Response to the reviewer 1

May 31, 2025

General comments

This manuscript presents the development of a new Python package that uses physics-informed neural networks (PINNs) to solve inverse problems in ice-sheet and ice-shelf dynamics. The authors provide a comprehensive description of their methods, including the network architecture, the physical constraints, the loss function, and the training hyperparameters, all of which are significant for successful training. The authors also provide concrete examples and technical details for using the package to solve various types of problems. Based on the provided example code, the package appears to be efficient and user-friendly. The manuscript is well-written and has a clear structure. I recommend this manuscript for publication, provided the authors address the following questions regarding novelty and reproducibility:
Response: We appreciate the reviewer for their thoughtful review and positive feedback on our manuscript. The comments and suggestions have been carefully addressed below.

1. **Loss function weight**: As mentioned in the manuscript, the PINN loss function comprises different terms, including data misfit and equation residuals. Each term requires a prefactor to weigh its contribution to the total loss. Table 2 provides the default values for these weights in PINNICLE, which show order-of-magnitude differences.

    - Regarding robustness, how can a user determine if these default values are suitable for their specific problem?
      Response: Thank you for this important question. As discussed in the manuscript, the selection of appropriate weights in the PINN loss function remains an open research topic, with little theoretical guidance currently available in the literature. The default weights we proposed in Table 2 were determined empirically through over 15,000 numerical experiments, and have been shown to provide robust performance in a variety of test cases. However, we acknowledge that the optimal weights can vary depending on the specific characteristics of the problem. Therefore, we recommend that users assess the suitability of these default weights by monitoring the convergence

history and the individual errors of each term during training, ensuring that the contributions from each term remain balanced for their specific problem. PINNICLE facilitates this process by recording the full training history, which can be used to adjust the weights accordingly.

- If adjustments are necessary, could the authors provide some rules-of-thumb for modifying these values, particularly for the equation residual weights?
  Response: Indeed, as mentioned in line 174 of the manuscript, our general rule-of-thumb is to adjust the weights so that the contributions from each term are approximately of the same order of magnitude—typically on the order of 1. We recognize that this recommendation could be more explicit, and we will revise the manuscript to clearly highlight this principle for the reader.
- Regarding implementation, could the authors include an example in one of Listings 1-3 demonstrating how to set weights different from the default values?
  Response: Thank you for this suggestion. We have provided detailed instructions for adjusting weights in our online documentation (`https://pinnicle.readthedocs.io/en/latest/training/lossfunctions.html`). We will add this in example 1 to demonstrate how users can customize weights directly.

2. **Fourier Feature**: The authors mention the use of Fourier Features in PINNICLE, expressed as $f(x) = [\cos(Bx), \sin(Bx)]$, where B is sampled from a Gaussian distribution $N(\theta, \sigma)$. According to Tancik et al., σ is a hyperparameter that users must carefully determine to capture high-frequency features in the data without overfitting.

   (a) Therefore, does PINNICLE automatically select the optimal σ value for generating the Fourier Feature net weights based on different training data, or should the user set this value manually?
   Response: This parameter has to be set manually, as demonstrated in example 2.

   (b) If the latter, what is the default value of σ in PINNICLE?
   Response: The default value in PINNICLE is a constant 10, and by default the FFT is turned off.

3. **Random Sampling**: In Line 197, the authors state that "...PINNICLE employs a random sampling strategy to automatically load data..."

   (a) Regarding the random sampling strategy, does PINNICLE exclusively use a uniform distribution for random sampling, or are other options available for users to select?
   Response: Currently, PINNICLE only supports uniform random sampling using *numpy.random.choice*. However, we plan to provide more options for sampling strategies in future versions.

(b) Does PINNICLE permit users to implement their own customized sampling strategies?
Response: Yes, since PINNICLE is open source, advanced users can implement and integrate their own customized sampling strategies as needed.

(c) Another question regarding random sampling is whether the data sampling points and the collocation points used for evaluating equation residuals are fixed throughout the training process or re-sampled at certain iteration intervals?
Response: By default, data sampling points and collocation points are fixed throughout the training process. However, we are planning to provided experimental resampling functions for advanced users who wish to re-sample at specified intervals.

4. **Mass Balance**: The mass balance equation (Equation 1) includes a net mass accumulation term, a.

   (a) When incorporating this equation into the PINN training, does the user need to provide data for a across the entire domain, or can a be treated as a constant?
   Response: Yes, the mass balance term $\dot{a}$ is treated as a spatially varying variable in PINNICLE, as demonstrated in Figure 7 and 8.

   (b) If a can be treated as a constant, what is its default value in PINNICLE?
   Response: Unfortunately, because $\dot{a}$ is an important variable in the mass balance equation, it is only considered as an output variable of the PINN and there is no default value provided by PINNICLE.

5. **Novelty**: Another Python package, DIFFICE-jax, which also uses PINNs to study ice dynamics, was recently published. Regarding novelty, could the authors elaborate on the differences between PINNICLE and DIFFICE-jax?
Response: We are indeed aware of DIFFICE-jax, and have had close collaborations with its authors. The two packages were developed almost simultaneously. In terms of differences, PINNICLE is designed to support multiple backends, including TensorFlow, PyTorch, and JAX, which offers users flexibility depending on their preferences, hardware compatibility, and experience.

Minor point

Figure 1 appears somewhat sparse. Consider shrinking the image or reorganizing its content to make it more compact.
Response: Thank you for this suggestion. We decide to remove Figure 1.

Suggested Additional Citations

The following relevant publications on PINNs in glaciology could also be cited:
Response: Change has been made.

Riel, Bryan, and Brent Minchew. "Variational inference of ice shelf rheology with physics-informed machine learning. " Journal of Glaciology 69.277 (2023): 1167-1186.
Wang, Yongji, et al. 1219-1224. "Deep learning the flow law of Antarctic ice shelves. " Science 387.6739 (2025):
Wang, Yongji, and Ching-Yao Lai. "DIFFICE-jax: Differentiable neural-network solver for data assimilation of ice shelves in JAX. " Journal of Open Source Software 10.109 (2025): 7254.