Physics Informed Neural Networks for Mountain Glaciers

Shreyas Sunil Gaikwad and Mohammad Afzal Shadab

Motivation for Ice Sheets and Oceans

- Sea level rise projections, climate change projections have large uncertainties. One of the reasons is that until recently *(D.N. Goldberg et. al 2018)*, there were no synchronously coupled models for ice sheets and oceans. Even for the MITgcm model, it is assumed that the ice velocities do not change too quickly and hence the ice velocities are not updated for each ocean time step. If we had a PINN emulator for the ice sheet model, we could perhaps achieve coupling at each time step. The process of finding coupled adjoint sensitivities could become easier too.
- Without atmospheric and oceanic circulation, the temperature difference between the equator and the poles would be ~$198^{\circ}F$ $(110^{\circ}C)$. The oceans contribute to ~$35\,\%$ in this temperature regulation through a mechanism called the MOC (Meridional Overturning Circulation). It is likely that freshwater input (ice melting) is weakening the MOC. Thus a good understanding of ice melting is important. *(Lozier 2012)*
- The first step could be to start with a local, non-linear ice model.

Motivation for UQ and Inverse Problems

- Part of UQ algorithms can be based on emulators, for which machine learning based on neural networks may be a compelling approach. For example, a typical posterior distribution when solving inverse problems under uncertainty using the Bayesian framework looks like this -

$$\Pi_{\text{post}}(m) \propto \exp\left(-\frac{1}{2}\|\zeta(m) - d\|^2_{\Gamma^{-1}_{\text{noise}}} - \frac{1}{2}\|m - m_{\text{pr}}\|^2_{\Gamma^{-1}_{\text{prior}}} \right)$$

- Typically, $\zeta(m)$ - the parameter to observable map is non-linear and requires a forward PDE solve. If we had a pre-trained PINN to emulate this PDE, we could perhaps get to sample from the exact posterior distribution instead of using a Gaussian about the MAP point (also called the Laplace approximation).
- There are also some recent developments in Bayesian PINNs, which finds a joint posterior distribution for model parameters and Neural network parameters, which then can be sampled using Hamiltonian MCMC or VI methods. *(L. Yang et. al 2021)*

Neural network specs

- All neural networks trained below use the Adam solver with the tanh activation function.
- L-BFGS is apparently way better than Adam for PINNs (we have used it), it is a limited memory second order method. Yet we ran out of memory trying to use it. We do believe for very complicated systems L-BFGS will be too costly anyways, or we will be forced to approximate the hessian with very low number of vectors, affecting results.
- We used a step learning rate with $\gamma = 0.9$ for every 100 steps. Since not much data was used, an entire epoch could fit in one batch. We found that it is really difficult to tune the learning rate to get good convergence.
- All training was done on TPUs on Google Colab. We used the PyTorch library since it offers a lot of control and flexibility for controlling neural network architectures.
- The entire code can be found here - https://github.com/Shreyas911/PINN

A Toy Problem

Simple 1D diffusion equation

Analytical solution

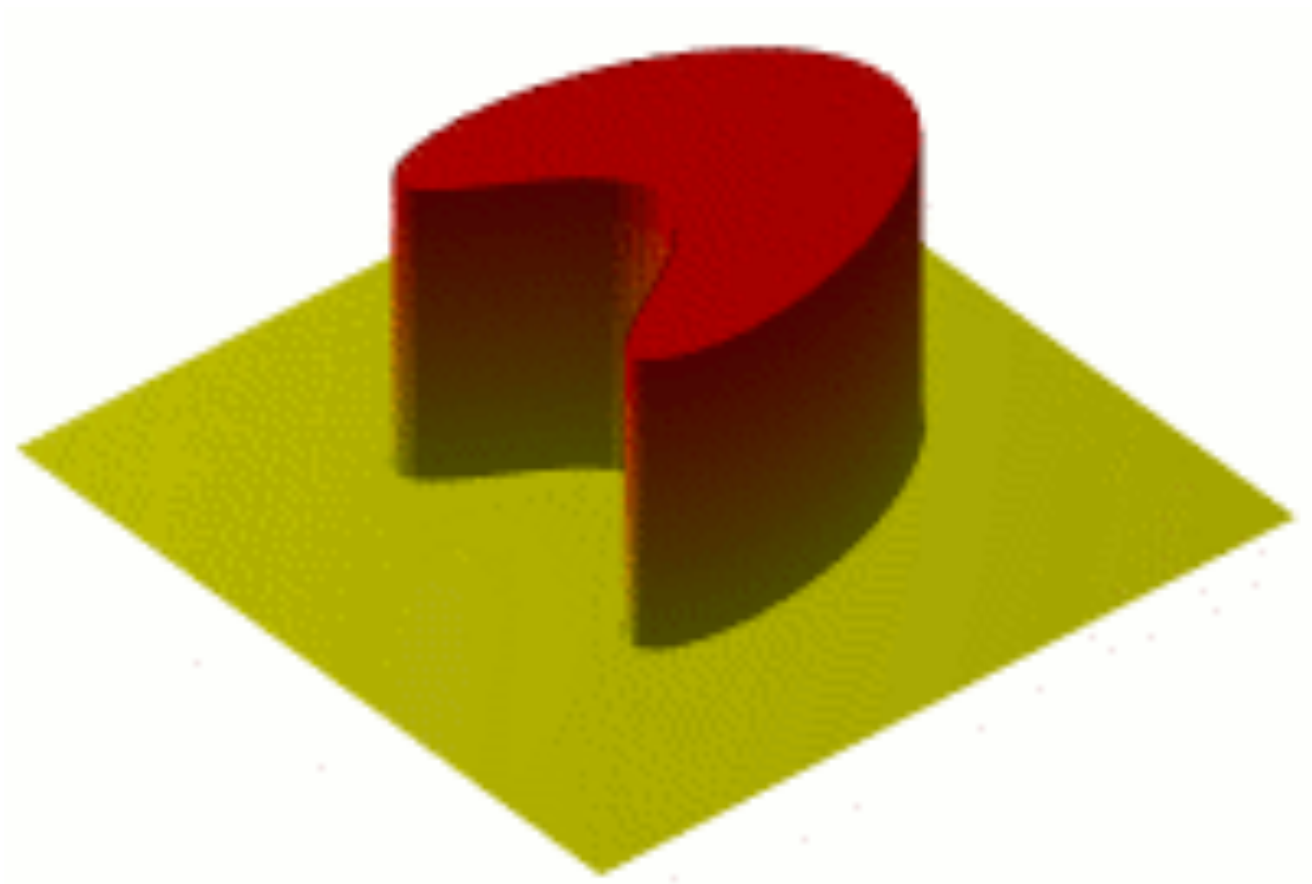
1D Diffusion Equation - Data generated using Finite Difference method to have some natural noise.

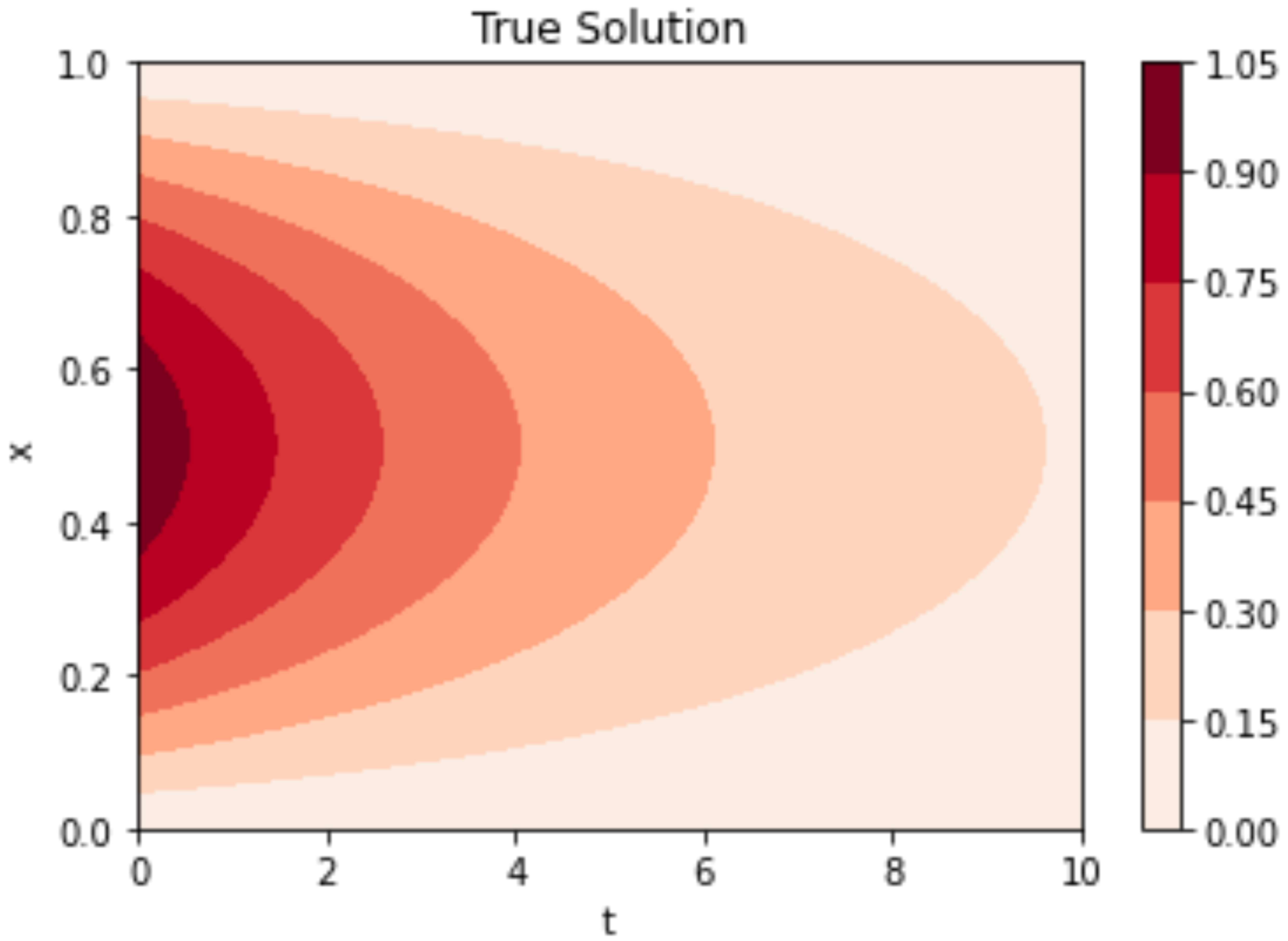
$$\frac{\partial u(x,t)}{\partial t} = D\frac{\partial^2 u(x,t)}{\partial x^2}$$

$$u(0,t) = 0, u(1,t) = 0 \quad \forall t \in [0, 10]$$

$$u(x,0) = \sin(x) \quad \forall x \in [0, 1]$$

$$D = 0.02$$



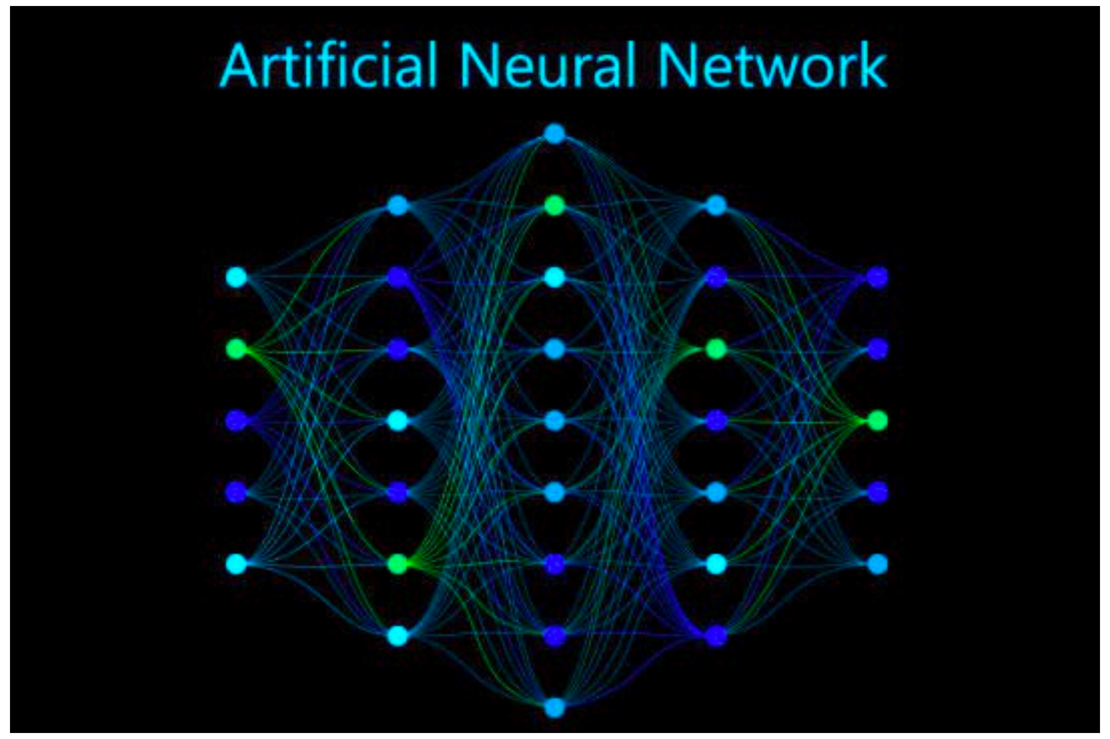


Task 1 - Emulate the PDE

We want the neural network to obey the heat equation, or some law of physics.
So we penalize the network for not obeying such laws.

x

t



$$u(x, t) = \mathcal{N}(x, t;\ \theta)$$

PDE operator $\quad \mathcal{F}(x, t) = u_t(x, t) - D u_{xx}(x, t)$

Loss function $\quad \mathcal{L} = \frac{1}{N_i} \sum_{i=1}^{N_i} \left(u^i - u(x^i, 0)\right)^2 + \frac{1}{N_b} \sum_{i=1}^{N_b} \left(u^i - u(x^i, t^i)\right)^2 + 100 \frac{1}{N_c} \sum_{i=1}^{N_c} \left(\mathcal{F}(x^i, t^i)\right)^2$

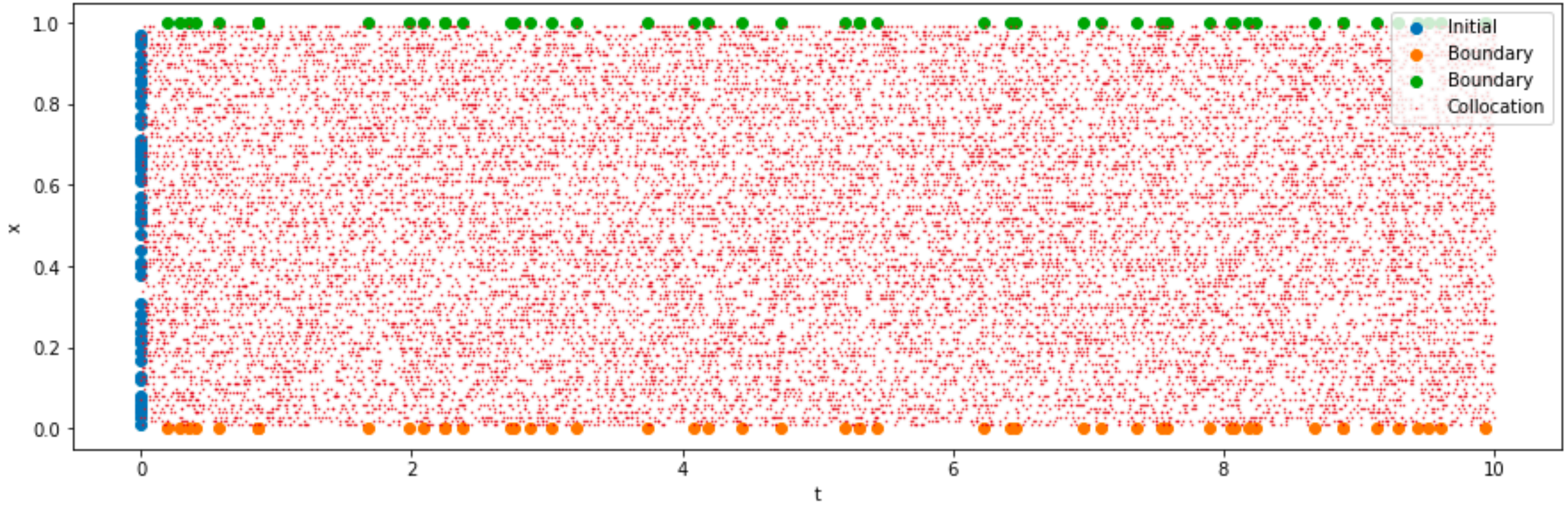
Initial conditions

Boundary conditions

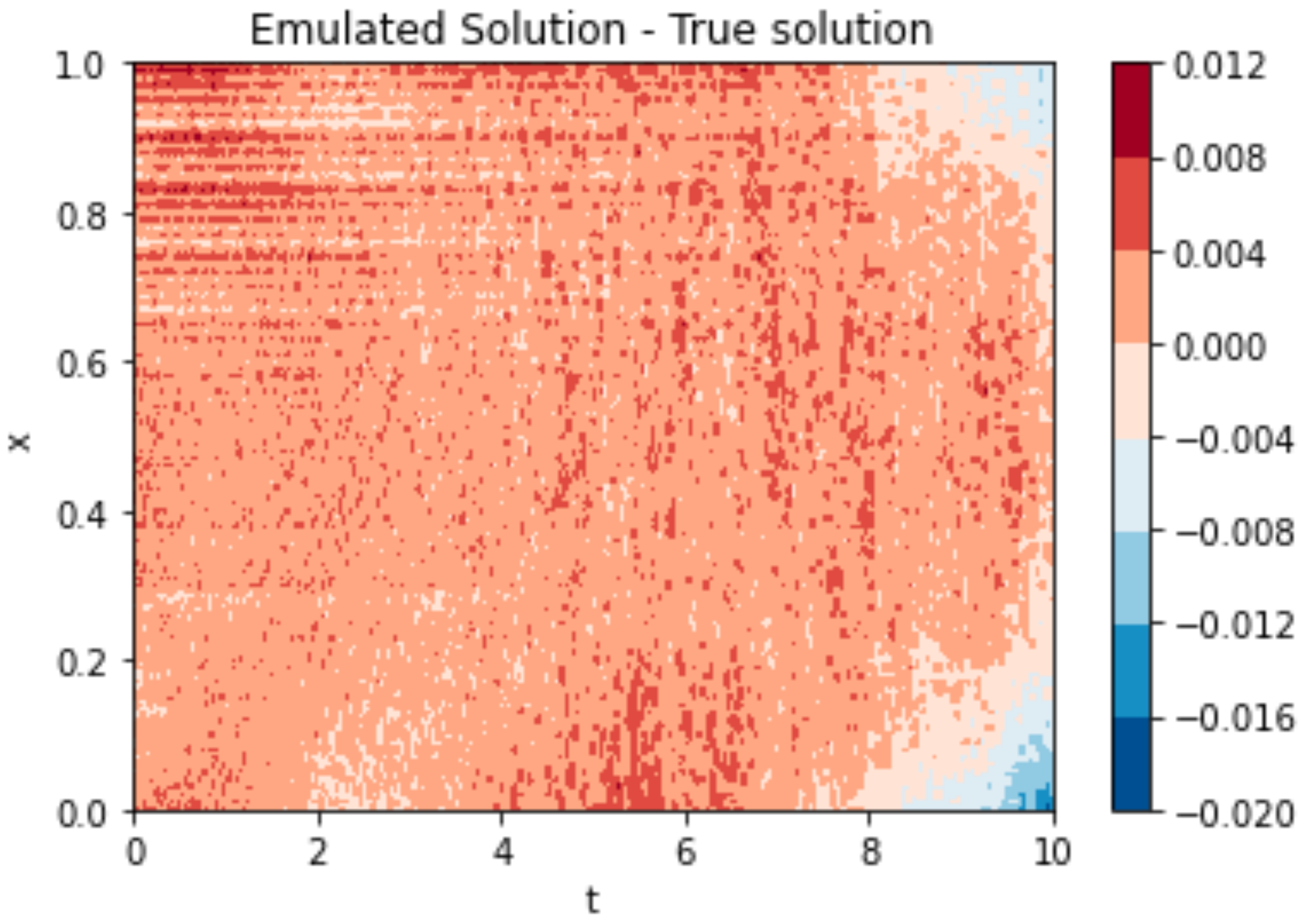
Interior collocation

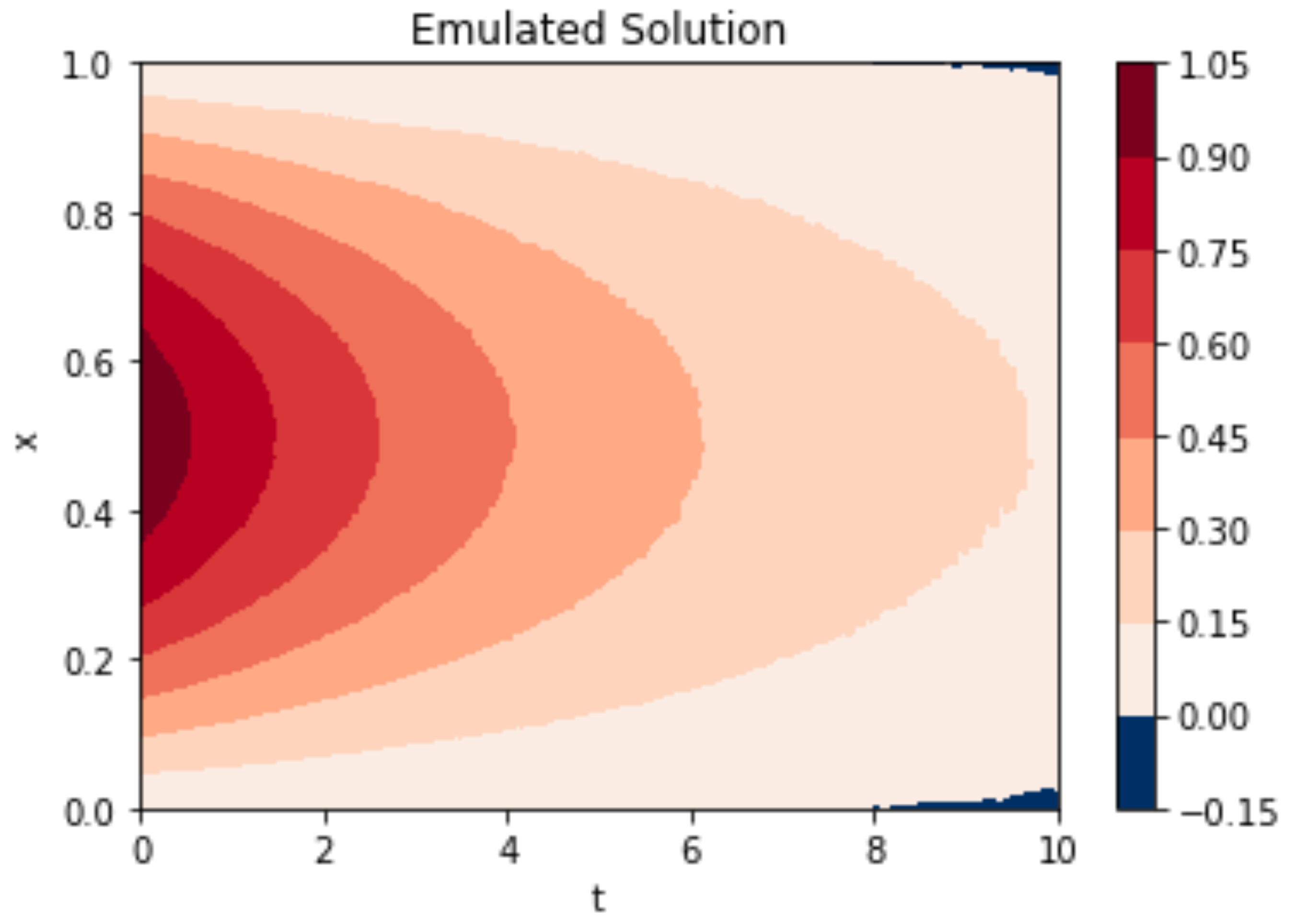
Sampling of data

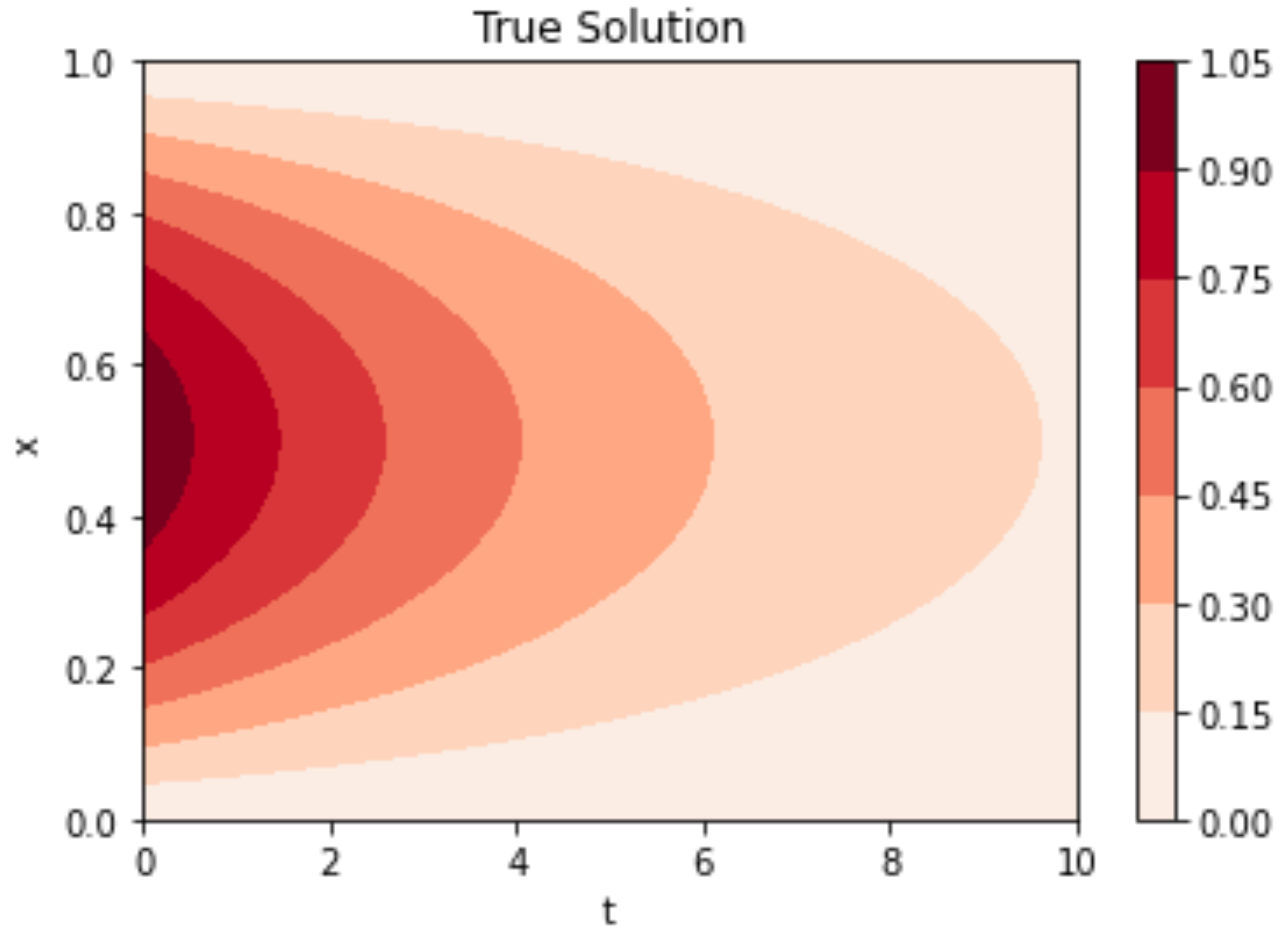
Fit the data at the blue, green, orange points and satisfy the heat equation at the red interior points.



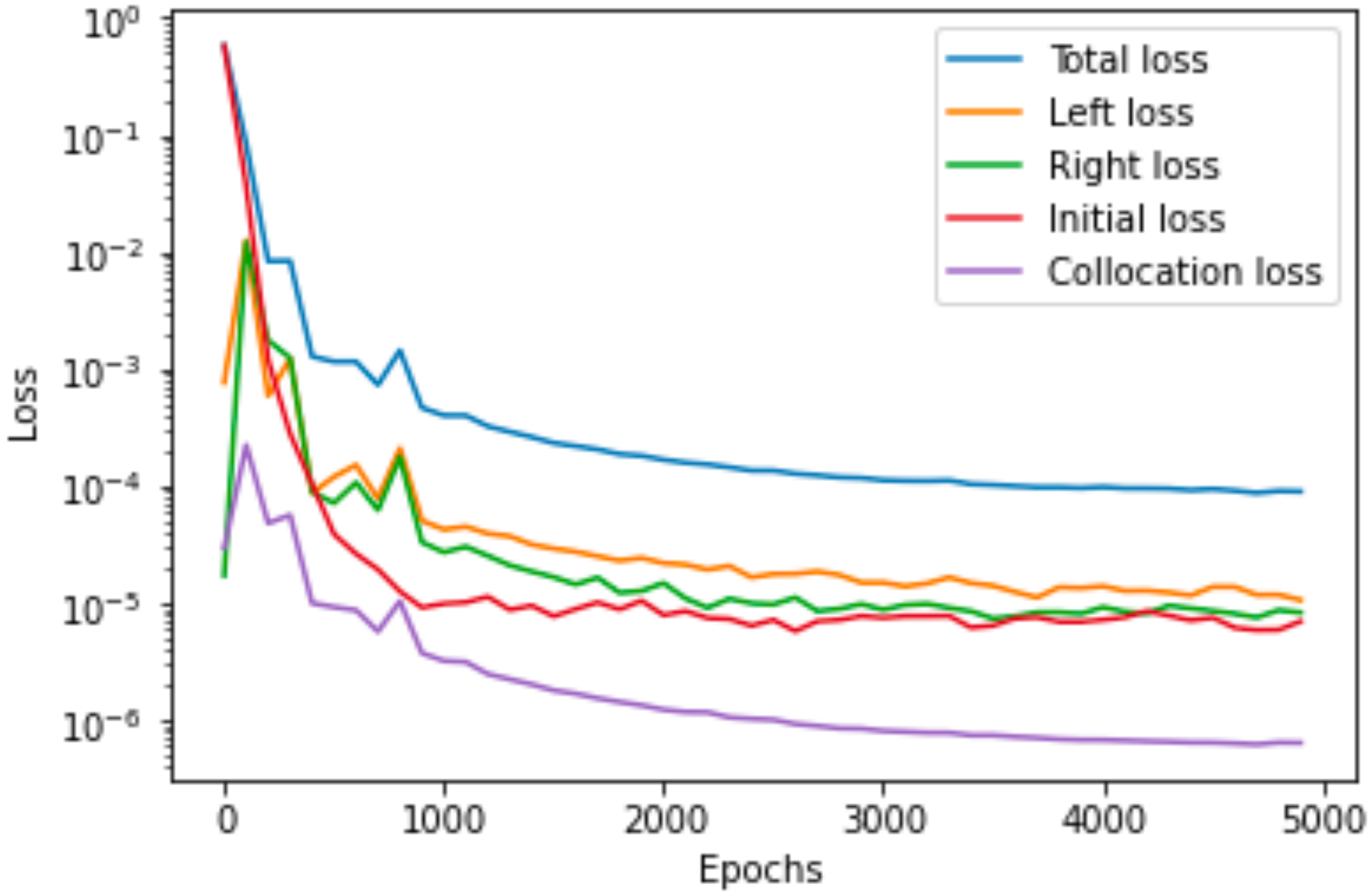
Results for the Emulator







Results for the emulator



Task 2 - Infer diffusion coefficient from data

$$u(x,t) = \mathcal{N}(x,t;\theta)$$

PDE operator $\quad \mathcal{F}(x,t) = u_t(x,t) - Du_{xx}(x,t)$

Loss function $\quad \mathcal{L} = \dfrac{1}{N_i}\displaystyle\sum_{i=1}^{N_i}\left(u^i - u(x^i,0)\right)^2 + \dfrac{1}{N_b}\sum_{i=1}^{N_b}\left(u^i - u(x^i,t^i)\right)^2 + 75\dfrac{1}{N_c}\sum_{i=1}^{N_c}\left(\mathcal{F}(x^i,t^i)\right)^2 + 25\dfrac{1}{N_d}\sum_{i=1}^{N_d}\left(u^i - u(x^i,t^i)\right)^2$

Initial conditions

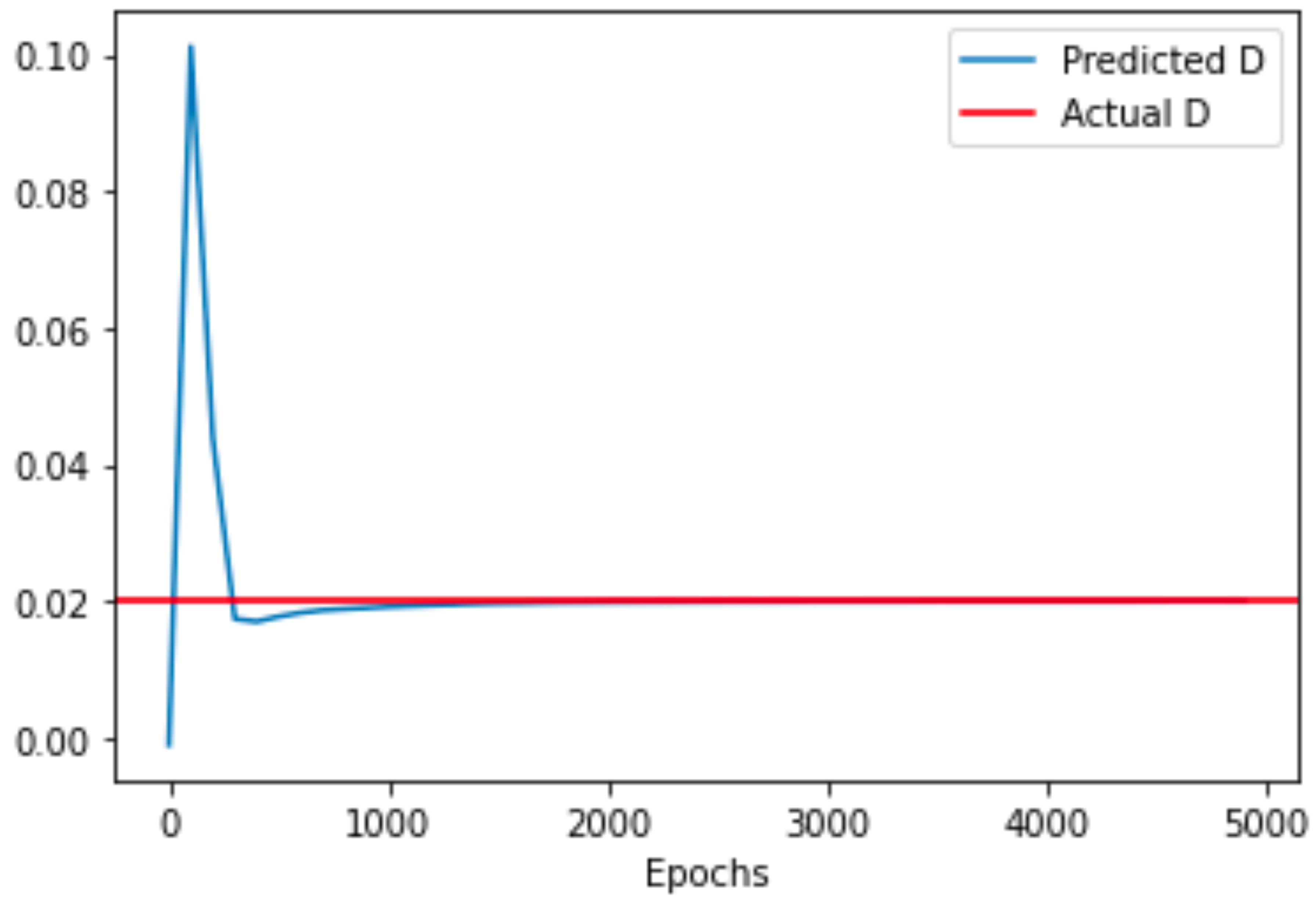
Boundary conditions

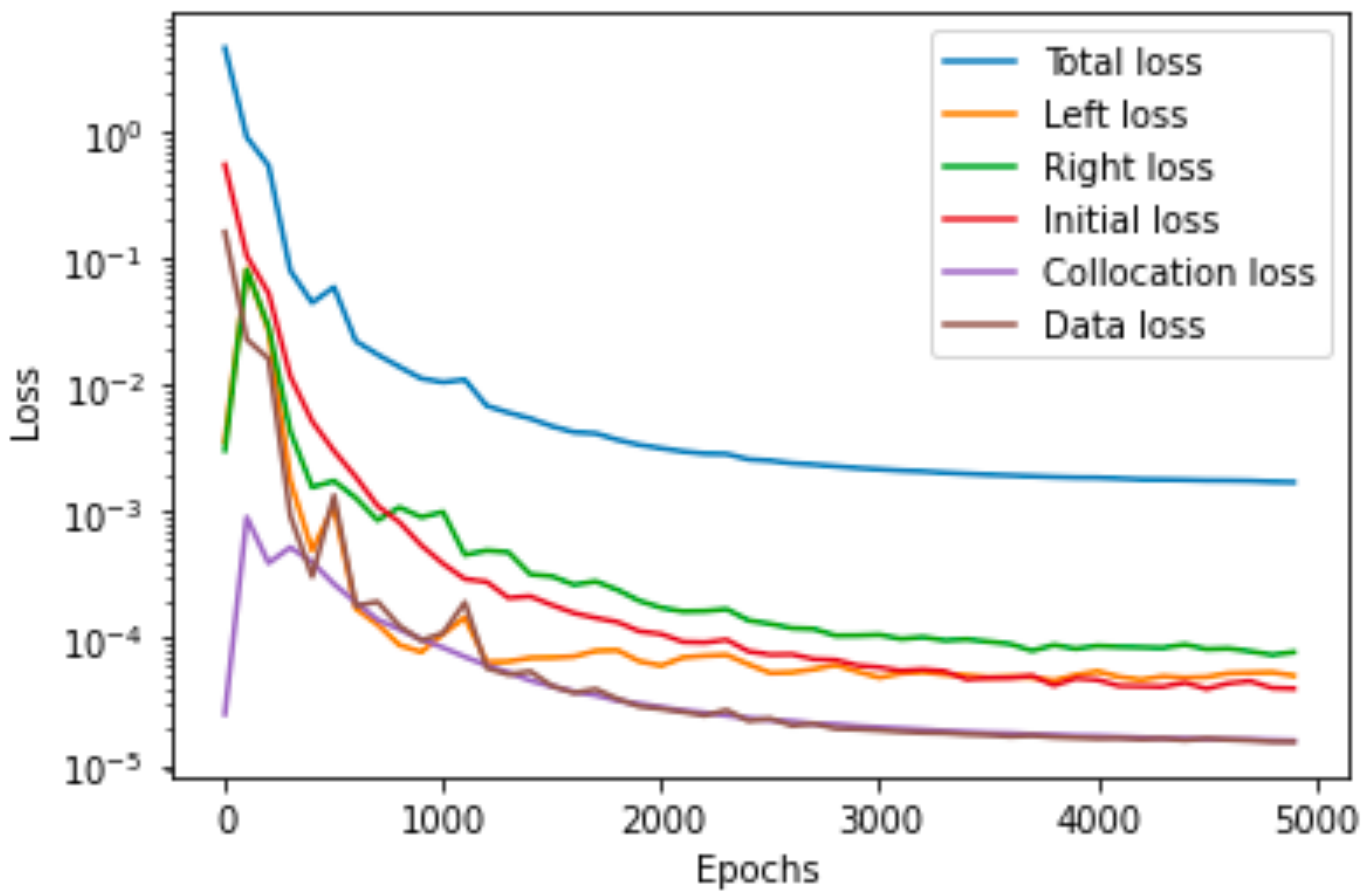
Interior collocation

Data misfit
NEW TERM

The diffusion coefficient D is unknown. Optimize the loss function wrt neural net parameters θ as well as D.

Results for inversion





True PDE: $u_t = 0.02u_{xx}$

Predicted PDE: $u_t = 0.02u_{xx}$

Simple Mountain Glacier model

1D highly non-linear diffusion equation

True solution

Mountain glacier model
Fundamentals of Glacier Dynamics, by CJ van der Veen
Data generated using Finite Volumes method on a staggered grid.

$$\frac{\partial H}{\partial t} = -\frac{\partial}{\partial x}\left(-D(x)\frac{\partial h}{\partial x}\right) + M$$

$$D(x) = CH^{n+2}\left|\frac{\partial h}{\partial x}\right|^{n-1}$$

$$C = \frac{2A}{n+2}(\rho g)^n$$

$$H(x,t) = h(x,t) - b(x)$$

$$H_l = 0, H_r > 0$$

PARAMETERS

$$\frac{\partial b}{\partial x} = -0.1$$

$M(x) = M_0 - xM_1$ (accumulation rate, essentially a source term)

$$M_0 = 4.0 \text{ m/yr}, \ M_1 = 0.0002 \text{ yr}^{-1}$$

$$\rho = 920 \text{ kg/m}^3$$

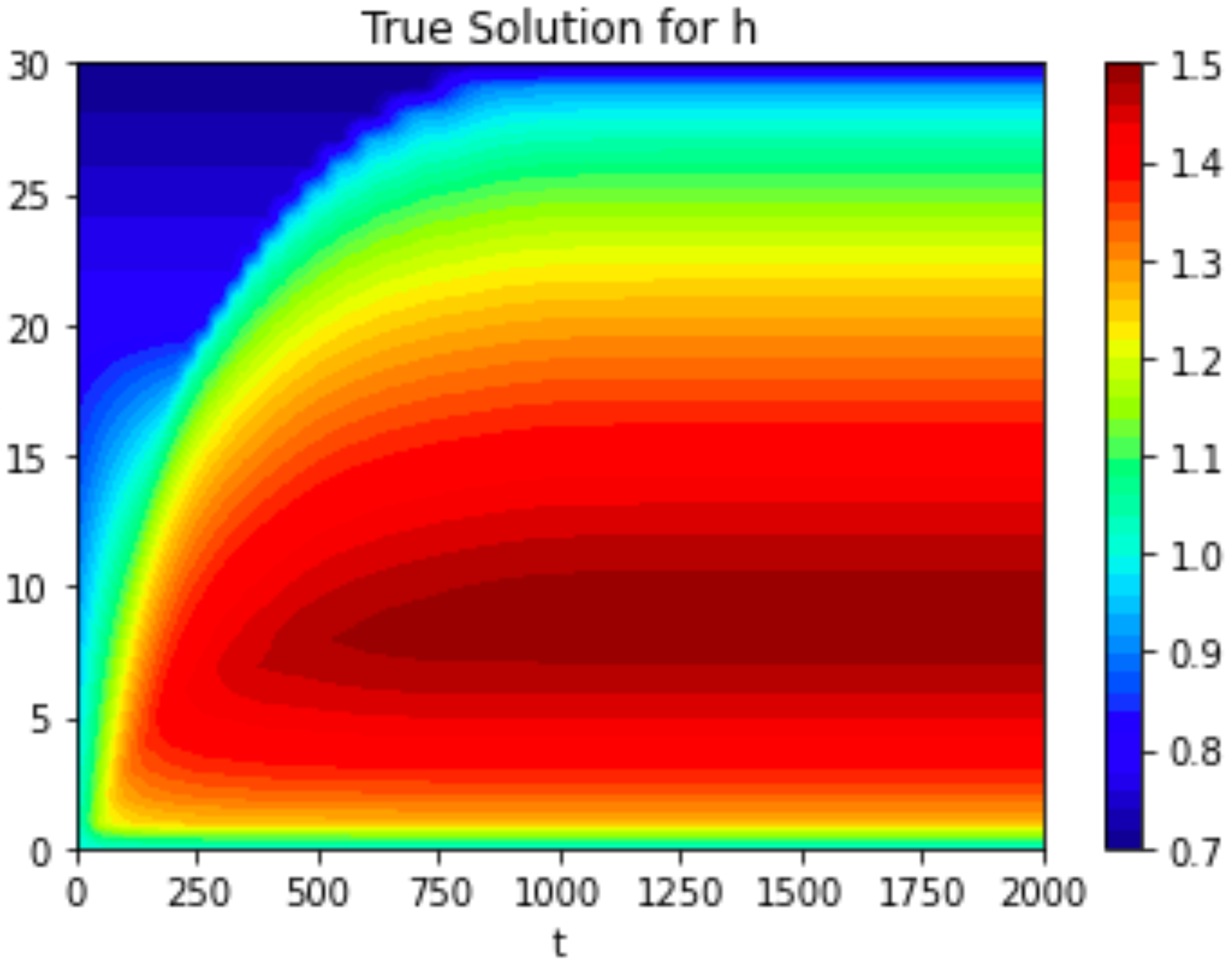
$$g = 9.8 \text{ m/s}^2$$

$$A = 10^{-16} \text{ Pa}^{-3}\text{a}^{-1}$$

$$n = 3$$

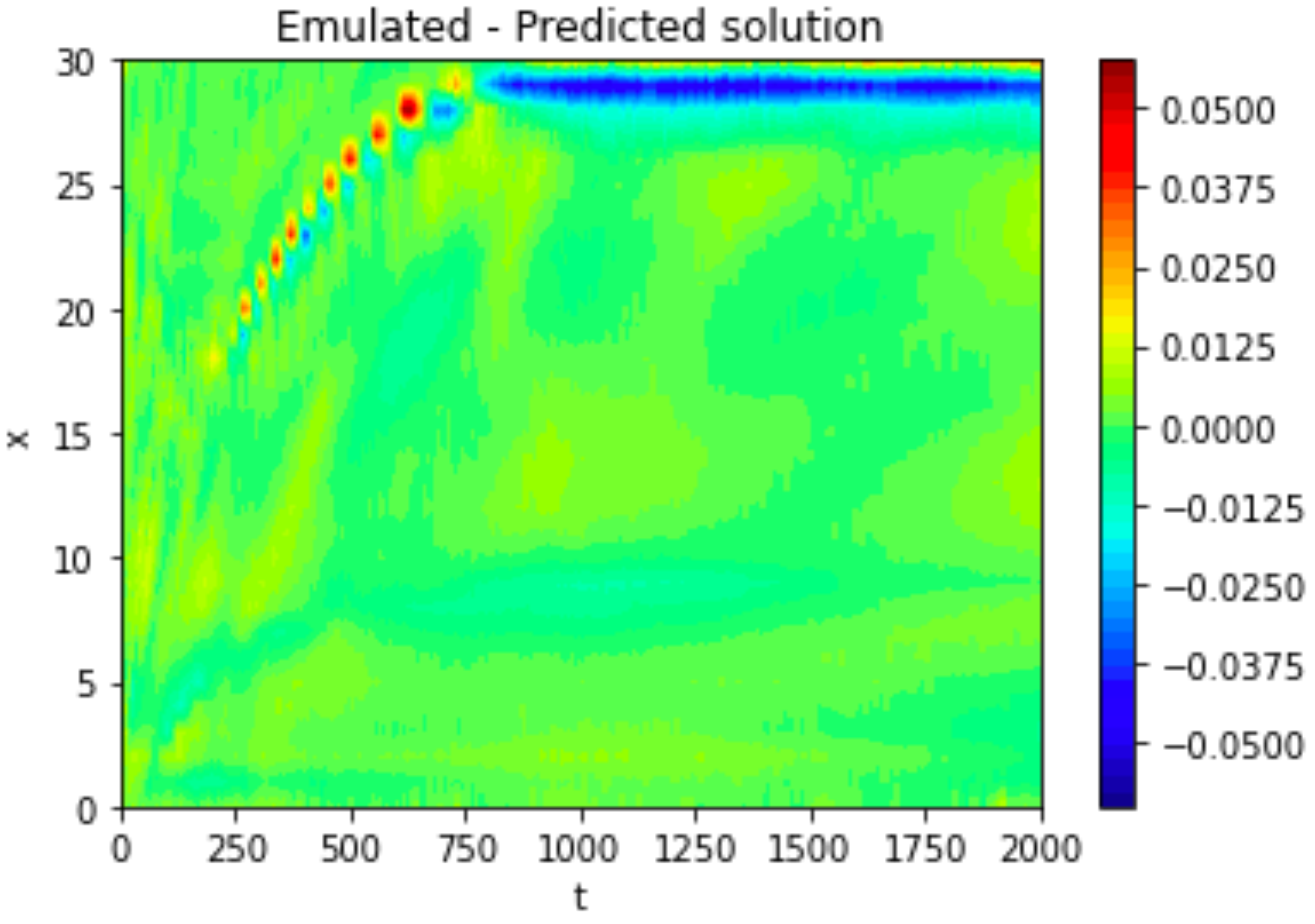
$$dx = 1.0 \text{ km}, \ L = 30 \text{ km}$$

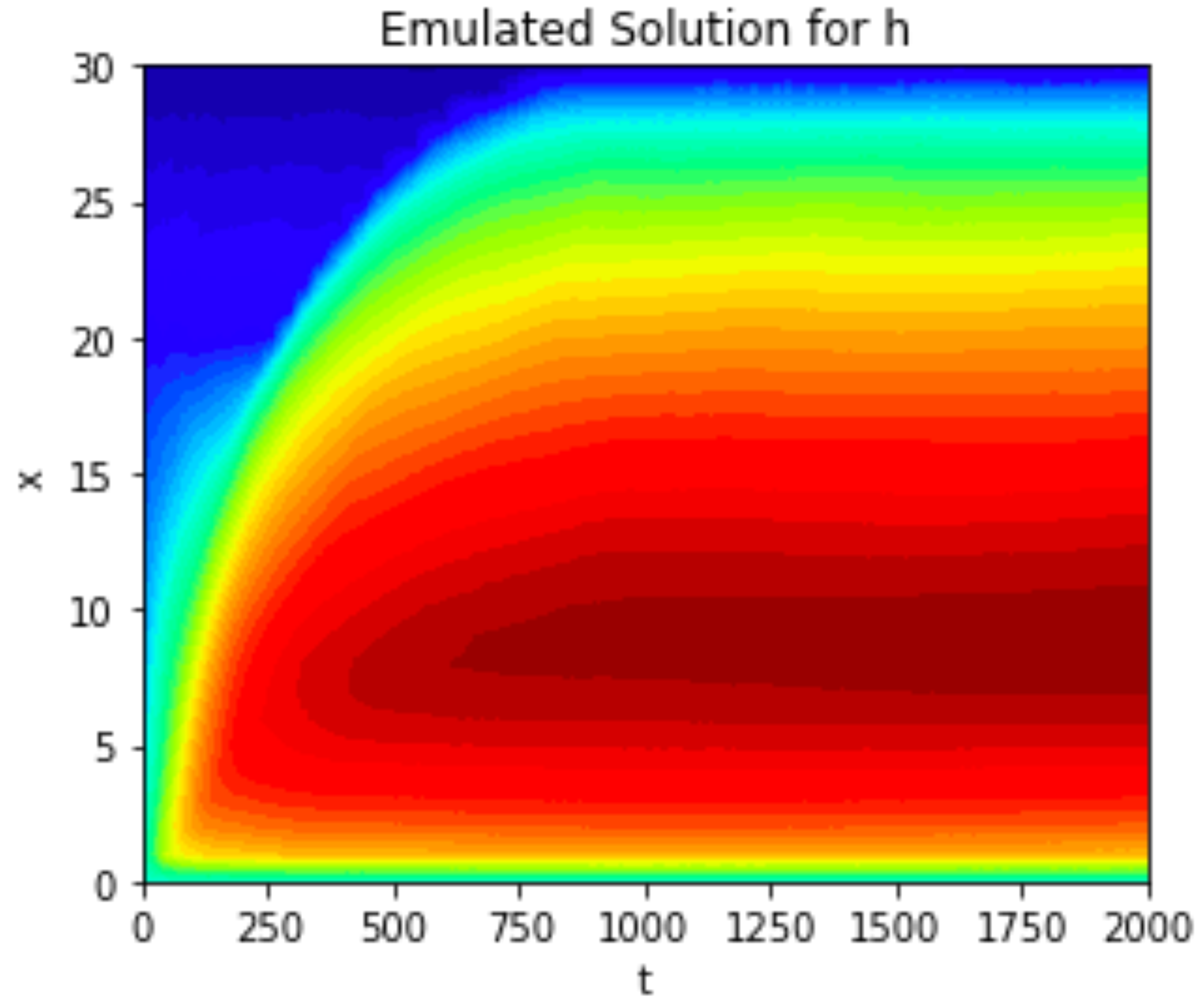
$$dt = 1 \text{ month}, \ T = 2000 \text{ yr}$$

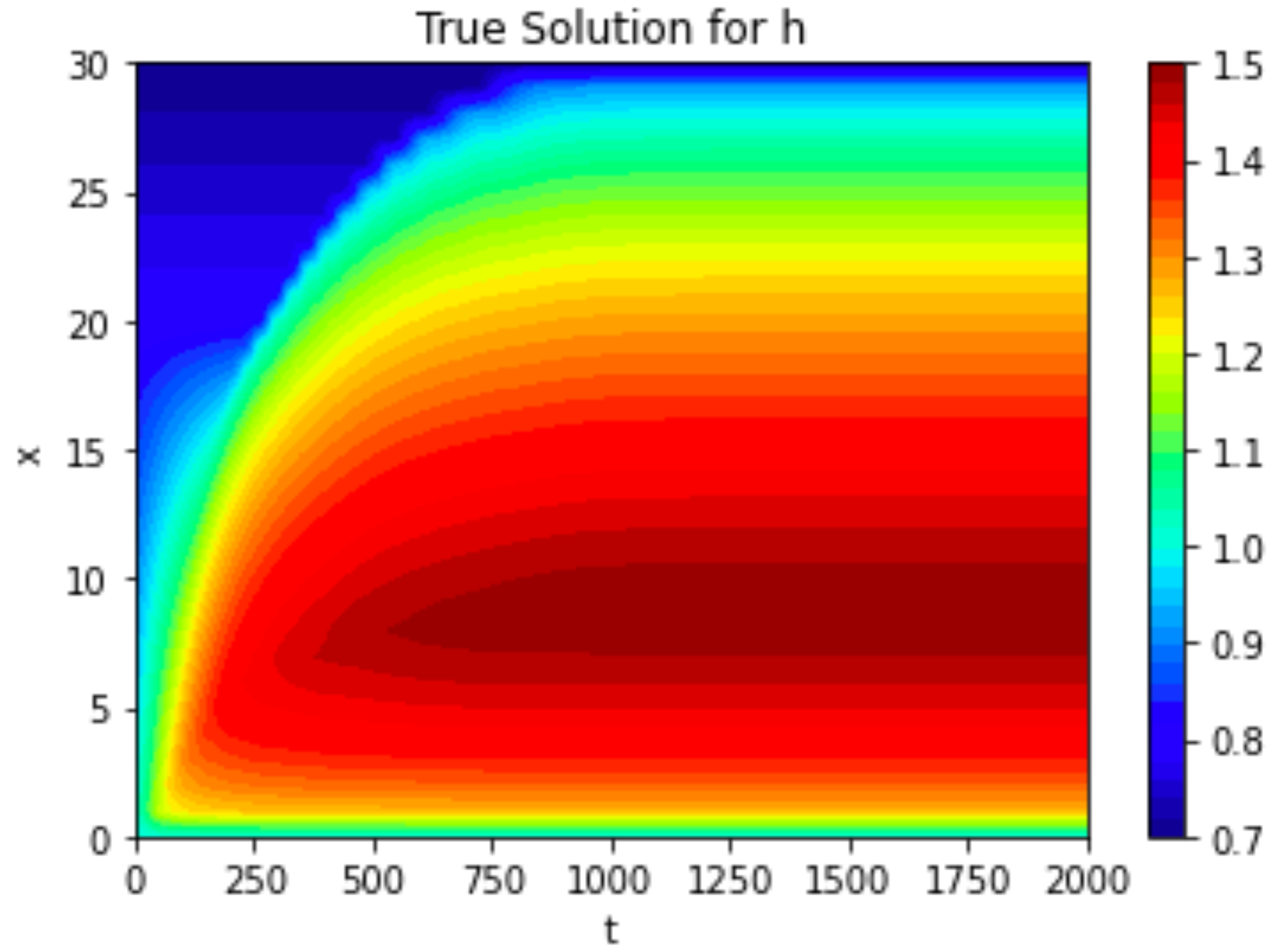


Results for the Emulator

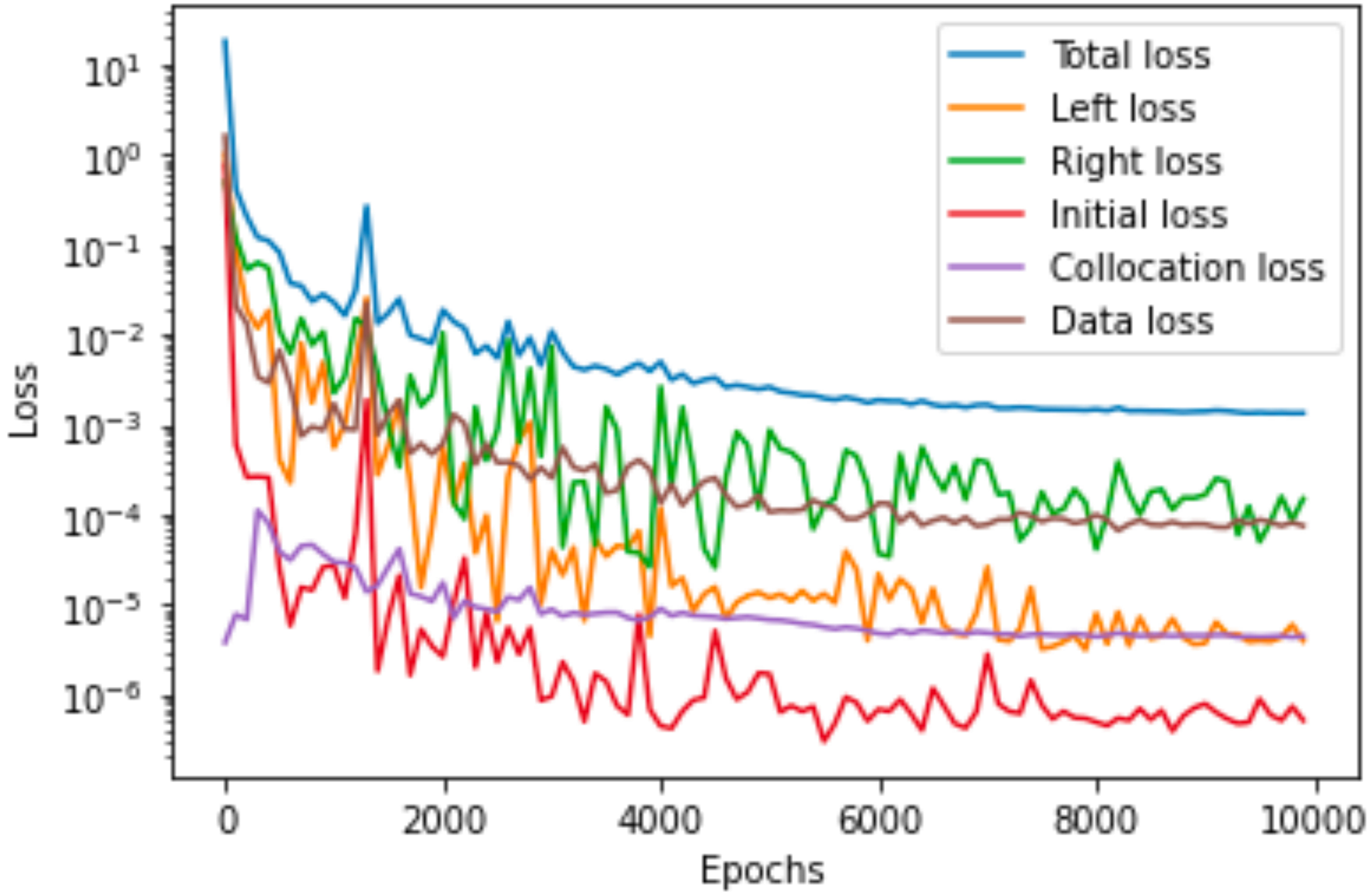
Had to give the emulator training some data to get some sensible results.



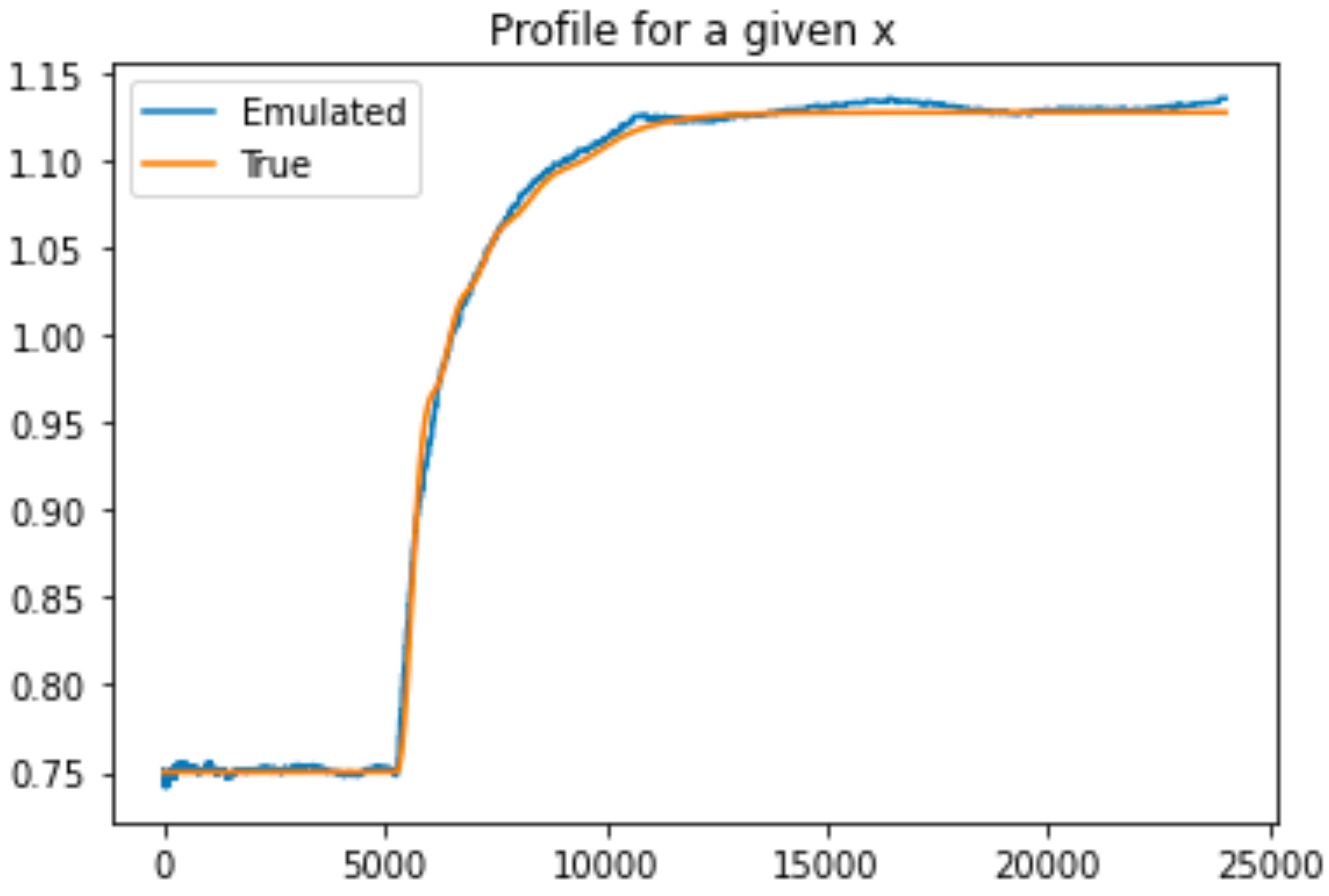


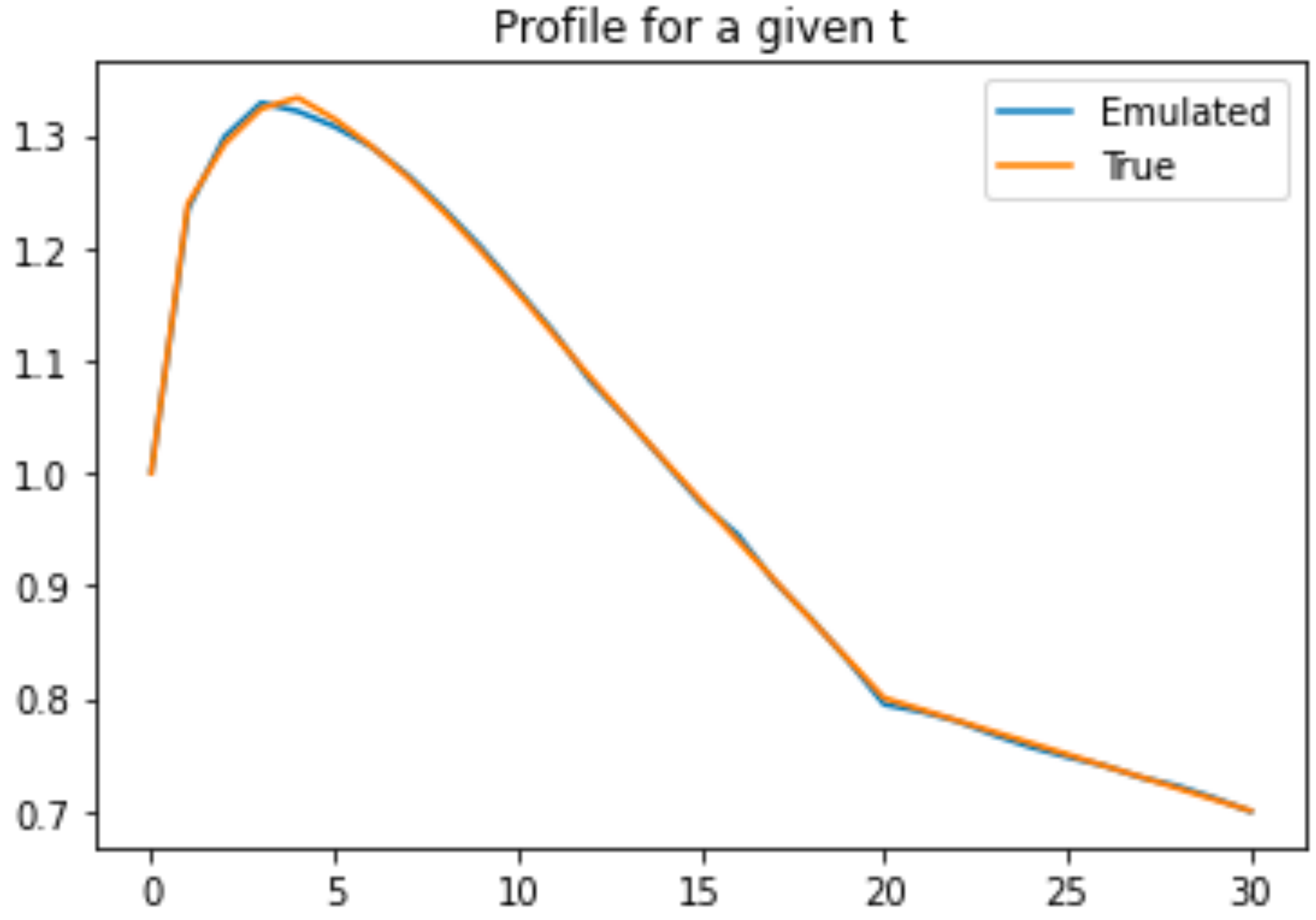


Results for the Emulator

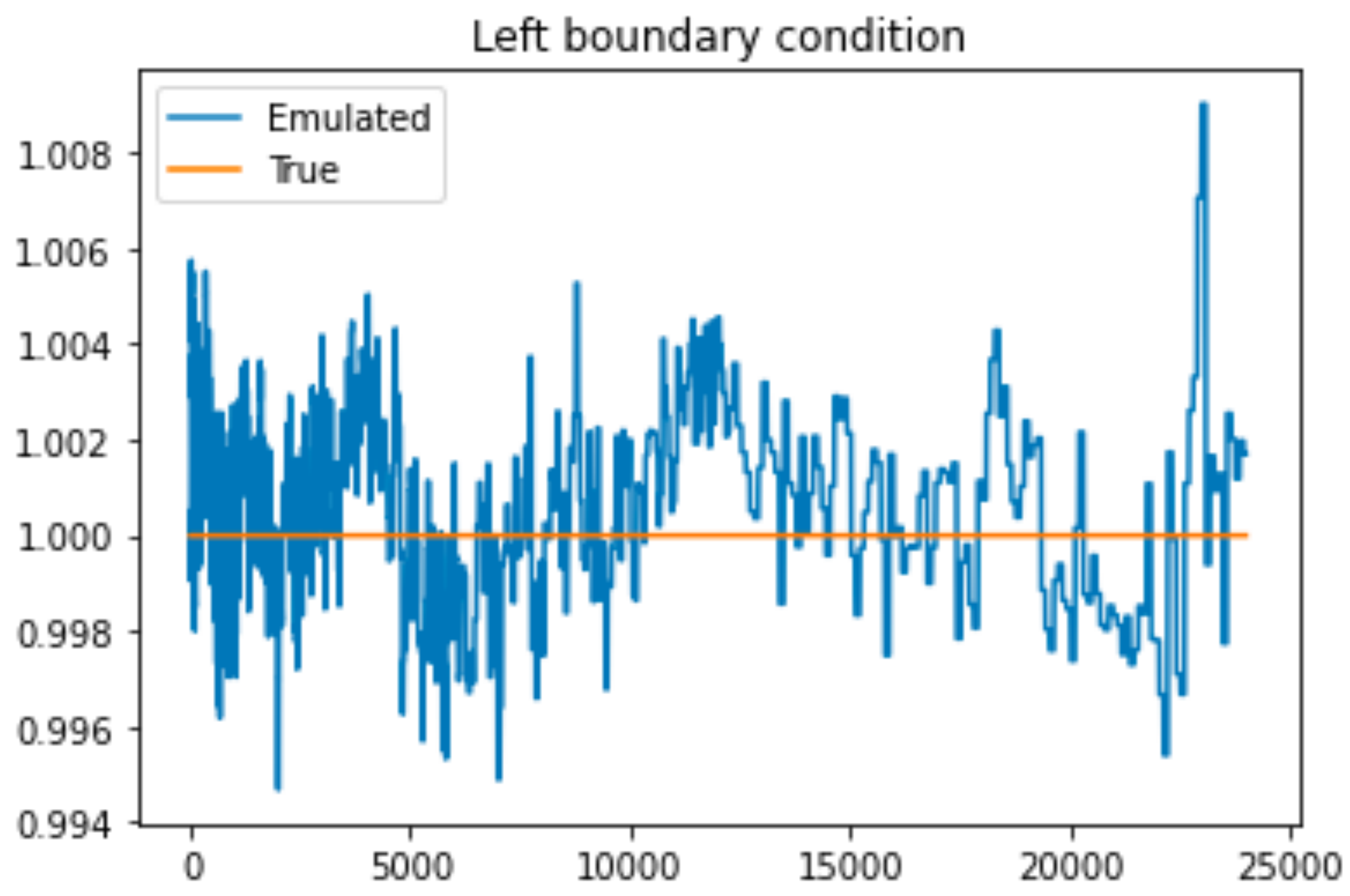


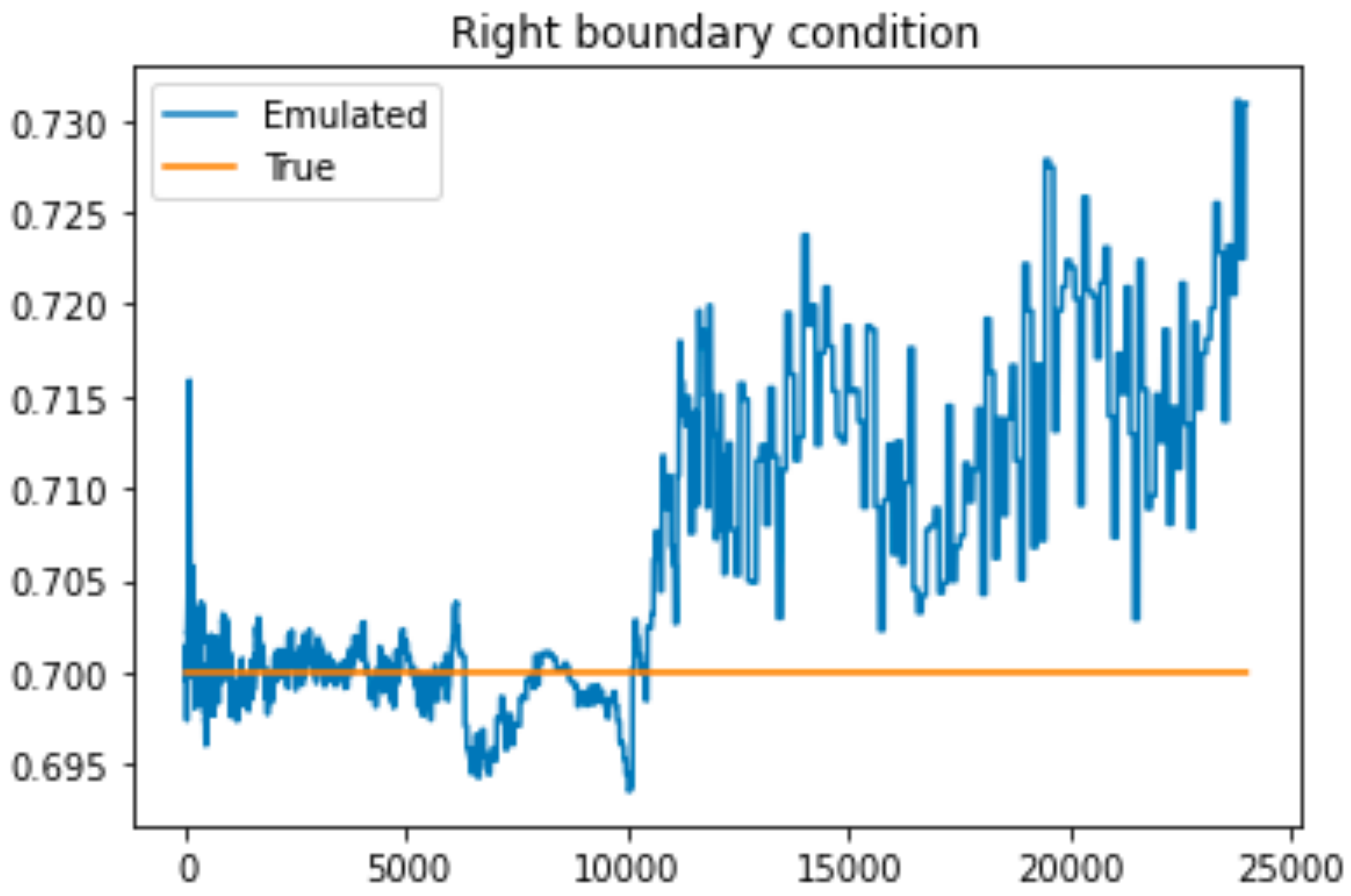
Results for the Emulator

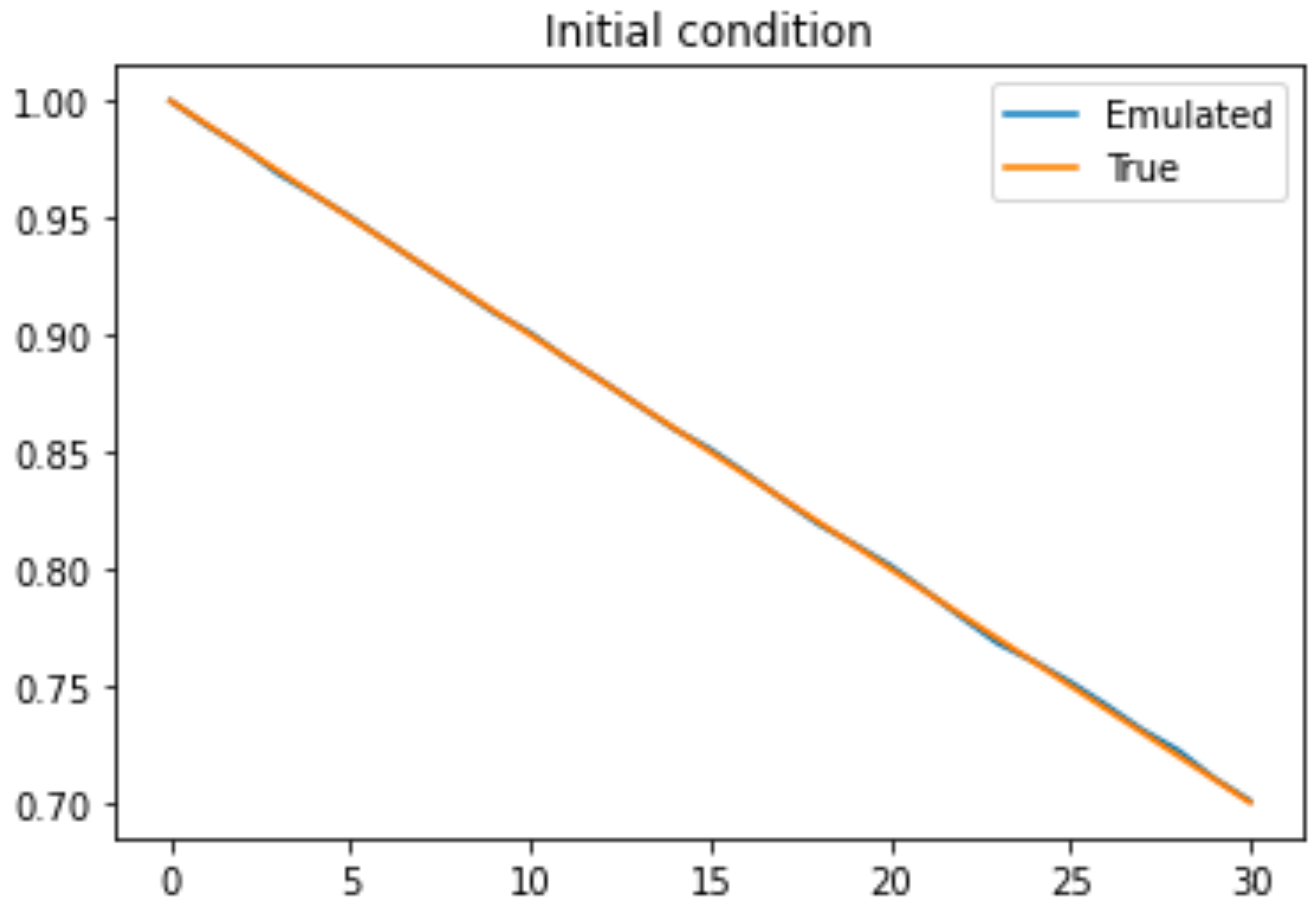




Results for the Emulator

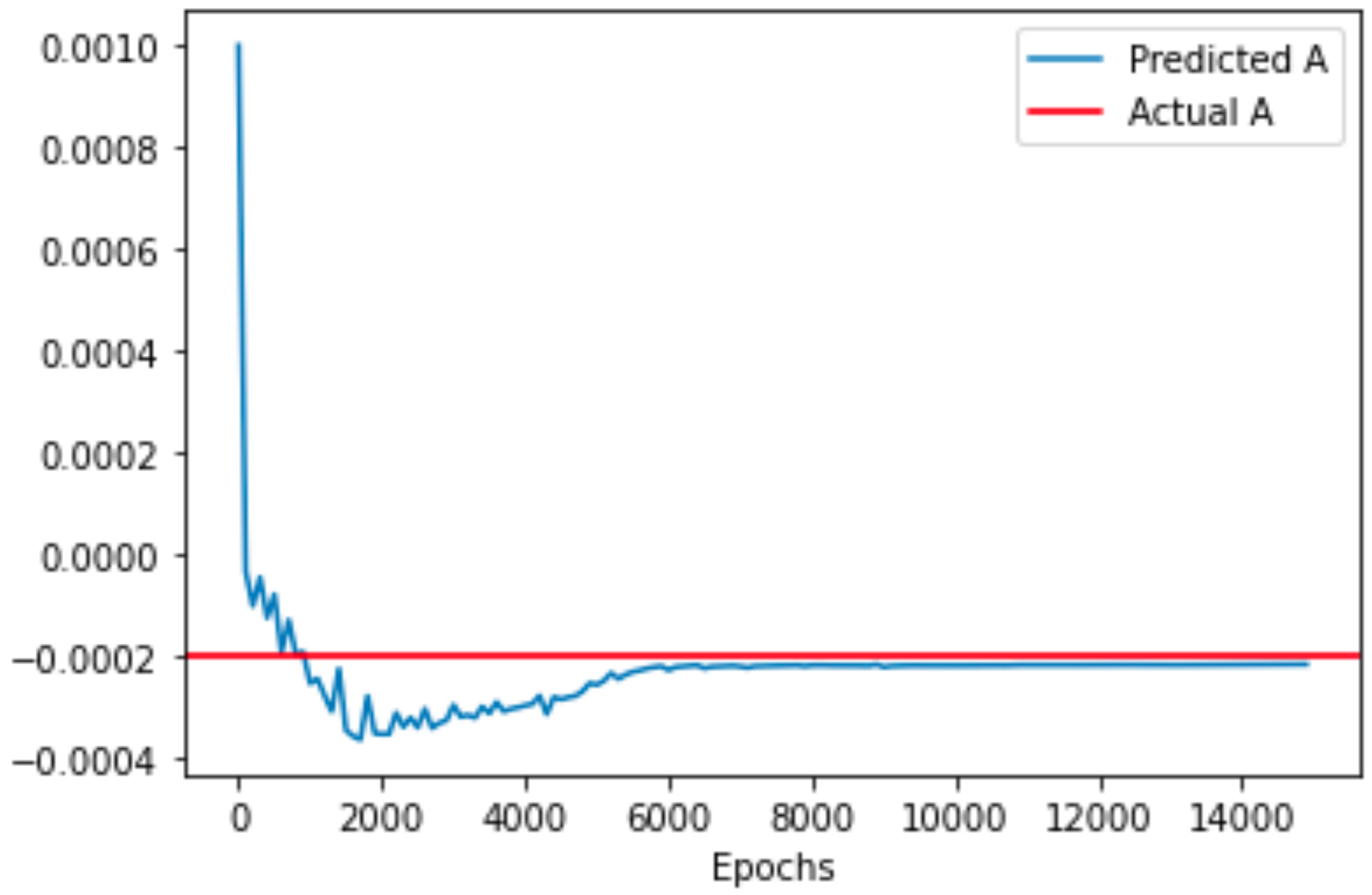


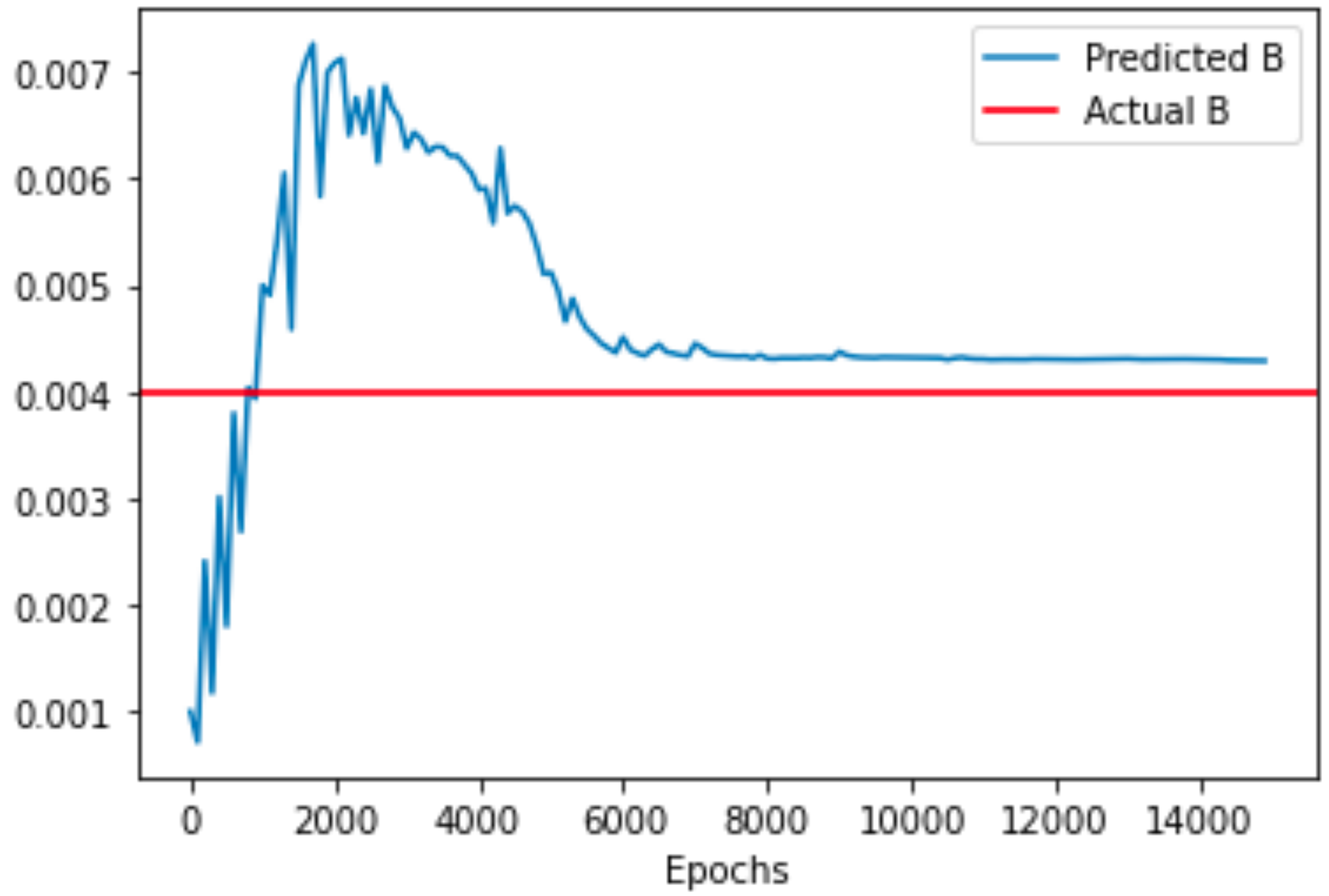




Task 2 - Inversion

We model the source term as M = B + Ax and try to infer A and B from data, just like we inferred D previously.



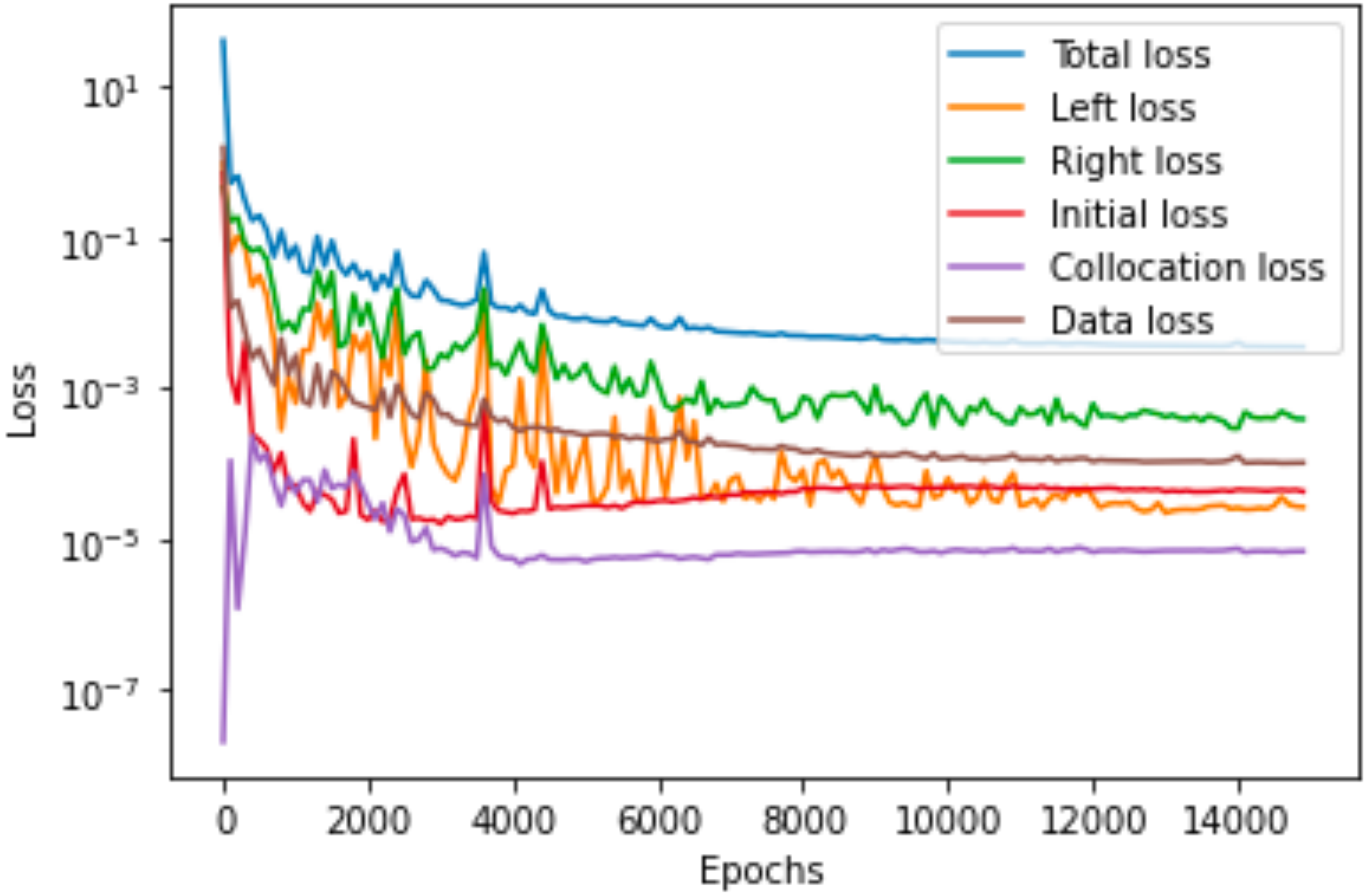


True PDE: $H_t(x,t) = 3(CH^5h_x^3)_x + 0.004 - 0.0002x$

Predicted PDE: $H_t(x,t) = 3(CH^5h_x^3)_x + 0.0043 - 0.00022x$

Not all runs give good results though, Bayesian inference might do wonders.

Task 2 - Inversion



Issues

- We only weakly imposed the physics, so the conservation laws don't hold up to arbitrary precision. One possible solution which has been recently explored is to fix your architectures such that these laws are automatically satisfied.
- Interpretability - huge problem for people who develop models for real-life physics applications
- Use for very complex systems - for example, SICOPOLIS (~20,000 lines FORTRAN code) is a simple ice sheet model for giant ice sheets and it is still highly non-linear and non-local. There are many ice-water, ice-air, sea-air, ice-lithosphere interfaces where jump conditions must be satisfied, and so many physics laws to hold to very high precision.
- One can perhaps only hope to find the parameters of a simplified model using data from a much more high-fidelity model.
- There is a belief in the modeling community (we think) that the ML guys are not honest about their training times and results are not reproducible, leading to skepticism.

PINNs for Learning Basal Mechanics*

- Used time dependent depth-averaged ice velocity **U** and elevation **H**
- Learnt time-invariant and dependent evolution of drag at glacier beds: c_b , $m(x)$
- Shallow Ice Shelf/Stream Approximation (SSA):

In 1D

$$2\frac{\partial}{\partial x}\left[h\eta\frac{\partial u}{\partial x}\right] - \tau_b = \rho_i g h\frac{\partial s}{\partial x},$$

$$\tau_b\left(x,t\right) = c_b\left(x,t\right)|u|^{\frac{1}{m(x)}-1}\ u.$$

In 2D

$$\frac{\partial}{\partial x}\left(2\eta h\left(2\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)\right)+\frac{\partial}{\partial y}\left(\eta h\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right) - \tau_{bx} = \rho_i g h\frac{\partial s}{\partial x},$$

$$\frac{\partial}{\partial y}\left(2\eta h\left(2\frac{\partial v}{\partial y}+\frac{\partial u}{\partial x}\right)\right)+\frac{\partial}{\partial x}\left(\eta h\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\right) - \tau_{by} = \rho_i g h\frac{\partial s}{\partial y},$$

$$\tau_{bx} = c_b\ \|\boldsymbol{u}\|^{\frac{1}{m}}\ \frac{u}{\|\boldsymbol{u}\|},$$

$$\tau_{by} = c_b\ \|\boldsymbol{u}\|^{\frac{1}{m}}\ \frac{v}{\|\boldsymbol{u}\|},$$

- Analyzed 1D SSA simulations and data from Rutford Ice Stream, Antarctica

* Riel, Minchew, Bischoff, JAMES, 2021

Misfits in the loss function

Data misfit (used to parametrize the uncertainty due to noise and model)

- standard deviations for the predictions $\hat{u}$ and $\hat{h}$ in addition to their mean values
- parameterize Gaussian probability distributions (independent for $\hat{h}$ and each component in $\hat{u}$) used to replace the MSE loss function with a negative log-likelihood function

$$\mathcal{L}_{nll}\left(\boldsymbol{\theta}\right) = \frac{1}{M}\sum_{k=1}^{M}\left[-\log p_{\hat{\boldsymbol{u}}^k}(\boldsymbol{u}^k) - \log p_{\hat{h}^k}(h^k)\right],$$

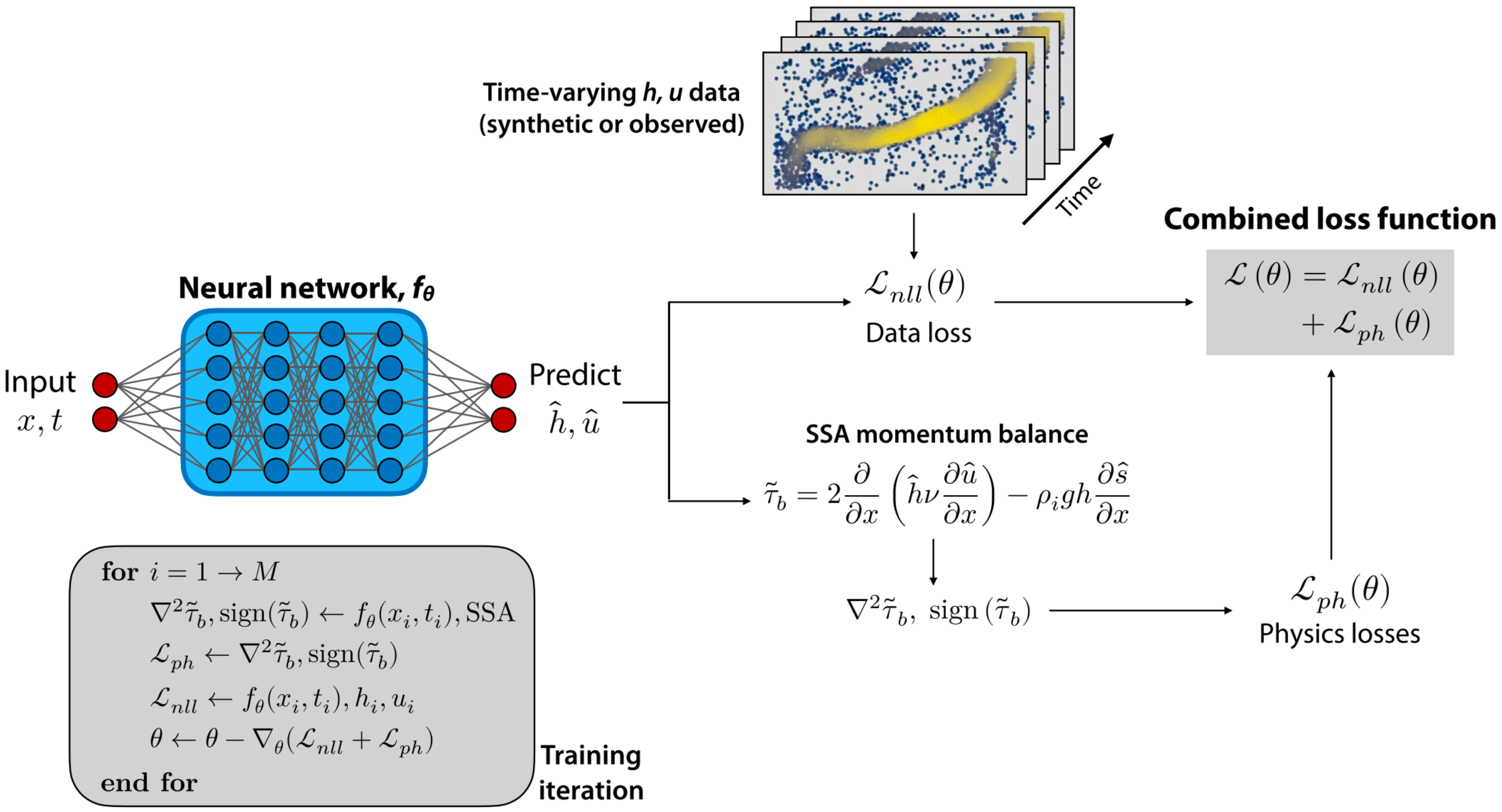
$p_{\hat{u}}$ and $p_{\hat{h}}$ are the likelihood functions for $\hat{u}$ and $\hat{h}$

Physics misfit (spatial roughness, sign of predicted flow drag)

$$\mathcal{L}_{ph}\left(\boldsymbol{\theta}\right) = \frac{1}{P}\sum_{k=1}^{P}\left[\lambda \cdot \left(\frac{\partial^2 \hat{\tau}_b^k}{\partial {x^k}^2} + \frac{\partial^2 \hat{\tau}_b^k}{\partial {y^k}^2}\right)^2 + \alpha \cdot \mathrm{ReLU}\left(\hat{\tau}_b^k\right)\right], \qquad where \qquad \widehat{\tau_b} = \hat{\boldsymbol{\tau}}_b \cdot \frac{\hat{\boldsymbol{u}}}{\|\hat{\boldsymbol{u}}\|}.$$

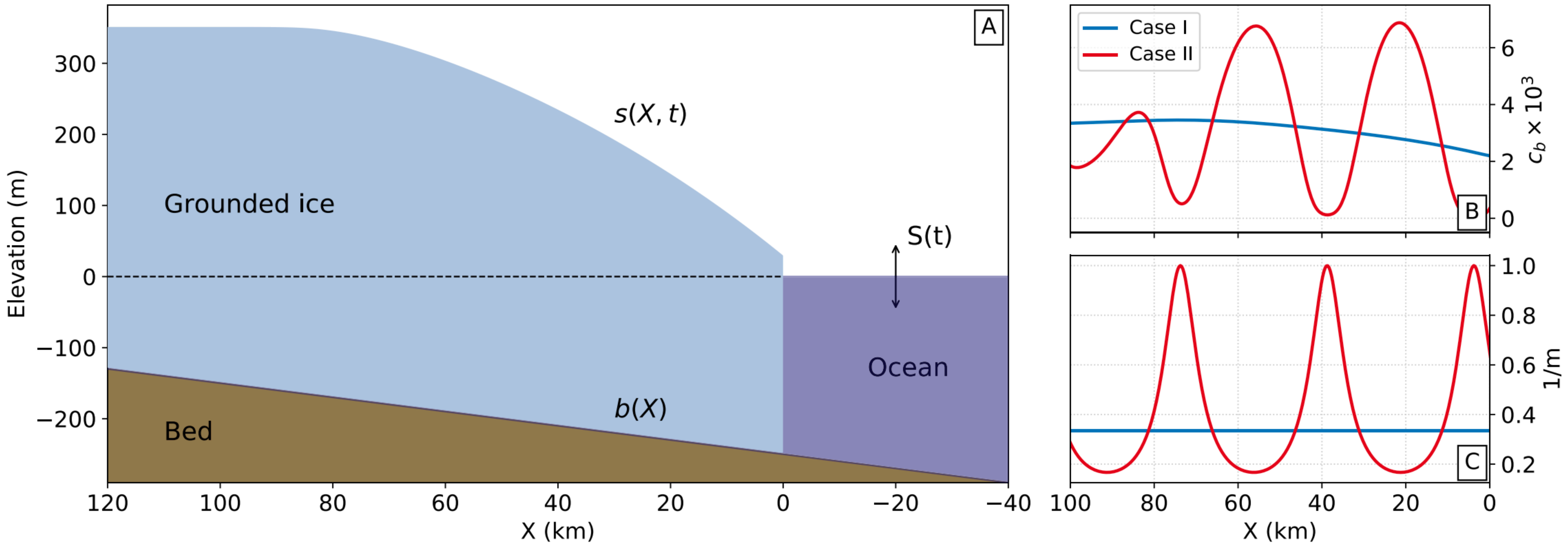
* Riel, Minchew, Bischoff, JAMES, 2021

NN architecture diagram



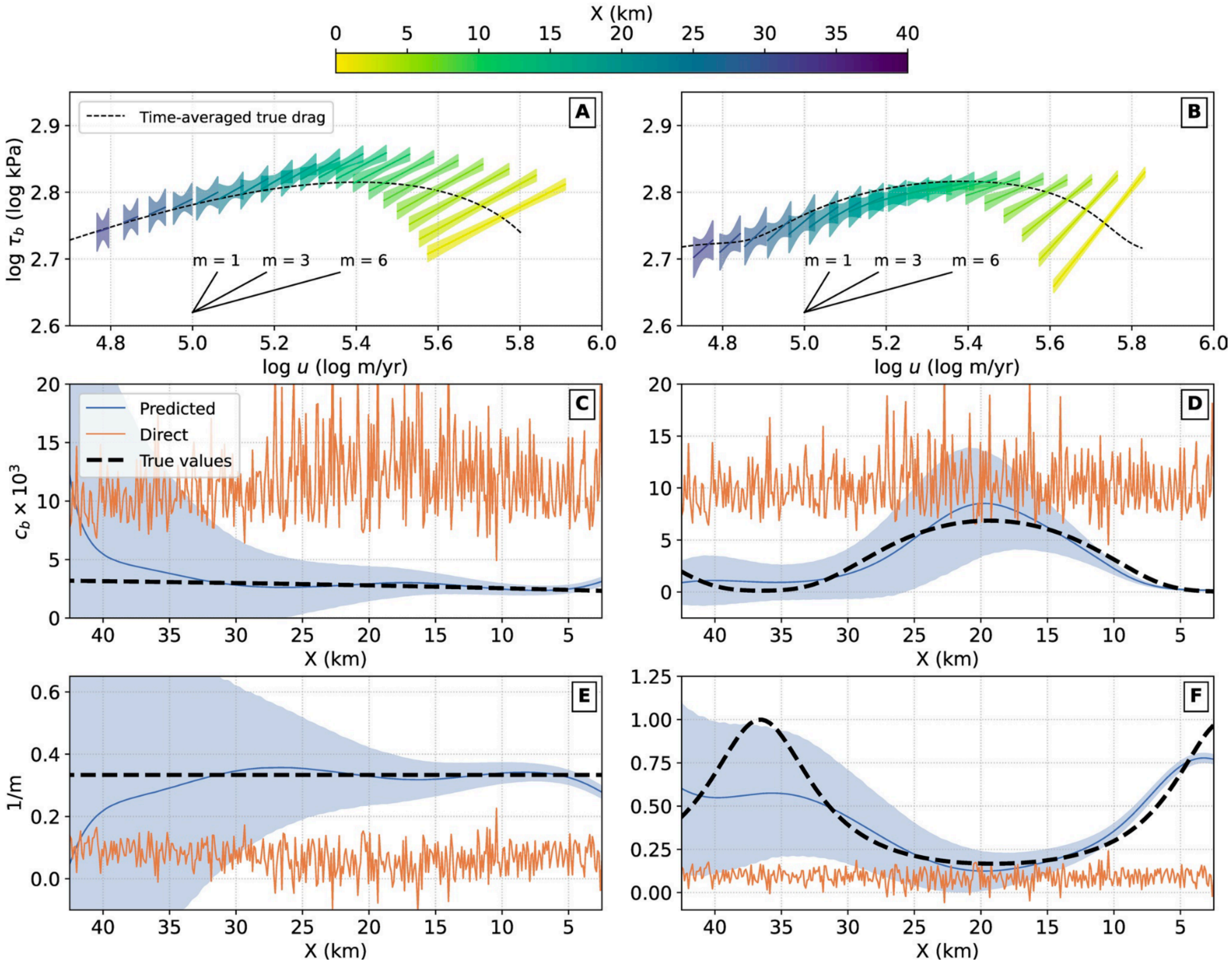
* Riel, Minchew, Bischoff, JAMES, 2021

1 D Problem - Spatially varying drag



* Riel, Minchew, Bischoff, JAMES, 2021

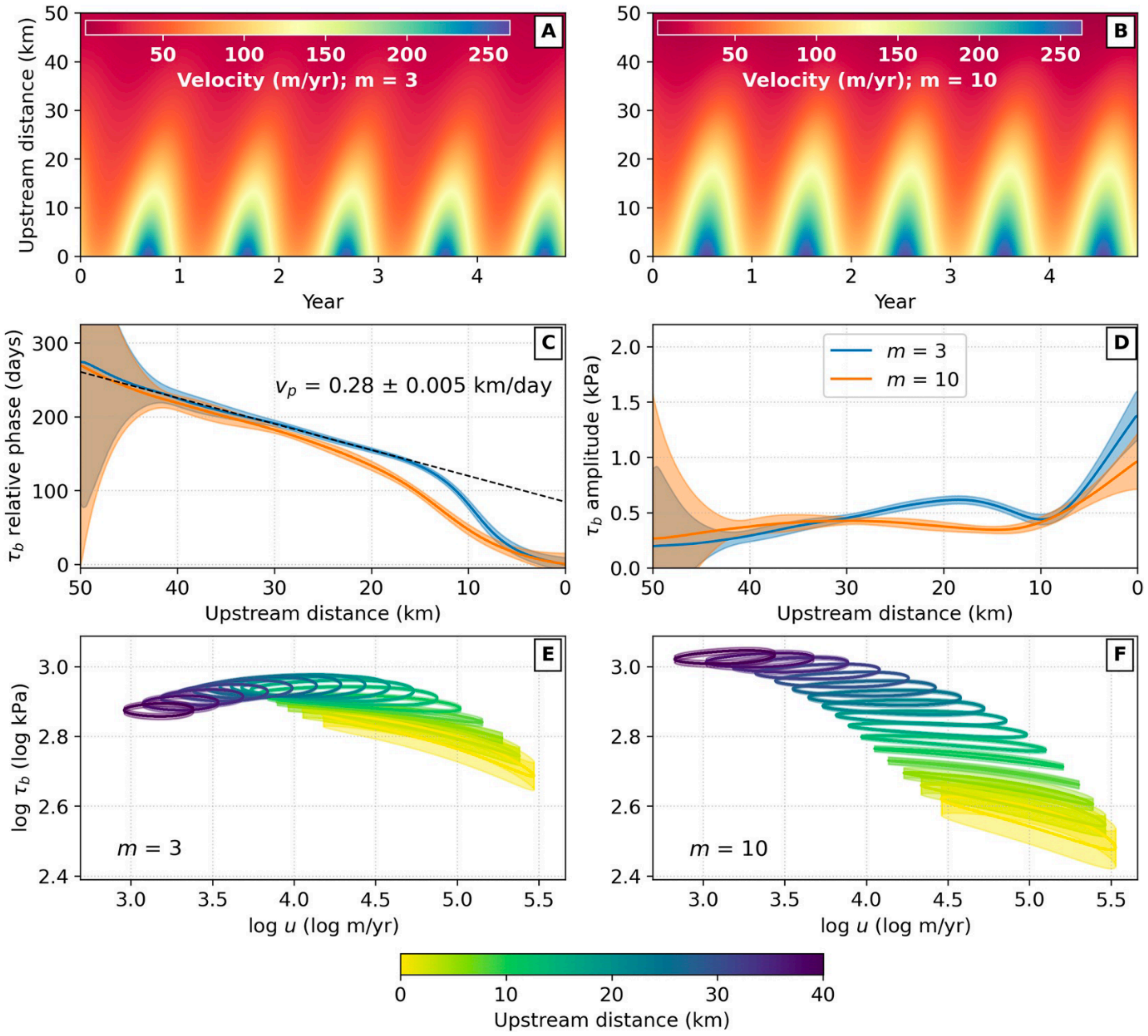
1D Problem - Spatially varying drag results



* Riel, Minchew, Bischoff, JAMES, 2021

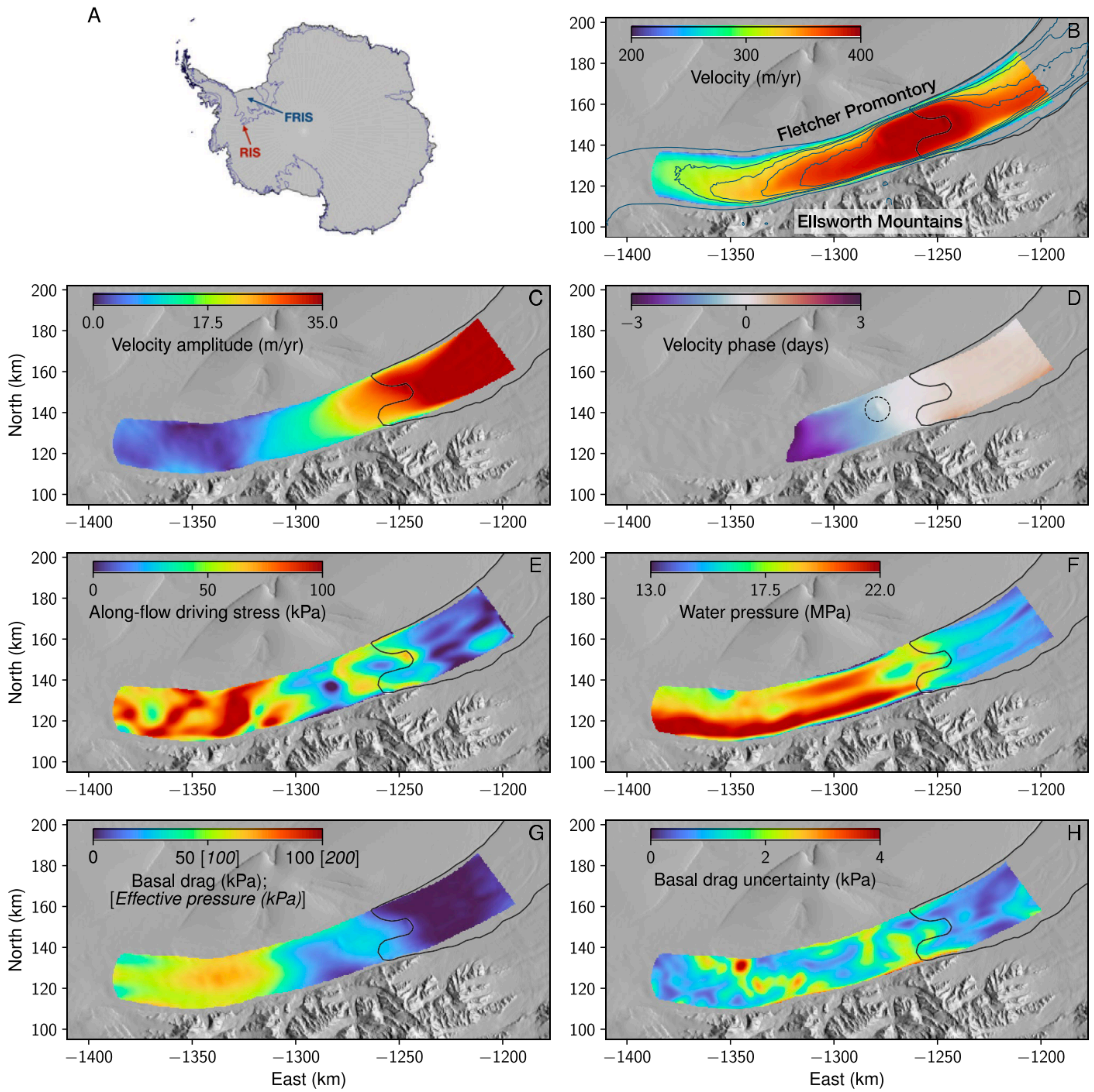
1 D Problem - temporally varying drag results

Power-law sliding law
Basal pressure propagates upstream as a periodic wave



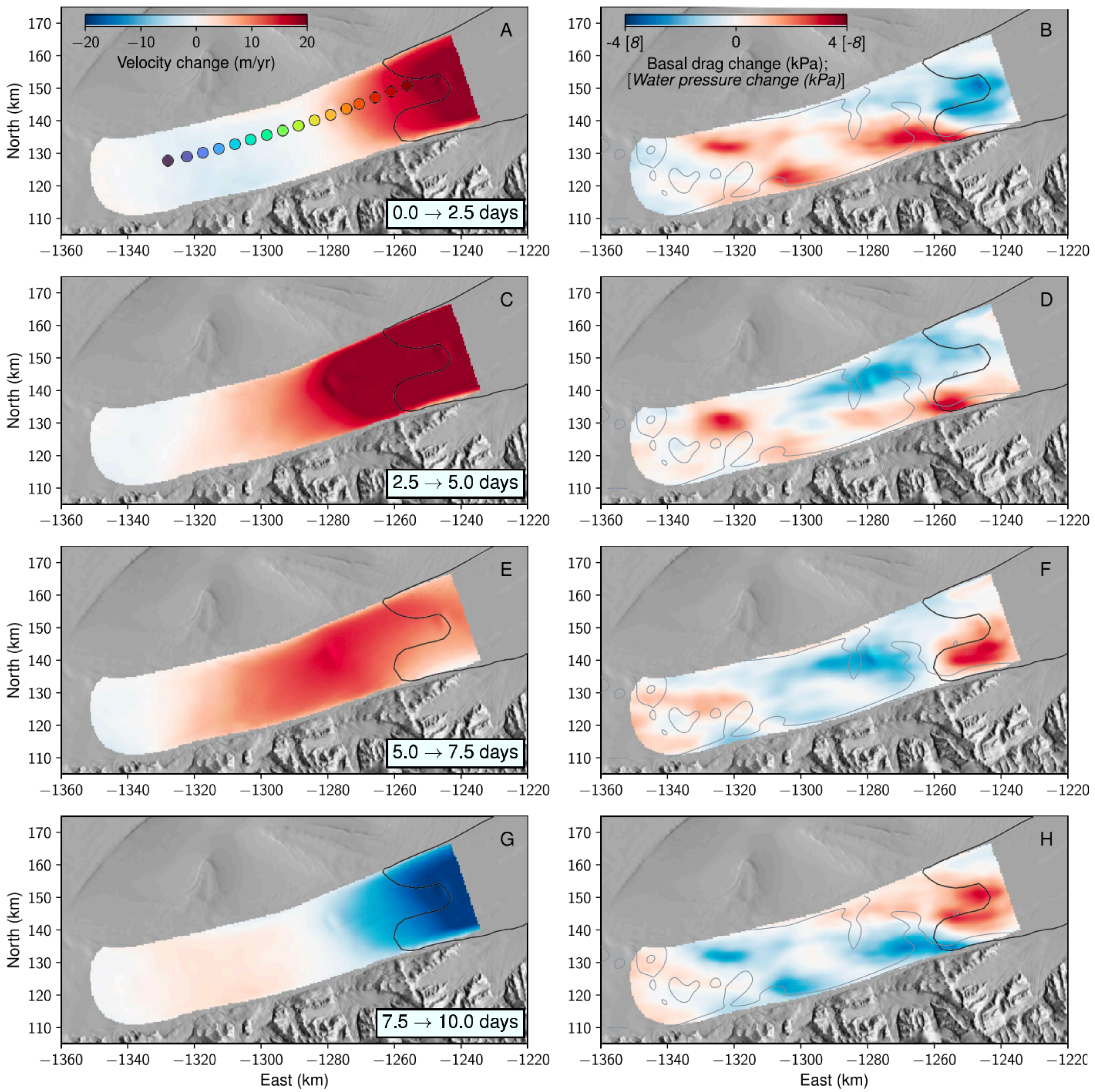
* Riel, Minchew, Bischoff, JAMES, 2021

Rutford Ice Stream, Antarctica - time-average U and drag



* Riel, Minchew, Bischoff, JAMES, 2021

Rutford Ice Stream, Antarctica - time-varying U and drag



* Riel, Minchew, Bischoff, JAMES, 2021

Conclusions

- **Inversion:** Time-invariant but spatially-varying parameters good for simultaneous inversions but not for time-varying parameters
- **Subglacial hydrology matters:** Ocean-tide-driven changes in subglacial water pressure drive changes in ice flow over tidal cycle
- **Utility of PINNs:**
    - Using physics info better constrains the solution
    - Can use more data per image
- **Uncertainty quantification:**
    - Smoothing penalty λ in loss function affects the uncertainties
    - Probabilistic framework is better for UQ

* Riel, Minchew, Bischoff, JAMES, 2021

Future directions (to be discussed)

- Availability of data
- Pressing issues to address
- More than inverse modeling?
- Joint prediction?
- Regularization of data from physics vs data


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS


THE #1 GAME AI DEV EXCUSE FOR LEGITIMATELY SLACKING OFF:
"MY BOT'S TRAINING."
HEY! GET BACK TO WORK!
TRAINING!
OH. CARRY ON.

References

1. Fundamentals of Glacier Dynamics, by CJ van der Veen
2. Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations Raissi, Maziar, Perdikaris, Paris, and Karniadakis, George E *Journal of Computational Physics 2019*
3. Yang, Liu, Meng, Xuhui, and Karniadakis, George E. B-PINNs: Bayesian Physics-informal Neural Networks for Forward and Inverse PDE Problems with Noisy Data. United States: N. p., 2021. Web. doi:10.1016/j.jcp.2020.109913.
4. D.N. Goldberg, K. Snow, P. Holland, J.R. Jordan, J.-M. Campin, P. Heimbach, R. Arthern, A. Jenkins, Representing grounding line migration in synchronous coupling between a marine ice sheet model and a z-coordinate ocean model, *Ocean Modelling*, Volume 125, 2018, Pages 45-60, ISSN 1463-5003, https://doi.org/10.1016/j.ocemod.2018.03.005.
5. Lozier, M.S., 2012. Overturning in the North Atlantic. *Annual Review of Marine Science*, **4**, 291-315.