolve the water layer thickness and allow for a transition between laminar and turbulent flow by evaluating the local Reynolds number. Beyer et al. (2018), based on Darcy flow, evolve the transmissivity and introduce a confined–unconfined aquifer system.

Some of these models are incorporated into ice sheet models or coupled to ice sheet models, which led to the first coupled simulations of ice sheet hydrology for individual ice stream or glacier systems with advanced hydrology models, such as in Smith-Johnsen et al. (2020), Cook et al. (2020, 2022), and Ehrenfeucht et al. (2023). However, future projection simulations for Greenland (Goelzer et al., 2020) and Antarctica (Seroussi et al., 2020) do not yet include any coupled ice sheet hydrology models.

Our own work is heavily inspired by the equivalent porous-media (EPM) approach for subglacial hydrology published by de Fleurian et al. (2014, 2016), which led to the development of the confined–unconfined aquifer system model (CUAS, Beyer et al., 2018) and its prototype implementation written in the Python language. This EPM approach offers a numerically relatively inexpensive way of incorporating different elements of the hydrological system, such as a thin water sheet, channels, cavities, and water transport within the subglacial sediments. The concept is sketched in Fig. 1. Those elements are not resolved individually, but are represented as an effective transmissivity of the equivalent porous media (right side of Fig. 1): hence, the ability to transport water.

While the main intention for developing a hydrological model was the influence on the dynamics of the ice sheet via sliding, other disciplines benefit from these simulations, too. Oceanographers are dealing with the simulated water flux across the grounding line or glacier terminus as freshwater input into the ocean–fjord system. Hydrologists need freshwater flux to estimate the discharge available for hydropower as in Ahlstrøm et al. (2009) and Braithwaite and Højmark Thomsen (1989).

Ice sheet dynamics significantly contribute to sea level rise, and projections show that this is going to increase in the next centuries (Nowicki et al., 2016; Goelzer et al., 2020; Seroussi et al., 2020). These projections are typically until the year 2100 as climate forcing for such simulations is available for this time period. As the subglacial hydrological system is having a big influence on the sliding of ice over the bed, projections of the contribution of ice sheets to sea level change will benefit from simulations of the subglacial system for the same time period. In order to solve problems such as the projection of the change in the hydrological system until 2100, reflecting the effects of alteration in the seasonal meltwater supply to the base, we need to compute many time steps at fine resolution, in particular around the ice sheet margins, where seasonal meltwater from the ice surface reaches the base. To be able to handle such large systems, we need simulation software capable of using parallel computers efficiently for their aggregate memory and computing power.

Therefore, we developed CUAS-MPI a parallel implementation of the Confined–Unconfined Aquifer Scheme (CUAS) presented by Beyer et al. (2018). The new implementation is written in C/C++ employing process parallelism using the Message Passing Interface (MPI) to enable the use of high-performance computers. CUAS-MPI allows file input and output in the Network Common Data Format (NetCDF, Rew and Davis, 1990) and uses the Portable, Extensible Toolkit for Scientific Computation (PETSc, to handle grids and equation systems; Balay et al., 2021). We validate the numerical results of CUAS-MPI with known analytical solutions of pumping tests in hydrology. Specifically, we consider solutions to flows in a confined aquifer setup (Theis, 1935; Ferris et al., 1962) and in an unconfined aquifer setup (Jacob,

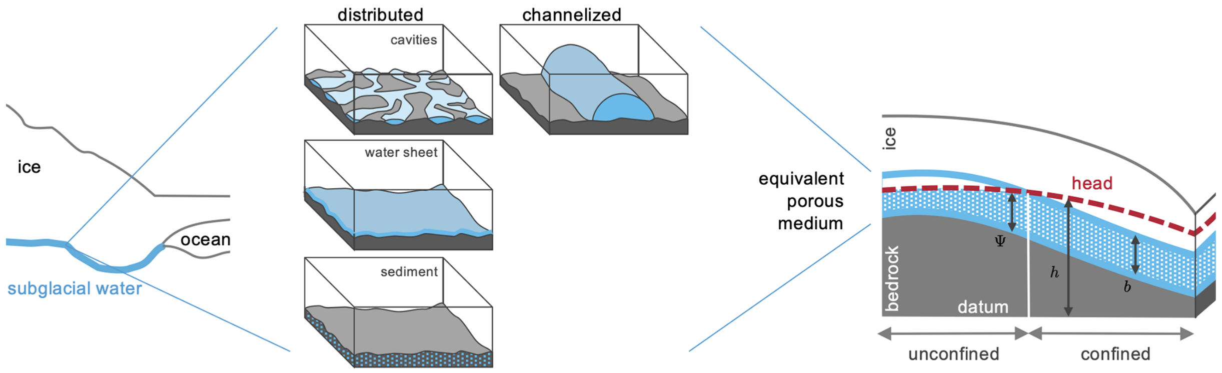

Figure 1. Modeling concept of CUAS-MPI. The left side shows the ice sheet and the underlying subglacial system. In the middle the different forms of the subglacial system are shown (after Benn and Evans, 2010). The right part represents the equivalent porous-media approach and selected variables of the model (Sect. 2.1).

1963). We show that the numerical results of CUAS-MPI are consistent with the analytical predictions for both the confined and unconfined cases and for different boundary conditions, indicating an accurate and credible implementation of the numerical scheme for those cases. On a realistic setup of Greenland, we then employed CUAS-MPI with up to 3840 processes, gathering runtime data on relevant code building blocks with Score-P (Knüpfer et al., 2012). The performance results of this model setup demonstrate that the CUAS-MPI software is able to harness the power of parallel computing to enable the simulation of relevant and challenging ice models.

The paper is structured as follows: we explain the underlying model and the software design of CUAS-MPI in Sect. 2. In Sect. 3 we describe a pumping test model problem where analytical solutions are known and compare them to the results of CUAS-MPI. We then describe a realistic Greenland setup in Sect. 4.1 and use this setup in an investigation of the performance and scaling behavior of CUAS-MPI in the remainder of Sect. 4. Finally, we discuss the model and its performance and conclude our work.

2 Implementation of CUAS-MPI

2.1 Description of the physical model

The subglacial hydrological system is simulated by a single EPM layer of thickness b between two confining layers – the ice sheet and the glacier bed (Fig. 1). We use the hydraulic head h, also recognized as the piezometric head, to express the water pressure p_w, where $h = p_w/(\rho_w g) + z_b$, with water density ρ_w, the acceleration due to gravity g, and the bed topography z_b. The void space in the aquifer is considered saturated, and thus the water transport can be described by Darcy flow. We use $\Psi = h - z_b$ and distinguish between confined ($b \leq \Psi$) and unconfined ($0 \leq \Psi < b$) conditions. Unconfined conditions may happen if the water supply from, e.g., ice sheet basal melt, is not sufficient to keep the water system fully saturated. In this case, Ψ describes the saturated thickness of the aquifer in which Darcy flow still applies.

The aquifer is described by its transmissivity T, a measure of the rate at which the water can spread. Very efficient water transport (high transmissivity) is thought to take place through a channelized system, while a distributed system is known to lead to inefficient transport (low transmissivity). An increase in T can be thought to be caused by an increase in the number of channels or an increase in the cross-sectional area of existing channels, although we do not distinguish between the two in our continuum description. The storativity S is another property of the aquifer and is the volume of water released from storage per unit surface area per unit decrease in the hydraulic head. Switching between the confined and unconfined aquifer conditions is facilitated by the effective variables for storativity, S_e, and transmissivity T_e. In the following, we summarize the relevant equations based on the arguments discussed in Ehlig and Halepaska (1976) and Beyer et al. (2018). The symbols and parameters used in the model are given in Table 1.

The vertical integrated mass balance for Darcy systems is given by Ehlig and Halepaska (1976):

$$S_e(h)\frac{\partial h}{\partial t} = -\nabla \cdot \mathbf{q} + Q = \nabla \cdot (T_e(h)\nabla h) + Q, \tag{1}$$

with the water input Q and the depth-integrated water flux $\mathbf{q}$. The effective storativity reads as $S_e(h) = S_s b + S'(h)$, with the specific storage S_s and

$$S'(h) = \begin{cases} 0, & b \leq \Psi \\ (S_y/d)(b - \Psi), & b - d \leq \Psi < b \\ S_y, & 0 \leq \Psi < b - d, \end{cases} \tag{2}$$

in which S_y is the yield storage, the ratio of water volume per unit volume which gets released once the aquifer drains. To allow a smooth transition between the confined and unconfined system, the parameter d, with $0 \leq d \leq \Psi$, is introduced (Ehlig and Halepaska, 1976). For the experiments presented here, we use $d = 0\,\mathrm{m}$ and thus do not apply smoothing in the vertical direction.

The effective transmissivity varies according to

$$T_\mathrm{e}(h) = \begin{cases} T, & b \le \Psi \\ \dfrac{T}{b}\Psi, & b > \Psi. \end{cases} \tag{3}$$

As soon as the head sinks below the aquifer height, only the saturated section contributes to the estimation of the transmissivity. The temporal change in transmissivity is computed by

$$\frac{\partial T}{\partial t} = \frac{g \rho_\mathrm{w} K T}{\rho_\mathrm{i} L} (\nabla h)^2 - 2 A n^{-n} |N|^{n-1} N T + \beta |\boldsymbol{v}_\mathrm{b}| K, \tag{4}$$

where K is the hydraulic conductivity, ρ_i is the ice density, and L is the latent heat of fusion. The first term on the right-hand side represents melting, the second term creep opening and closure, and the last term the formation of cavities. The creep term incorporates the creep rate factor A, the creep exponent n, and the effective pressure N. The effective pressure is related to the ice overburden pressure $p_\mathrm{i} = \rho_\mathrm{i} g H$ as $N = p_\mathrm{i} - p_\mathrm{w}$. Cavity opening is related to the basal ice velocity v_b and a parameter β that represents the bed undulation.

We have only two state variables in the model (h and T), and all other quantities are either parameters or can be derived from the state variables using equations outlined in Beyer et al. (2018). Boundary conditions are either Dirichlet boundaries with a prescribed head or no-flow boundaries $T_\mathrm{e} \nabla h = 0$ (homogeneous Neumann boundary condition). The latter are approximated by setting the $T_\mathrm{e} = T_\text{no flow}$ outside the margin and using the harmonic mean for T_e directly on the boundary (Patankar, 1980). For all ocean grid nodes we use a very high transmissivity $T = T_\mathrm{e} = T_\mathrm{max}$.

The transient flow equation (Eq. 1) is a two-dimensional diffusion equation with nonuniform and time-dependent hydraulic diffusivity $\alpha(x, y, t) = T_\mathrm{e}/S_\mathrm{e}$ and a time-dependent source term $Q(x, y, t)$. The equation is discretized spatially on a regular square grid using a second-order central difference scheme (e.g., Ferziger and Perić, 2002). All quantities are co-located on the grid. The implementation allows for first-order approximation (fully implicit or fully explicit) and second-order (Crank–Nicolson) approximation in time for the hydraulic head. Nevertheless, only the fully implicit time stepping is used in this study to make the solution less dependent on the initial conditions for the hydraulic head. The transmissivity is updated using an explicit Euler step. The discretization of Eq. (1) leads to a system of linear equations. In each time step this system of linear equations is solved using one of the PETSc solvers configured by the user. If an iterative solver is used, convergence is decided by the decrease in the residual norm relative to the norm of the right-hand side and the initial residuum (PETSc option `-ksp_converged_use_min_initial_residual_norm`). If the linear system is written as $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$, where $\mathbf{A}$ denotes the matrix representation of a linear operator, $\boldsymbol{b}$ is the right-hand-side vector, $\boldsymbol{x}$ is the solution vector, and $\boldsymbol{r}_k = \boldsymbol{b} - \mathbf{A}\boldsymbol{x}_k$ is the residuum vector of for the kth iteration, then convergence is detected if $||\boldsymbol{r}_k||_2 < \max(\mathrm{rtol} * \min(||\boldsymbol{b}_k||_2, ||\boldsymbol{r}_0||_2), \mathrm{atol})$. The iterative solver will also stop after a maximum number of iterations (maxits) if neither rtol nor atol is reached.

2.2 Software design

The starting point of this project was a serial implementation of CUAS (Beyer et al., 2018), partially written in Python. While producing good numerical results, its performance was too low for larger setups such as for the subglacial hydrology under the whole Greenland Ice Sheet. Our new software design for CUAS-MPI is based on this earlier implementation: CUAS-MPI again uses regular two-dimensional grids and the physics kernels are implemented analogously to the Python implementation. But we designed the data structures and computation of CUAS-MPI to run in parallel on large HPC systems. Each kernel represents an individual equation of CUAS (see Fig. 2 and corresponding equations in Sect. 2.1). While a kernel reads and writes data from and to grids, the CUAS-MPI system matrix creates the equation system (matrix and vector) based on the previously computed grids. This system of linear equations is then solved by the PETSc linear solver. Finally, the solution is stored in a grid and the time stepping sequence starts over.

The order of operations in each time step (red box in Fig. 2) may seem odd at first, but under normal circumstances, the program would stop after writing the NetCDF output (3) as usual if no more time steps are required. There are two main reasons for the chosen loop structure: first, the model has only two state variables (h, T), and all derived quantities can be computed using the methods implemented in CUAS-MPI system kernels (lower blue box in Fig. 2). This is called "do some diagnostics" in the figure. We want the model output to always be consistent with the state variables without recomputing them prior to the output (e.g., ∇h is needed for Eq. 4 and later for computing the water flux in Eq. 1). Second, the model can run with minimal output (option "small" in Table 2) for many time steps. If the user later decides that additional output fields are required, a time slice from the model output can be selected using command line tools for manipulating the NetCDF file, and CUAS-MPI can be restarted from that slice configured with a zero time step length to compute only the requested derived quantities (see, e.g., option "normal" in Table 2).

The goal of our code development was to arrive at a software artifact that can perform on current HPC systems but is also maintainable in light of future HPC system evolution. To that end, we based our development on the well-known PETSc (Balay et al., 2021) parallel math library. PETSc is open-source software supported by a wide user base and, for example, part of the software stack supported

Table 1. List of symbols and parameters used in the model. The values given in the lower part of the table refer to the Greenland setup.

Symbol	Name	Value/units
h	hydraulic head	m
T	transmissivity	$\mathrm{m^2\,s^{-1}}$
S	storage	–
t	time	s
z_b	bed elevation	$\mathrm{m^{-1}}$
Q	water supply per unit area	$\mathrm{m\,s^{-1}}$
Q^*	water supply	$\mathrm{m^3\,s^{-1}}$
$\mathbf{q}$	depth-integrated water flux	$\mathrm{m^2\,s^{-1}}$
T_e	effective transmissivity	$\mathrm{m^2\,s^{-1}}$
S_e	effective storage	–
a_melt	opening by melt	$\mathrm{m^2\,s^{-2}}$
a_cavity	opening by sliding	$\mathrm{m^2\,s^{-2}}$
a_creep	opening/closure by creep	$\mathrm{m^2\,s^{-2}}$
p_w	water pressure	Pa
p_i	ice pressure	Pa
N	effective pressure	Pa
H	ice thickness	m
s	drawdown	m
r_pm	distance between the pumping well and measurement positions	m
r_im	distance between an image well and measurement positions	m
T_min	min./transmissivity	$10^{-8}\,\mathrm{m^2\,s^{-1}}$
T_max	max./transmissivity	$100\,\mathrm{m^2\,s^{-1}}$
S_s	specific storage	$9.8\times10^{-5}\,\mathrm{m^{-1}}$
S_y	specific yield	10^{-6}
b	EPM thickness	$0.1\,\mathrm{m}$
d	confined–unconfined transition	$0\,\mathrm{m}$
A	creep rate factor	$5\times10^{-25}\,\mathrm{Pa^{-3}\,s^{-1}}$
K	hydraulic conductivity	$10\,\mathrm{m\,s^{-1}}$
L	latent heat of fusion	$334\,\mathrm{kJ\,kg^{-1}}$
n	flow law exponent	3
g	gravitational acceleration	$9.81\,\mathrm{m\,s^{-2}}$
v_b	basal ice velocity	$10^{-6}\,\mathrm{m\,s^{-1}}$
β	cavity opening parameter	5×10^{-4}

by the US exascale project https://www.exascaleproject.org/research-project/petsc-tao/ (last access: 5 September 2023). Similarly, to parallelize the NetCDF output, we used the HDF5 library, an open-source product maintained by the not-for-profit HDF group. By basing our development work on these software infrastructures, we profit both from their maturity and performance on current systems, but also, in particular, from performance improvements that will be implemented by the community supporting these software libraries down the road.

In CUAS-MPI, we use PETSc with an object-oriented interface we designed to handle our matrices, vectors, and grids. In particular, we employ the distributed array (DMDA) features of PETSc for grid creation, ghost cell update, and optimal matrix and vector distribution in the context of two-dimensional structured grids. The distribution pattern is generated by `DMDACreate`. Grids and vectors are directly derived from this pattern, and the matrix is distributed and initialized accordingly using `DMCreateMatrix`. This ensures that all data structures use the same distribution, which lowers communication costs for assembly operations and the linear solver. We implemented our own grid class, which is based on PETSc DMDA vectors and provides some sanity features; e.g., the grid access is guarded by read and write handles, which automatically trigger ghost cell updates between the different parallel processes after each kernel execution. Thus, we are guaranteed not to miss necessary ghost cell updates. The solver can be selected by the user from the list of available direct and iterative PETSc solvers. We use the iterative generalized minimal residual method (GMRES) solver for the Greenland simulations. For smaller-scale tests, we also employed the direct solver MUMPS (MUltifrontal Massively Parallel sparse direct Solver; Amestoy et al., 2001, 2019).

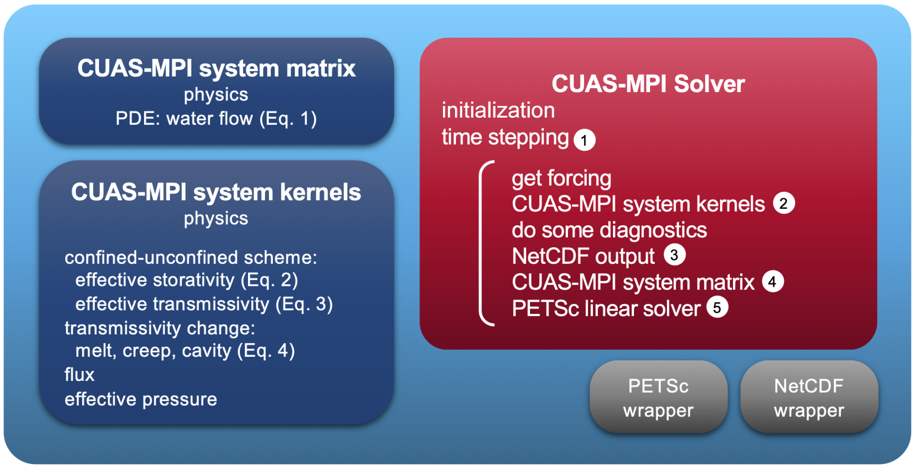

Figure 2. Components of CUAS-MPI. The physics modules of the mathematical model are shown in dark blue. The actual solver is sketched in the red box. Grey boxes denote wrappers interfacing with PETSc and NetCDF. The numbers 1–5 in white circles are used again in Sect. 4, e.g., Fig. 8, for cross-referencing.

The regular two-dimensional grids of CUAS-MPI are stored in the standardized file format NetCDF for input and output. Different libraries implement parallel read and write operations for NetCDF files: the NetCDF implementation (Rew and Davis, 1990) with HDF5, PnetCDF (Latham et al., 2003), and ParallelIO (Hartnett and Edwards, 2021). As all three parallel NetCDF libraries have advantages and disadvantages, we implement an adapter class which provides a uniform interface of the features we require in CUAS-MPI. Thereby we ensure the flexibility to switch between different I/O implementations and allow easy adaption to future developments. The abstraction layer is able to handle scalars, vectors, and grids according to the format used in CUAS-MPI and pass it to the selected NetCDF implementation. The NetCDF library is always used for reading, and we also employ it in our experiments. CUAS-MPI supports four levels of output: small, normal, large, and x-large (Table 2). The small output includes only the absolutely necessary fields. All other fields can be derived. In the three more complex output configurations we write additional analytical information, which is computed in the CUAS-MPI solver pipeline.

3 Validation of CUAS-MPI using analytical solutions

Validating the results of CUAS-MPI is an essential task to quantify the computational errors which arise while solving the continuous differential equations in a computationally discrete environment. Specifically, we check the consistency of the numerical solutions of CUAS-MPI with suitable analytical solutions. By doing so, we verify the implementation of the governing equation (Eq. 1) in the confined and unconfined case, as well as the implementation of Neumann and Dirichlet boundary conditions, and we quantify their error range. The analytical solutions on which our verification is based were developed in the field of hydrology and are commonly known as pumping tests. The problem we simulate in a pumping test involves a horizontal aquifer of uniform thickness b, constant conductivity K, and a pump of constant rate Q^* that is located at the center of the domain and that fully penetrates the aquifer. We compare the numerical results against the analytical solutions in two cases: one in which the aquifer is confined and the model is linear and another in which the aquifer is unconfined and the model is nonlinear. Both the confined and unconfined CUAS-MPI simulations were performed with the direct (MUMPS) and the iterative (GMRES) solver, and the results of the two computational methods were indistinguishable.

3.1 Confined case

For the simulation in a confined case the transmissivity is constant and the model (Eq. 1) is linear. Assuming that the aquifer has an infinite extent, the model has the analytical solution (Theis, 1935)

$$s_{\mathrm{ub}}(r_{\mathrm{pm}}) = \frac{Q^*}{4\pi T} W\left(\frac{r_{\mathrm{pm}}^2 S}{4Tt}\right), \tag{5}$$

where $s \equiv h(x,y,0) - h(x,y,t)$ is the drawdown, r_{pm} is the distance between the pump and the measurement positions, and W is the dimensionless well function. This exact solution can be tested despite the bounded numerical domain, as long as the flow remains far from the boundaries. Indeed, the CUAS-MPI results are in good agreement with the analytical prediction s_{ub} (Eq. 5) until $\approx 3 \times 10^4$ s, when the flow begins to be affected by the finite boundaries of the numerical domain (Fig. 3).

Moreover, an analytical solution that accounts for a bounded domain can be constructed based on the unbounded

Table 2. Output options of CUAS-MPI; the categories are inclusive, and each option includes the options above.

Configuration	Enabled fields for output	Size per time step	
		G600	G150
small	head, transmissivity	188 MB	3 GB
normal	bed elevation, water input, dT/dt due to channel wall melt, creep opening and cavity opening, effective pressure, flux	752 MB	12 GB
large	ice thickness, effective layer transmissivity and storativity	940 MB	15 GB
X-large	ice pressure	940 MB	15 GB

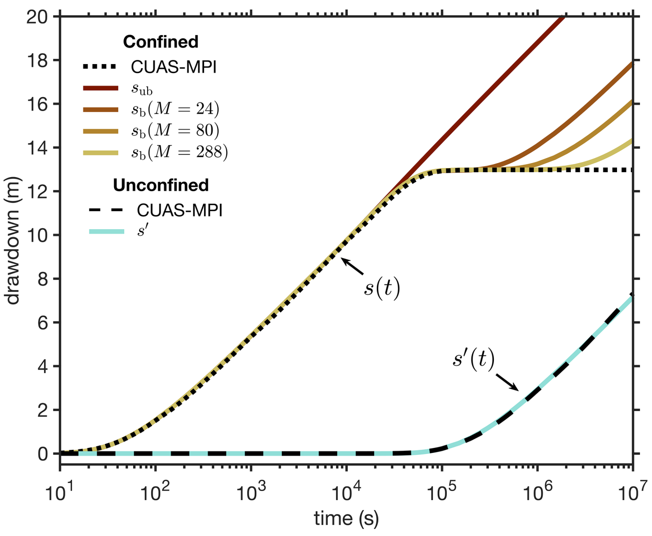

Figure 3. Comparison of the CUAS-MPI results for the drawdown to the analytical solutions for both confined and unconfined aquifers. In the confined case the results are compared with analytical solutions for an unbounded domain (Eq. 5) and with the analytical solutions for a bounded domain (Eq. 6) with a growing number of image wells ($M = 24$, 80, and 288). In the unconfined case, the results are compared with the approximated solution (Eq. 7). The point of drawdown measurement is 80 m away from the pumping well for both simulations.

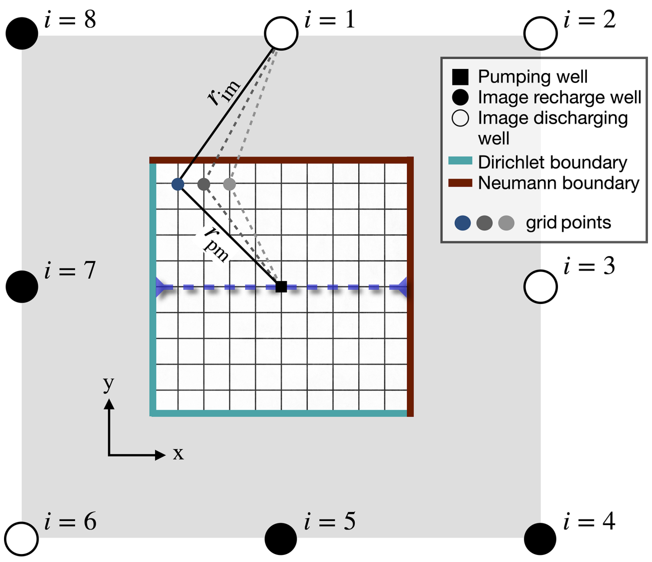

Figure 4. Configuration for numerical validation using analytical solutions (pumping test), with Dirichlet conditions at the southern and western numerical boundaries (blue) and Neumann conditions at the northern and eastern numerical boundaries (brown). The wells outside of the numerical domain (○, ●) are the image wells, each described by the unbounded analytical solution (Eq. 5), which collectively construct the solution (Eq. 6) for the bounded domain (Ferris et al., 1962). The dashed blue line indicates the profile in Fig. 5.

solution (Eq. 5) using the method of images, which can be applied due to the linearity of the model in this case (Ferris et al., 1962). Specifically, we consider the case in which a pump is at equal distance from two Dirichlet boundaries (zero drawdown) and two Neumann boundaries (zero drawdown gradient) (Fig. 4), which are the two types of boundary conditions implemented in CUAS-MPI. The analytical solution for such a configuration consists of a superposition of well solutions to image wells placed across the domain boundaries, as illustrated in Fig. 4 (Ferris et al., 1962). Therefore, the analytical solution for the drawdown in a bounded domain s_{b} is the series

$$s_{\mathrm{b}} = s_{\mathrm{ub}}(r_{\mathrm{pm}}) + \sum_{i=1}^{M} s_{\mathrm{ub}}(r_{\mathrm{im}}), \tag{6}$$

where r_{im} is the distance from the ith image well to the point of measurement. The accuracy of the solution grows with the number of image wells M.

The numerical simulation for the confined case is set up with a domain size of 2000 m × 2000 m, $b = 100\,\mathrm{m}$, $S_{\mathrm{s}} = 10^{-6}\,\mathrm{m}^{-1}$, $S = S_{\mathrm{s}}\,b$, $K = 4.16 \times 10^{-5}\,\mathrm{m\,s}^{-1}$, and an initial hydraulic head of $h(x, y, t = 0) = 300\,\mathrm{m}$. The pumping well has a constant rate of $Q^* = 0.1\,\mathrm{m}^3\,\mathrm{s}^{-1}$. We find that the CUAS-MPI results are in good agreement with the bounded analytical solution s_{b} (Eq. 6) for a period that grows longer with the number M of image wells (Figs. 3, 5).

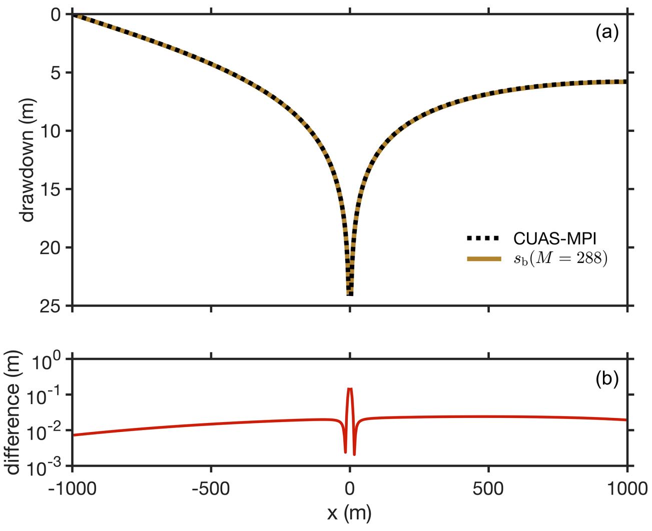

Figure 5. (a) The drawdown along the dashed blue line in Fig. 4 computed by CUAS-MPI and compared with the prediction s_b (Eq. 6), which accounts for the Dirichlet and Neumann boundary conditions on the left and right, respectively. **(b)** The difference between the computed and predicted drawdown, shown in the top panel, indicates an overall small discrepancy between the numerical and theoretical values.

3.2 Unconfined case

For the simulation in an unconfined aquifer the transmissivity is proportional to the head and consequently the model (Eq. 1) is nonlinear. To validate the results of CUAS-MPI in this case, we approximate the drawdown s' of the unconfined aquifer using the drawdown of an equivalent confined aquifer s_{ub} (Eq. 5) with $T = Kh(t = 0)$ through the relation

$$s' = b - \sqrt{b\Big(b - 2s_{\mathrm{ub}}(r_{\mathrm{pm}})\Big)}, \tag{7}$$

which is based on the Dupuit–Forchheimer assumption that the horizontal flux is greater than the vertical flux and provides a good prediction of the drawdown at large distances from the pump compared to the aquifer thickness (Jacob, 1963).

For this unconfined case, the domain size is set to 4000 m × 4000 m and the initial hydraulic head is $h(x, y, t = 0) = 99$ m. We set the specific yield to $S_y = 0.4$, which represents a subglacial hydrology system with an EPM approach (de Fleurian et al., 2014), and set the pumping well with a constant rate of $Q^* = 0.1\,\mathrm{m}^3\,\mathrm{s}^{-1}$. We find that the simulation results agree well with the approximated analytical solution for the unconfined case (Fig. 3).

4 Performance of CUAS-MPI running a representative Greenland setup

CUAS-MPI was developed to enable high-performance and high-throughput simulations on up-to-date HPC systems, which typically consist of many-core nodes connected by a high-speed network. To assess the performance of CUAS-MPI, we present performance data from CUAS-MPI for the Greenland setup described in Sect. 4.1. We run the model for 24 time steps (1 model day) using different grid resolutions to compare the model performance and scaling behavior. We use the GMRES linear solver with a Jacobi preconditioner.

We employ the compiler GCC 11.2, the message passing library OpenMPI 4.1.4, the math library PETSc 3.17.4, and the data format libraries NetCDF 4.7.4 and HDF5 1.8.22 for our performance experiments. To instrument the code for measurements, we use Score-P 7.1. Score-P is a highly scalable performance measurement infrastructure for profiling and event tracing. The code is compiled using optimization level "-O3" and native processor architecture flags "-march=cascadelake". All experiments are conducted on dedicated compute nodes of the Lichtenberg HPC system (https://www.hrz.tu-darmstadt.de/hlr/hochleistungsrechnen/index.en.jsp, last access: 5 September 2023) with two 48-core Intel Xeon Platinum 9242 processors and 384 GB of main memory each, connected with an InfiniBand HDR100 fat-tree network providing point-to-point connections between nodes. The Slurm scheduling system is used for workload management. As such, the Lichtenberg HPC system is representative of the currently most commonly used type of HPC system. Due to temporary energy saving measures, turbo mode has been disabled and the base frequency of the chips reduced by 100 MHz to 2.2 GHz. Turbo mode allows the CPU to exceed its base frequency for a short time but also considerably increases power consumption. Every experiment is repeated three times, and the average runtime is reported. For all runs of the models with 1200 m or finer resolution, we see a relative standard deviation of less than 5 %. For the coarsest 2400 m model, we observed a relative standard deviation of 15 %, which we believe to be in part due to the very short running time of the code in this case.

4.1 Description of the Greenland setup

The model domain consists of a rectangular area containing the Greenland Ice Sheet. Grid points of the hydrological system for which Eq. (1) is solved are colored red in Fig. 6. Boundary conditions are determined by the type of grid cell next to the margin grid cells of the subglacial system. In the case of a land-terminating margin, we use a zero-flux Neumann boundary condition (no flow). A transition to the ocean at the grounding line is a Dirichlet boundary condition for the head. We use $h = 0$ at the grounding line assuming the water pressure in the aquifer equals the hydrostatic ocean pressure. We ignore the slight density difference between seawater (about $1028\,\mathrm{kg\,m^{-3}}$) and fresh water ($1000\,\mathrm{kg\,m^{-3}}$). By our choice of boundary conditions, the subglacial water is drained into fjords only, as land-terminating margins prohibit outflow. This is not very realistic, as subglacial water could also feed into ice-marginal lakes and rivers and would decrease the hydraulic head (water pressure) in the subglacial system. Outflow could be indirectly simulated by applying Dirichtlet conditions for the hydraulic head at those river and ice-marginal lake locations. The model is capable of doing so, but to our knowledge, no dataset of the river and ice-marginal lake locations and fluxes for the Greenland Ice Sheet exists. Even though outflow along the land-terminating margin is not considered, we think the setup is realistic enough to draw conclusions about the model performance.

The type of grid cell (mask), ice thickness, and bedrock and surface topographies are based on a preliminary version of the BedMachine Greenland dataset as used in Christmann et al. (2021) that later became BedMachine version 4 (Morlighem, 2021; Morlighem et al., 2017). The BedMachine dataset is originally available at 150 m resolution (G150), which we regrid to 300 m (G300), 600 (G600), 1200 m (G1200), and 2400 m (G2400) resolution using conservative remapping for the geometry and nearest-neighbor interpolation for the mask. Ice sheet basal melt from the Parallel Ice Sheet Model (PISM) output (Aschwanden et al., 2016, 1200 m resolution) is regridded to the CUAS meshes using bilinear interpolation. Large areas consisting of small glaciers get a spatially uniform ice thickness of 1 m assigned in the BedMachine dataset due to insufficient data coverage. This is particularly visible along the southeastern and eastern margin of the ice sheet. Those areas are also not well resolved in the PISM. Therefore, a minimum ice thickness constraint of 10 m is applied to eliminate thin marginal areas. The observed surface velocity (MEaSUREs, Joughin et al., 2016, 2018) is used to check if areas with at least $30\,\mathrm{m\,a^{-1}}$ are included in the mask. If not, the minimum ice thickness constraint is applied and the bed elevation is adjusted to be consistent with the surface elevation from BedMachine. Further, a flood-filling algorithm (van der Walt et al., 2014) is used to select only the connected grid points from the main ice sheets without peripheral ice caps and glaciers to ensure consistent coverage from BedMachine and the PISM output. This step is important because missing data in the basal melt forcing would degrade the solver performance in CUAS-MPI.

The resulting numbers of total and active grid points are given in Table 3. Water input is the basal melt rate presented in Aschwanden et al. (2016). The model parameters (see Table 1) are mainly taken from Beyer et al. (2018) with only two exceptions. We use $S_\mathrm{y} = 10^{-6}$ instead of 0.4 and a minimum transmissivity bound T_min of 10^{-8} instead of $10^{-7}\,\mathrm{m^2\,s^{-1}}$. Those changes are found to result in a smoother hydraulic head in areas of no or only very little basal water supply. For the purpose of this study, the exact representation of the Greenlandic hydrological system is not of primary importance, as we analyze the performance of the model, but we represent the Greenland Ice Sheet sufficiently realistically to be able to infer from the outcome in this study the computational costs for other simulations. The hydraulic head is initialized to be equal to the ice overburden pressure at each grid point and the initial transmissivity is spatially uniform ($T(t=0) = 0.2\,\mathrm{m^2\,s^{-1}}$). The convergence criteria for the iterative GMRES solver are configured as rtol $= 10^{-7}$ (relative norm) and atol $= 10^{-5}$ (absolute norm) with a maximum number of iterations set to 10^5.

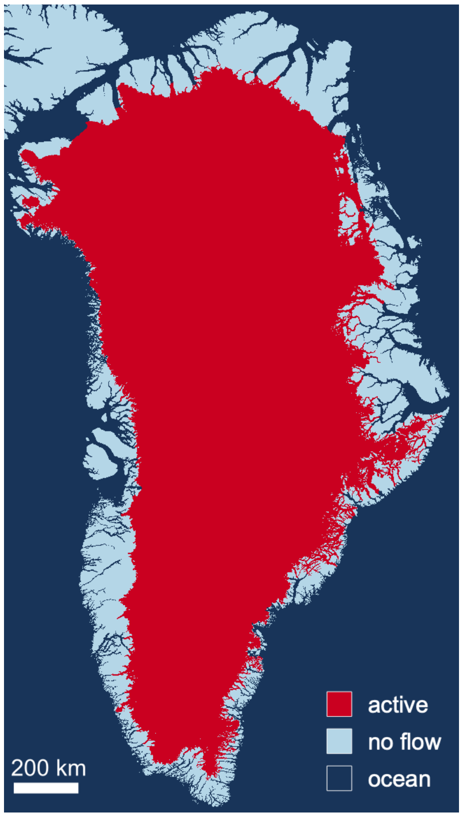

Figure 6. The Greenland setup used for this study. Red represents the area where the hydrological system is computed, dark blue denotes ocean, and pale blue is land area. Ocean and land lead to different boundary conditions.

4.2 Thread occupancy of compute nodes

The two processors on one node share access paths to main memory, and they have a sophisticated cache architecture. The sparse matrix setup employed in the numerical solvers requires a significant amount of memory accesses and, when employing all cores, will more than saturate the available memory bandwidth. Thus, it is worthwhile to explore whether a lower thread occupancy on a node, which provides more individual bandwidth to the re-

Table 3. Characteristics of the Greenland setups, where “active” is the number of active grid points (red in Fig. 6) relative to the total number of grid points.

Name	Resolution	Grid points	Active (%)
G150	150 m	187 459 428	39.08
G300	300 m	46 879 140	39.05
G600	600 m	11 726 928	39.03
G1200	1200 m	2 935 305	39.01
G2400	2400 m	734 720	39.06

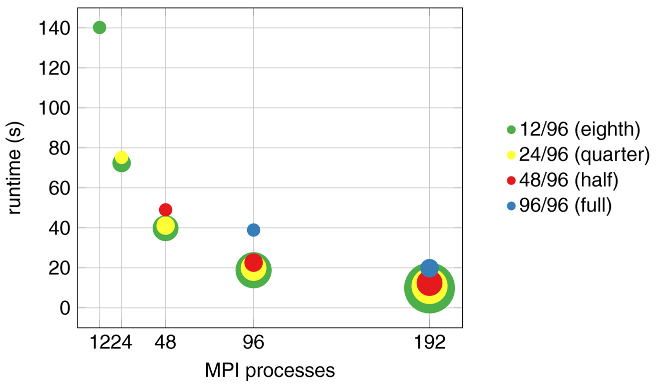

Figure 7. Performance of CUAS-MPI on full (96/96), half (48/96), quarter (24/96), and eighth (12/96) populated nodes of the Lichtenberg HPC system without file output. The center of the circle shows the number of spawned MPI processes (x axis) and the runtime (y axis). For each color the smallest circle indicates the usage of 1 compute node; the next size up indicates 2 nodes, the third 4 nodes, then 8, and lastly 16 nodes.

maining threads, is not, in the end, the better choice. To this end, we tested CUAS-MPI on the Greenland setup with G600 resolution using full, half, quarter, and one-eighth populated compute nodes of the Lichtenberg HPC system. Each thread is pinned to one CPU core and realizes one MPI process. Hence, in our discussion and for our setup, the notions of thread and MPI process are used interchangeably.

The result is shown in Fig. 7. Here, the color of a circle indicates the thread occupancy ratio from one-eighth (green), i.e., 12 threads of a 96-core node, up to the use of all 96 threads on a node (blue). The size of the circle indicates the hardware investment, i.e., the resources requested in the Slurm script. The smallest circle indicates that only 1 node was used; the next size up indicates 2 nodes, then 4 nodes, and lastly 16 nodes. The center of the respective circle indicates the runtime that was achieved with this setup.

Hence, the leftmost green circle (12 threads on one node, resulting in a wall-clock time of about 140 s) is as large as the blue circle in the middle (96 threads one node, resulting in a runtime of about 40 s). On the other hand, the rightmost blue circle is bigger because here we need to employ two nodes at full occupancy to realize the 192 threads, resulting in a runtime of about 20 s. Almost concentric circles, such as the red, yellow, and green circles for 192 processes, then indicate that the additional hardware investment does not pay off, as the corresponding runtime can be achieved with the setup corresponding to the smallest circle.

We see that there is not much difference in the runtime of one node with 48 MPI processes (red circle at 48 processes) to one node with 96 MPI processes (blue circle at 96 processes), but 96 MPI processes on two nodes (red circle at 96 processes) are about twice as fast. The rightmost circles show that 192 MPI processes are faster than 96 MPI processes and that we should use a distribution of four nodes (red circle) because it is faster than two nodes (blue circle) and not significantly slower than four (yellow circle) or eight (green circle) nodes. Similar observations can be made for 96 MPI processes, where an occupancy lower than half results in some, but not very significant, speedup. Hence, we consider 48 MPI processes per node, i.e., half the thread occupancy of a node, to be a good trade-off between getting the solution fast and using the hardware reasonably and employ 48 MPI processes on each node to analyze the throughput, i.e., how many simulated years we can run in a day of computing time (simulated years per day, SYPD), and scalability of CUAS-MPI in the following subsections.

4.3 Runtime and throughput measurements

We identified five functional parts in CUAS-MPI, whose individual performances are worth differentiating: CUAS-MPI setup, CUAS-MPI system kernels, CUAS-MPI system matrix, PETSc linear solver, and NetCDF output. The CUAS-MPI setup contains initial model loading from disk and the setup of necessary grids. The runtime of this code component is not significant in large productive runs and we do not consider it further. The four remaining functional parts are identified in Fig. 2. In particular, the category of the CUAS-MPI system kernels includes all kernels running on CUAS-MPI grids (see the blue box in Fig. 2). The grids are necessary for the computation of the system matrix entries and for diagnostic purposes. The CUAS-MPI system matrix denotes the creation of the equation system, i.e., the computation of the matrix entries and the matrix assembly, that is solved in the PETSc linear solver. The PETSc linear solver is a library call to the solver, in our case the preconditioned GMRES implementation. Finally, the NetCDF output category includes all calls which write data to file during the simulation. The file output performance, in general, greatly depends on the storage and network capabilities of the current HPC system and on the output frequency, which depends on the particular experimental setup in question.

In Fig. 8 we show the runtime of different code categories for 24 time steps of CUAS-MPI with one output to disk. The top line of the runtime plots shows the runtime of the entire time stepping sequences, i.e., the aggregate runtime of all other routines listed as well as forcing calculations and di-

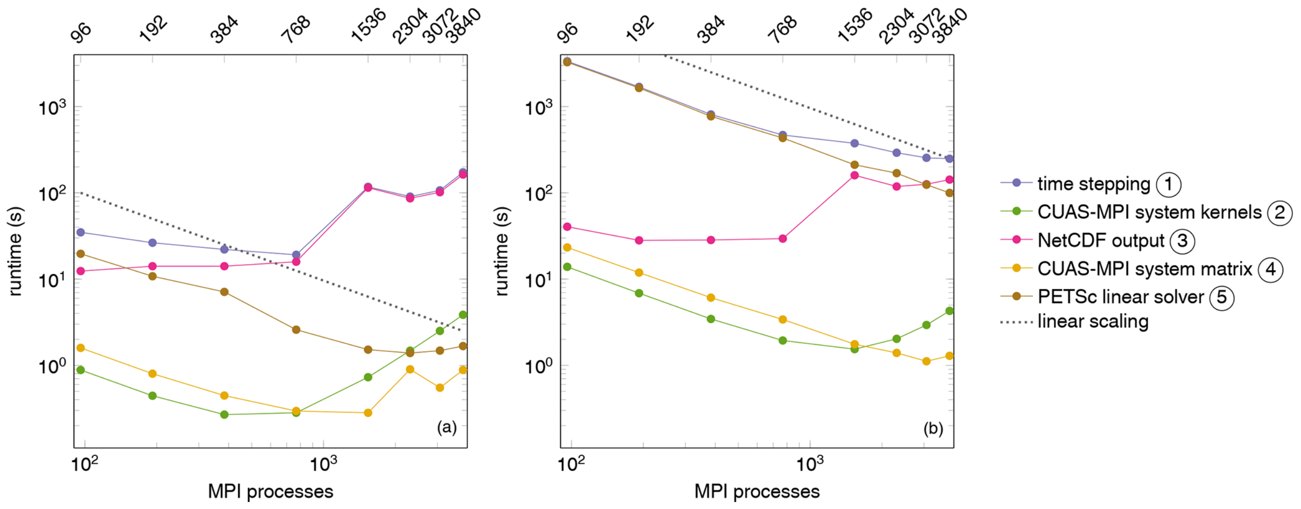

Figure 8. Runtime of 24 time steps of the CUAS-MPI pipeline writing a single output running G600 **(a)** and G150 **(b)**.

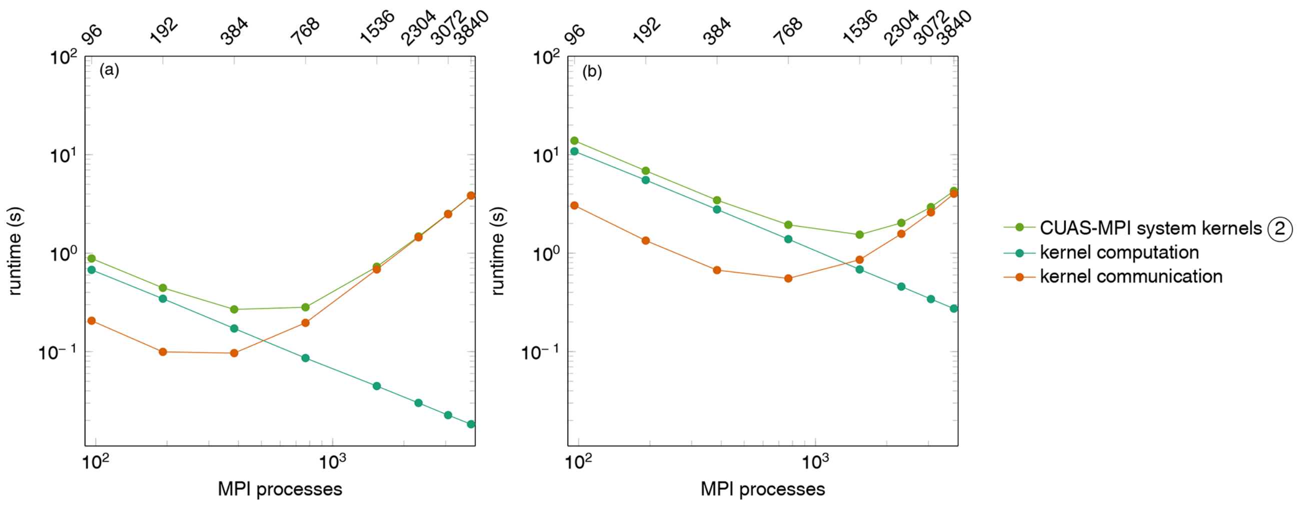

Figure 9. Sum of the runtime of the CUAS-MPI system kernels of 24 time steps running G600 **(a)** and G150 **(b)**.

agnostics, whose runtime is negligible compared to the code categories shown. First, we note that NetCDF output routine does not scale at all. Its runtime is, for both G600 and G150, essentially flat up to 768 MPI processes and then increases by an order of magnitude. The general file output performance is highly system- and load-dependent, as shown by Orgogozo et al. (2014). Therefore, we do not investigate this issue further but keep it in mind for experiment planning.

Considering the coarser G600 grid, we note that, ignoring NetCDF output, the PETSc linear solver dominates runtime and scales up quasi-linearly up to 1536 MPI processes, reaching a minimum at 2304 MPI processes. Finally, it is overtaken by the CUAS-MPI system kernels, whose runtime increases past 768 MPI processes, due to the increasing communication overhead between MPI processes and decreasing computation performed on each MPI process.

For the finer grid (G150), on the other hand, we see a continual decrease in runtime as we increase the number of MPI processes. Ignoring NetCDF output, the PETSc linear solver dominates the runtime but scales in an almost linear fashion up to 3840 MPI processes. In particular, we also see approximately linear scaling for the CUAS-MPI system kernels up to 768 MPI processes and the system matrix routine up to 3072 MPI processes. Further scaling increases the runtime of these two parts due to communication overhead.

Disregarding file output operations, the category of the CUAS-MPI system kernels is the first component to reach its scaling limit. As all kernels behave in the same way, we consider them as a group. Figure 9 shows the accumulated runtime and the separated runtime of kernel computation and kernel communication for grid data exchange caused by PETSc. We notice that the computation scales linearly as expected, but the grid exchange prevents further scaling.

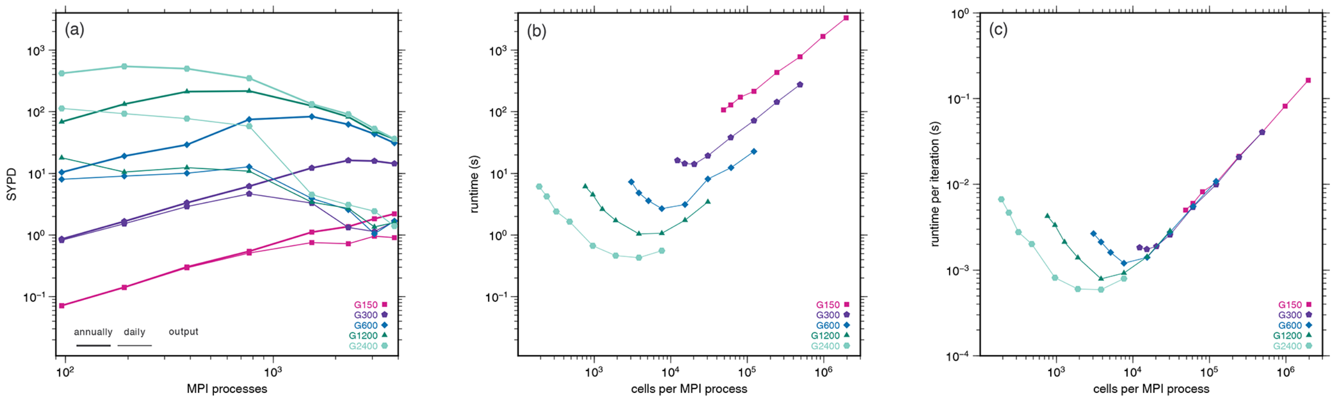

Figure 10. Throughput **(a)** versus MPI process, runtime **(b)**, and runtime per linear solver iteration **(c)** versus number of cells per MPI process.

In our studies of throughput, i.e., how many simulated years we can run in a day of computing time (simulated years per day, SYPD), we employ the same time step (1 h) for the different resolutions in order to allow a sensible comparison of the various code components across resolutions and process counts. This is conservative for the lower resolutions, as coarser grids generally allow larger time steps, because time steps are adapted to fit the stability conditions of the simulation. The different grid resolutions require different numbers of iterations for convergence.

We simulate 1 d (24 time steps) and write one output per day using the large configuration (see Table 2) and measure the runtime. Based on that measurement we compute the runtime for 365 d and compute SYPD.

In Fig. 10, we compare the throughput and runtime scaling of different grid resolutions on the Greenland setup, from 96 up to 3840 MPI processes and for different output frequency. Figure 10a intends to provide users with the data basis for their simulation planning. As an example, it allows assessing how more or less costly a higher or lower spatial resolution is if the output frequency is fixed. We have chosen daily and annual output to cover the upper and lower bounds of user requirements.

The peak throughput of the coarsest grid (G2400) of about 550 simulated years per day is reached at 192 MPI processes and annual writing, while the finest grid (G150) can efficiently utilize more than 3840 MPI processes for both daily and annual writing. For G150, we derive a maximum throughput of approximately 1 simulated year per day if daily output is written and 2 simulated years per day if output is written annually.

In general, there is no sense in using a large number of processes for coarse grids, as there is not enough work to be done for an efficient parallelization. On the other hand, for the G150 grid (which has almost 100 times as many grid points as the G1200 resolution, see Table 3), it does make sense to use more processes to increase throughput: with 768 and 1536 processes, we can compute 198 and 402 model days per compute day. So, in particular, going from 768 to 1536 processes, we increase throughput by a factor of 2.

The fact that there is no sense in using many processes for coarse grids is underscored by Fig. 10b, which shows the runtime of the CUAS-MPI pipeline without output against the number of cells per MPI process. For any given resolution, the leftmost data point corresponds to the run with 3840 processes, i.e., the maximum number of processes employed in our study, whereas the rightmost data point corresponds to the run with 96 processes, i.e., the minimum number of processes employed in our study. The line for the coarsest grid (G2400) is the leftmost one, and we see that here, using more than 192 processes (the inflection point of the curve) makes no sense as we simply exercise communication overhead and do not have enough work on each process to offset this overhead. The lines for finer resolutions then progressively shift right, and at the finest resolution (G150) we no longer have an inflection point – even for 3840 processes there is enough work to be done locally. So, in particular the finest grid (G150) still has potential to effectively use even more processes.

The number of iterations taken by the iterative linear solver varies with grid resolution, and more iterations are needed for solving the linear equation system for finer grids. While this is certainly an issue to consider for throughput, for scalability, i.e., the question of whether using more processes reduces computation time, the time of one iteration is the deciding factor. Hence, we also show in Fig. 10c the runtime per linear solver iteration versus the number of cells per MPI process. The minimum of the curves does not change, but the curves move closer together, as most of the computing time is dependent on the number of cells per MPI process. However, the minimum of the total runtime per linear solver iteration increases as the resolution gets finer because we use more MPI processes at the respective minima, and more MPI processes cause more communication overhead than fewer MPI processes, e.g., for global communications for norm computations.

Figure 10 also has practical use for planning simulations. For most tasks, scientists have a constraint on SYPD for a simulation, such as simulations must not take longer than a certain number of hours or days. The research question we want to answer also determines the output frequency, e.g., investigations of seasonality need daily output. Using Fig. 10 one can also, e.g., estimate the affordable spatial resolution and select the optimal number of MPI cores or estimate computing time for proposals for computing resources.

5 Discussion

The throughput shown by CUAS-MPI enables ice-sheet-wide simulations at high (600 m), but potentially not the highest, resolution (150 m BedMachine grid resolution) due to the computational costs. The number of time steps required for a seasonal cycle limits the number of years that can be simulated within a few days of computing time, as can be seen in Fig. 10. Derived from the timings we measure on the Lichtenberg HPC system, a simulation covering the 90 years from 2010 to 2100 at 600 m spatial and 1 h temporal resolution requires an estimated wall-clock time of 74 h (3 d) on 384 MPI processes. Such a simulation can be performed a few times but is not feasible for ensemble simulations for different atmospheric inputs. Moreover, high computational demand arises from spin-ups to have a proper initial state for simulations and a control run needed for assessment of the projection run. So, with CUAS-MPI, Greenland-wide simulations are still challenging, but they are feasible. The computational costs also emphasize the need for efficient coupling of ice sheet and hydrology models.

Although we have applied the code to an entire ice sheet, applications to alpine glaciers might be of interest for a larger community. As an example we consider the Kanderfirn Glacier (Switzerland), which has a size of about 12 km^2. To compute this glacier at 10 m resolution would result in 120 000 grid points. Assuming hourly time steps and daily output, 1 year would require about 3 min of wall-clock time on 384 MPI processes (based on G150 performance).

The physical (and mathematical) model of CUAS-MPI may, of course, not be powerful enough to simulate the complex physics sufficiently well in other cases of interest. However, with the CUAS-MPI domain scientists now have a tool to conduct simulations on relevant areas of interest to identify strengths and weaknesses of the physical basis of the model. The code has been designed in a modular fashion to allow for extensions of the underlying physical model.

The next step will be the coupling of CUAS-MPI to an ice sheet model, in our case the finite-element-based Ice-sheet and Sea-level System Model (ISSM, https://issm.jpl.nasa.gov/, last access: 5 September 2023, Larour et al., 2012). There are inherent system dependencies between the physical quantities in both models (ice sheet and hydrology); hence, a coupling needs to consider the ingestion of a larger number of fields from the ice sheet code into CUAS-MPI, while there is only the effective normal pressure to be fed into the ice sheet model. To this end, we plan to employ the preCICE coupling library for partitioned multi-physics simulations (Chourdakis et al., 2022).

As an alternative to coupled simulations, one can also consider the possibility of implementing the subglacial hydrology model of CUAS-MPI directly in ice sheet codes like ISSM. Such a monolithic implementation would avoid the necessity of inter-model communication. However, we see various advantages in coupling the two codes versus a monolithic solution. First, a stand-alone implementation supporting a generalized coupling interface is more flexible and can be used in many other projects. Second, it is fully independent of other code discretization and thus prevents inconsistencies. Finally, we see a huge advantage in the implicit additional parallelization potential. While a monolithic implementation will most likely execute the ice and hydrology modules one after the other in a multi-physics environment, a coupled implementation can easily run ice sheet and subglacial hydrology simulation on dedicated nodes. So both modules can run in parallel and exchange data after computing the next time sequence. Certainly the data exchange of coupled simulations generates additional overhead, but as we see no necessity for high-frequency data exchange, e.g., each time step, but consider less frequent exchange intervals, e.g., daily or even seasonal, the overhead of coupled simulations will be affordable and the benefit of running hydrology in parallel to other modules will prevail.

For this endeavor, it is worth comparing the computational costs of the ice sheet and hydrology models. Comparing the SYPD for a Greenland setup in the ice sheet model at the highest tested resolution (minimum edge length of 250 m) presented in Fischler et al. (2022) with the finest grid of the CUAS-MPI setup (edge length of 150 m), we find that the computational costs are comparable for both models. As a consequence, both simulations would need about the same time per year, thus achieving low idle time at synchronization points in coupled runs. Nevertheless, the coupling frequency depends on the phenomena investigated in both models. If someone wants to study the effect of ocean tides on the subglacial environment, time steps in the hydrology model need to resolve the tides, and the ice sheet model may or may not be informed about the changes depending on the scientific goals and ice sheet model capabilities.

Considering the performance of CUAS-MPI, we see that the NetCDF performance plays an important role and limits scalability of CUAS-MPI on the Lichtenberg HPC system that we are running on. Hence, we suggest using low-frequency output, e.g., annual output. In addition, the smallest output configuration can be used to reduce the output costs, and then the CUAS-MPI restart capabilities can be used to compute additional fields afterwards. We have not investigated NetCDF performance further, as I/O performance is highly system-dependent. We have, however, encapsulated

I/O in our software design so that other NetCDF implementations can be linked in easily.

Outside of I/O, the runtime of CUAS-MPI is dominated by the runtime of the PETSc linear solver (unless there is too little local work). Our scaling tests have shown that the solver still has more scaling potential in the case of large setups, but we reached the scaling limit of coarser grids. Future throughput improvements will be enabled by improvements of the PETSc software infrastructure, which is being continually improved and updated, e.g., using persistent MPI communication.

The scaling of the CUAS-MPI system kernels, which is generally of lower importance overall, might be further improved by asynchronous communication or additional thread parallelism per MPI process to increase computational granularity, i.e., the amount of work which is performed by a task. Hybrid parallelism would also enable the opportunity to use less parallelism in less scalable regions like file output and more parallelism in the computationally dominant linear solver.

6 Conclusions

CUAS-MPI is a newly designed process-parallel code based on the model of Beyer et al. (2018) that is software-engineered for performance and portability, which so far has been implemented as a prototype. The results of CUAS-MPI are consistent with analytical solutions of pumping tests in hydrology, implying successful code implementations of the Beyer et al. (2018) model. The code we present here can be widely applied for glaciological applications, from simulating water discharge of smaller glaciers for hydropower production to estimating seasonality of water flux of rivers that are fed by subglacial water.

The code is instrumented for performance measurements, which we conducted on the Lichtenberg HPC system at TU Darmstadt. Our study demonstrates that CUAS-MPI performs well and scales up to 3840 MPI processes, enabling the simulation of challenging setups at finer resolutions. Some performance limitations result from the current implementation of PETSc and NetCDF, and we anticipate that CUAS-MPI will profit from performance improvements in these software infrastructures in the future.

The insights gained from the performance measurements can be used to plan simulations. Given a constraint on available computing hours, the numbers presented here allow assessing the choices of spatial resolution and output frequency, as well as selecting the optimum number of MPI cores.

Subglacial hydrology and ice sheet evolution are strongly related to each other, so runtime coupling of CUAS-MPI and ice sheet simulations like ISSM is an important topic. As the subglacial hydrological system delivers fresh water into fjord systems and ice shelf cavities, the coupling to ocean models will also be needed in the future. Therefore, we see necessary enhancements in the implementation of inter-simulation data exchange and the adaption of the model to support changing simulation domains in transient runs.

In contrast to the Greenland Ice Sheet, the hydrology below the Antarctic Ice Sheet is expected to be driven by basal water production without additional water from the ice sheet surface draining to the base during the summer season. Nevertheless, subglacial hydrology seems very important, and two-dimensional subglacial hydrology models have already been applied to large Antarctic catchments (Dow et al., 2022; Dow, 2023). The performance of a subglacial hydrology model is vital if model simulations are extended to the entire Antarctic Ice Sheet, which is several times larger than the Greenland Ice Sheet.

Our study demonstrates that CUAS-MPI fulfills a number of the demands of the Earth system modeling community: allowing simulations from global to local scale, performance portability, scalable workflows, and being open-source.

Appendix A: Workflow

We usually prepare a setup script to pre-process available datasets and set up the simulations. In that script, the model domain and the grid are defined and used to create the mask (see Fig. 6). This mask is one of the input fields and informs the model about the locations and types of boundaries. In addition, the fields for bedrock topography, ice thickness, and water input are prepared into one NetCDF file.

The next step is initialization. The initial head and thus the initial water pressure can be selected via specific named CUAS-MPI options to be spatially uniform, following the bed elevation, or to be equal to the ice overburden pressure. The initial transmissivity can be set to a spatially uniform value via a command line option. Full control over the initial conditions is further given using the CUAS-MPI "restart from file" capabilities. A restart could be done from, e.g., the last time slice of a previous run, from arbitrary fields for head and transmissivity provided by the user, or from the output of a coarse-resolution spin-up that has to be remapped onto the new grid outside of CUAS-MPI. All file names and parameters used for a model run are stored in the NetCDF output file for later reference.

The last step is to set up the actual run script that serves the needs of the cluster environment. In our case this is the Slurm scheduler. This step includes setting up the command line options to control CUAS-MPI's physics and time stepping and setting up the PETSc solver via the environment variable `PETSC_OPTIONS`, as well as the memory, node, core, and runtime configuration on the cluster. The time stepping strategy can be provided with an optional parameter specifying either the constant time step or providing the sequence of time steps to be applied in a time step file. Our implementation is backwards-compatible with the setup of the serial version, supporting input data in NetCDF format and the same command line parameters. The comprehensive list of the parameters is displayed when executing the CUAS-MPI executable with the "–help" option. In addition, CUAS-MPI is able to restart from previous runs.

A typical use case may be the simulation of a seasonal hydrological cycle. To this end, a spin-up would be conducted first to retrieve a steady-state system. This may be done at a coarse spatial resolution first, followed by a refinement using another grid using the restart option. At this point, the actual simulation with seasonal forcing starts using the refined mesh. It is highly recommended to also run a control simulation on the refined mesh by further using the steady-state forcing. This allows us to account for model transients that may still be contained in the model after the spin-up. If a particular target area needs to be simulated in even higher resolution, a regional subdomain nesting can be employed to reduce the computational costs of a model run. If a drainage basin can only be covered partially, it is very beneficial to embed this area into a large-scale and probably also coarse-resolution run that provides boundary conditions along the subdomain margin (Dirichlet conditions for h and T). The desired output frequency, as well as the variables to be stored, can be adjusted to the needs of the user depending on the domain and resolution.

Another typical application is a projection of the change in the hydrological system over a longer time period. For this purpose a spin-up is required, too. To balance the costs for the long simulation period, the user can reduce spatial resolution if the science questions allows or reduce the output to only annually write files with the essential physical fields. In all cases the user needs to choose an adequate time step.

Code and data availability. We published our repository at https://github.com/tudasc/CUAS-MPI (last access: 5 September 2023). For the measurements presented in this paper, version 0.1.0 was used, which is available at GitHub and through the following address: https://doi.org/10.5281/zenodo.7554686 (Fischler et al., 2023). Our profiling data are available at https://doi.org/10.48328/tudatalib-1034 (Fischler, 2022).

Author contributions. YF, CB, TK, and AH planned the project. YF and CB developed the software design. YF, LFO, RE, and TK implemented the code. YF conducted all performance measurements. YF, CB, TK, and AH analyzed the performance measurements. TK developed the Greenland setup. TK and AH ran the code for polar applications. JS and RS developed the pumping test concept, and JS conducted the tests. All authors discussed the performance analysis and wrote the paper.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Acknowledgements. The authors gratefully acknowledge the computing time provided to them on the high-performance computer Lichtenberg at the NHR Center NHR4CES at TU Darmstadt under grant p0020118. NHR4CES is funded by the Federal Ministry of Education and Research and the state governments participating on the basis of the resolutions of the GWK for national high-performance computing at universities.

The authors thank Vadym Aizinger (University of Bayreuth) and Alexander Hück (TU Darmstadt) for helpful discussions.

Financial support. This research has been supported by the German-Israeli Foundation for Scientific Research and Development (GIF) (grant no. I-1493-301.8/2019).

Review statement. This paper was edited by Ludovic Räss and reviewed by two anonymous referees.

References

Ahlstrøm, A. P., Mottram, R., Nielsen, C., Reeh, N., and Andersen, S. B.: Evaluation of the future hydropower potential at Paakitsoq, Ilulissat, West Greenland, Tech. Rep. 31, GEUS, https://doi.org/10.22008/gpub/27154, 2009.

Amestoy, P., Duff, I. S., Koster, J., and L'Excellent, J.-Y.: A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling, SIAM J. Matrix Anal. A., 23, 15–41, 2001.

Amestoy, P., Buttari, A., L'Excellent, J.-Y., and Mary, T.: Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures, ACM T. Math. Software, 45, 2:1–2:26, 2019.

Arnold, N. and Sharp, M.: Flow variability in the Scandinavian ice sheet: modelling the coupling between ice sheet flow and hydrology, Quaternary Sci. Rev., 21, 485–502, https://doi.org/10.1016/S0277-3791(01)00059-2, 2002.

Aschwanden, A., Fahnestock, M. A., and Truffer, M.: Complex Greenland outlet glacier flow captured, Nat. Commun., 7, 10524, https://doi.org/10.1038/ncomms10524, 2016.

Balay, S., Abhyankar, S., Adams, M. F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W. D., Karpeyev, D., Kaushik, D., Knepley, M. G., May, D. A., McInnes, L. C., Mills, R. T., Munson, T., Rupp, K., Sanan, P., Smith, B. F., Zampini, S., Zhang, H., and Zhang, H.: PETSc Users Manual, Tech. Rep. ANL-95/11 – Revision 3.15, Argonne National Laboratory, https://petsc.org/release/ (last access: 5 September 2023), 2021.

Benn, D. I. and Evans, D. J. A.: Glaciers and glaciation, Routledge, 2nd edn., https://doi.org/10.4324/9780203785010, 2010.

Beyer, S., Kleiner, T., Aizinger, V., Rückamp, M., and Humbert, A.: A confined–unconfined aquifer model for subglacial hydrology and its application to the Northeast Greenland Ice Stream, The

Cryosphere, 12, 3931–3947, https://doi.org/10.5194/tc-12-3931-2018, 2018.

Bindschadler, R. A., Vornberger, P. L., King, M. A., and Padman, L.: Tidally driven stick-slip motion in the mouth of Whillans Ice Stream, Antarctica, Ann. Glaciol., 36, 263–272, https://doi.org/10.3189/172756403781816284, 2003.

Braithwaite, R. J. and Højmark Thomsen, H.: Simulation of Run-Off from the Greenland Ice Sheet for Planning Hydro-Electric Power, Ilulissat/Jakobshavn, West Greenland, Ann. Glaciol., 13, 12–15, https://doi.org/10.3189/S0260305500007540, 1989.

Budd, W. F. and Jenssen, D.: Numerical modelling of the large scale basal water flux under the west Antarctic ice sheet., in: Dynamics of the west Antarctic ice sheet, edited by: Van der Veen, C. J. and Oerlemens, J., Dordrecht, Reidel, 293–320, https://doi.org/10.1007/978-94-009-3745-1_16, 1987.

Bueler, E. and van Pelt, W.: Mass-conserving subglacial hydrology in the Parallel Ice Sheet Model version 0.6, Geosci. Model Dev., 8, 1613–1635, https://doi.org/10.5194/gmd-8-1613-2015, 2015.

Chourdakis, G., Davis, K., Rodenberg, B., Schulte, M., Simonis, F., Uekermann, B., Abrams, G., Bungartz, H.-J., Yau, L. C., Desai, I., Eder, K., Hertrich, R., Lindner, F., Rusch, A., Sashko, D., Schneider, D., Totounferoush, A., Volland, D., Vollmer, P., and Koseomur, O. Z.: preCICE v2: A sustainable and user-friendly coupling library, Open Research Europe, 2, 51, https://doi.org/10.12688/openreseurope.14445.2, 2022.

Christmann, J., Helm, V., Khan, S. A., Kleiner, T., Müller, R., Morlighem, M., Neckel, N., Rückamp, M., Steinhage, D., Zeising, O., and Humbert, A.: Elastic deformation plays a non-negligible role in Greenland's outlet glacier flow, Commun. Earth Environ., 2, 232, https://doi.org/10.1038/s43247-021-00296-3, 2021.

Colosio, P., Tedesco, M., Ranzi, R., and Fettweis, X.: Surface melting over the Greenland ice sheet derived from enhanced resolution passive microwave brightness temperatures (1979–2019), The Cryosphere, 15, 2623–2646, https://doi.org/10.5194/tc-15-2623-2021, 2021.

Cook, S. J., Christoffersen, P., Todd, J., Slater, D., and Chauché, N.: Coupled modelling of subglacial hydrology and calving-front melting at Store Glacier, West Greenland , The Cryosphere, 14, 905–924, https://doi.org/10.5194/tc-14-905-2020, 2020.

Cook, S. J., Christoffersen, P., and Todd, J.: A fully-coupled 3D model of a large Greenlandic outlet glacier with evolving subglacial hydrology, frontal plume melting and calving, J. Glaciol., 68, 486–502, https://doi.org/10.1017/jog.2021.109, 2022.

de Fleurian, B., Gagliardini, O., Zwinger, T., Durand, G., Le Meur, E., Mair, D., and Råback, P.: A double continuum hydrological model for glacier applications, The Cryosphere, 8, 137–153, https://doi.org/10.5194/tc-8-137-2014, 2014.

de Fleurian, B., Morlighem, M., Seroussi, H., Rignot, E., van den Broeke, M. R., Munneke, P. K., Mouginot, J., Smeets, P. C. J. P., and Tedstone, A. J.: A modeling study of the effect of runoff variability on the effective pressure beneath Russell Glacier, West Greenland, J. Geophys. Res.-Earth, 121, 1834–1848, https://doi.org/10.1002/2016jf003842, 2016.

Dow, C. F.: The role of subglacial hydrology in Antarctic ice sheet dynamics and stability: a modelling perspective, Ann. Glaciol., 1–6, https://doi.org/10.1017/aog.2023.9, 2023.

Dow, C. F., Ross, N., Jeofry, H., Siu, K., and Siegert, M. J.: Antarctic basal environment shaped by high-pressure flow through a subglacial river system, Nat. Geosci., 15, 892–898, https://doi.org/10.1038/s41561-022-01059-1, 2022.

Ehlig, C. and Halepaska, J. C.: A numerical study of confined-unconfined aquifers including effects of delayed yield and leakage, Water Resour. Res., 12, 1175–183, https://doi.org/10.1029/WR012i006p01175, 1976.

Ehrenfeucht, S., Morlighem, M., Rignot, E., Dow, C. F., and Mouginot, J.: Seasonal Acceleration of Petermann Glacier, Greenland, From Changes in Subglacial Hydrology, Geophys. Res. Lett., 50, e2022GL098009, https://doi.org/10.1029/2022GL098009, 2023.

Engelhardt, H. and Kamb, B.: Basal sliding of Ice Stream B, West Antarctica, J. Glaciol., 44, 223–230, https://doi.org/10.3189/S0022143000002562, 1997.

Ferris, J. G., Knowles, D. B., Brown, R. H., and Stallman, R. W.: Theory of Aquifer Tests, Tech. rep., U.S. Government Print. Office, https://doi.org/10.3133/wsp1536E, 1962.

Ferziger, J. H. and Perić, M.: Computational Methods for Fluid Dynamics, Springer, 3rd edn., https://doi.org/10.1007/978-3-642-56026-2, 2002.

Fischler, Y.: GMD 2022 CUAS-MPI on HHLR Scaling Profiles, TUdatalib [data set], https://doi.org/10.48328/tudatalib-1034, 2022.

Fischler, Y., Rückamp, M., Bischof, C., Aizinger, V., Morlighem, M., and Humbert, A.: A scalability study of the Ice-sheet and Sea-level System Model (ISSM, version 4.18), Geosci. Model Dev., 15, 3753–3771, https://doi.org/10.5194/gmd-15-3753-2022, 2022.

Fischler, Y., Kleiner, T., Bischof, C., Schmiedel, J., Sayag, R., Emunds, R., Oestreich, L. F., and Humbert, A.: A parallel implementation of the confined-unconfined aquifer system model for subglacial hydrology: design, verification, and performance analysis (CUAS-MPI v0.1.0) (0.1.0), Zenodo [code], https://doi.org/10.5281/zenodo.7554686, 2023.

Flowers, G. E.: Modelling water flow under glaciers and ice sheets, P. Roy. Soc. A, 471, 20140907, https://doi.org/10.1098/rspa.2014.0907, 2015.

Flowers, G. E. and Clarke, G. K. C.: A multicomponent coupled model of glacier hydrology 1. Theory and synthetic examples, J. Geophys. Res.-Sol. Ea., 107, 2287, https://doi.org/10.1029/2001JB001122, 2002.

Fountain, A. G. and Walder, J. S.: Water flow through temperate glaciers, Rev. Geophys., 36, 299–328, https://doi.org/10.1029/97RG03579, 1998.

Fowler, A. C.: A theory of glacier surges, J. Geophys. Res., 92, 9111–9120, https://doi.org/10.1029/JB092iB09p09111, 1987.

Franke, S., Jansen, D., Beyer, S., Neckel, N., Binder, T., Paden, J., and Eisen, O.: Complex Basal Conditions and Their Influence on Ice Flow at the Onset of the Northeast Greenland Ice Stream, J. Geophys. Res.-Earth, 126, e2020JF005689, https://doi.org/10.1029/2020jf005689, 2021.

Goelzer, H., Nowicki, S., Payne, A., Larour, E., Seroussi, H., Lipscomb, W. H., Gregory, J., Abe-Ouchi, A., Shepherd, A., Simon, E., Agosta, C., Alexander, P., Aschwanden, A., Barthel, A., Calov, R., Chambers, C., Choi, Y., Cuzzone, J., Dumas, C., Edwards, T., Felikson, D., Fettweis, X., Golledge, N. R., Greve, R., Humbert, A., Huybrechts, P., Le clec'h, S., Lee, V., Leguy, G., Little, C., Lowry, D. P., Morlighem, M., Nias, I., Quiquet, A., Rückamp, M., Schlegel, N.-J., Slater, D. A., Smith, R. S., Straneo, F., Tarasov, L., van de Wal, R., and van den Broeke, M.: The future sea-level contribution of the Greenland ice sheet: a multi-model ensemble study of ISMIP6, The Cryosphere, 14, 3071–3096, https://doi.org/10.5194/tc-14-3071-2020, 2020.

Hartnett, E. and Edwards, J.: The parallelio (pio) c/fortran libraries for scalable hpc performance, in: 37th Conference on Environmental Information Processing Technologies, American Meteorological Society Annual Meeting,, 2021.

Hewitt, I. J.: Modelling distributed and channelized subglacial drainage: the spacing of channels, J. Glaciol., 57, 302–314, https://doi.org/10.3189/002214311796405951, 2011.

Hoffman, M. J., Catania, G. A., Neumann, T. A., Andrews, L. C., and Rumrill, J. A.: Links between acceleration, melting, and supraglacial lake drainage of the western Greenland Ice Sheet, J. Geophys. Res.-Earth, 116, F04035, https://doi.org/10.1029/2010JF001934, 2011.

Jacob, C. E.: Determining the permeability of water-table aquifers, in: Methods of determining permeability, transmissibility and drawdown, edited by Bentall, R., no. 1536 in Geological survey water-supply paper, chap. I, 272–292, US Government Printing Office, https://pubs.usgs.gov/wsp/1536i/report.pdf (last access: 5 September 2023), 1963.

Joughin, I., Smith, B., Howat, I., and Scambos, T.: MEaSUREs Multi-year Greenland Ice Sheet Velocity Mosaic, Version 1, Boulder, Colorado USA, NASA National Snow and Ice Data Center Distributed Active Archive Center, https://doi.org/10.5067/QUA5Q9SVMSJG, 2016.

Joughin, I., Smith, B. E., and Howat, I. M.: A complete map of Greenland ice velocity derived from satellite data collected over 20 years, J. Glaciol., 64, 243, https://doi.org/10.1017/jog.2017.73, 2018.

Kleiner, T. and Humbert, A.: Numerical simulations of major ice streams in western Dronning Maud Land, Antarctica, under wet and dry basal conditions, J. Glaciol., 60, 215–232, https://doi.org/10.3189/2014JoG13J006, 2014.

Knüpfer, A., Rössel, C., Mey, D. a., Biersdorff, S., Diethelm, K., Eschweiler, D., Geimer, M., Gerndt, M., Lorenz, D., Malony, A., Nagel, W. E., Oleynik, Y., Philippen, P., Saviankou, P., Schmidl, D., Shende, S., Tschüter, R., Wagner, M., Wesarg, B., and Wolf, F.: Score-P: A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir, in: Tools for High Performance Computing 2011, edited by: Brunst, H., Müller, M. S., Nagel, W. E., and Resch, M. M., Springer Berlin Heidelberg, Berlin, Heidelberg, 79–91, https://doi.org/10.1007/978-3-642-31476-6_7, 2012.

Larour, E., Seroussi, H., Morlighem, M., and Rignot, E.: Continental scale, high order, high spatial resolution, ice sheet modeling using the Ice Sheet System Model (ISSM), J. Geophys. Res.-Earth, 117, F01022, https://doi.org/10.1029/2011JF002140, 2012.

Latham, R., Zingale, M., Thakur, R., Gropp, W., Gallagher, B., Liao, W., Siegel, A., Ross, R., Choudhary, A., and Li, J.: Parallel netCDF: A High-Performance Scientific I/O Interface, in: SC Conference, p. 39, IEEE Computer Society, Los Alamitos, CA, USA, SC Conference 2003, 15–21 November, https://doi.org/10.1109/SC.2003.10053, 2003.

Le Brocq, A. M., Payne, A. J., Siegert, M. J., and Alley, R. B.: A subglacial water-flow model for West Antarctica, J. Glaciol., 55, 879–888, https://doi.org/10.3189/002214309790152564, 2009.

Lliboutry, L.: Glissement d'un glacier sur un plan de parsemé d'obstacles hémisphériques, Ann. Géophys., 34, 147–162, 1978.

MacGregor, J. A., Chu, W., Colgan, W. T., Fahnestock, M. A., Felikson, D., Karlsson, N. B., Nowicki, S. M. J., and Studinger, M.: GBaTSv2: a revised synthesis of the likely basal thermal state of the Greenland Ice Sheet, The Cryosphere, 16, 3033–3049, https://doi.org/10.5194/tc-16-3033-2022, 2022.

Morlighem, M., Williams, C. N., Rignot, E., An, L., Arndt, J. E., Bamber, J. L., Catania, G., Chauché, N., Dowdeswell, J. A., Dorschel, B., Fenty, I., Hogan, K., Howat, I., Hubbard, A., Jakobsson, M., Jordan, T. M., Kjeldsen, K. K., Millan, R., Mayer, L., Mouginot, J., Noël, B. P. Y., O'Cofaigh, C., Palmer, S., Rysgaard, S., Seroussi, H., Siegert, M. J., Slabon, P., Straneo, F., van den Broeke, M. R., Weinrebe, W., Wood, M., and Zinglersen, K. B.: BedMachine v3: Complete Bed Topography and Ocean Bathymetry Mapping of Greenland From Multibeam Echo Sounding Combined With Mass Conservation, Geophys. Res. Lett., 44, 11051–11061, https://doi.org/10.1002/2017gl074954, 2017.

Morlighem, M. E. A.: IceBridge BedMachine Greenland, Version 4, IceBridge BedMachine Greenland, Version 4 [data Set], https://doi.org/10.5067/VLJ5YXKCNGXO, 2021.

Neckel, N., Zeising, O., Steinhage, D., Helm, V., and Humbert, A.: Seasonal Observations at 79°N Glacier (Greenland) From Remote Sensing and in situ Measurements, Front. Earth Sci., 8, 142, https://doi.org/10.3389/feart.2020.00142, 2020.

Nowicki, S. M. J., Payne, A., Larour, E., Seroussi, H., Goelzer, H., Lipscomb, W., Gregory, J., Abe-Ouchi, A., and Shepherd, A.: Ice Sheet Model Intercomparison Project (ISMIP6) contribution to CMIP6, Geosci. Model Dev., 9, 4521–4545, https://doi.org/10.5194/gmd-9-4521-2016, 2016.

Orgogozo, L., Renon, N., Soulaine, C., Hénon, F., Tomer, S., Labat, D., Pokrovsky, O., Sekhar, M., Ababou, R., and Quintard, M.: An open source massively parallel solver for Richards equation: Mechanistic modelling of water fluxes at the watershed scale, Comput. Phys. Commun., 185, 3358–3371, https://doi.org/10.1016/j.cpc.2014.08.004, 2014.

Patankar, S. V.: Numerical Heat Transfer and Fluid Flow, McGraw-Hill, New York, 1980.

Rew, R. and Davis, G.: NetCDF: an interface for scientific data access, IEEE Comput. Graph., 10, 76–82, https://doi.org/10.1109/38.56302, 1990.

Schröder, L., Neckel, N., Zindler, R., and Humbert, A.: Perennial Supraglacial Lakes in Northeast Greenland Observed by Polarimetric SAR, Remote Sensing, 12, 2798, https://doi.org/10.3390/rs12172798, 2020.

Seroussi, H., Nowicki, S., Payne, A. J., Goelzer, H., Lipscomb, W. H., Abe-Ouchi, A., Agosta, C., Albrecht, T., Asay-Davis, X., Barthel, A., Calov, R., Cullather, R., Dumas, C., Galton-Fenzi, B. K., Gladstone, R., Golledge, N. R., Gregory, J. M., Greve, R., Hattermann, T., Hoffman, M. J., Humbert, A., Huybrechts, P., Jourdain, N. C., Kleiner, T., Larour, E., Leguy, G. R., Lowry, D. P., Little, C. M., Morlighem, M., Pattyn, F., Pelle, T., Price, S. F., Quiquet, A., Reese, R., Schlegel, N.-J., Shepherd, A., Simon, E., Smith, R. S., Straneo, F., Sun, S., Trusel, L. D., Van Breedam, J., van de Wal, R. S. W., Winkelmann, R., Zhao, C., Zhang, T., and Zwinger, T.: ISMIP6 Antarctica: a multi-model ensemble of the Antarctic ice sheet evolution over the 21st century, The Cryosphere, 14, 3033–3070, https://doi.org/10.5194/tc-14-3033-2020, 2020.

Siegert, M. J.: Antarctic subglacial lakes, Earth-Sci. Rev., 50, 29–50, https://doi.org/10.1016/S0012-8252(99)00068-9, 2000.

Smith-Johnsen, S., de Fleurian, B., and Nisancioglu, K. H.: The role of subglacial hydrology in ice streams with elevated geothermal heat flux, J. Glaciol., 66, 303–312, https://doi.org/10.1017/jog.2020.8, 2020.

Sommers, A., Rajaram, H., and Morlighem, M.: SHAKTI: Subglacial Hydrology and Kinetic, Transient Interactions v1.0, Geosci. Model Dev., 11, 2955–2974, https://doi.org/10.5194/gmd-11-2955-2018, 2018.

Theis, C. V.: The relation between the lowering of the Piezometric surface and the rate and duration of discharge of a well using ground-water storage, Eos, Transactions American Geophysical Union, 16, 519–524, https://doi.org/10.1029/TR016i002p00519, 1935.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T., and the scikit-image contributors: scikit-image: image processing in Python, PeerJ, 2, e453, https://doi.org/10.7717/peerj.453, 2014.

Werder, M. A., Hewitt, I. J., Schoof, C. G., and Flowers, G.: Modeling channelized and distributed subglacial drainage in two dimensions, J. Geophys. Res.-Earth, 118, 2140–2158, https://doi.org/10.1002/jgrf.20146, 2013.

Young, T. J., Christoffersen, P., Bougamont, M., and Stewart, C. L.: Rapid basal melting of the Greenland Ice Sheet from surface meltwater drainage, P. Natl. Acad. Sci. USA, 119, e2116036119, https://doi.org/10.1073/pnas.2116036119, 2022.

Zeising, O. and Humbert, A.: Indication of high basal melting at the EastGRIP drill site on the Northeast Greenland Ice Stream, The Cryosphere, 15, 3119–3128, https://doi.org/10.5194/tc-15-3119-2021, 2021.