The Journal of Open Source Software

ODINN.jl: Scientific machine learning glacier modelling

Jordi Bolibar[1,2¶], **Facundo Sapienza**[3,4], **Alban Gossard**[1], **Mathieu le Séac'h**[1], **Vivek Gajadhar**[2], **Fabien Maussion**[5,6], **Bert Wouters**[2], and **Fernando Pérez**[4]

1 Univ. Grenoble Alpes, CNRS, IRD, G-INP, Institut des Géosciences de l'Environnement, Grenoble, Franc **2** Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, The Netherlands **3** Department of Geophysics, Stanford University, Stanford, United States **4** Department of Statistics, University of California, Berkeley, United States **5** Bristol Glaciology Centre, School of Geographical Sciences, University of Bristol, Bristol, UK **6** Department of Atmospheric and Cryospheric Sciences, University of Innsbruck, Innsbruck, Austria ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

`ODINN.jl` is a glacier model leveraging scientific machine learning (SciML) methods to perform forward and inverse simulations of large-scale glacier evolution. It can simulate both surface mass balance and ice flow dynamics through a modular architecture which enables the user to easily modify model components. For this, `ODINN.jl` is in fact an ecosystem composed of multiple packages, each one handling a specific task:

- `Sleipnir.jl`: Handles all the basic types, functions and datasets, common through the whole ecosystem, as well as data management tasks.
- `Muninn.jl`: Handles surface mass balance processes, via different types of models.
- `Huginn.jl`: Handles ice flow dynamics, by solving the ice flow partial differential equations (PDEs) using numerical methods. It can accommodate multiple types of ice flow models.
- `ODINN.jl`: Acts as the interface to the whole ecosystem, and provides the necessary tools to differentiate and optimize any model component. It can be seen as the SciML layer, enabling different types of inverse methods, using hybrid models combining differential equations with data-driven models.

The ODINN ecosystem extends beyond this suite of Julia (Bezanson et al., 2017) packages, by leveraging the data preprocessing tools of the Open Global Glacier Model (OGGM, Maussion et al. (2019)). We do so via an auxiliary Python library named `Gungnir`, which is responsible for downloading all the necessary data to force and initialize the model, such as glacier outlines from the Randolph Glacier Inventory (RGI Consortium (2023), RGI), digital elevation models (DEMs), ice thickness observations from GlaThiDa (Consortium, 2020), ice surface velocities from different studies (Millan et al., 2022), and many different sources of climate reanalyses and projections (Eyring et al., 2016; Lange, 2019). This implies that `ODINN.jl`, like OGGM, is virtually capable of simulating any of the ~274,000 glaciers on Earth (RGI Consortium, 2023).

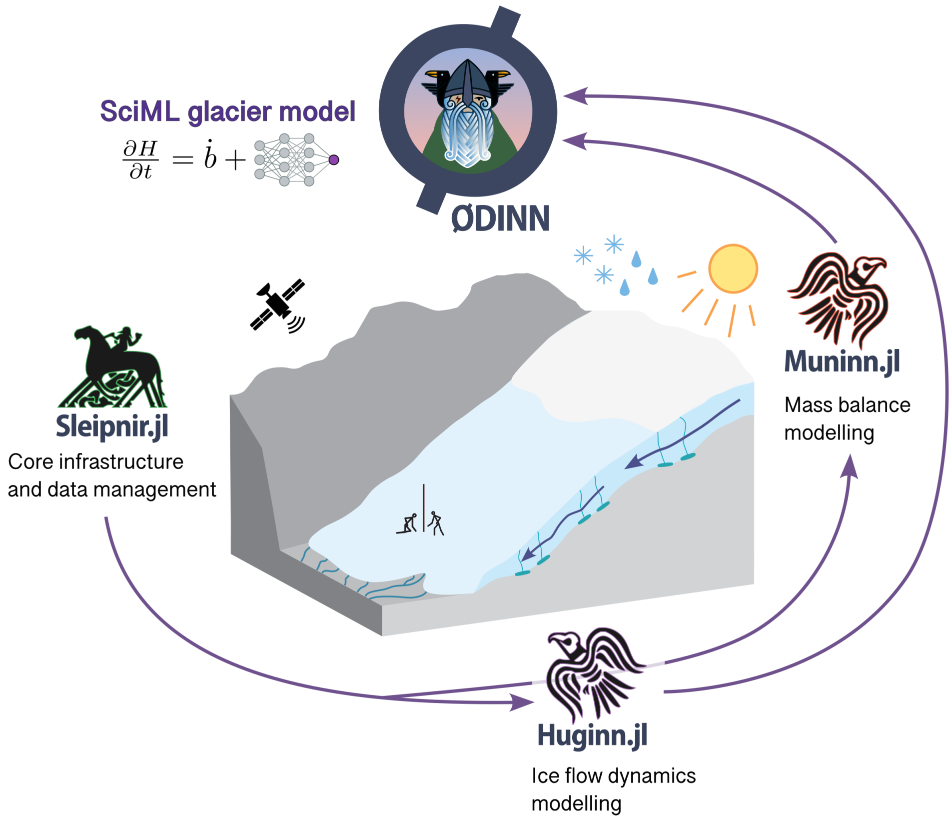

Figure 1: Figure 1: Overview of the ODINN.jl ecosystem.

`ODINN.jl` provides a high-level user-friendly interface, enabling the user to swap and replace most elements of a glacier simulation in a very modular fashion. The main elements of a simulation, such as the `Parameters`, a `Model` and a `Simulation` (either a `Prediction` or an `Inversion`), are all objects that can be easily modified and combined. In a few lines of code, the user can automatically retrieve all necessary information for most glaciers on Earth, compose a `Model` based on a specific combination of surface mass balance and ice flow models, and incorporate data-driven models (e.g. a neural network) to parametrize specific physical processes of any of these components. Both forward and inverse simulations run in parallel using multiprocessing, leveraging Julia's speed and performance. GPU compatibility is still not ready, due to the difficulties of making everything compatible with automatic differentiation (AD). Nonetheless, it is planned for future versions.

The most unique aspect of `ODINN.jl` is its differentiability and capabilities of performing all sorts of different hybrid modelling. Since the whole ecosystem is differentiable, we can optimize almost any model component, providing an extremely powerful framework to tackle many scientific problems (Bolibar et al., 2023). `ODINN.jl` can optimize, separately or together, in a steady-state or transient way:

- The initial or intermediate state of glaciers (i.e. their ice thickness) or the equivalent ice surface velocities.
- Model parameters (e.g. the Glen coefficient `A` related to ice viscocity in a 2D Shallow Ice Approximation (Hutter, 1983)), in a gridded or scalar format. This can be done for multiple time steps where observations (e.g. ice surface velocities) are available.
- The parameters of a regressor (e.g. a neural network), used to parametrize a subpart or one or more coefficients of an ice flow or surface mass balance mechanistic model.

This enables the exploration of empirical laws describing physical processes of glaciers, leveraging Universal Differential Equations (UDEs, Christopher Rackauckas et al. (2021)).

For this, it is necessary to differentiate (that is, computing gradients or derivatives) through complex code, including numerical solvers, which is a non-trivial task (Sapienza et al., 2024). We use reverse differentation based on the adjoint method to achive this. We have two strategies for computing both the adjoint and the required vector-jacobian products (VJPs): (1) manual adjoints, which have been implemented using AD via `Enzyme.jl` (Moses et al., 2021), as well as fully manual implementations of the discrete and continuous adjoints; and (2) automatic adjoints using `SciMLSensitivity.jl` (Chris Rackauckas et al., 2019), providing both continuous and discrete versions and available with different AD back-ends. These two approaches are complementary, with the manual adjoints being ideal for high-performance tasks, and serving as a ground truth for benchmarking and testing automatic adjoint methods from `SciMLSensitivity.jl`.

Beyond all these inverse modelling capabilities, `ODINN.jl` can also act as a more conventional forward glacier model, simulating glaciers in parallel, and easily customizing almost every possible detail of the simulation. Its high modularity, combined with the easy access to a vast array of datasets coming from OGGM, makes it very easy to run simulations, even with a simple laptop. `Huginn.jl` is responsible for the ice flow dynamics models, with an architecture capable of integrating and easily swapping various models. Models based on partial differential equations (PDEs) are solved using `DifferentialEquations.jl` (Christopher Rackauckas & Nie, 2017), which provides access to a huge amount of numerical solvers. For now, we have implemented a 2D Shallow Ice Approximation (SIA, Hutter (1983)), but in the future we plan to incorporate other models, such as the Shallow Shelf Approximation (SSA, Weis et al. (1999)). Validation of numerical forward simulations are evaluated in the test suite based on exact analytical solutions of the SIA equation for some simpler cases (Bueler et al., 2005). In terms of surface mass balance, `Muninn.jl` incorporates for now simple temperature-index models. Nonetheless, the main addition of the upcoming version will be the machine learning-based models from the `MassBalanceMachine` (Sjursen et al., 2025), which will become the de-facto solution. Frontal ablation (i.e. calving) and debris cover are not available for now, but we plan to add it to future versions of the model.

Statement of need

`ODINN.jl` has been designed to address the need for a glacier model which can leverage both the interpretability and established knowledge coming from the literature in the form of mechanistic models based on differential equations, with the flexibility and data-assimilation capabilities of data-driven models (Bolibar et al., 2023). The combination of these two paradigms enables a targeted approach to inverse methods for learning parametrizations of glacier physical processes, learning only the unknown physics and keeping a reliable structure in the dynamics in the form of a differential equation. While purely mechanistic and data-driven modelling approaches exist in glaciology, there is a need for flexible models which can leverage existing widely available observations at the glacier surface, to simulate complex physical processes of glaciers, such as basal sliding, creep or calving. Existing laws do not necessarily map available observations with these physical processes, difficulting the finding and calibration of parametrizations and laws. Approaches based on functional inversions and differentiable programming offer the needed flexibility to derive new empirical laws based on carefully chosen input proxies, which can help to test hypothesis of what can constitute and drive new parametrizations.

At the same time, a good representation of this complex and poorly represented physical processes is key to accurate predictions of glacier evolution, crucial for their impact to both freshwater resources and sea-level rise. Therefore, with `ODINN.jl`, we provide a unified modelling ecosystem, capable of both flexible and advance inverse methods for model calibration, as well as efficient and modular methods for forward simulations for large-scale glacier modelling.

Acknowledgements

We acknowledge the help of Chris Rackauckas for the debugging and discussion of issues related to the SciML Julia ecosystem, Redouane Lguensat for scientific discussions on the first prototype of the model, and Julien le Sommer for scientific discussions around differentiable programming. JB acknowledges financial support from the Nederlandse Organisatie voor Wetenschappelijk Onderzoek, Stichting voor de Technische Wetenschappen (Vidi grant 016.Vidi.171.063) and a TU Delft Climate Action grant. FS acknowledges funding from the National Science Foundation (EarthCube programme under awards 1928406 and 1928374). AG acknowledges funding from the MIAI cluster and Agence Nationale de la Recherche (ANR) in the context of France 2030 (grant ANR-23-IACL-0006).

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Bolibar, J., Sapienza, F., Maussion, F., Lguensat, R., Wouters, B., & Pérez, F. (2023). Universal differential equations for glacier ice flow modelling. *Geoscientific Model Development*, *16*(22), 6671–6687. https://doi.org/10.5194/gmd-16-6671-2023

Bueler, E., Lingle, C. S., Kallen-Brown, J. A., Covey, D. N., & Bowman, L. N. (2005). Exact solutions and verification of numerical models for isothermal ice sheets. *Journal of Glaciology*, *51*(173), 291–306. https://doi.org/10.3189/172756505781829449

Consortium, G. (2020). *Glacier Thickness Database 3.1.0*. World Glacier Monitoring Service, Zurich, Switzerland.

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., & Taylor, K. E. (2016). Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization. *Geoscientific Model Development*, *9*(5), 1937–1958. https://doi.org/10.5194/gmd-9-1937-2016

Hutter, K. (1983). *Theoretical Glaciology*. Springer Netherlands. https://doi.org/10.1007/978-94-015-1167-4

Lange, S. (2019). *WFDE5 over land merged with ERA5 over the ocean (W5E5)*. GFZ Data Services. https://doi.org/10.5880/PIK.2019.023

Maussion, F., Butenko, A., Champollion, N., Dusch, M., Eis, J., Fourteau, K., Gregor, P., Jarosch, A. H., Landmann, J., Oesterle, F., Recinos, B., Rothenpieler, T., Vlug, A., Wild, C. T., & Marzeion, B. (2019). The Open Global Glacier Model (OGGM) v1.1. *Geoscientific Model Development*, *12*(3), 909–931. https://doi.org/10.5194/gmd-12-909-2019

Millan, R., Mouginot, J., Rabatel, A., & Morlighem, M. (2022). Ice velocity and thickness of the world's glaciers. *Nature Geoscience*, *15*(2), 124–129. https://doi.org/10.1038/s41561-021-00885-z

Moses, W. S., Churavy, V., Paehler, L., Hückelheim, J., Narayanan, S. H. K., Schanen, M., & Doerfert, J. (2021). Reverse-mode automatic differentiation and optimization of GPU kernels via enzyme. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. https://doi.org/10.1145/3458817.3476165

Rackauckas, Chris, Innes, M., Ma, Y., Bettencourt, J., White, L., & Dixit, V. (2019). DiffEqFlux.jl - A Julia Library for Neural Differential Equations. *arXiv:1902.02376 [Cs, Stat]*. http://arxiv.org/abs/1902.02376

Rackauckas, Christopher, Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., Skinner, D., Ramadhan, A., & Edelman, A. (2021). *Universal Differential Equations for Scientific*

Machine Learning. arXiv. https://doi.org/10.48550/arXiv.2001.04385

Rackauckas, Christopher, & Nie, Q. (2017). DifferentialEquations.jl – A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia. *Journal of Open Research Software*, *5*, 15. https://doi.org/10.5334/jors.151

RGI Consortium. (2023). *Randolph Glacier Inventory - A Dataset of Global Glacier Outlines, Version 7*. National Snow; Ice Data Center. https://doi.org/10.5067/F6JMOVY5NAVZ

Sapienza, F., Bolibar, J., Schäfer, F., Groenke, B., Pal, A., Boussange, V., Heimbach, P., Hooker, G., Pérez, F., Persson, P.-O., & Rackauckas, C. (2024). *Differentiable Programming for Differential Equations: A Review*. arXiv. http://arxiv.org/abs/2406.09699

Sjursen, K. H., Bolibar, J., Meer, M. van der, Andreassen, L. M., Biesheuvel, J. P., Dunse, T., Huss, M., Maussion, F., Rounce, D. R., & Tober, B. (2025). Machine learning improves seasonal mass balance prediction for unmonitored glaciers. *EGUsphere*, 1–39. https://doi.org/10.5194/egusphere-2025-1206

Weis, M., Greve, R., & Hutter, K. (1999). Theory of shallow ice shelves. *Continuum Mechanics and Thermodynamics*, *11*(1), 15–50. https://doi.org/10.1007/s001610050102