Geosci. Model Dev., 18, 5311–5327, 2025
https://doi.org/10.5194/gmd-18-5311-2025
© Author(s) 2025. This work is distributed under
the Creative Commons Attribution 4.0 License.

Geoscientific Model Development
Open Access
EGU

A Python library for solving ice sheet modeling problems using physics-informed neural networks, PINNICLE v1.0

Gong Cheng, Mansa Krishna, and Mathieu Morlighem

Department of Earth Sciences, Dartmouth College, Hanover, NH 03755, USA

Correspondence: Gong Cheng (gong.cheng@dartmouth.edu)

Received: 13 March 2025 – Discussion started: 2 April 2025
Revised: 10 June 2025 – Accepted: 11 June 2025 – Published: 26 August 2025

Abstract. Predicting the future contributions of the ice sheets to sea-level rise remains a significant challenge due to our limited understanding of key physical processes (e.g., basal friction, ice rheology) and the lack of observations of critical model inputs (e.g., bed topography). Traditional numerical models typically rely on data assimilation methods to estimate these variables by solving inverse problems based on conservation laws of mass, momentum, and energy. However, these methods are not versatile and require extensive code development to incorporate new physics. Moreover, their dependence on data alignment within computational grids hampers their adaptability, especially in the context of sparse data availability in space and time. To address these limitations, we developed PINNICLE (Physics-Informed Neural Networks for Ice and CLimatE), an open-source Python library dedicated to ice sheet modeling. PINNICLE seamlessly integrates observational data and physical laws, facilitating the solution of both forward and inverse problems within a single framework. PINNICLE currently supports a variety of conservation laws, including the Shelfy-Stream Approximation (SSA), MOno-Layer Higher-Order (MOLHO) models, and mass conservation equations, for both time-independent and time-dependent simulations. The library is user-friendly, requiring only the setting of a few hyperparameters for standard modeling tasks, while advanced users can define custom models within the framework. Additionally, PINNICLE is based on the DeepXDE library, which supports widely used machine learning packages such as TensorFlow, PyTorch, and JAX, enabling users to select the backend that best fits their hardware. We describe here the implementation of PINNICLE and showcase this library with examples across the Greenland and Antarctic ice sheets for a range of forward and inverse problems.

1 Introduction

Ice sheet modeling is essential for projecting future sea-level rise and understanding the complex dynamics that drive ice sheet behavior under changing climate conditions (e.g., Larour et al., 2012; Aschwanden et al., 2019; Goelzer et al., 2020; Seroussi et al., 2020). These numerical models provide insights into fundamental glaciological processes, advancing our understanding of ice flow and its response to and interactions with the climate system (e.g., Schoof, 2007; Joughin et al., 2021; Morlighem et al., 2024). However, developing models that accurately capture current ice sheet behavior and mass change remains challenging, further complicated by limited observational data and an incomplete understanding of critical physical processes. Despite decades of development, significant uncertainties persist in ice sheet models, impacting the accuracy of sea-level projections (Robel et al., 2019; Edwards et al., 2019; Aschwanden et al., 2021). A major source of uncertainty in models comes from parameters that are challenging to measure directly in the field, such as the bed topography (Morlighem et al., 2017, 2020); basal conditions (Gagliardini et al., 2007; Morlighem et al., 2013; Wernecke et al., 2023); and material properties of ice, including its rheology and thermal structure (e.g., Furst et al., 2015; Colgan et al., 2021).

Traditionally, these parameters are estimated by solving inverse problems, often framed as partial differential equation (PDE)-constrained optimization problems (MacAyeal, 1993; Morlighem et al., 2010; Goldberg and Sergienko, 2011). These inverse problems are difficult to solve because they require customized numerical algorithms that depend on the specific PDEs governing the system (Goldberg and Heimbach, 2013; Cheng and Lotstedt, 2020) and often demand regularization to avoid overfitting and facilitate convergence

due to their inherent ill-posedness (Tikhonov, 1943). As data availability and quality continue to improve, integrating these data into traditional inverse frameworks presents new challenges, both in terms of computational demands and the complexity of the required numerical methods. This increase in data complexity calls for new approaches that can flexibly integrate observational data while remaining computationally efficient.

Recent advances in machine learning offer promising alternatives for handling the shortcomings of traditional ice sheet models, particularly when dealing with sparse or noisy data (Karniadakis et al., 2021). In particular, Physics-Informed Neural Networks (PINNs) have gained attention as a powerful tool for combining physical laws with observational data (Raissi et al., 2019; Karniadakis et al., 2021; Lu et al., 2021). Unlike conventional approaches that require extensive customization for each specific problem, PINNs allow for the flexible integration of various physical constraints, making them highly adaptable to different modeling scenarios. In glaciology, PINNs have been applied to a range of challenging modeling tasks recently, including inferring basal conditions, simulating complex ice flow dynamics, and testing novel hypotheses about physical processes in ice shelves (Riel et al., 2021; Wang et al., 2025; Jouvet and Cordonnier, 2023; Iwasaki and Lai, 2023; Cheng et al., 2024). The incorporation of physical knowledge within the neural network structure has the potential to introduce a regularizing effect, which is particularly beneficial for handling sparse and noisy observational data, supporting mesh-free modeling, and enabling flexible constraints beyond standard boundary conditions (Seo, 2024). This framework provides a balance between model complexity and accuracy, making it a robust tool for advancing ice sheet modeling (Raissi et al., 2020; Cheng et al., 2024).

In this context, we describe here PINNICLE (Physics-Informed Neural Networks for Ice and CLimatE), an open-source Python library for ice sheet modeling designed to facilitate the solution of both forward and inverse problems using PINNs. PINNICLE provides a unified framework to integrate observational data directly with the governing physical laws. Leveraging the DeepXDE library (Lu et al., 2021), PINNICLE supports machine learning platforms like TensorFlow, PyTorch, and JAX, giving users the flexibility to choose the backend that best suits their hardware and computational resources. This paper outlines the methodological framework of PINNICLE and illustrates its capabilities through applications on glaciers in the Greenland and Antarctic ice sheets.

2 Physics-informed neural networks

In recent years, physics-informed machine learning techniques have become increasingly popular in the field of ice sheet modeling (e.g., Riel et al., 2021; Brinkerhoff, 2022; Jouvet and Cordonnier, 2023; Bolibar et al., 2023; He et al., 2023). Ice flow is governed by a set of PDEs derived from fundamental conservation laws. The extent to which these governing equations are satisfied in different machine learning frameworks varies across studies. For example, the approaches described in He et al. (2023), Bolibar et al. (2023), and Koo et al. (2024) employ neural networks as emulators or surrogate models, learning internal relationships derived from numerical solutions of the governing PDEs, with minimal enforcement of physical constraints. In Jouvet and Cordonnier (2023), the neural network also acts as an emulator but incorporates the PDE residual directly into the training loss function, embedding physical principles into the optimization process for a more physically consistent model. Similarly, Riel et al. (2021) combines observational data with physics-inspired constraints, such as smoothness and sign of the basal drag, though without explicitly enforcing physical laws on the model's outputs.

This study takes an alternative approach by leveraging neural networks to directly learn the physical constraints and relationships governing ice dynamics, aiming for a more comprehensive integration of data and physics. The architecture of PINNICLE follows the same framework as in Raissi et al. (2019), Wang et al. (2025), Iwasaki and Lai (2023), and Cheng et al. (2024), where a neural network is trained under the constraints of data misfit and PDE loss. The inputs of the neural network are the independent variables of the PDEs, which can be the spatial coordinates (x, y, z) and/or the time variable (t), depending on the governing equations. The outputs of the neural network are all the dependent variables of the PDEs. This architecture enables the neural network, through backpropagation, to compute the required spatial and temporal derivatives of these dependent variables for evaluating the PDEs. The loss function of the PINN consists of two parts, data and physical loss, which are both constructed using these output variables. The data loss measures the misfit between the data and the corresponding output variable at the location and time of the data acquired. The physical loss is computed by evaluating the residual of the PDEs on a set of randomly generated collocation points. Then, the training procedure is to minimize the loss function, such that the output of the neural network satisfies the governing PDEs and also matches the data provided. However, it is important to note that this formulation represents an idealized scenario. In practice, the residuals typically do not reach zero, nor does the data misfit, as discussed in Cheng et al. (2024).

3 Physics

PINNICLE is designed to be flexible so that diverse physical laws along with relevant observational data can be easily integrated. In ice sheet modeling, conservation of mass and momentum form the governing equation of ice dynamics. PINNICLE currently supports these laws by implementing

widely used stress balance approximations. Users can readily apply or extend the framework to include additional physical constraints as required by specific modeling objectives.

3.1 Conservation of mass

In ice sheet modeling, the large aspect ratio of the horizontal dimensions (x, y) to the vertical dimension (z) leads most dynamics to occur in the x–y plane. Therefore, it is common practice to model the ice sheet using a depth-averaged approach that effectively captures horizontal behavior, simplifying the model by considering a two-dimensional domain (x, y) and focusing on the horizontal ice velocity components, denoted by $\mathbf{u} = (u, v)^T$.

Mass conservation is fundamental to simulating ice sheet dynamics, as it controls the variations in ice thickness over time. Since ice behaves as an incompressible fluid, the rate of change in ice thickness, H, is equal to the sum of the flux divergence in the horizontal direction and vertical mass exchange processes:

$$\frac{\partial H}{\partial t} + \frac{\partial (uH)}{\partial x} + \frac{\partial (vH)}{\partial y} = a, \tag{1}$$

where a represents the net mass balance field from surface and basal processes, such as surface accumulation (e.g., snowfall) and losses like surface or basal melting.

3.2 Conservation of momentum

We describe here two approximations of the conservation of momentum that are widely used in glaciology. First, the Shelfy-Stream Approximation (SSA), provides a simplified form of the full Stokes equations by neglecting vertical shear stresses (MacAyeal, 1989). SSA is an excellent approximation of ice sheet flow in regions of fast sliding, such as ice streams, and floating ice shelves. This reduction allows us to describe the horizontal motion of ice through a system of PDEs, which balances gravitational driving forces with internal stress gradients and basal drag as follows:

$$\begin{aligned}
&\frac{\partial}{\partial x}\left(4\mu H\frac{\partial u}{\partial x} + 2\mu H\frac{\partial v}{\partial y}\right) + \frac{\partial}{\partial y}\left(\mu H\frac{\partial u}{\partial y} + \mu H\frac{\partial v}{\partial x}\right) - \tau_\mathrm{b}\frac{u}{|\mathbf{u}|} = \rho_\mathrm{i} g H\frac{\partial s}{\partial x},\\
&\frac{\partial}{\partial x}\left(\mu H\frac{\partial u}{\partial y} + \mu H\frac{\partial v}{\partial x}\right) + \frac{\partial}{\partial y}\left(2\mu H\frac{\partial u}{\partial x} + 4\mu H\frac{\partial v}{\partial y}\right) - \tau_\mathrm{b}\frac{v}{|\mathbf{u}|} = \rho_\mathrm{i} g H\frac{\partial s}{\partial y},
\end{aligned} \tag{2}$$

where s is the ice surface elevation, ρ_i denotes ice density, g is gravitational acceleration, τ_b is the basal shear stress, and μ denotes the ice viscosity, which follows Glen's flow law (Glen, 1958), reflecting the nonlinear behavior of ice deformation:

$$\mu = \frac{B}{2}\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 + \frac{1}{4}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)^2 + \frac{\partial u}{\partial x}\frac{\partial v}{\partial y}\right)^{\frac{1-n}{2n}}, \tag{3}$$

where $n = 3$ is the flow-law exponent, and B is a temperature-dependent pre-factor (Cuffey and Paterson, 2010).

To characterize the relationship between basal shear stress and sliding velocity, we use Weertman's friction law (Weertman, 1957; Fowler, 1981; Cuffey and Paterson, 2010) as an example:

$$\tau_\mathrm{b} = C^2 |\mathbf{u}|^m, \tag{4}$$

where C is a spatially varying friction coefficient, and $m = 1/3$.

In regions where vertical deformation significantly influences ice flow, such as in the ice sheet interior, we incorporate a MOno-Layer Higher-Order (MOLHO) model into PINNICLE, following the approach in dos Santos et al. (2022). This formulation represents ice velocity $\mathbf{u}$ as the sum of a basal component $\mathbf{u}^\mathrm{b}$ and a shear velocity $\mathbf{u}^\mathrm{sh}$, modulated by a normalized depth factor ζ as follows:

$$\mathbf{u} = \mathbf{u}^\mathrm{b} + \mathbf{u}^\mathrm{sh}\left(1 - \zeta^{n+1}\right), \tag{5}$$

where $\zeta(z) = \frac{s-z}{H}$ scales with depth with z varying between the ice base b and the ice surface s.

Both basal and shear velocities are defined on a 2D domain, with vertical variations accounted for by the polynomial in ζ. The MOLHO model extends the SSA equations by accounting for vertical deformation, expressed as

$$\begin{aligned}
&\frac{\partial}{\partial x}\left(\overline{\mu}_1 H\epsilon^\mathrm{b}_{11} + \overline{\mu}_2 H\epsilon\mathrm{sh}_{11}\right) + \frac{\partial}{\partial y}\left(\overline{\mu}_1 H\epsilon^\mathrm{b}_{12} + \overline{\mu}_2 H\epsilon^\mathrm{sh}_{12}\right) - \tau_\mathrm{b}\frac{u_\mathrm{b}}{|\mathbf{u}_\mathrm{b}|} = \rho_\mathrm{i} g H\frac{\partial s}{\partial x},\\
&\frac{\partial}{\partial x}\left(\overline{\mu}_1 H\epsilon^\mathrm{b}_{12} + \overline{\mu}_2 H\epsilon^\mathrm{sh}_{12}\right) + \frac{\partial}{\partial y}\left(\overline{\mu}_1 H\epsilon^\mathrm{b}_{22} + \overline{\mu}_2 H\epsilon^\mathrm{sh}_{22}\right) - \tau_\mathrm{b}\frac{v_\mathrm{b}}{|\mathbf{u}_\mathrm{b}|} = \rho_\mathrm{i} g H\frac{\partial s}{\partial y},\\
&\frac{\partial}{\partial x}\left(\overline{\mu}_2 H\epsilon^\mathrm{b}_{11} + \overline{\mu}_3 H\epsilon^\mathrm{sh}_{11}\right) + \frac{\partial}{\partial y}\left(\overline{\mu}_2 H\epsilon^\mathrm{b}_{12} + \overline{\mu}_3 H\epsilon^\mathrm{sh}_{12}\right) + \overline{\mu}_4 H u^\mathrm{sh} = \frac{(n+1)}{(n+2)}\rho_\mathrm{i} g H\frac{\partial s}{\partial x},\\
&\frac{\partial}{\partial x}\left(\overline{\mu}_2 H\epsilon^\mathrm{b}_{12} + \overline{\mu}_3 H\epsilon^\mathrm{sh}_{12}\right) + \frac{\partial}{\partial y}\left(\overline{\mu}_2 H\epsilon^\mathrm{b}_{22} + \overline{\mu}_3 H\epsilon^\mathrm{sh}_{22}\right) + \overline{\mu}_4 H v^\mathrm{sh} = \frac{(n+1)}{(n+2)}\rho_\mathrm{i} g H\frac{\partial s}{\partial y},
\end{aligned} \tag{6}$$

where ϵ^b_{ij} and $\epsilon^\mathrm{sh}_{ij}$ represent the strain rate components for basal and shear velocities, respectively. These components are detailed by

$$\begin{aligned}
&\epsilon^\mathrm{b}_{11} = 4\frac{\partial u^\mathrm{b}}{\partial x} + 2\frac{\partial v^\mathrm{b}}{\partial y}, \quad \epsilon^\mathrm{b}_{12} = \frac{\partial u^\mathrm{b}}{\partial y} + \frac{\partial v^\mathrm{b}}{\partial x}, \quad \epsilon^\mathrm{b}_{22} = 2\frac{\partial u^\mathrm{b}}{\partial x} + 4\frac{\partial v^\mathrm{b}}{\partial y},\\
&\epsilon^\mathrm{sh}_{11} = 4\frac{\partial u^\mathrm{sh}}{\partial x} + 2\frac{\partial v^\mathrm{sh}}{\partial y}, \quad \epsilon^\mathrm{sh}_{12} = \frac{\partial u^\mathrm{sh}}{\partial y} + \frac{\partial v^\mathrm{sh}}{\partial x}, \quad \epsilon^\mathrm{sh}_{22} = 2\frac{\partial u^\mathrm{sh}}{\partial x} + 4\frac{\partial v^\mathrm{sh}}{\partial y},
\end{aligned} \tag{7}$$

and the vertically integrated viscosities are calculated as follows:

$$\begin{aligned}
&\overline{\mu}_1 = \frac{1}{2}\int_b^s \mu \mathrm{d}z,\ \overline{\mu}_2 = \frac{1}{2}\int_b^s \mu\left(1 - \zeta^{n+1}\right)\mathrm{d}z,\\
&\overline{\mu}_3 = \frac{1}{2}\int_b^s \mu\left(1 - \zeta^{n+1}\right)^2\mathrm{d}z,\ \overline{\mu}_4 = \frac{1}{2}\int_b^s \mu\left(\frac{n+1}{H}\zeta^n\right)^2\mathrm{d}z,
\end{aligned} \tag{8}$$

where μ is the effective viscosity defined as in Eq. (3).

The basal shear stress τ_b is defined using Weertman's friction law (Weertman, 1957), as in the SSA model. Alternative friction laws exist (Budd et al., 1979; Gagliardini et al., 2007), and PINNICLE can be easily adjusted to support these friction laws.

4 Data

PINNICLE supports a variety of data formats, including CSV,[1] NetCDF, and MATLAB data files. These files can contain scattered data or structured data, such as the model `struct` from the Ice-sheet and Sea-level System Model (ISSM). The spatial and temporal coordinates of the data can be different between different variables, offering flexibility to accommodate any complex situations. Depending on the underlying problem to solve, users can specify the amount of data needed for training. The selected data contribute to the calculation of the data misfit component within the loss function.

To evaluate the data misfit, PINNICLE provides multiple metrics, integrating both the built-in data misfit functions from the DeepXDE package (Lu et al., 2021) and commonly employed functions from other machine learning libraries. Table 1 summarizes the data misfit functions currently available in PINNICLE. In the table, d represents the predicted solution from PINNICLE, $\hat{d}$ denotes the "reference solution" from the data, and $\varepsilon = 2.2204 \times 10^{-16}$ corresponds to the machine epsilon for double precisions. For advanced users, PINNICLE also provides a flexible interface to define custom misfit functions to specific modeling needs.

5 Neural networks

The fundamental neural network architecture implemented in PINNICLE is the fully connected neural network (FNN), which takes spatial and temporal coordinates as input and the dependent variables of the PDEs as output, through one single neural network. This architecture allows the neural network to implicitly capture the internal relationship among the dependent variables, and it has been widely used in many applications (Raissi et al., 2019; Iwasaki and Lai, 2023; Wang et al., 2025; Lu et al., 2021; Karniadakis et al., 2021; Teisberg et al., 2021). However, in some cases, the training process may not converge, as the relationship between some dependent variables may be too complex for the neural network to capture (Cheng et al., 2024). To address these problems when they occur, we also provide a parallel fully connected neural network (PFNN) (Lu et al., 2021). The PFNN uses multiple FNNs for each dependent variable with the same independent variables as input. In this case, each neural network only needs to learn its corresponding dependent variable from the data and the PDE constraints.

By default, we use the hyperbolic tangent function as the activation function for the neural network. To accommodate different physical problems, PINNICLE applies min–max normalization before the input layer and a min–max denormalization after the output layer. These normalization processes are strictly limited to the inputs and outputs of the neural network, while the governing PDE residuals and data misfit terms are all defined and calculated in the International System of Units (SI). Consequently, there is no need for the user to scale the physical equations or data misfit functions themselves.

While glacier ice typically exhibits low-pass filtering behavior, some variables, such as surface elevation, contain high-frequency components. To capture these variations accurately, we implemented Fourier feature transform (FFT) (Tancik et al., 2020; Wang et al., 2021) in PINNICLE. Assuming the dimension of the input variables $\mathbf{x}$ is d, the FFT for m features is written as $f(\mathbf{x}) = [\cos(\mathbf{Bx}), \sin(\mathbf{Bx})]^T$, where $\mathbf{B} \in \mathbb{R}^{m \times d}$ is sampled from a Gaussian distribution $\mathcal{N}(0, \sigma)$; the functions $\cos(\cdot)$ and $\sin(\cdot)$ are applied coordinate-wise on the vector $\mathbf{Bx}$. This transformation is applied between the min–max normalization and the input layer of the neural network.

6 Loss function

The loss function in PINNICLE integrates both data misfit and physical constraints. The general form of the loss function is given by

$$\mathcal{L} = \mathcal{L}_{\mathrm{d}} + \mathcal{L}_{\varphi}, \tag{9}$$

where $\mathcal{L}_{\mathrm{d}}$ and $\mathcal{L}_{\varphi}$ represent the weighted sum of data misfit and PDE residual, respectively. The specific formulation of the loss function depends on the underlying physical problem, the data, and the misfit functions (Table 1) employed. Examples of detailed loss function formulations are provided in Sect. 8, specifically in Eqs. (10), (11), and (12).

Physical variables in glaciological applications span multiple orders of magnitude; for example, ice velocity is typically around $10^{-5}\,\mathrm{m\,s^{-1}}$, ice thickness is around $10^{3}\,\mathrm{m}$, and driving stress of a glacier is approximately $10^{5}\,\mathrm{Pa}$. These differences can lead to imbalanced contributions in the optimization process, where certain variables dominate the loss function while others are underrepresented. To address this issue, PINNICLE applies weights to each individual term in the loss function, ensuring balanced contributions from different data sources and governing equations. The impact of weighting strategies on ice sheet modeling problems has been examined in previous studies (Iwasaki and Lai, 2023; Cheng et al., 2024). While no theoretical guidelines exist for determining optimal weights, an extensive set of over 15 000 numerical experiments described in Cheng et al. (2024), including an L-curve analysis, demonstrated that the best performance is achieved when the contributions of all terms are weighted to approximately the same order of magnitude (i.e., around order 1). Based on this empirical finding, PINNICLE assigns default weights for each term in the loss function accordingly. The default weights and their corresponding typical values are provided in Table 2. For greater flexibility, users can modify the weights to meet their specific model-

[1] comma separated value

Table 1. Data misfit functions implemented in PINNICLE.

Key	Formula	Description
`MAE`	$E_{\mathrm{MAE}}(\hat{d}, d) = \frac{1}{n}\sum_{i=1}^{n} \lvert\hat{d}_i - d_i\rvert$	mean absolute error
`MSE`	$E_{\mathrm{MSE}}(\hat{d}, d) = \frac{1}{n}\sum_{i=1}^{n} \left(\hat{d}_i - d_i\right)^2$	mean square error
`MAPE`	$E_{\mathrm{MAPE}}(\hat{d}, d) = \frac{100}{n}\sum_{i=1}^{n} \left\lvert\frac{\hat{d}_i - d_i}{\hat{d}_i}\right\rvert$	mean absolute percentage error
`VEL_LOG`	$E_{\mathrm{VLOG}}(\hat{d}, d) = \frac{1}{n}\sum_{i=1}^{n} \frac{\ln(\lvert d_i\rvert+\varepsilon)}{\ln\left(\lvert\hat{d}_i\rvert+\varepsilon\right)}$	mean relative logarithmic error
`MEAN_SQUARE_LOG`	$E_{\mathrm{MSLOG}}(\hat{d}, d) = \frac{1}{n}\sum_{i=1}^{n} \left(\ln\left(\lvert\hat{d}_i\rvert+1\right) - \ln\left(\lvert d_i\rvert+1\right)\right)^2$	mean-squared logarithmic error

ing requirements. Examples of loss function construction and weight selection strategies are provided in Sect. 8.

7 Structure of the package

PINNICLE is configured by the user based on a nested dictionary to define the neural network architecture, governing equations, computational domains, data, and other experiment-specific settings. This design ensures flexibility for incorporating new functionalities, allowing the model to evolve alongside future advances in physics-informed machine learning. To ensure reproducibility, all user-defined configurations are saved in a JSON[2] file, which can be reloaded to replicate simulations with the same parameters.

The core structure of PINNICLE is built upon five key modules: Physics, Neural Network, Data, Domain, and PINN. The interconnected structure of these modules is depicted in Fig. 1. The Physics module constructs the PDE constraints, gathering independent and dependent variables from the governing equations. These variables are unified to establish a well-defined mapping between inputs and outputs for the PINN framework. The module assigns these variables to the Neural Network and Data modules, enabling integration of physics-based constraints into the computational pipeline.

The Neural Network module constructs the architecture for the neural network based on user configurations. It supports several popular machine learning libraries, including TensorFlow (Abadi et al., 2015), PyTorch (Paszke et al., 2019), and JAX (Bradbury et al., 2018). To enable seamless transitions between these libraries without altering the codebase, the module employs the DeepXDE framework (Lu et al., 2021) as its backend. PINNICLE supports Python 3.8 or higher and requires minimal library versions, including TensorFlow 2.11.0, PyTorch 2.5, and JAX 0.4, ensuring compatibility with current machine learning standards.

The Data module manages data integration by loading datasets and assigning them to the model with specific loss functions as described in Sect. 4. Users can control the volume of training data, enabling the PINNICLE to handle forward or inverse problems based on the combination of data and PDEs provided. To prevent overfitting and generalization, PINNICLE employs a random sampling strategy to automatically load data from the data file depending on their types, which also acts as a form of implicit regularization in the optimization problem. For mesh data (e.g., ISSM or NetCDF), it applies uniform random sampling, while scattered data are downsampled using Cartesian grids to maintain spatial coverage and avoid over-clustering before random sampling.

The Domain module defines the computational domain by generating a polygon based on a list of user-defined vertices. Using Hammersley sequence sampling (Wong et al., 1997), it produces quasi-random collocation points within the domain. These points are then utilized during training to evaluate residuals of the governing PDEs.

After configuring all four modules, the PINN module integrates them using the DeepXDE package, which provides a comprehensive framework for compiling and training PINNs. By leveraging DeepXDE's built-in capabilities, PINNICLE streamlines the training process, allowing users to focus on model setup and interpretation rather than low-level implementation details.

8 Example of applications

8.1 Example 1: an SSA inverse problem on Helheim Glacier

In this example, we use PINNICLE to reproduce results of the inverse problem described in Cheng et al. (2024). The problem involves solving the 2D SSA equation outlined in Eq. (2) for the fast-flowing region of Helheim Glacier in Southeast Greenland. We use a 6-layer fully connected neural network, with 20 neurons per layer. A schematic representation of the problem, including the data used for training, is shown in Fig. 2, and the corresponding Python implementa-

[2] JavaScript Object Notation

Table 2. Default weights and typical values of physical variables in PINNICLE.

Name	Weight	Default value	Variable	Typical value of the variable
Velocity	$\gamma_{\mathbf{u}}$	$10^{-8} \times (31\,536\,000^2)\ \mathrm{m^{-2}\,s^2}$	$u, v, \lvert\mathbf{u}\rvert$	$10^4\ \mathrm{m\,yr^{-1}}$
Thickness	γ_H	$10^{-6}\ \mathrm{m^{-2}}$	H	$10^3\ \mathrm{m}$
Surface elevation	γ_s	$10^{-6}\ \mathrm{m^{-2}}$	s	$10^3\ \mathrm{m}$
Mass balance	γ_a	$31\,536\,000^2\ \mathrm{m^{-2}\,s^2}$	a	$1\ \mathrm{m\,yr^{-1}}$
Friction coefficient	γ_C	$10^{-8}\ \mathrm{Pa^{-1}\,m^{1/3}\,s^{-1/3}}$	C	$10^4\ \mathrm{Pa^{1/2}\,m^{-1/6}\,s^{1/6}}$
Rheology pre-factor	γ_B	$10^{-18}\ \mathrm{Pa^{-2}\,s^{-2/3}}$	B	$10^9\ \mathrm{Pa\,s^{1/3}}$
Driving stress	$\gamma_{\boldsymbol{\tau}}$	$10^{-10}\ \mathrm{Pa^{-2}}$	$\rho g H \lvert\nabla s\rvert$	$10^5\ \mathrm{Pa}$
Dynamic thinning	$\gamma_{H/t}$	$10^{10}\ \mathrm{m^{-2}\,s^2}$	$\frac{\partial H}{\partial t}$	$10^3\ \mathrm{m}\,/\,31\,536\,000\ \mathrm{s}$

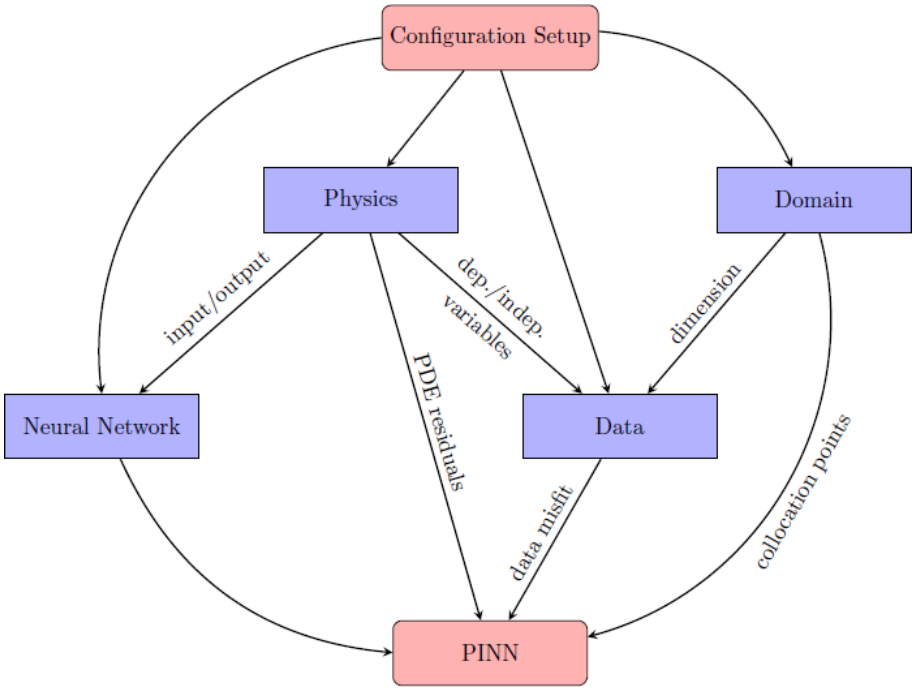

Figure 1. Flowchart illustrating the structure of PINNICLE, with arrows representing the flow of information between modules.

tion is provided in Listing 1. We configure PINNICLE to randomly sample $N_{\mathbf{u}} = N_H = N_\mathrm{s} = 4000$ data points for each input variables (lines 19 to 22), including ice velocities, surface elevation, and ice thickness, along with $N_\varphi = 9000$ collocation points over the entire basin of Helheim Glacier for evaluating the PDE residuals (lines 12 to 13). The unknown friction coefficient is inferred by specifying `"C":None` in the data section (line 20), ensuring that only boundary data are used to constrain the inversion. It is important to note that the friction coefficient within the red box in Fig. 2 is not included in the training processes but is retained as the "reference solution" for validation purposes in Fig. 3b and f.

Once the setup is complete, PINNICLE assembles the loss function based on these user-defined inputs. In this case, the complete formulation of the loss function is written as

$$\begin{aligned}\mathcal{L} = &\underbrace{\frac{\gamma_{\mathbf{u}}}{N_{\mathbf{u}}}\sum_{i=1}^{N_{\mathbf{u}}}\left(\left(\hat{u}_i - u_i\right)^2 + \left(\hat{v}_i - v_i\right)^2\right)}_{\mathcal{L}_\mathrm{u}} \\ &+ \underbrace{\frac{\gamma_H}{N_H}\sum_{i=1}^{N_H}\left(\hat{H}_i - H_i\right)^2}_{\mathcal{L}_H} + \underbrace{\frac{\gamma_\mathrm{s}}{N_\mathrm{s}}\sum_{i=1}^{N_\mathrm{s}}\left(\hat{s}_i - s_i\right)^2}_{\mathcal{L}_\mathrm{s}} \\ &+ \underbrace{\frac{\gamma_C}{N_C}\sum_{i=1}^{N_C}\left(\hat{C}_i - C_i\right)^2}_{\mathcal{L}_C} \\ &+ \underbrace{\frac{\gamma_{\boldsymbol{\tau}}}{N_\varphi}\sum_{i=1}^{N_\varphi}\left|\nabla\cdot\boldsymbol{\sigma}_\mathrm{SSA} - \boldsymbol{\tau}_\mathrm{b} - \rho_\mathrm{i} g H \nabla s\right|^2}_{\mathcal{L}_\varphi},\end{aligned} \tag{10}$$

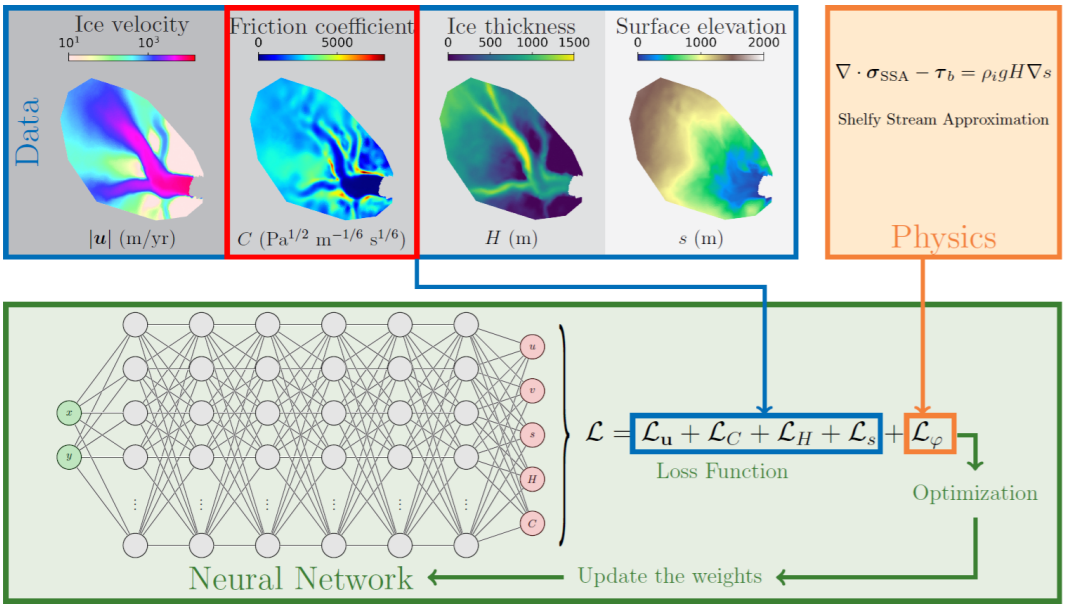

Figure 2. The PINNICLE framework for Example 1: inverse problem on Helheim Glacier. The training data include ice velocity, ice thickness, and surface elevation. The friction coefficient C, highlighted in the red box, represents the numerical solution from ISSM and is used only on the domain boundary as a boundary condition. It also serves as the "reference solution" for comparison. The governing equation in the box labeled Physics is the vector form of the SSA in Eq. (2).

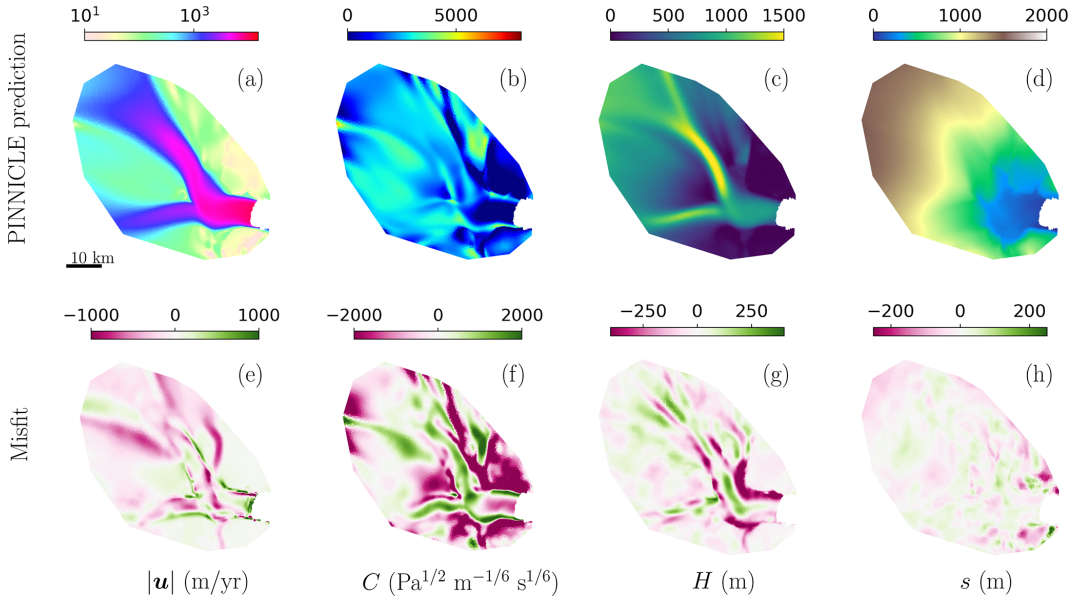

Figure 3. The PINNICLE predictions and the misfit of Example 1. **(a–d)** The predictions of surface velocity, friction coefficient, ice thickness, and surface elevation. **(e–h)** The misfit between the "reference solution" in Fig. 2 and the corresponding PINNICLE predictions in **(a)**–**(d)**.

where the weights $\gamma_{(\cdot)}$ are the default weights in PINNICLE as shown in Table 2, and $N_{\mathrm{C}} = 541$ is the number of boundary points used for constraining friction coefficient C. The terms $\mathcal{L}_{\mathbf{u}}$, $\mathcal{L}_H$, and $\mathcal{L}_{\mathrm{s}}$ are the weighted MSEs of the data misfit for ice velocity $|\mathbf{u}|$, ice thickness H, and surface elevation s, respectively. The term $\mathcal{L}_C$ accounts for the MSE of the boundary condition imposed on the friction coefficient. However, it is possible to treat C as a free parameter in the inversion. In such cases, users can exclude C from the dataset by removing the entry `"C":None` from the `issm["data_size"]` configuration (e.g., line 20 in Listing 1). PINNICLE will then automatically omit the $\mathcal{L}_C$ term from the loss function in Eq. (10). The PDE residual, denoted as $\mathcal{L}_\varphi$, is expressed in the vector form of the SSA in Eq. (2)

```python
import pinnicle

# General parameters
hp = {}
hp["epochs"] = 100000

# NN
hp["num_neurons"] = 20
hp["num_layers"] = 6

# domain
hp["shapefile"] = "Helheim.exp"
hp["num_collocation_points"] = 9000

# physics
hp["equations"] = {"SSA":{}}

# data
issm = {}
issm["data_size"] = {"u":4000, "v":4000, "s":4000, "H":4000, "C":None}
issm["data_path"] = "Helheim.mat"
hp["data"] = {"ISSM":issm}

# create experiment
experiment = pinnicle.PINN(hp)
experiment.compile()

# Train
experiment.train()
```

Listing 1. Python code of Example 1: inverse problem on Helheim Glacier.

and is weighted according to the driving stress to ensure balanced contributions in the loss function.

After training for 100 000 epochs (line 5 in Listing 1), the solution is presented in Fig. 3. The root-mean-square error (RMSE) for each variable comparing to the "reference solution" is as follows: velocity at 273.86 m a^{-1}, surface elevation at 22.21 m, ice thickness at 29.07 m, and friction coefficient at 1101.96 $Pa^{1/2}$ $m^{-1/6}$ $s^{1/6}$. These results closely align with those reported in Cheng et al. (2024), with similar spatial patterns of errors. However, our errors are approximately twice as large, due to training for only one-tenth of the epochs compared to Cheng et al. (2024). Despite this, the overall consistency demonstrates the capability of PINNICLE to reproduce and approximate inverse problem results effectively.

8.2 Example 2: simultaneous inference of basal friction and ice rheology for Pine Island Glacier

This second example demonstrates PINNICLE's capability to simultaneously infer two spatially varying parameters: the basal friction coefficient beneath grounded ice and the ice rheology, described by the flow-law pre-factor, within the floating ice shelf. We employ the SSA in Eq. (2) on Pine Island Glacier, Antarctica. To capture ice rheology variations, we incorporate a spatially varying pre-factor B in Eq. (3). The Python implementation of this example is provided in Listing 2. The neural network architecture consists of a 6-layer fully connected network with 40 neurons per layer. To effectively capture high-frequency variations, we apply a Fourier feature transformation with $\sigma = 10$ and $m = 30$, as described in Sect. 5 (lines 10 to 12 in Listing 2). Given the larger domain compared to Example 1 in Sect. 8.1, we utilize $N_{\mathbf{u}} = N_H = N_s = 8000$ data points and $N_\varphi = 18\,000$ collocation points to ensure comprehensive spatial coverage and resolution. The training dataset comprises surface velocity, surface elevation, and ice thickness data across the entire domain from `PIG.mat`. To constrain the model, we impose Dirichlet boundary conditions: setting $C = 0$ on the floating ice shelf with $N_C = 4000$ and prescribing $B = 1.41 \times 10^8$ Pa $s^{1/3}$ for grounded ice with $N_B = 4000$, corresponding to ice at $-10\,^\circ$C (Cuffey and Paterson, 2010). These two boundary conditions are imposed using a separate data file

```python
import pinnicle

# hyperparameters
hp = {}
hp["epochs"] = 1000000

# NN
hp["num_neurons"] = 40
hp["num_layers"]  = 6
hp["fft"] = True
hp['sigma'] = 10
hp['num_fourier_feature'] = 30

# domain
hp["shapefile"] = "PIG.exp"
hp["num_collocation_points"] = 18000

# physics
hp["equations"] = {"SSA_VB": {}}

# data
issm = {}
issm["data_size"] = {"u":8000, "v":8000, "s":8000, "H":8000}
issm["data_path"] = "PIG.mat"
BC = {}
BC["data_size"] = {"C":4000, "B":4000}
BC["data_path"] = "BC.mat"
BC["source"] = "mat"
hp["data"] = {"ISSM":issm, "BC":BC}

# create experiment
experiment = pinnicle.PINN(hp)
experiment.compile()

# Train
experiment.train()
```

Listing 2. Python code of Example 2: inferring basal friction and ice rheology for Pine Island Glacier.

`BC.mat`, which contains scatter data points of C and B, as well as their corresponding coordinates.

In this case, the loss function is formulated as

$$\mathcal{L} = \mathcal{L}_{\mathbf{u}} + \mathcal{L}_H + \mathcal{L}_{\mathrm{s}} + \underbrace{\frac{\gamma_C}{N_C}\sum_{i=1}^{N_C}\left(\hat{C}_i - C_i\right)^2}_{\mathcal{L}_C} + \underbrace{\frac{\gamma_B}{N_B}\sum_{i=1}^{N_B}\left(\hat{B}_i - B_i\right)^2}_{\mathcal{L}_B} + \mathcal{L}_\varphi, \tag{11}$$

where $\mathcal{L}_{\mathbf{u}}$, $\mathcal{L}_H$, $\mathcal{L}_{\mathrm{s}}$, and $\mathcal{L}_\varphi$ have the same formulation as in Eq. (10). The additional terms $\mathcal{L}_C$ and $\mathcal{L}_B$ account for the data misfit of the friction coefficient C on the floating ice shelf and the ice rheology pre-factor B on the grounded ice, respectively.

After 1 000 000 training epochs, the results are presented in Fig. 4. The RMSE for each variable, compared to the "reference solution", is as follows: surface velocity at 173.13 m a^{-1}, surface elevation at 18.28 m, ice thickness at 24.38 m, basal friction coefficient at 1242.05 Pa$^{1/2}$ m$^{-1/6}$ s$^{1/6}$, and ice rheology pre-factor at 3.81×10^7 Pa s$^{1/3}$. The magnitudes of these errors are consistent with those observed in Example 1. Larger misfits are observed in two areas: the friction coefficient in slow-moving regions and the rheology near the ice front. Despite these localized discrepancies, the overall performance demonstrates

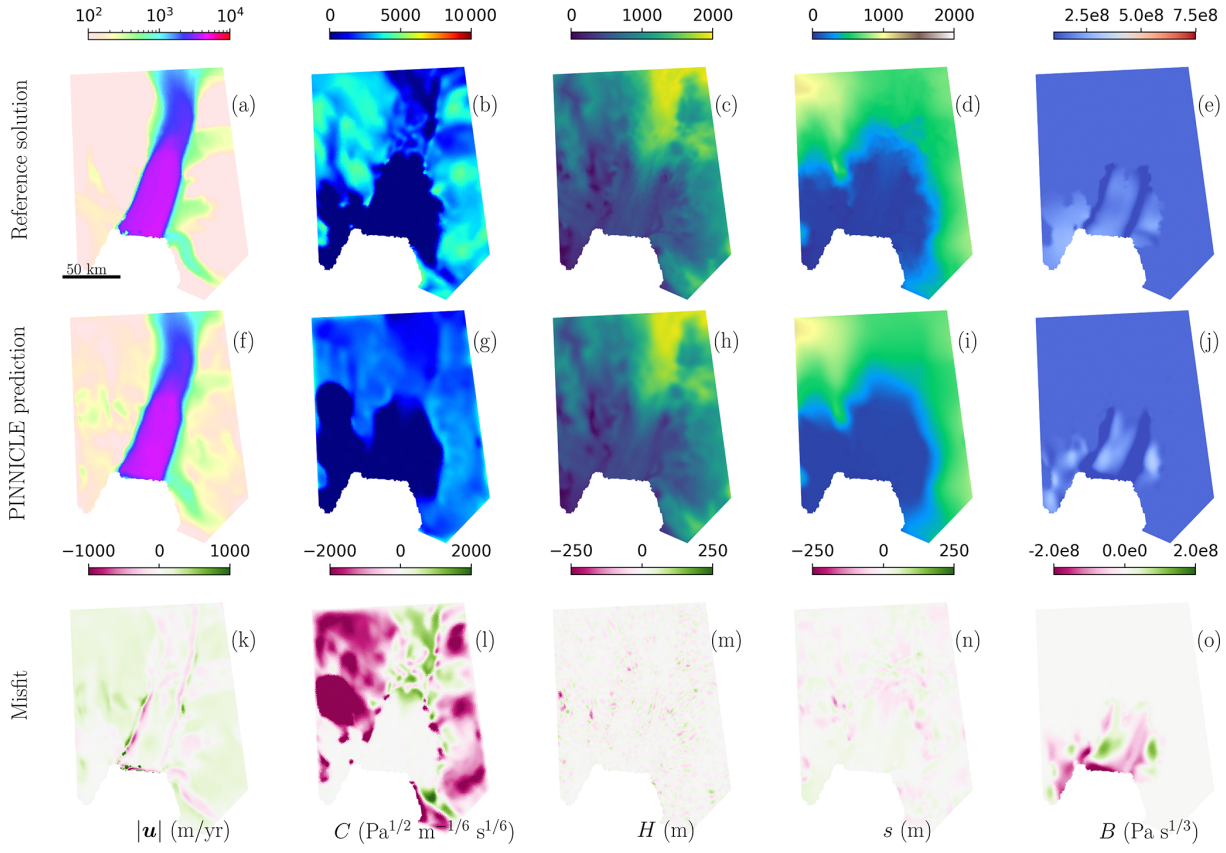

Figure 4. The comparison of the "reference solution", PINNICLE predictions, and misfit of Example 2 to infer the basal friction coefficient and ice rheology simultaneously for Pine Island Glacier. **(a–e)** The "reference solution" of surface velocity, friction coefficient, ice thickness, surface elevation, and the temperature-dependent pre-factor. **(f–j)** The PINNICLE predictions of surface velocity, friction coefficient, ice thickness, surface elevation, and the temperature-dependent pre-factor. **(k–o)** The misfit between the "reference solution" and the corresponding PINNICLE predictions.

PINNICLE's capacity to simultaneously infer multiple variables.

8.3 Example 3: time-dependent forward modeling of Helheim Glacier

In this final example, we demonstrate PINNICLE's capability to solve a transient problem by modeling the evolution of ice thickness over time using the mass transport equation in Eq. (1). Specifically, we simulate the dynamics of Helheim Glacier from 2008 to 2009. The framework for this example is illustrated in Fig. 5, and the corresponding Python implementation is provided in Listing 3. Since the problem is defined as time-dependent (lines 8 to 10), PINNICLE automatically constructs a spatiotemporal domain and sets the input variables of the neural network to include spatial coordinates x, y, and time t.

To drive the forward model, we provide a time series of ice velocity and surface mass balance at 0.1-year intervals ($N_t = 11$), with each time slice containing $N_{\mathbf{u}} = N_a = 3000$ data points for each variable. The initial condition of the ice thickness is specified at $t = 2008$, also using $N_H = 3000$ data points. These data are extracted from the transient simulation results of Cheng et al. (2022). Additionally, we randomly select $N_\varphi = 10\,000$ collocation points distributed across the spatiotemporal domain to enforce the governing equation. The loss function is written as

$$\begin{aligned}\mathcal{L} = &\underbrace{\frac{\gamma_{\mathbf{u}}}{N_{\mathbf{u}} N_t}\sum_{j=1}^{N_t}\sum_{i=1}^{N_{\mathbf{u}}}\left(\left(\hat{u}_{i,j}-u_{i,j}\right)^2+\left(\hat{v}_{i,j}-v_{i,j}\right)^2\right)}_{\mathcal{L}_{\mathbf{u}}}\\ &+\underbrace{\frac{\gamma_H}{N_H}\sum_{i=1}^{N_H}\left(\hat{H}_i-H_i\right)^2}_{\mathcal{L}_H}+\underbrace{\frac{\gamma_a}{N_a N_t}\sum_{j=1}^{N_t}\sum_{i=1}^{N_a}\left(\hat{a}_{i,j}-a_{i,j}\right)^2}_{\mathcal{L}_a}\\ &+\underbrace{\frac{\gamma_{H/t}}{N_\varphi}\sum_{i=1}^{N_\varphi}\left|\frac{\partial H}{\partial t}+\nabla\cdot(\overline{v}H)-a\right|^2}_{\mathcal{L}_\varphi},\end{aligned} \tag{12}$$

where $\mathcal{L}_{\mathbf{u}}$ and $\mathcal{L}_a$ represent the weighted data misfit functions of ice velocity and mass balance across the entire spatial and

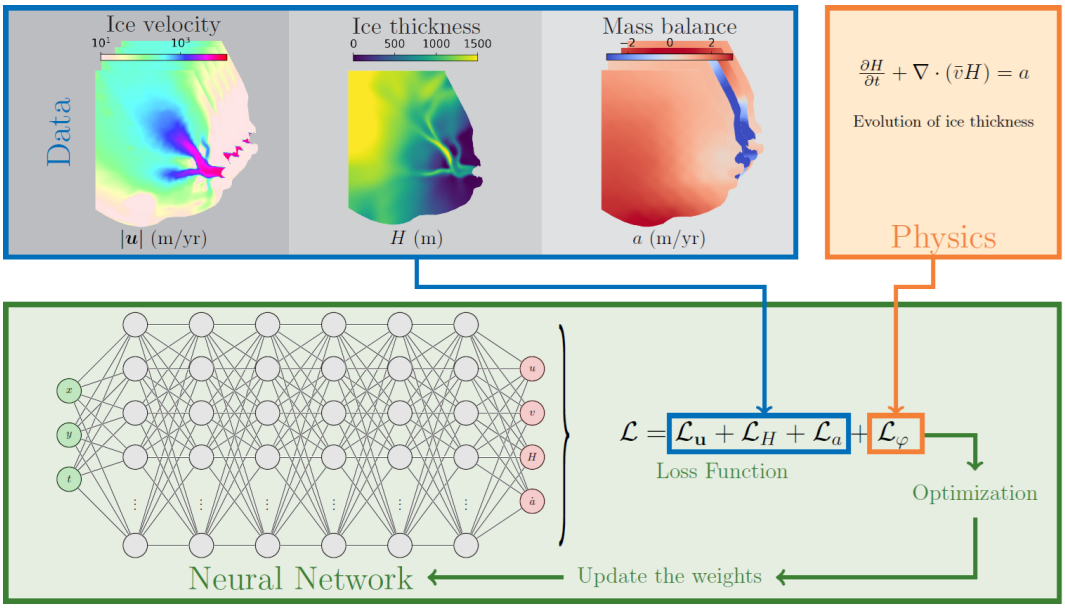

Figure 5. The PINNICLE framework for Example 3: the forward time-dependent problem. The training data consist of the initial ice thickness, a time series of ice velocity, and apparent mass balance. Notably, the neural network inputs are automatically adapted to incorporate both spatial and temporal variables.

temporal domain, $\mathcal{L}_H$ accounts for the misfit in the initial ice thickness, and $\mathcal{L}_\varphi$ corresponds to the PDE residual of the mass conservation in Eq. (1).

After training for 800 000 epochs using a fully connected neural network with 6 layers and 32 neurons per layer, the results at the initial time step ($t = 2008$) are shown in Fig. 6, while those at the final time step ($t = 2009$) are presented in Fig. 7. The RMSE values for each variable over the entire simulation period (2008–2009), compared to the "reference solution", are as follows: surface velocity at 186.18 m a^{-1}, surface mass balance at 0.03 m a^{-1}, and ice thickness at 53.11 m. These errors remain within the same order of magnitude as the other two examples, demonstrating PINNICLE's performance in solving transient problems.

9 Performance

We present the performance of the examples in Table 3. All experiments were conducted on the Texas Advanced Computing Center Lonestar6 system, using nodes equipped with NVIDIA A100 GPUs with 40 GB of high-bandwidth memory each. The reported wall time (in hours) represents the average of five identical runs for each numerical experiment. The computational cost is influenced by the number of epochs, neural network parameters, data points, and collocation points, among other factors. It is essential to recognize that problems similar to Examples 1 and 3 typically can be resolved more quickly with traditional numerical models, such as ISSM (Larour et al., 2012), often within just a few minutes on multicore CPUs to achieve comparable accuracy (Cheng et al., 2024, 2022). However, solving complex mixed inverse problems, like the one presented in Example 2, using traditional adjoint methods is nontrivial, since it would include the derivation of new adjoint equations with respect to the two unknown variables, simultaneously. Furthermore, we highlight that PINNICLE's current computational performance has not been fully optimized yet, and rapid advances in hardware technology will help further increase its performance. For example, migrating our experiments from NVIDIA V100 GPUs (Cheng et al., 2024) to NVIDIA A100 GPUs has led to at least a twofold reduction in computation time.

Additionally, direct comparisons between machine learning frameworks (on GPU) and traditional data assimilation methods (on CPU) remain limited even in the broader climate science community. Lai et al. (2024), for example, points out critical differences between PINNs and conventional ensemble-based data assimilation methods, such as ensemble Kalman filters (Bauer et al., 2015). PINNs offer the distinct advantage of simultaneously addressing forward and inverse problems, significantly reducing computational demands compared to ensemble methods, which require multiple simulation runs. Conversely, ensemble methods, despite being computationally intensive, provide explicit uncertainty quantification through ensemble spread. Relative to classical adjoint methods, PINNs demonstrate enhanced robustness to noisy observational data, effectively resolving inversion problems with smoother, physically consistent solutions (Wang et al., 2025; Cheng et al., 2024).

```python
import pinnicle
import numpy as np
# hyperparameters
hp = {}
hp["epochs"] = 800000

# time dependent problem
hp["time_dependent"] = True
hp["start_time"]     = 2008
hp["end_time"]       = 2009

# NN
hp["num_neurons"] = 32
hp["num_layers"]  = 6

# domain
hp["shapefile"] = "Helheim_Basin.exp"
hp["num_collocation_points"] = 10000

# physics
hp["equations"] = {"Mass transport":{}}

# data
hp["data"] = {}
for t in np.linspace(2008,2009,11):
    issm = {}
    if t == 2008:
        issm["data_size"] = {"u":3000, "v":3000, "a":3000, "H":3000}
    else:
        issm["data_size"] = {"u":3000, "v":3000, "a":3000, "H":None}

    issm["data_path"]         = "Helheim_Transient_" + "%g"%t + ".mat"
    issm["default_time"]      = t
    issm["source"]            = "ISSM"
    hp["data"]["ISSM"+"%g"%t] = issm

# create experiment
experiment = pinnicle.PINN(hp)
experiment.compile()

# Train
experiment.train()
```

Listing 3. Python code of Example 3: time-dependent problem of Helheim Glacier.

Table 3. Performance of the examples in PINNICLE.

Name	Equations	Neural network	Epochs	# of parameters	# of data	# of collocation points	Average wall time
Example 1	SSA	6×20	10^5	2286	4000	9000	0.48 h
Example 2	SSA_VB	6×40	10^6	10 886	8000	18 000	14.97 h
Example 3	Mass transport	6×32	8×10^5	5540	33 000	10 000	4.22 h

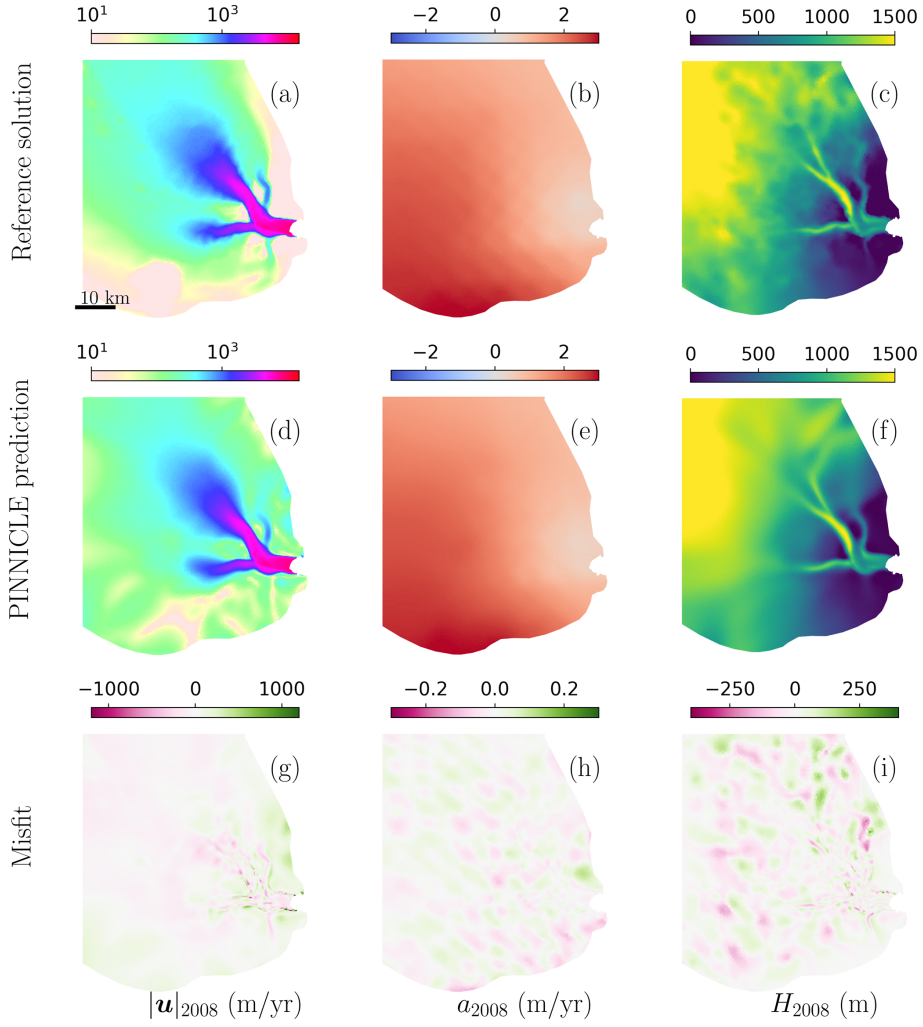

Figure 6. The comparison of the "reference solution", PINNICLE predictions, and misfits of the transient simulation in Example 3 at the initial time step $t = 2008$. **(a–c)** The "reference solution" of surface velocity, mass balance, and ice thickness. **(d–f)** The PINNICLE predictions. **(g–i)** The misfit between the "reference solution" and the corresponding PINNICLE predictions.

For the current version of PINNICLE, all spatial, temporal, and parameter derivatives are computed using reverse-mode automatic differentiation (AD) as implemented in the DeepXDE backend. Although it has been shown that using forward-mode AD for spatial derivatives (e.g., with respect to t, x, and y) and reverse-mode AD for network parameters can significantly improve memory and computational efficiency (Cho et al., 2023), the current version of DeepXDE does not yet support this separable AD approach. We are actively following the ongoing development of this feature and plan to integrate it into PINNICLE as soon as it becomes available. We anticipate that the incorporation of this strategy in future versions of PINNICLE will further enhance the scalability and efficiency, particularly for large-scale simulations.

10 Conclusions

In this study, we introduced PINNICLE, a flexible and robust framework designed to solve a wide range of glaciological problems using Physics-Informed Neural Networks. PINNICLE integrates observational data with physical laws, enabling both inverse and forward modeling across diverse spatial and temporal scales. The framework demonstrates consistent performance in inferring spatially and temporally varying parameters. PINNICLE's flexibility allows users to customize neural network architectures, incorporate various types of data, and apply different physical constraints, making it suitable for complex modeling tasks in glaciology. Additionally, we emphasize that the PINNICLE framework is not intended to replace traditional numerical methods for solving standard forward or inverse problems. Instead, it is best viewed as a complementary tool, especially useful

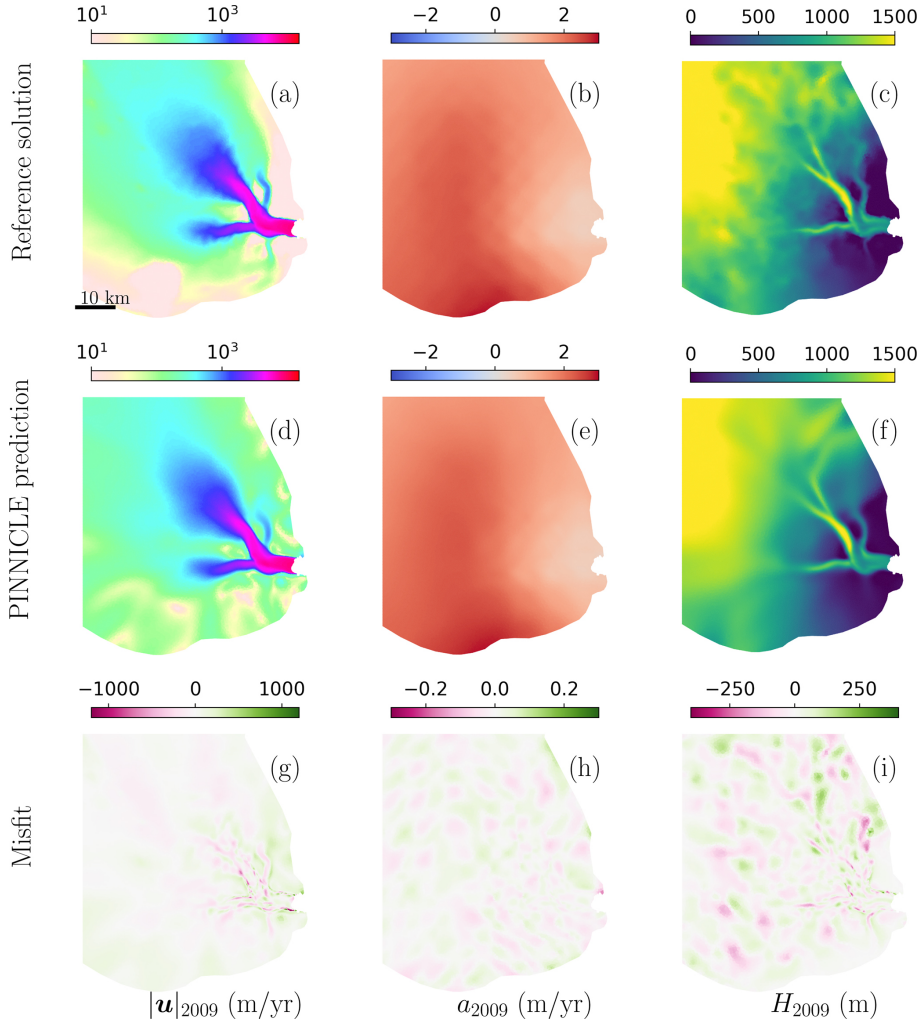

Figure 7. The comparison of the "reference solution", PINNICLE predictions, and misfits of the transient simulation in Example 3 at the final time step $t = 2009$. **(a–c)** The "reference solution" of surface velocity, mass balance, and ice thickness. Note that **(c)** is not exposed to the training. **(d–f)** The PINNICLE predictions. **(g–i)** The misfit between the "reference solution" and the corresponding PINNICLE predictions.

in scenarios involving the integration of diverse physical processes and the exploration of innovative scientific concepts. PINNICLE's ability to seamlessly integrate observational data with physical laws within a unified framework enhances flexibility and can significantly simplify the implementation and evaluation of new physical models. Future developments will focus on incorporating more advanced neural network architectures, optimization techniques, and additional physical constraints to further enhance accuracy and computational efficiency. PINNICLE's modular design ensures adaptability to evolving machine learning methodologies and glaciological challenges, positioning it as a valuable tool for the cryosphere community.

Code and data availability. The source code and development history are hosted on GitHub at https://github.com/ISSMteam/PINNICLE (last access: 22 August 2025). The specific version of PINNICLE used in this study, including all examples, input data used for training, and neural network weights after training, has been archived on Zenodo and is available at https://doi.org/10.5281/zenodo.15643042 (Cheng et al., 2025). All examples mentioned in this study are organized in the folder: `PINNICLE/examples/`. The code used in this work is available as a Python package on PyPI. It can be installed using `pip install pinnicle`. This software is licensed under the GNU Lesser General Public License v2 (LGPLv2).

Author contributions. GC, MK, and MM designed the study. GC did the numerical computations. GC wrote the manuscript with input from MK and MM.

Competing interests. The authors declare that they have no conflict of interest.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

Acknowledgements. This work was supported by the National Science Foundation, #2118285 and #2147601. Gong Cheng acknowledges support from the Novo Nordisk Foundation under the Challenge Programme 2023 (grant NNF23OC00807040). The authors gratefully acknowledge the data and models obtained from projects supported by the Heising-Simons Foundation under grants 2019-1161 and 2021-3059.

Financial support. This research has been supported by the Novo Nordisk Fonden (grant no. NNF23OC00807040), the National Science Foundation (grant nos. 2147601 and 2118285), and Heising-Simons Foundation (grant nos. 2019-1161 and 2021-3059).

Review statement. This paper was edited by Ludovic Räss and reviewed by Facundo Sapienza and one anonymous referee.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems, https://www.tensorflow.org (last access: 14 August 2025), 2015.

Aschwanden, A., Fahnestock, M. A., Truffer, M., Brinkerhoff, D. J., Hock, R., Khroulev, K., Mottram, R., and Khan, S. A.: Contribution of the Greenland Ice Sheet to sea level over the next millennium, Sci. Adv., 5, eaav9396, https://doi.org/10.1126/sciadv.aav9396, 2019.

Aschwanden, A., Bartholomaus, T. C., Brinkerhoff, D. J., and Truffer, M.: Brief communication: A roadmap towards credible projections of ice sheet contribution to sea level, The Cryosphere, 15, 5705–5715, https://doi.org/10.5194/tc-15-5705-2021, 2021.

Bauer, P., Thorpe, A., and Brunet, G.: The Quiet Revolution of Numerical Weather Prediction, Nature, 525, 47–55, https://doi.org/10.1038/nature14956, 2015.

Bolibar, J., Sapienza, F., Maussion, F., Lguensat, R., Wouters, B., and Pérez, F.: Universal differential equations for glacier ice flow modelling, Geosci. Model Dev., 16, 6671–6687, https://doi.org/10.5194/gmd-16-6671-2023, 2023.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q.: JAX: composable transformations of Python+NumPy programs, GitHub [code], http://github.com/jax-ml/jax (last access: 14 August 2025), 2018.

Brinkerhoff, D. J.: Variational inference at glacier scale, J. Comput. Phys., 459, 111095, https://doi.org/10.1016/j.jcp.2022.111095, 2022.

Budd, W. F., Keage, P. L., and Blundy, N. A.: Empirical studies of ice sliding, J. Glaciol., 23, 157–170, 1979.

Cheng, G. and Lotstedt, P.: Parameter sensitivity analysis of dynamic ice sheet models – numerical computations, The Cryosphere, 14, 673–691, https://doi.org/10.5194/tc-14-673-2020, 2020.

Cheng, G., Morlighem, M., Mouginot, J., and Cheng, D.: Helheim Glacier's Terminus Position Controls Its Seasonal and Inter-Annual Ice Flow Variability, Geophys. Res. Lett., 49, e2021GL097085, https://doi.org/10.1029/2021GL097085, 2022.

Cheng, G., Morlighem, M., and Francis, S.: Forward and Inverse Modeling of Ice Sheet Flow Using Physics-Informed Neural Networks: Application to Helheim Glacier, Greenland, J. Geophys. Res. Mach. Learn. Comput., 1, e2024JH000169, https://doi.org/10.1029/2024JH000169, 2024.

Cheng, G., Krishna, M., and Francis, S.: ISSMteam/PINNICLE: Official release version of PINNICLE, Zenodo [code], https://doi.org/10.5281/zenodo.15643042, 2025.

Cho, J., Nam, S., Yang, H., Yun, S.-B., Hong, Y., and Park, E.: Separable Physics-Informed Neural Networks, arXiv [preprint], https://arxiv.org/abs/2306.15969 (last access: 14 August 2025), 2023.

Colgan, W., MacGregor, J. A., Mankoff, K. D., Haagenson, R., Rajaram, H., Martos, Y. M., Morlighem, M., Fahnestock, M. A., and Kjeldsen, K. K.: Topographic Correction of Geothermal Heat Flux in Greenland and Antarctica, J. Geophys. Res., 126, e2020JF005598, https://doi.org/10.1029/2020JF005598, 2021.

Cuffey, K. M. and Paterson, W. S. B.: The Physics of Glaciers, in: 4th Edn., Elsevier, Oxford, ISBN 978-0123694614, 2010.

dos Santos, T. D., Morlighem, M., and Brinkerhoff, D.: A new vertically integrated, MOno-Layer Higher-Order ice flow model (MOLHO), The Cryosphere, 16, 179–195, https://doi.org/10.5194/tc-16-179-2022, 2022.

Edwards, T. L., Brandon, M. A., Durand, G., Edwards, N. R., Golledge, N. R., Holden, P. B., Nias, I. J., Payne, A. J., Ritz, C., and Wernecke, A.: Revisiting Antarctic ice loss due to marine ice-cliff instability, Nature, 566, 58–64, https://doi.org/10.1038/s41586-019-0901-4, 2019.

Fowler, A. C.: A theoretical treatment of the sliding of glaciers in the absence of cavitation, Philos. T. Roy. Soc. A, 298, 637–685, https://doi.org/10.1098/rsta.1981.0003, 1981.

Furst, J. J., Durand, G., Gillet-Chaulet, F., Merino, N., Tavard, L., Mouginot, J., Gourmelen, N., and Gagliardini, O.: Assimilation of Antarctic velocity observations provides evidence for uncharted pinning points, The Cryosphere, 9, 1427–1443, https://doi.org/10.5194/tc-9-1427-2015, 2015.

Gagliardini, O., Cohen, D., Raback, P., and Zwinger, T.: Finite-element modeling of subglacial cavities and re-

lated friction law, J. Geophys. Res.-Earth, 112, 1–11, https://doi.org/10.1029/2006JF000576, 2007.

Glen, J. W.: The flow law of ice: A discussion of the assumptions made in glacier theory, their experimental foundations and consequences, IASH Publ., 47, 171–183, 1958.

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.

Goldberg, D. N. and Heimbach, P.: Parameter and state estimation with a time-dependent adjoint marine ice sheet model, The Cryosphere, 7, 1659–1678, https://doi.org/10.5194/tc-7-1659-2013, 2013.

Goldberg, D. N. and Sergienko, O. V.: Data assimilation using a hybrid ice flow model, The Cryosphere, 5, 315–327, https://doi.org/10.5194/tc-5-315-2011, 2011.

He, Q., Perego, M., Howard, A. A., Karniadakis, G. E., and Stinis, P.: A hybrid deep neural operator/finite element method for ice-sheet modeling, J. Comput. Phys., 492, 112428, https://doi.org/10.1016/j.jcp.2023.112428, 2023.

Iwasaki, Y. and Lai, C.-Y.: One-dimensional ice shelf hardness inversion: Clustering behavior and collocation resampling in physics-informed neural networks, J. Comput. Phys., 492, 112435, https://doi.org/10.1016/j.jcp.2023.112435, 2023.

Joughin, I., Shapero, D., Dutrieux, P., and Smith, B.: Ocean-induced melt volume directly paces ice loss from Pine Island Glacier, Sci. Adv., 7, eabi5738, https://doi.org/10.1126/sciadv.abi5738, 2021.

Jouvet, G. and Cordonnier, G.: Ice-flow model emulator based on physics-informed deep learning, J. Glaciol., 69, 1941–1955, https://doi.org/10.1017/jog.2023.73, 2023.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L.: Physics-informed machine learning, Nat. Rev. Phys., 3, 422–440, https://doi.org/10.1038/s42254-021-00314-5, 2021.

Koo, Y., Cheng, G., Morlighem, M., and Rahnemoonfar, M.: Calibrating calving parameterizations using graph neural network emulators: Application to Helheim Glacier, East Greenland, EGUsphere [preprint], https://doi.org/10.5194/egusphere-2024-1620, 2024.

Lai, C.-Y., Hassanzadeh, P., Sheshadri, A., Sonnewald, M., Ferrari, R., and Balaji, V.: Machine Learning for Climate Physics and Simulations, Annu. Rev. Condens. Matt. Phys., 16, 343–365, https://doi.org/10.1146/annurev-conmatphys-043024-114758, 2024.

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., 117, 1–20, https://doi.org/10.1029/2011JF002140, 2012.

Lu, L., Meng, X., Mao, Z., and Karniadakis, G. E.: DeepXDE: A Deep Learning library for solving differential equations, SIAM Rev., 63, 208–228, https://doi.org/10.1137/19M1274067, 2021.

MacAyeal, D. R.: Large-scale ice flow over a viscous basal sediment: Theory and application to Ice Stream B, Antarctica, J. Geophys. Res., 94, 4071–4087, 1989.

MacAyeal, D. R.: A tutorial on the use of control methods in ice-sheet modeling, J. Glaciol., 39, 91–98, 1993.

Morlighem, M., Rignot, E., Seroussi, H., Larour, E., Ben Dhia, H., and Aubry, D.: Spatial patterns of basal drag inferred using control methods from a full-Stokes and simpler models for Pine Island Glacier, West Antarctica, Geophys. Res. Lett., 37, 1–6, https://doi.org/10.1029/2010GL043853, 2010.

Morlighem, M., Seroussi, H., Larour, E., and Rignot, E.: Inversion of basal friction in Antarctica using exact and incomplete adjoints of a higher-order model, J. Geophys. Res., 118, 1746–1753, https://doi.org/10.1002/jgrf.20125, 2013.

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 multi-beam echo sounding combined with mass conservation, Geophys. Res. Lett., 44, 11,051–11,061, https://doi.org/10.1002/2017GL074954, 2017GL074954, 2017.

Morlighem, M., Rignot, E., Binder, T., Blankenship, D., Drews, R., Eagles, G., Eisen, O., Ferraccioli, F., Forsberg, R., Fretwell, P., Goel, V., Greenbaum, J. S., Gudmundsson, H., Guo, J., Helm, V., Hofstede, C., Howat, I., Humbert, A., Jokat, W., Karlsson, N. B., Lee, W. S., Matsuoka, K., Millan, R., Mouginot, J., Paden, J., Pattyn, F., Roberts, J., Rosier, S., Ruppel, A., Seroussi, H., Smith, E. C., Steinhage, D., Sun, B., v. d. Broeke, M. R., v. Ommen, T. D., v. Wessem, M., and Young, D. A.: Deep glacial troughs and stabilizing ridges unveiled beneath the margins of the Antarctic ice sheet, Nat. Geosci., 13, 132–137, https://doi.org/10.1038/s41561-019-0510-8, 2020.

Morlighem, M., Goldberg, D., Barnes, J. M., Bassis, J. N., Benn, D. I., Crawford, A. J., Gudmundsson, G. H., and Seroussi, H.: The West Antarctic Ice Sheet may not be vulnerable to marine ice cliff instability during the 21st century, Sci. Adv., 10, eado7794, https://doi.org/10.1126/sciadv.ado7794, 2024.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library, arXiv [preprint], https://doi.org/10.48550/arXiv.1912.01703, 2019.

Raissi, M., Perdikaris, P., and Karniadakis, G. E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys., 378, 686–707, https://doi.org/10.1016/j.jcp.2018.10.045, 2019.

Raissi, M., Yazdani, A., and Karniadakis, G. E.: Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science, 367, 6481, 1026–1030, https://doi.org/10.1126/science.aaw4741, 2020.

Riel, B., Minchew, B., and Bischoff, T.: Data-driven inference of the sechanics of slip slong glacier beds using physics-informed neural networks: case study on Rutford Ice Stream, Antarctica, J. Adv. Model. Earth Syst., 13, e2021MS002621, https://doi.org/10.1029/2021MS002621, 2021.

Robel, A. A., Seroussi, H., and Roe, G. H.: Marine ice sheet instability amplifies and skews uncertainty in projections of future sea-level rise, P. Natl. Acad. Sci. USA, 116, 14887–14892, https://doi.org/10.1073/pnas.1904822116, 2019.

Schoof, C.: Ice sheet grounding line dynamics: Steady states, stability, and hysteresis, J. Geophys. Res., 112, 1–19, https://doi.org/10.1029/2006JF000664, 2007.

Seo, J.: Solving real-world optimization tasks using physics-informed neural computing, Sci. Rep., 14, 202, https://doi.org/10.1038/s41598-023-49977-3, 2024.

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.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains, Adv. Neural Inf. Process. Syst., 33, 7537–7547, 2020.

Teisberg, T. O., Schroeder, D. M., and MacKie, E. J.: A machine learning approach to mass-conserving ice thickness interpolation, in: 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 2021, 8664–8667, https://doi.org/10.1109/IGARSS47720.2021.9555002, 2021.

Tikhonov, A. N.: On the stability of inverse problems, C. R. (Dokl.) Acad. Sci. SSSR (N.S.), 39, 176–179, 1943.

Wang, S., Teng, Y., and Perdikaris, P.: Understanding and mitigating gradient flow pathologies in physics-informed neural networks, SIAM J. Sci.Comput., 43, A3055–A3081, 2021.

Wang, Y., Lai, C.-Y., Prior, D. J., and Cowen-Breen, C.: Deep learning the flow law of Antarctic ice shelves, Science, 387, 1219–1224, https://doi.org/10.1126/science.adp3300, 2025.

Weertman, J.: On the sliding of glaciers, J. Glaciol., 3, 33–38, 1957.

Wernecke, A., Edwards, T. L., Holden, P. B., Edwards, N. R., and Cornford, S. L.: Quantifying the Impact of Bedrock Topography Uncertainty in Pine Island Glacier Projections for This Century, Geophys. Res. Lett., 49, e2021GL096589, https://doi.org/10.1029/2021GL096589, 2023.

Wong, T.-T., Luk, W.-S., and Heng, P.-A.: Sampling with Hammersley and Halton points, J. Graph. Tools, 2, 9–24, 1997.