INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 22, Number 1, Pages 1–20

doi: 10.4208/ijnam2025-1001

FROM OBSTACLE PROBLEMS TO NEURAL INSIGHTS: FEEDFORWARD NEURAL NETWORK MODELING OF ICE THICKNESS

KAPIL CHAWLA, WILLIAM HOLMES, AND ROGER TEMAM

Abstract. In this study, we integrate the established obstacle problem formulation from ice sheet modeling [1, 2] with cutting-edge deep learning methodologies to enhance ice thickness predictions, specifically targeting the Greenland ice sheet. By harmonizing the mathematical structure with an energy minimization framework tailored for neural network approximations, our method's efficacy is confirmed through both 1D and 2D numerical simulations. Utilizing the NSIDC-0092 dataset for Greenland [22] and incorporating bedrock topography for model pre-training, we register notable advances in prediction accuracy. Our research underscores the potent combination of traditional mathematical models and advanced computational techniques in delivering precise ice thickness estimations.

Key words. Neural networks, ice thickness estimation, obstacle problems, feedforward neural networks, mathematical modeling, partial differential equations.

1. Introduction

The melting of ice sheets, driven by climate change, is a topic of mounting concern across various scientific disciplines. This phenomenon is pivotal for understanding the dynamic processes of Earth's climate, particularly in regions such as Greenland. The melting of Greenland's ice sheet not only contributes to global sea level rise but also provides insights into intricate climate interactions and feedback loops. As such, mathematicians have developed complex models to delve deeper into ice sheet dynamics. Among these models, obstacle problems [3] offer a unique lens, presenting challenges in partial differential equations (PDEs). Over the years, numerous numerical methods have been devised to address these challenges. Most of these methods focus on providing approximation solutions to the weak variational inequality. Techniques like the Galerkin least squares finite element method ([5], [4], [6]), multigrid algorithm ([8], [7]), piecewise linear iterative algorithm [9], the first-order least-squares method [10], the level set method [11], and the dynamical functional particle method [12] have been employed with varying degrees of success.

In the wake of technological advancements, deep learning has emerged as a promising tool in many scientific applications. It has recently gained significant traction in solving differential equations and inverse problems ([13], [14], [15], [16], [17], [18]). Despite this momentum, its application to variational inequalities remains in its infancy. Some studies ([20], [21]) have innovatively applied deep learning to the traditional obstacle problem, whereas others [19] have ventured into using deep learning techniques for elliptic hemivariational inequalities. A common observation, however, is that many of these studies prioritize computations over theoretical insights.

With this backdrop, our paper endeavors to bridge this gap. We explore both traditional ice-sheet models [1, 2] and introduce a computational approach using

Received by the editors on May 14, 2024 and, accepted on September 3, 2024.
2000 *Mathematics Subject Classification.* 68T01, 35R35, 65N99.

deep learning to address the obstacle problem, deriving inspiration from its variational form. A central theme of our work is to discern the influence of parameters such as network size and training samples on the outcomes. Through rigorous numerical experiments, we substantiate the efficacy of our proposed method.

This article is organized as follows: Section 2 introduces the mathematical formulation of the model and elucidates the ice-thickness variational inequality. Section 3 sheds light on the energy minimization formulation. Section 4 delineates the approximation of the solution using fully connected feedforward deep neural networks, detailing its architecture, universality as an approximator, and the composite loss function tailored for optimal training and optimization. Section 5 showcases numerical experiments for one and two-dimensional problems, accompanied by solution visualizations and error analysis. Section 6 applies our model to data sourced from Greenland. We wrap up in Section 7, offering a concise summary of our study's principal insights and findings.

2. Mathematical Formulation of the Model

In this section, we present the mathematical formulation of the model, which is adapted from the work presented in [1, 2].

Let $\mathbb{R}^n$ denote the n-dimensional Euclidean space, equipped with the standard Euclidean norm. A domain Ω in $\mathbb{R}^n$ is defined as a bounded and connected open subset of $\mathbb{R}^n$, whose boundary is Lipschitz continuous. Consider a subset Ω residing within $\mathbb{R}^2$. For any point $x = (x_1, x_2)$ contained within the closure of Ω, denoted as $\bar{\Omega}$, we will utilize common mathematical operators without going into their detailed definitions here.

The bedrock elevation is denoted by the function $b : \bar{\Omega} \to \mathbb{R}$. It's noteworthy that a positive value of b represents elevations above sea level, while negative values correspond to depths below the sea level.

Similarly, the elevation of the top surface of the ice sheet is characterized by the function $h : \bar{\Omega} \to \mathbb{R}$. It is imperative to emphasize that throughout the domain Ω, h always maintains a value greater than or equal to b. A visual representation of this relationship is provided in Figure 1. Consequently, the thickness of the ice, denoted as $H : h - b$, consistently remains nonnegative throughout $\bar{\Omega}$. This insight underscores the observation that studying changes in ice thickness is tantamount to addressing an obstacle problem, where the bedrock acts as the primary constraint.

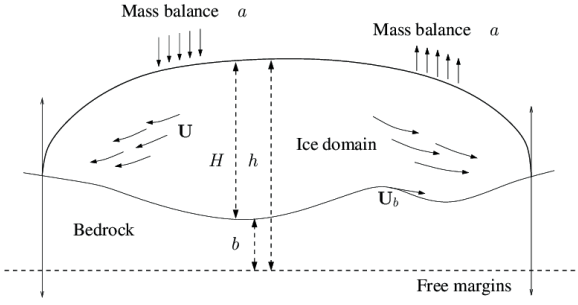

FIGURE 1. Cross-sectional view of an ice sheet with the respective notation, based on Jouvet et al. (2012).

This particular constraint implies the existence of a free boundary. Let's define

$$\Omega^+ = \{h > b\} = \{H > 0\} \tag{1}$$

as the region within Ω where the ice is present. Accordingly, the associated free boundary can be expressed as:

$$\Gamma_f = \Omega \cap \partial\Omega^+. \tag{2}$$

For a continuous ice sheet, wherein Ω^+ is populated, the source function a should inherently display a positive value, denoting ice accumulation, within specified regions of Ω^+. However, considering a defined free margin and the continuity of a, a is expected to yield a negative value, indicative of ablation, outside Ω^+, especially within $\Omega^- = \Omega\backslash(\Omega^+ \cup \Gamma)$. It is in this dynamic that the ice migrates from accumulation zones to areas dominated by ablation, resulting in the ice sheet tapering to an imperceptible thickness at its edges.

Moreover, the ice thickness is also influenced by the basal sliding velocity, U_b. This velocity can be conceptualized as a specific vector field within $\mathbb{R}^2$, which is contingent solely on its lateral position. This can be formally described as:

$$U_b : \bar{\Omega} \to \mathbb{R}^2.$$

As documented in references such as [23], it is generally accepted that, when the ice base is in a frozen state, U_b inherently equals zero.

The horizontal ice flow velocity is represented by the vector field $U : \bar{\Omega} \times \mathbb{R} \to \mathbb{R}^2$. The viscosity of the ice is described in terms of the Glen power law with ice softness coefficient $A(x, z)$ and exponent $2.8 \leq p \leq 5$. The attainable values for p are suggested by laboratory experiments [26]. The vector field U can be computed using the surface elevation h and its gradient as discussed in [23].

$$U(x, z) = -(2\rho g)^{p-1} \left[\int_b^z A(s)(h - s)^{p-1} ds \right] |\nabla h|^{p-2} \nabla h + U_b. \tag{3}$$

Here, ρ denotes the ice's density and g symbolizes the gravitational acceleration.

The morphology of ice sheets is independent of the surface equation, which correlates the ice surface's motion with the ice velocity and the mass balance data $a = a(x, z)$ [24]. Equivalently for this incompressible flow, in steady state a continuity equation applies to the flow where the volume flux is characterized by the vector field $\mathbf{q}$, defined as the integral of the ice flow velocity U concerning the vertical direction [23]:

$$\nabla \cdot \mathbf{q} = a. \tag{4}$$

Given by:

$$\mathbf{q} = \int_b^h U(z) dz. \tag{5}$$

By inclusion of (3) and (5) into (4), the equation for surface elevation h can be formulated as:

$$-\nabla \cdot \left(-(2\rho g)^{p-1} \left[\int_b^z A(s)(h - s)^{p-1} ds \right] |\nabla h|^{p-2} \nabla h - (h - b) U_b \right) = a. \tag{6}$$

For a scenario where $A(x, z) = A_0$ remains constant, Equation (6) simplifies to:

$$-\nabla \cdot \left(K(h - b)^{p+1} |\nabla h|^{p-2} \nabla h - (h - b) U_b \right) = a, \tag{7}$$

with K being a positive constant. It's imperative to note that Equations (6) and (7) are only applicable within the domain containing ice.

To form a weak solution for problem (7) over the entire domain Ω, one must posit the absence of a volumetric ice flow towards Ω^-:

$$\mathbf{q} \cdot \nu = 0 \text{ and } H = 0 \text{ on } \Gamma_f. \tag{8}$$

Here, ν represents the outward unit normal vector along Γ. Taking (8) into account, it's intuitive to extrapolate the volume flux $\mathbf{q}$ as zero outside Ω^+:

$$\mathbf{q} = 0 \text{ in } \Omega^-. \tag{9}$$

Let's introduce v as a smooth test function that satisfies $v \geq b$ throughout $\bar{\Omega}$ and multiply (5) by $(v-h)$, subsequently integrating over Ω. Applying Gauss's theorem yields:

$$-\int_\Omega \mathbf{q}\cdot\nabla(v-h) = \int_{\Omega^-}(\nabla\cdot\mathbf{q})(v-h) + \int_{\Omega^+}(\nabla\cdot\mathbf{q})(v-h) - \int_{\Gamma_f}[(v-h)\mathbf{q}\cdot\nu]^+_-. \tag{10}$$

Here, the difference across Γ_f is represented by $[\ ]^+_-$. Given the conditions $\mathbf{q} = 0$ and $a \leq 0$ within Ω^-, it follows that $\nabla\cdot\mathbf{q} \geq a$ in Ω^-. Moreover, since $\mathbf{q}\cdot\nu = 0$ on Γ_f, we deduce:

$$-\int_\Omega \mathbf{q}\cdot\nabla(v-h) \geq \int_\Omega a(v-h). \tag{11}$$

Expressing $\mathbf{q}$ based on (7), we encounter the subsequent variational inequality:

$$\left((2\rho g)^{p-1}\left[\int_b^z A(s)(h-s)^{p-1}ds\right]|\nabla h|^{p-2}\nabla h - (h-b)U_b\right)\cdot\nabla(v-h)$$
$$\geq \int_\Omega a(v-h). \tag{12}$$

An essential observation to make is that $\mathbf{q}$ displays a degenerate behavior at the free boundary. To navigate the gradient degeneracy proximate to the free boundary, a transformation was introduced in [27]:

$$H = u^{(p-1)/(2p)}. \tag{13}$$

This results in a transformed version of Equation (7):

$$-\nabla\cdot\big(\mu(x,u)|\nabla u - \Phi(x,u)|^{p-2}\nabla u - (h-b)U_b\big) = a, \tag{14}$$

where $\mu(x,u)$ and $\Phi(x,u)$ are specifically defined functions. Given this transformation, further study would reveal the implications and applications of the aforementioned equations.

In the present work, we focus on simplifying the nonlinear dynamics by making specific assumptions to address a more tractable problem. Specifically, we consider the scenario where $U_b = 0$. Under this assumption, we set $\Psi(x,u) = 0$ and treat $\mu(x,u)$ as a constant, denoted μ_0. By adopting these assumptions, the inequality expressed in (12) simplifies to the following:

$$\int_\Omega \big(\mu_0|\nabla u - \Phi(x,u)|^{p-2}(\nabla u - \Phi(x,u))\big)\cdot\nabla(v-u) \geq \int_\Omega a(v-u). \tag{15}$$

This inequality holds for all $v \geq 0$ defined over Ω and $p \geq 2$. For simplification, we take $\mu_0 = 1$.

In the seminal work by the original authors, several inherent nonlinearities were identified but not thoroughly explored. Recognizing the significance and implications of these nonlinearities, we have taken steps in this paper to address and analyze them more holistically. Specifically, we have chosen to focus on two primary nonlinearities that were presented but not fully addressed in the original study [1]:

(1) A p-Laplace-type nonlinearity arising from Glens flow law. Notably, while the original authors noted the simplification of this nonlinearity when $p = 2$, resulting in $|\nabla u - \Phi(x, u)|^{p-2} = 1$, our investigation delves deeper into cases where $p > 2$, thereby introducing and emphasizing the complexities of this nonlinearity.
(2) An intricate nonlinearity in $\Phi(x, u)$ due to the bedrock gradient, ∇b. In our research, we consider scenarios where the bedrock elevation isn't flat, which introduces another layer of nonlinearity. This is in contrast to the original paper where the implications of a non-flat bedrock weren't fully explored, resulting in $\Phi(x, u) = 0$.

By focusing on these areas, our aim is to provide a more comprehensive understanding of the problem, shedding light on aspects that were previously left in the shadows. As we delve deeper into our discourse, our approach becomes particularly evident. We will explicitly address these nonlinearities, offering insights and solutions that not only build upon but also augment and enhance the foundation set by the original author, especially as we transition into the energy minimization formulation.

3. Energy Minimization Formulation

Consider a Hilbert space V such that $V \in H^1(\Omega)$. Let the subspace U be defined by

$$U = \left\{ v \in V : v\big|_{\partial\Omega} = 0 \right\}.$$

Furthermore, we introduce the constraint set K as

$$K = \left\{ v \in V : v \geq b \text{ in } \Omega \text{ and } v\big|_{\partial\Omega} = 0 \right\}.$$

Given these definitions, our variational inequality problem, as specified in (15), can be equivalently posed as an energy minimization problem:

$$\text{Find } u \in K \text{ such that } J(u) \leq J(v) \text{ for all } v \in K, \tag{16}$$

with the energy functional J given by

$$J(v) = B(v, v) - \langle a, v \rangle,$$

where

$$B(v, v) = \frac{1}{p} \int_\Omega |\nabla v - \Psi|^p \, dx.$$

We introduce a regularization term to the energy functional and define the augmented energy functional as

$$L(v) = J(v) + \alpha \int_\Omega [b(x) - v(x)]_+^2 \, dx, \tag{17}$$

where α is a positive constant and

$$[b(x) - v(x)]_+ = \max\{0, b(x) - v(x)\}.$$

Based on the findings from [28], the unique solution u of problem (17) approaches the solution of (16) as α grows significantly large. This leads us to the following minimization problem:

$$\min_{v \in V} L(v), \tag{18}$$

with the energy functional given by

$$L(v) = \frac{1}{p} \int_\Omega |\nabla v - \Psi|^p \, dx - \int_\Omega av \, dx + \alpha \int_\Omega [b - v]_+^2 \, dx, \tag{19}$$

where α is again a positive constant.

In numerical computations, this continuous energy functional is approximated by its discrete counterpart:

$$\bar{L}(v) = \frac{|\Omega|}{N} \sum_{i=1}^{N} \left[\frac{1}{p} \|\nabla v(X_i) - \Psi(X_i)\|_2^p - a(X_i)v(X_i) + \alpha[b(X_i) - v(X_i)]_+^2 \right], \tag{20}$$

where $\{X_k\}_{k=1}^N$ are i.i.d random variables sampled from the uniform distribution $U(\Omega)$.

If the solution on the boundary is non-zero, the minimization problem can be generalized to:

$$\min_{v \in V} L(v), \tag{21}$$

with the modified energy functional:

$$L(v) = \frac{1}{p} \int_\Omega |\nabla v - \Psi|^p \, dx - \int_\Omega av \, dx + \alpha \int_\Omega [b - v]_+^2 \, dx + \beta \int_{\partial\Omega} (v - h)^2, \tag{22}$$

where h denotes the solution on the boundary.

With the energy functional clearly defined, the next step is to seek efficient computational or approximation methods for solutions. Here, Deep Neural Networks (DNNs) emerge as an especially promising tool. Their ability to capture intricate nonlinear relationships and represent functions make them apt for approximating our energy functional. In the following section, we delve into how we can leverage these networks for our purpose.

4. Solution Approximation Using Deep Neural Networks

We utilize a fully connected feedforward neural network, denoted as $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^{N_D}$, to approximate our solution. This neural network comprises multiple layers, with each layer introducing nonlinearity through an activation function. This ensures that the network can function as a universal approximator, capturing the complex relationships inherent to this problem. The weights and biases in the network are iteratively refined using backpropagation based on the minimization of a loss function associated with the energy minimization formulation above.

To measure the accuracy and efficacy of our deep neural network approximation, we employ a composite loss function derived from three primary components:

- **Residual Loss ($loss_1$)**:

$$loss_1 = \frac{1}{N} \sum_{i=1}^{N} \left[\frac{1}{p} \|\nabla u^*(X_i) - \Psi(X_i)\|^p - a(X_i)u^*(X_i) \right]. \tag{23}$$

  This term captures the degree to which the current state of the solution fails to satisfy the PDE. Minimizing this is equivalent to convergence to a solution of the PDE.

- **Obstacle Loss ($loss_2$)**:

$$loss_2 = \frac{1}{N} \sum_{i=1}^{N} [b(X_i) - u^*(X_i)]_+^2 . \tag{24}$$

  This term penalizes the discrepancy between the network's output and the obstacle function, b. This component helps ensure the resulting solution lies above the constraint.

- **Boundary Condition Loss ($loss_3$)**:

$$loss_3 = \frac{1}{M}\sum_{j=1}^{M}\left[u^*(Y_j) - h(Y_j)\right]^2. \tag{25}$$

This component ensures our solution adheres to known values on the boundary of our domain, effectively representing the boundary condition loss.

The total loss function is then a weighted combination of the aforementioned losses:

$$L(\theta) = loss_1 + \alpha \cdot loss_2 + \beta \cdot loss_3. \tag{26}$$

The weighting coefficients α and β balance the importance of loss components for PDE satisfaction, constraint satisfaction, and boundary condition satisfaction respectively. Their optimal values, attained through iterative trial and error, ensure a balanced representation of all problem constraints in the loss function. Our training uses a batch approach, where data subsets iteratively refine the network parameters. With this foundational understanding, we now explore the neural network's practical applications in subsequent sections, showcasing its versatility and efficacy.

5. Numerical Illustrations: 1D and 2D Examples

Before describing the specific case studies, we outline the sequence of numerical experiments we will perform. Initially, we will use the "method of manufactured solutions" (MMS) to test this approach. This technique, commonly used in computational science, involves creating an exact solution (the "manufactured solution") for an augmented form of partial differential equation. This is done by choosing a solution function (that satisfies obstacle and boundary constraints), computing the residual arising from that function not satisfying the PDE, and augmenting that PDE with an inhomogeneity that offsets that residual exactly. The manufactured solution is then a solution to the inhomogeneity augmented PDE. The MMS allows for a comprehensive assessment of the numerical method in question as it provides a "ground truth" against which numerical approximations can be rigorously compared. By applying MMS in our preliminary experiments, we aim to validate the robustness and accuracy of this deep learning approach.

This problem is comprised of both a complex PDE and an obstacle constraint. We will thus first apply this approach to a simpler PDE with an obstacle constratint followed by the more complex PDE of interest with the constraint. This will be performed on both 1D and 2D domains. Following this we will test this approach on the motivating problem of trying to find the elevation profile of ice above the bedrock of Greenland.

For the solution approximator, we adopt an architecture comprising a minimum of 5 hidden layers, each equipped with 128 neurons. We utilized the squared Rectified Linear Unit ($ReLU^2$) in our study to leverage its improved differentiability and empirical performance, particularly enhancing the accuracy and convergence in solving complex Partial Differential Equations (PDEs).Furthermore, to foster a consistent learning environment, layer normalization techniques are integrated into the model. The optimization phase employs the Adam variant of the stochastic gradient descent (SGD) method. The learning rate is initially set to 5×10^{-4} for the first 500 iterations. After which, it reduces by half from the 500 to 750 iterations, and then further reduces by half for the remaining iterations. This learning

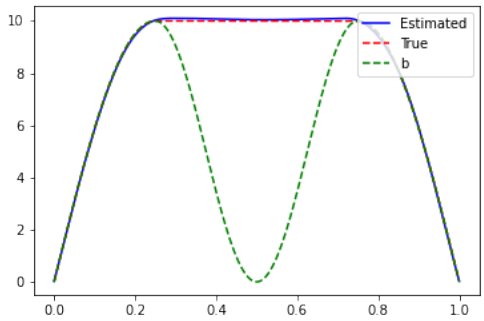

FIGURE 2. Comparison of the 1D solution, obstacle function (denoted b), and the exact solution.

rate schedule was determined through trial and error. The primary objective remains the minimization of the associated loss function. For the implementation and training processes, we relied on the capabilities offered by the Pytorch library [25]. We note that there are many variations on this network and training structure that could be investigated. Here we are mainly investigating whether this approach is feasible and refinement is left for future study.

5.1. 1D Example: Case with $p = 2$. To initiate our numerical exploration, we consider a one-dimensional problem with $p = 2$. For this instance, we use for the following obstacle function:

$$b(x) = \begin{cases} 10\sin(2\pi x), & \text{for } 0 \le x \le 0.25, \\ 5\cos(\pi(4x-1)) + 5, & \text{for } 0.25 \le x \le 0.75, \\ 10\sin(2\pi(1-x)), & \text{for } 0.75 \le x \le 1.0. \end{cases} \tag{27}$$

The exact solution is characterized as:

$$u_{\text{exact}}(x) = \begin{cases} 10\sin(2\pi x), & \text{for } 0 \le x \le 0.25, \\ 10, & \text{for } 0.25 \le x \le 0.75, \\ 10\sin(2\pi(1-x)), & \text{for } 0.75 \le x \le 1.0. \end{cases} \tag{28}$$

Refer to Figure 2 for the corresponding visualization. To train a solution approximator for this problem, we follow the approach outlined above with appropriate modifications for the problems specifics. The points $\{X_k\}_{k=1}^N$ for evaluation are drawn from a uniform distribution over the interval $[0, 1]$.

In our experimental setup, the neural network undergoes training over 2000 iterations with the regularization parameter α and β fixed at 4000. We then assess the performance of our methodology on a mesh grid, uniformly spaced and marked by a fine resolution of 10^{-3}. The resultant numerical solution is presented in Figure 2, where it is juxtaposed against both the obstacle function and the exact solution. Delving deeper into the training dynamics, Figure 3 charts the evolution of various loss metrics, including the total loss, Loss 1, Loss 2 and Loss 3. A notable observation from our experiments is the efficiency of the proposed technique; it typically converges within a mere 500 iterations, underscoring its robustness and efficacy.

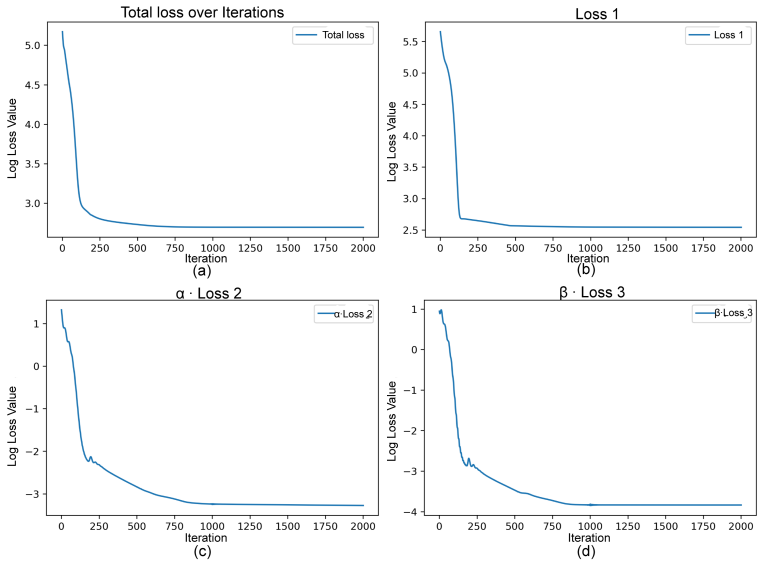

FIGURE 3. Evolution of different losses during training, plotted against the number of iterations on a logarithmic scale (with base 10). (a) *Total Loss*, represented as "Log Loss Value" on the y-axis and computed from Equation (26), symbolizes the amalgamation of all individual loss terms. (b) *Loss 1*, denoted as "Log Loss Value" on the y-axis from Equation (23), illustrates the residual loss term. (c) *Loss 2*, weighted by coefficient α and articulated as "Log Loss Value" on the y-axis in Equation (24), encapsulates the obstacle loss term. (d) *Loss 3*, accentuated by coefficient β and reflected as "Log Loss Value" on the y-axis from Equation (25), highlights the boundary loss inherent in the problem.

In our analysis, the difference between the exact and approximated solutions is quantified using the L^1 norm, defined as:

$$\text{loss}_{\text{exact}} = \frac{1}{N}\sum_{i=1}^{N} |U_{\text{inner\_pred},i} - U_{\text{exact\_inner},i}|,$$

where N denotes the total number of grid points or samples, and $U_{\text{inner\_pred},i}$ and $U_{\text{exact\_inner},i}$ represent the approximated and exact solutions, respectively, at the i-th sample. A plot of this loss versus the iterations provides insight into the convergence behavior and accuracy of the neural network's predictions. These results demonstrate that this approach learns a neural network approximator that accurately learns to solve the PDE subject to the given constraints.

5.1.1. Analysis of Relative Error in the Context of Sample Variability. We next analyze the accuracy of this approach as a function of the number of sampled points at which the PDE loss is computed. This analysis provides insights into how the sampling granularity can influence the accuracy of the derived solution. These sample points are systematically chosen from a uniform grid, ensuring consistent and unbiased assessment. Figure 5 portrays the relationship between sample size and the resultant relative error. In our analysis regarding the number of sam-

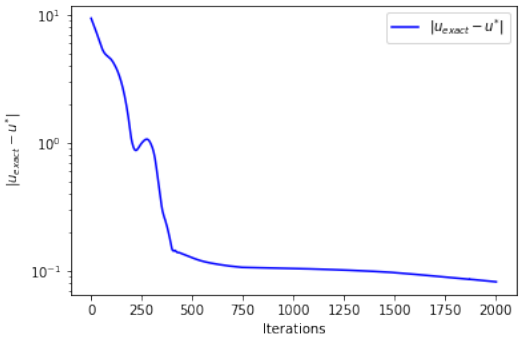

FIGURE 4. Evolution of the L^1 error between the approximated solution u^* and the exact solution u_{exact} over training iterations. The plot showcases the difference in magnitudes as training progresses, shedding light on the convergence and accuracy of the neural network's predictions.

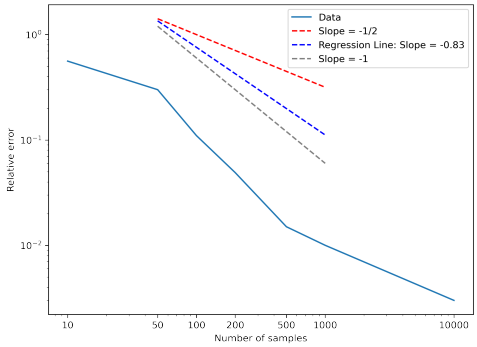

FIGURE 5. Graphical representation of relative error against varying sample sizes. The blue dashed line represents the regression line with a slope of -0.83, which lies between the theoretical slopes of $-1/2$ (red) and -1 (gray).

ples, Figure 5 highlights the reduction in relative error as sample count augments. To better visualize the expected linear scaling of error with the number of points, a line with a -1 slope was added to the plot. This represents the ideal trend where, if the error scales as $\frac{1}{N}$, then log(error) directly corresponds to $-\log(N)$.

To further analyze the error scaling, we performed a linear regression on the log-transformed data points, resulting in a slope of approximately -0.83. This regression line, shown in the updated plot, lies between the theoretical slopes of -1 and $-1/2$. Therefore, the actual slope from our data indicates that the error reduction behavior in our analysis is intermediate between these two theoretical expectations.

In the analysis (Figure 5) illustrates the reduction in relative error as the computational grid is refined. Common computational PDE approaches (finite difference, finite element, etc.) produce errors that scale as $1/N$ [29] while Monte Carlo approaches commonly lead to errors that scale as $1/sqrt(N)$ [30]. To visualize the

TABLE 1. Relative errors corresponding to a spectrum of regularization parameters.

Parameter (α)	**Parameter** (β)	**Relative Error**
100	100	0.40
500	100	0.015
1000	500	0.0035
4000	4000	0.0010
5000	4000	0.0023

scaling relationship between error and number of points (N) Figure 5 is plotted on log-log axes and a regression line is fit to the error magnitude to compute the scaling slope. Results indicate that errors decay faster than $1/sqrt(N)$ but more slowly than $1/N$ for the range used.

In our comprehensive analysis, we not only explored the impact of sampling but also assessed the influence of varying regularization parameters and loss weighting parameters α and β on relative error. Understanding these dynamics is vital, as they shed light on the model's sensitivity to its central hyperparameters, ensuring that it captures underlying patterns effectively without the pitfalls of overfitting or underfitting.

Table 1 illustrates the relative errors for different values of the regularization parameter. These results illustrates that a proper balance between the different loss components is needed to ensure optimal training.

5.2. 2D Problem: General Case with Any $p \geq 2$. We now consider the 2D problem with $p \geq 2$. For this analysis, we use the collocation points that are uniformly sampled from the unit square. For simplicity and to facilitate the use of the MMS approach, we use a circularly symmetric obstacle function:

$$b(r) = \begin{cases} 1 - \left(r^{\frac{p}{p-1}} - (1-r)^{\frac{p}{p-1}} + 1 - \frac{p}{p-1} r \right) & \text{for } r \leq r^* \\ 0 & \text{for } r > r^* \end{cases}, \tag{29}$$

where r represents the radial distance in our 2D space and $r^* = 0.75$ is a threshold value defining the boundary of the obstacle's influence.

We construct an exact solution within this domain $\Omega = [0,1] \times [0,1]$ given by:

$$u_{exact}(r) = \begin{cases} 1 - (F(r) - G(r) + 1 - E(r)) & \text{for } r \leq r^* \\ -\left(\frac{p}{p-1}\right) r^{*\frac{1}{p-1}} + (1-r^*)^{(1/(p-1))-1}(r - r^*) & \\ +1 - r^{*(p/(p-1))} - (1-r^*)^{(p/(p-1))} + 1 - E(r^*) & \text{for } r > r^* \end{cases}, \tag{30}$$

where

$$F(r) = r^{\frac{p}{p-1}}, \tag{31}$$

$$G(r) = (1-r)^{\frac{p}{p-1}}, \tag{32}$$

$$E(r) = \frac{p}{p-1} r. \tag{33}$$

u_{exact} is continuous and differentiable (i.e., C^1). Moreover, the value of u_{exact} can be freely chosen for $r > r^*$ without violating its C^1 smoothness. The obstacle function, exact solution, and the neural network's approximation are shown for $p = 3$ (Figure 6):

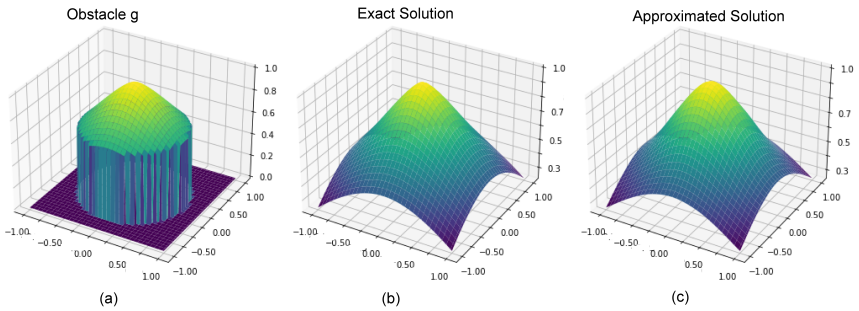

FIGURE 6. (a) Obstacle function plot, (b) Exact solution plot, and (c) Approximated solution plot.

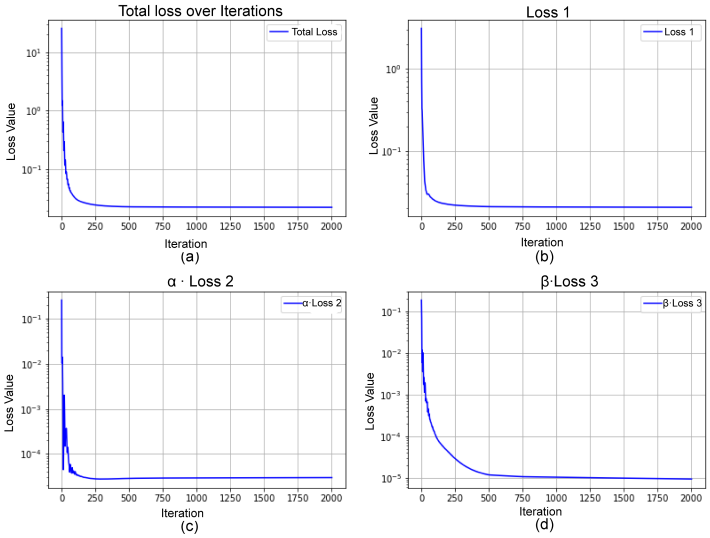

FIGURE 7. Evolution of different losses during training plotted on a log scale versus the number of iterations. (a) *Total Loss*, as described in Equation (26). (b) *Loss 1* from Equation (23). (c) *Loss 2* influenced by coefficient α, from Equation (24). (d) *Loss 3*, affected by coefficient β, as per Equation (25).

In this study, we set weighting parameters α and β both to 100 and train the network for 2000 iterations. All losses are shown on a logarithmic scale against the number of iterations to illustrate the efficiency of the training process (Figure 7). The L^1 norm of the difference between the approximated and true solutions as a function of training iteration is further shown in Figure 8.

For the case $p = 4$, Building upon our previous methodologies, we present the results for this scenario. Visual representations of the obstacle function, exact solution, and the neural network's approximation for $p = 4$ are shown in Figure 9.

Post-training, we juxtapose the exact solution and the neural network's approximation for a more comprehensive visual comparison for $p = 4$ (Figure 9). Similar to the previous cases, we showcase the loss behavior during training for $p = 4$ in

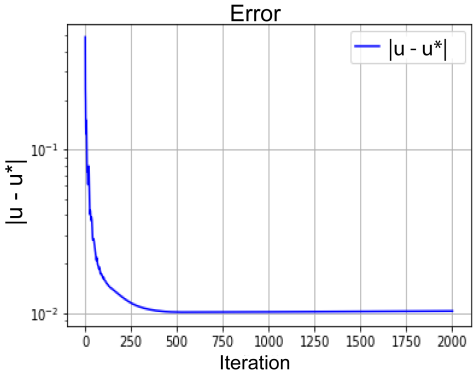

FIGURE 8. Evolution of the L^1 error for $p = 3$ between the approximated solution u^* and the exact solution u over training iterations. As with the previous case, this plot gives insight into the neural network's convergence behavior and prediction accuracy for this parameter setting.

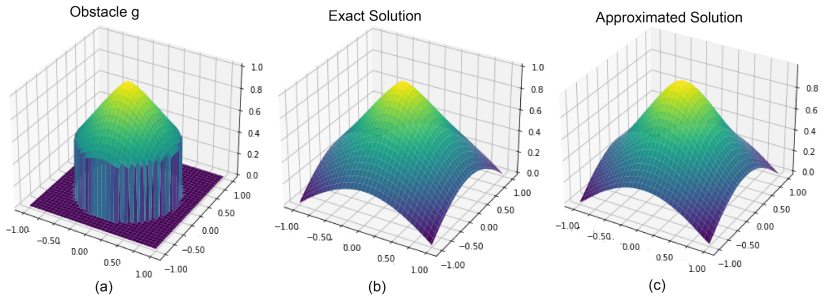

FIGURE 9. (a) Obstacle function plot, (b) Exact solution plot, and (c) Approximated solution plot.

the Figure 10. To assess the efficiency of the neural network's predictions for $p = 4$, we compute the L^1 error (Figure 11):

These results demonstrate this approach can well approximate the solution of the obstacle constrained ice sheet PDE in simplified 1D and 2D domains. We next apply this approach to solve this PDE with a realistic obstacle geometry: the Greenland bedrock elevation taken from real-world data. The challenges and nuances of this dataset offer a rigorous testbed to evaluate the adaptability and robustness of our proposed algorithms.

6. Application to Greenland Data

In this section, we apply the techniques developed earlier to a dataset derived from Greenland. This dataset, available from the National Snow and Ice Data Center (NSIDC), offers a comprehensive view of various parameters related to Greenland's ice melt and movement. Specifically, we utilize data from the [NSIDC-0092 dataset [22]](`https://nsidc.org/data/nsidc-0092/versions/1`), which captures detailed information on ice-thickness. The NSIDC-0092 dataset

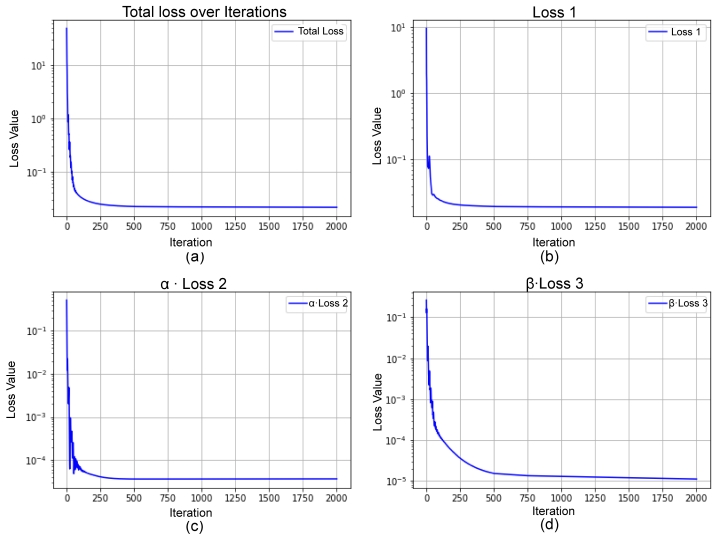

FIGURE 10. Evolution of different losses during training plotted on a log scale versus the number of iterations. (a) *Total Loss*, as described in Equation (26). (b) *Loss 1* from Equation (23). (c) *Loss 2* influenced by coefficient α, from Equation (24). (d) *Loss 3*, affected by coefficient β, as per Equation (25).

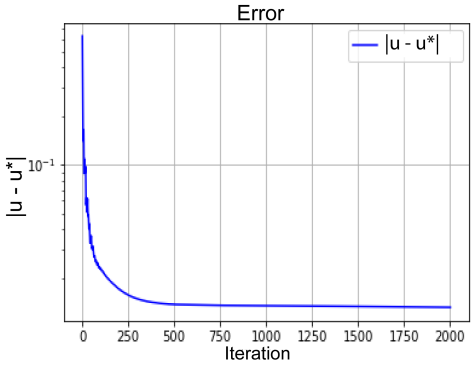

FIGURE 11. Evolution of the L^1 error for $p = 4$ between the approximated solution u^* and the exact solution u_{exact} over iterations, highlighting the convergence and prediction accuracy.

offers detailed insights into Greenland's topography, presenting Digital Elevation Models (DEMs), ice thickness, and bedrock elevation data. The parameter range is extensive: DEM values range from -0.1 m to 3278.3 m, ice thickness measurements extend from 0 m to 3366.5 m, and bedrock elevation data vary between -963.1 m and 3239.0. These data are organized into ASCII text grids with a spatial resolution of 5 km, comprising 301 columns by 561 rows. This grid resolution ensures that the dataset provides a comprehensive and granular view of the ice sheet and bedrock topography across Greenland, suitable for precise climatological and glaciological

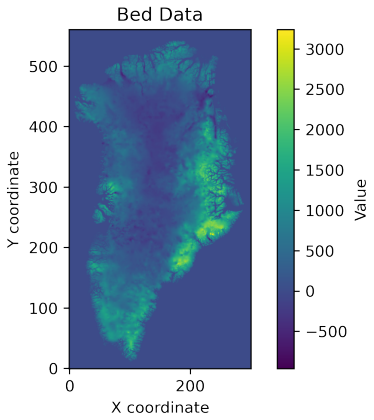

FIGURE 12. Bedrock topography of Greenland derived from the NSIDC-0092 dataset. This representation serves as our obstacle function for the subsequent analysis.

analyses. The interpolation to a 5 km grid underscores the dataset's utility in modeling and analyzing the dynamics of Greenland's ice sheet and underlying bedrock with considerable detail.

To provide context, we begin by visualizing the bedrock topography from the dataset (Figure 12):

6.1. Neural Network Initialization and Training. Translating this method to the bedrock obstacle data from the Greenland's ice-thickness data introduces a specific challenge. Notably, when employing conventional random initialization, the neural network behaved erratically, at least in part because it failed to satisfy the obstacle, leading to numerical issues. The resulting solutions were not only characterized by high error rates but were also noticeably distant from the anticipated results. Such deviations and unpredictable behavior underscored the need for an alternative approach to initialize the neural network.

To overcome this, we took a two-stage modeling approach. In the first, we intelligently initialize the NN. In the second, we train that initialized network to approximate the PDE (the same method as previously).

To initialize the network, we trained it to mimic the bedrock elevation as illustrated in Figure 12. This preliminary training phase enabled the neural network to learn and approximate the patterns of the bedrock topography. This is not a solution to the problem, only a method of initializing the network.

Once this stage achieved satisfactory convergence, we retained the optimized weights of the network. These pre-trained weights were then employed as initialization parameters. By harnessing the pre-trained weights, the neural network exhibited a more efficient learning process and showed considerable improvements in error reduction. To summarize, we pre-trained the network to mimic a reasonable starting function, then used a second phase of training (same as previously discussed) to construct a solution to the PDE.

For the training process, we set the penalty parameters as $\alpha = \beta = 4000$. This was chosen by trail and error to ensure all weighted components of the loss are the same order of magnitude. For the solution approximation, we adopt an architecture comprising 15 hidden layers, each equipped with 320 neurons. The neural network

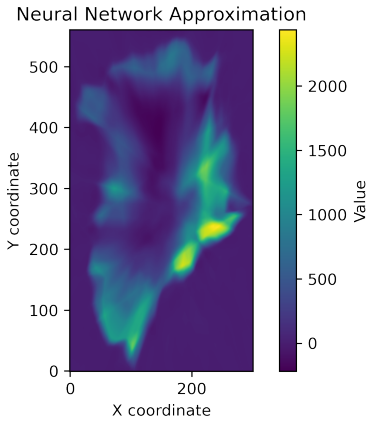

FIGURE 13. Visualization of the neural network after pre-training on the bedrock elevation. This pre-training produces an initialization for the network that is subsequently used as a starting point for training.

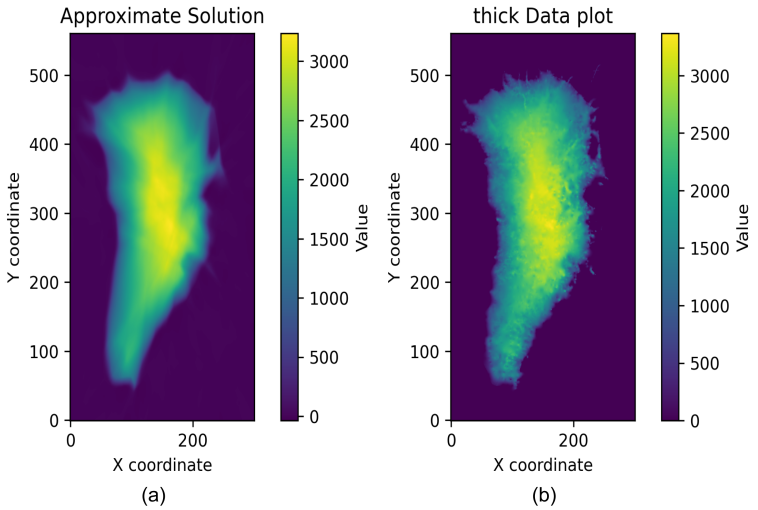

FIGURE 14. Comparison over Greenland: (a) Neural network-derived approximation for $p = 4$ and (b) precise data from the NSIDC-0092 dataset.

underwent training for a total of 22000 iterations to ensure accurate convergence and approximation of the Greenland ice-thickness data (Figure 15).

Following the training process and the application of our methodology, we obtained the approximated solution. For a clear comparison, we juxtapose this solution alongside the exact ice thickness data derived from the NSIDC-0092 dataset (Figure 14).

6.2. Analysis of Training Losses. Determining how well the approximated solution does at satisfying the PDE is challenging for two reasons: 1) there is no exact

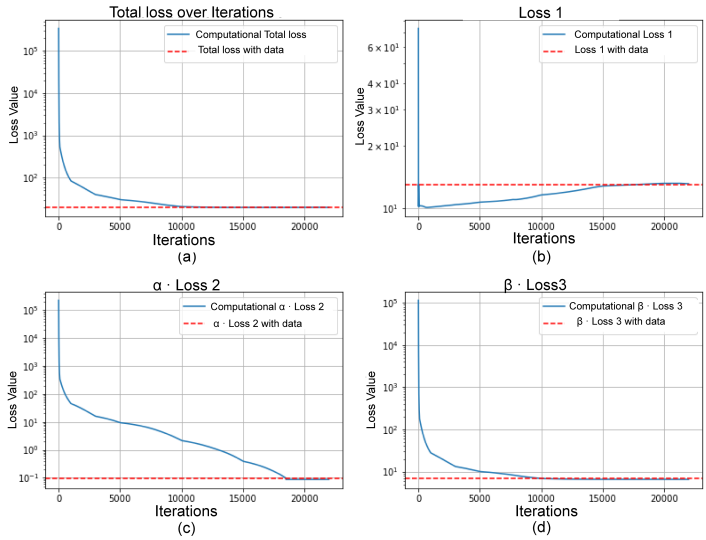

FIGURE 15. Comparison of various loss components for $p = 3$. Each plot showcases the loss trajectory over iterations, with the red dashed line representing the loss upon data input, and the blue solid line symbolizing computational loss.

solution to compare against and 2) the magnitude of training losses are not interpretable in absolute terms. To circumvent this issue, we compare the properties of the neural network solution to this problem with the ice-sheet thickness data. Note however that the thickness data is itself not necessarily a solution to the PDE, a research question that is beyond the scope of this article. Thus we will not directly compare the solution to the data. Instead, to get a benchmark for comparison, we compute the losses that would be found from inserting the measured ice sheet data into the three loss terms. These form the dashed lines against which training losses are compared in Figures 15 and 16. These dashed lines provide a scale reference to compare the model training losses against.

Figure 15 shows the trajectory of the losses as a function of training iteration. Results show that all losses converge to an absolute level that is consistent with the real ice sheet thickness data's satisfaction of the PDE. This, in combination with the visual comparison of the approximated solution to the real data suggests this approach well approximates the solution to this PDE. Similar analysis was carried out for $p = 4$ with similar results (Figures 16).

7. Summary

This paper embarked on an exploration of the integration between the realm of mathematical modeling and deep learning to address intricate problems related to the dynamics of Greenland's ice sheet. We first established a robust theoretical framework, illustrating the underpinnings of the variational inequalities and obstacle problems. Adopting a novel approach, we leveraged the power of neural networks to seek solutions, demonstrating both its theoretical and practical applications.

A key highlight of our methodology was the integration of pre-training. By first acquainting the neural network with a representation of the bedrock topography, we ensured a more efficient learning process when the actual ice-thickness data was

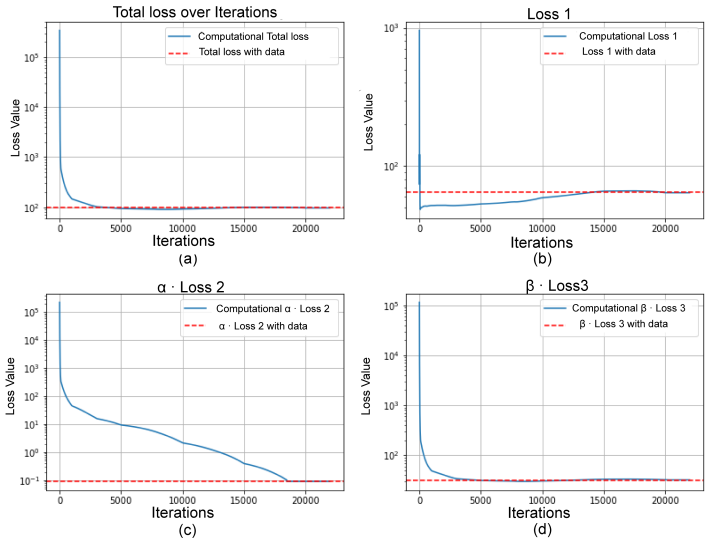

FIGURE 16. Comparison of various loss components for $p = 4$. Each plot showcases the loss trajectory over iterations, with the red dashed line representing the loss upon data input, and the blue solid line symbolizing computational loss.

introduced. This strategic move not only optimized our results but also underscores the importance of judicious model initialization in complex problem-solving.

When applied to the NSIDC-0092 dataset, our method displayed remarkable accuracy. The approximated solutions closely mirrored the exact data, which stands as a testament to the promise and potential of this interdisciplinary approach.

Furthermore, this study illuminates the broader implications for climate science. As the melting of Greenland's ice sheet plays a pivotal role in global climate dynamics, having precise and efficient computational methods becomes paramount. The methodologies explored here can be adapted and extended to other similar domains, making it a versatile tool in the scientific community's arsenal.

In this paper, we've shown how deep learning can work together with mathematical models to study Greenland's ice sheet dynamics. As we collect more data and face more complex challenges in the future, using both these tools together will help us find better solutions. This combination is promising for future research and problem-solving.

Acknowledgments

The authors acknowledge useful discussions on the mathematical background with Professor Ed. Bueler and Paolo Piersanti. This work was supported in part by the Research Fund of Indiana University.

References

[1] G. Jouvet and E.D. Bueler, Steady, shallow ice sheets as obstacle problems: Well-posedness and finite element approximation, SIAM J. Appl. Math., 72 (2012), 1292–1314.
[2] P. Piersanti and R. Temam, On the dynamics of grounded shallow ice sheets: Modeling and analysis, Adv. Nonlinear Anal., 12 (2023), Article 20220280.
[3] J.F. Rodrigues, Obstacle Problems in Mathematical Physics, Elsevier, 1987.

[4] I. Harari and T.J.R. Hughes, Galerkin/least-squares finite element methods for the reduced wave equation with non-reflecting boundary conditions in unbounded domains, Comput. Methods Appl. Mech. Engrg., 98 (1992), 411–454.
[5] K. Xia and H. Yao, A Galerkin/least-square finite element formulation for nearly incompressible elasticity/Stokes flow, Appl. Math. Model., 31 (2007), 513–529.
[6] E. Burman, P. Hansbo, M.G. Larson, and R. Stenberg, Galerkin least squares finite element method for the obstacle problem, Comput. Methods Appl. Mech. Engrg., 313 (2017), 362–374.
[7] R. Kornhuber, Monotone multigrid methods for elliptic variational inequalities II, Numer. Math., 72 (1996), 481–499.
[8] R. Kornhuber, Monotone multigrid methods for elliptic variational inequalities I, Numer. Math., 69 (1994), 167–184.
[9] D.M. Yuan and X.L. Cheng, An iterative algorithm based on the piecewise linear system for solving bilateral obstacle problems, Int. J. Comput. Math., 89 (2012), 2374–2384.
[10] T. Fhrer, First-order least-squares method for the obstacle problem, Numerische Mathematik, 144 (2020), 55–88.
[11] K. Majava and X.-C. Tai, A level set method for solving free boundary problems associated with obstacles, Int. J. Numer. Anal. Model., 1 (2004), 157–172.
[12] Q. Cheng, X.L. Cheng, and S. Abide, A dynamical method for solving the obstacle problem, Numer. Math. Theory Meth. Appl., 13 (2020), 353–371.
[13] S.H. Rudy, J.N. Kutz, and S.L. Brunton, Deep learning of dynamics and signal-noise decomposition with time-stepping constraints, J. Comput. Phys., 396 (2019), 483–506.
[14] M. Raissi, P. Perdikaris, and G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys., 378 (2019), 686–707.
[15] W. E and B. Yu, The Deep Ritz Method: A deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat., 6 (2018), 1–12.
[16] N. Winovich, K. Ramani, and G. Lin, ConvPDE-UQ: Convolutional neural networks with quantified uncertainty for heterogeneous elliptic partial differential equations on varied domains, J. Comput. Phys., 394 (2019), 263–279.
[17] R.K. Tripathy and I. Bilionis, Deep UQ: Learning deep neural network surrogate models for high dimensional uncertainty quantification, J. Comput. Phys., 375 (2018), 565–588.
[18] Y. Khoo and L. Ying, SwitchNet: A neural network model for forward and inverse scattering problems, SIAM J. Sci. Comput., 41 (2019), A3182–A3201.
[19] J. Huang, C. Wang, and H. Wang, Adaptive learning on the grids for elliptic hemivariational inequalities, arXiv:2104.04881, (2021).
[20] X. Cheng, S. Xing, X. Wang, and K. Liang, A deep neural network-based method for solving obstacle problems, Nonlinear Analysis: Real World Appl., 72 (2023), 103864.
[21] R. Scholz, Numerical solution of the obstacle problem by the penalty method, Computing, 32 (1984), 297–306.
[22] J. Bamber, Greenland 5 km DEM, ice thickness, and bedrock elevation grids, Version 1, NASA Natl. Snow and Ice Data Center Distributed Active Archive Center, (2001), DOI: 10.5067/01A10Z9BM7KP.
[23] R. Greve and H. Blatter, Dynamics of Ice Sheets and Glaciers, Springer-Verlag, 2009.
[24] G. Jouvet, J. Rappaz, E. Bueler, and H. Blatter, Existence and stability of steady-state solutions of the shallow-ice-sheet equation by an energy-minimization approach, J. Glaciol., 57 (2011), 345–354.
[25] A. Paszke et al., PyTorch: An Imperative Style, High-Performance Deep Learning Library, NeurIPS, (2019).
[26] D.L. Goldsby and D.L. Kohlstedt, Superplastic deformation of ice: Experimental observations, J. Geophys. Res., 106 (2001), 11017–11030.
[27] N. Calvo et al., On a doubly nonlinear parabolic obstacle problem modeling ice sheet dynamics, SIAM J. Appl. Math., 63 (2002), 683–707.
[28] D. Kinderlehrer and G. Stampacchia, An Introduction to Variational Inequalities and Their Applications, Classics in Appl. Math. 31, SIAM, Philadelphia, 2000.
[29] J.C. Strikwerda, Finite Difference Schemes and Partial Differential Equations, Society for Industrial and Applied Mathematics (SIAM), 2nd ed., 2004.
[30] P. Glasserman, Monte Carlo Methods in Financial Engineering, Springer Science & Business Media, 2004.

Department of Mathematics and the Institute for Scientific Computing and Applied Mathematics, Indiana University, Indiana 47405, USA

E-mail: kapchaw@iu.edu

URL: https://math.indiana.edu/about/graduate-students/Chawla-Kapil.html

Department of Mathematics and Cognitive Science Program, Indiana University, Indiana 47405, USA

E-mail: wrholmes@iu.edu

URL: https://math.indiana.edu/about/faculty/holmes-william.html

Department of Mathematics and the Institute for Scientific Computing and Applied Mathematics, Indiana University, Indiana 47405, USA

E-mail: temam@iu.edu

URL: https://math.indiana.edu/about/faculty/temam-roger.html