Advancements in PINNs for Fluid Mechanics
The PirateNet Architecture and Training Strategies

Azzeddine Soulaïmani[1], Hamza Kamil[1,2], Mathieu Mullins[1], Adil Fahsi[3], and Abdelaziz Beljadid[2,4]

Accuracy and Efficiency in Scientific Machine Learning Workshop
Montréal, June 25–27, 2025
Organized by IVADO and CRM

[1]École de technologie supérieure, Montréal, Canada
[2]University Mohamed VI Polytechnic, Benguerir, Morocco
[3]Faculty of Science and Technology, Beni Mellal, Morocco
[4]University of Ottawa, Canada

Table of contents

Introduction

Leverage modern deep learning methods and integrate them with traditional numerical solvers to tackle key challenges in computational mechanics, including:

- High-dimensional, long-time integration, and parameter-sensitive problems (curse of dimensionality);
- Accurate predictions, extrapolations, and forecasting.
- Inverse modeling and parameter identification.
- Uncertainty quantification;
- Scalability and computational efficiency.

Plain PINN pathologies

Plain PINNs as first implemented by (Raissi et al., 2019).
Loss function:

$$\mathcal{L}(\theta) = \lambda_{ic}\mathcal{L}_{ic}(\theta) + \lambda_{bc}\mathcal{L}_{bc}(\theta) + \lambda_{r}\mathcal{L}_{r}(\theta) + \lambda_{data}\mathcal{L}_{data}(\theta), \tag{1}$$

Loss terms:

$$\mathcal{L}_{ic}(\theta) = \frac{1}{N_{ic}} \sum_{i=1}^{N_{ic}} |\boldsymbol{u}_\theta(x_{ic}^i, 0) - g(x_{ic}^i)|^2, \tag{2}$$

$$\mathcal{L}_{bc}(\theta) = \frac{1}{N_{bc}} \sum_{i=1}^{N_{bc}} |\mathcal{B}[\boldsymbol{u}_\theta](x_{bc}^i, t_{bc}^i)|^2, \tag{3}$$

$$\mathcal{L}_{r}(\theta) = \frac{1}{N_{r}} \sum_{i=1}^{N_{r}} |\mathcal{R}_\theta(x_r^i, t_r^i)|^2, \tag{4}$$

$$\mathcal{L}_{data}(\theta) = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \left| \boldsymbol{u}_\theta(x_{data}^i, t_{data}^i) - \hat{\boldsymbol{u}}^i \right|^2, \tag{5}$$

Plain PINN paradigm

- Loss function terms imbalance
- Trouble with high-frequency components
- Training instabilities
- Performance loss with deeper networks
- Temporal causality violation

PINN Training Strategies

1. **Random Fourier Features** (Tancik et al., 2020): A technique to improve the learning of high-frequency functions by mapping input coordinates to a higher-dimensional space using random Fourier basis functions.

2. **Loss Balancing** (Wang et al., 2023): A method to dynamically balance multiple loss components during training.

3. **Random Weight Factorization** (Wang, Wang, et al., 2022): A normalization that factorizes weight matrices using random projections.

4. **Adaptive sampling** (Wang, Sankaran, and Perdikaris, 2022).

5. **Causal Training** (Wang, Sankaran, and Perdikaris, 2022): A training strategy that enforces causal ordering in temporal domains, where the model is trained sequentially in time.

6. **Non-dimensionalization of the PDE**.

PINN Training strategies

1. Sequence-to-sequence (Krishnapriyan et al., 2021): (can be complementary to causal training)

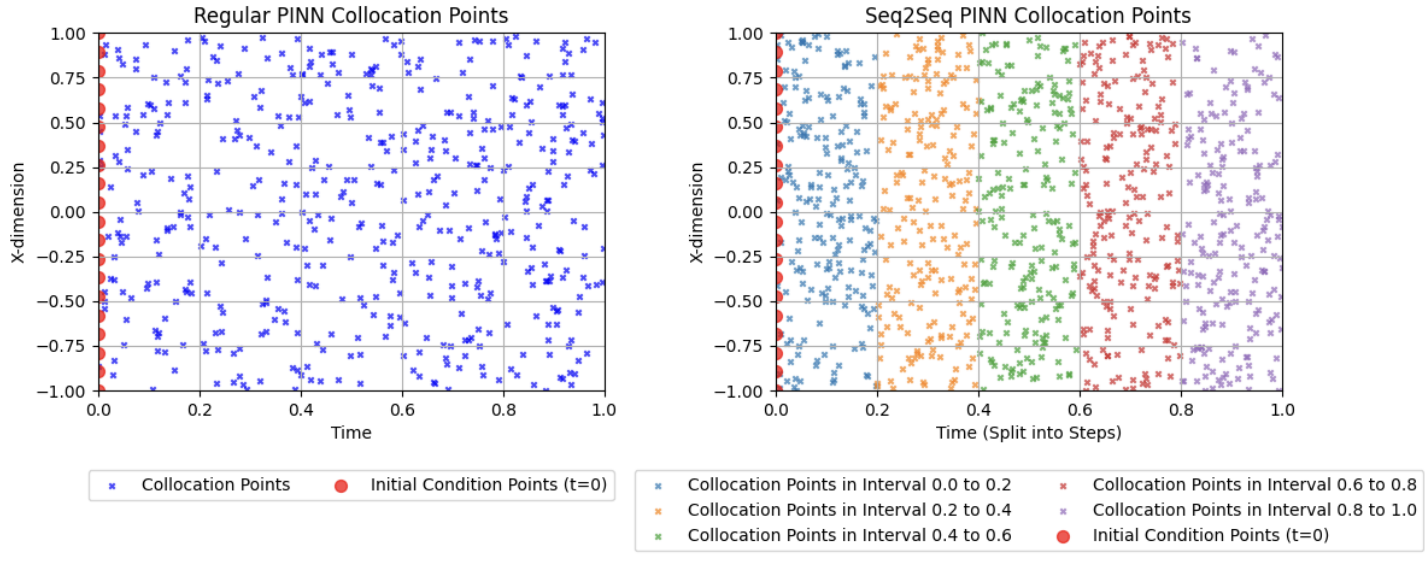


Figure: Sequence-to-sequence training

2. Curriculum training (Krishnapriyan et al., 2021):
   - Transfer learning based
   - Gradual increase in problem difficulty (ex: Reynolds number)

Optimizers

1. Sequential Optimization using Adam and L-BFGS.
   - Adam aids in escaping saddle points, while L-BFGS ensures fast local convergence.
   - This hybrid approach mitigates ill-conditioning in PINN training (Rathore et al., 2024).
2. Shampoo with Adam on the preconditioner's eigenbasis (SOAP) (Vyas et al., 2025).
   - Quasi second-order method.
   - Addresses conflicts in gradient alignement of composite loss functions.

Advanced Architecture

Physics-Informed AdapTivE Networks (PirateNets, Wang et al., 2024): Mitigate the gradient stiffness, use of a suitable initialization scheme, enable deeper networks.

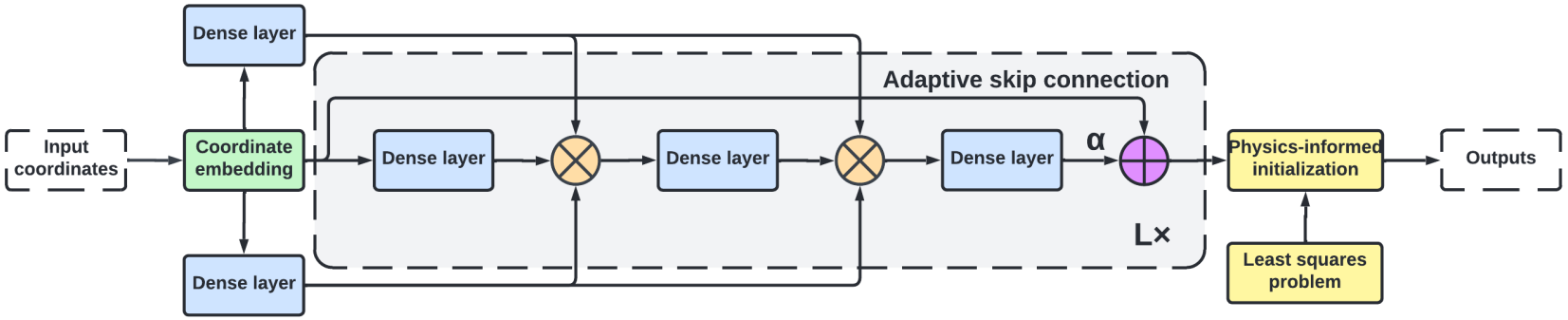


Figure: PirateNet architecture

$$
\begin{aligned}
f^{(l)} &= \sigma(\boldsymbol{W}_1^{(l)}\boldsymbol{x}^{(l)} + \boldsymbol{b}_1^{(l)}), && \text{(Dense layer)}\\
z_1^{(l)} &= f^{(l)} \odot U + (1 - f^{(l)}) \odot V, && \text{(Gating operation)}\\
g^{(l)} &= \sigma(\boldsymbol{W}_2^{(l)} z_1^{(l)} + \boldsymbol{b}_2^{(l)}), && \text{(Dense layer)}\\
z_2^{(l)} &= g^{(l)} \odot U + (1 - g^{(l)}) \odot V, && \text{(Gating operation)}\\
h^{(l)} &= \sigma(\boldsymbol{W}_3^{(l)} z_2^{(l)} + \boldsymbol{b}_3^{(l)}), && \text{(Dense layer)}\\
\boldsymbol{x}^{(l+1)} &= \alpha^{(l)} h^{(l)} + (1 - \alpha^{(l)})\boldsymbol{x}^{(l)}, && \text{(Adaptive skip connection)}\\
f_\theta(\mathbf{x}) &= \mathbf{W}^{(L+1)} \cdot \mathbf{x}^{(L)}, && \text{(Final output)}.
\end{aligned}
$$

Flows in Unsaturated soils

Modeling water flow and solute transport in unsaturated soils is key for optimizing irrigation and effective fertilizer management.

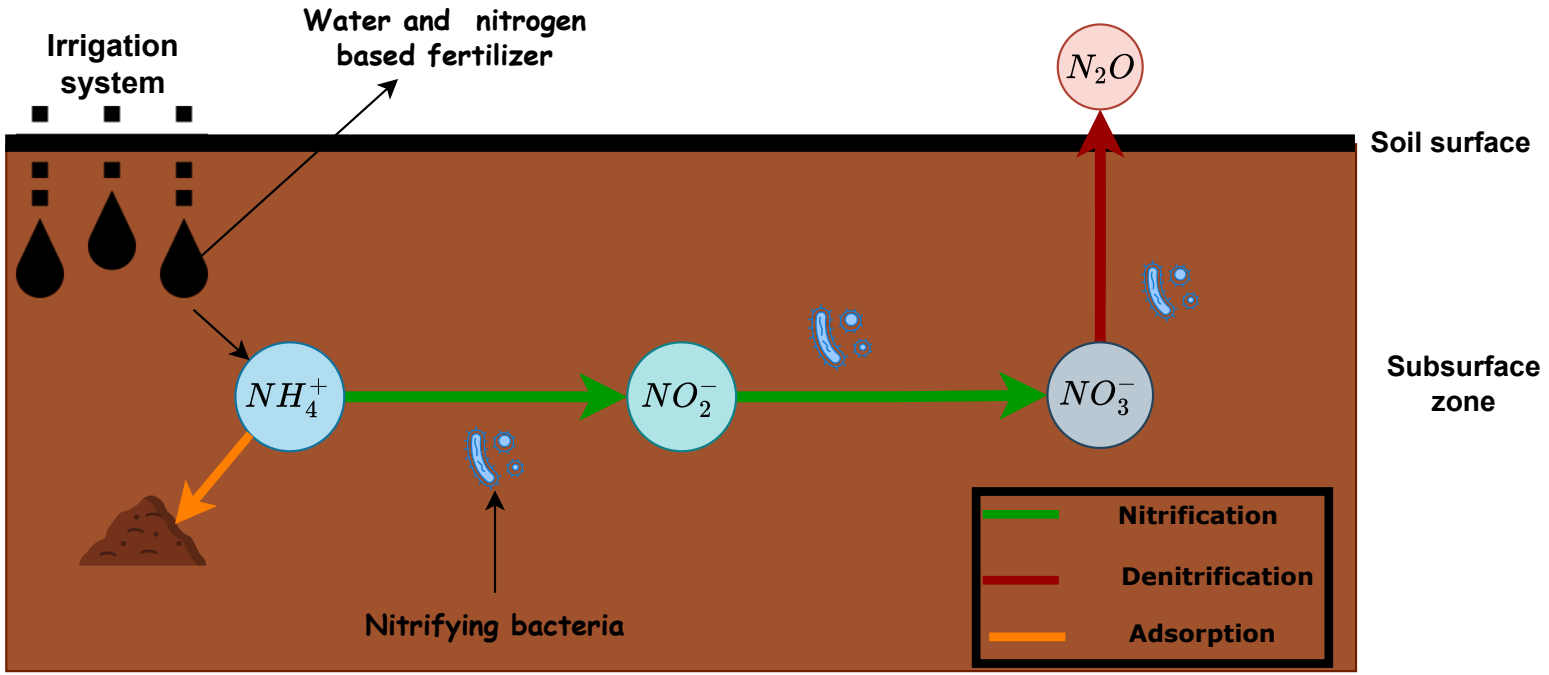


Figure: Illustration of the nitrogen species first-order decay chain (Kamil, Soulaïmani, and Beljadid, 2025).

Governing equations

Modeling water flow and solute transport in unsaturated soils involves solving a complex multi-physics system of coupled PDEs.

Water infiltration

$$\begin{cases} \dfrac{\partial \theta}{\partial t} + \nabla \cdot \mathbf{q} = 0, \\ \mathbf{q} = -K\nabla(\Psi + z), \end{cases} \tag{6}$$

Solute species

$$\begin{cases} \dfrac{\partial R\theta c_1}{\partial t} = \nabla \cdot (\theta \mathbf{D} \nabla c_1 - c_1 \mathbf{q}) - \mu_1 \theta c_1, \text{ Ammonium} \\ \dfrac{\partial \theta c_2}{\partial t} = \nabla \cdot (\theta \mathbf{D} \nabla c_2 - c_2 \mathbf{q}) + \mu_1 \theta c_1 - \mu_2 \theta c_2, \text{ Nitrite} \\ \dfrac{\partial \theta c_3}{\partial t} = \nabla \cdot (\theta \mathbf{D} \nabla c_3 - c_3 \mathbf{q}) + \mu_2 \theta c_2 - \mu_3 \theta c_3, \text{ Nitrate} \end{cases} \tag{7}$$

System (6) contains five parameters to calibrate, while System (7) involves at least six parameters.

Transfer learning training

We propose solving the fully coupled model using a sequential transfer learning approach, enabling more efficient and accurate training than standard PINNs (Kamil et al., 2024).

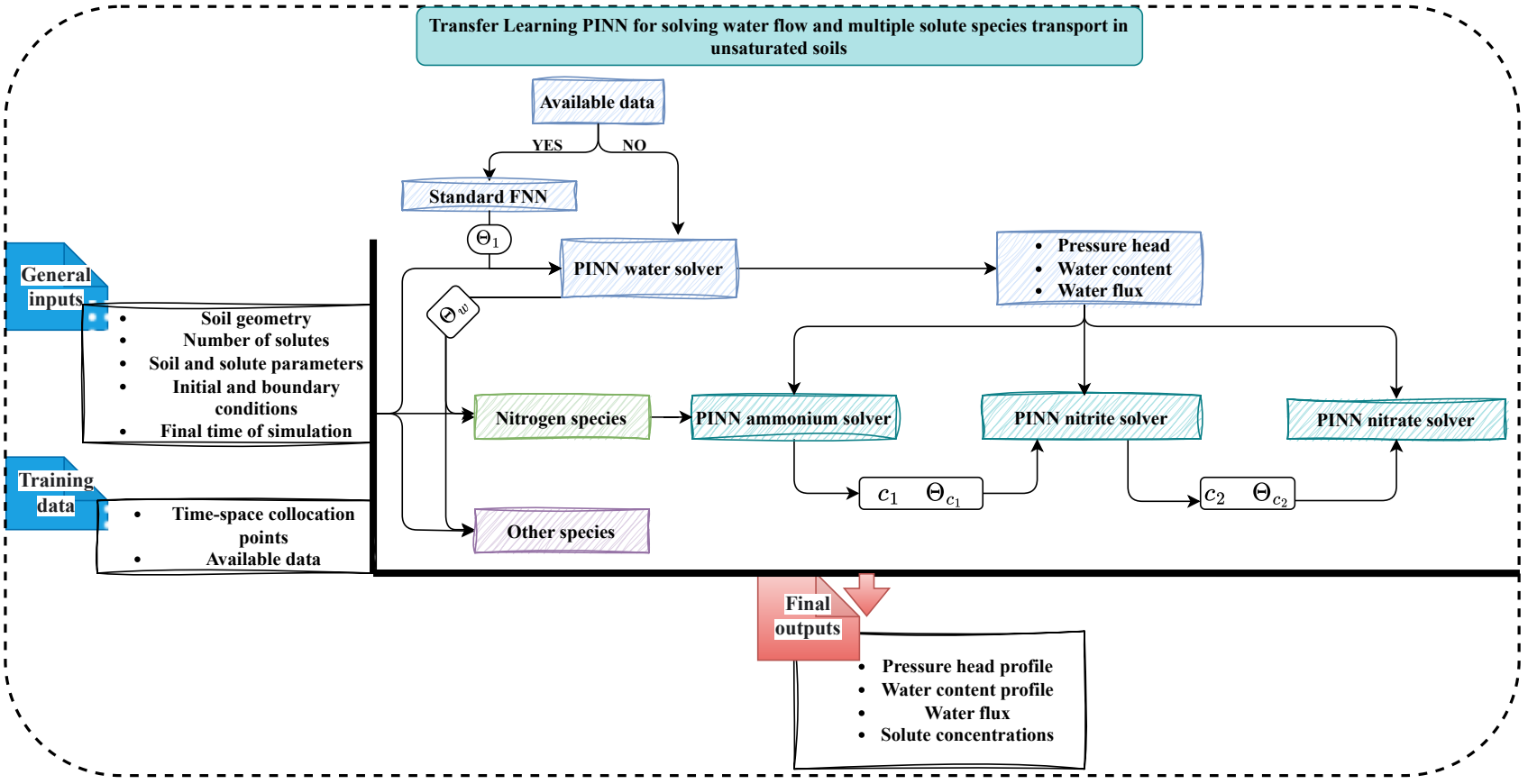
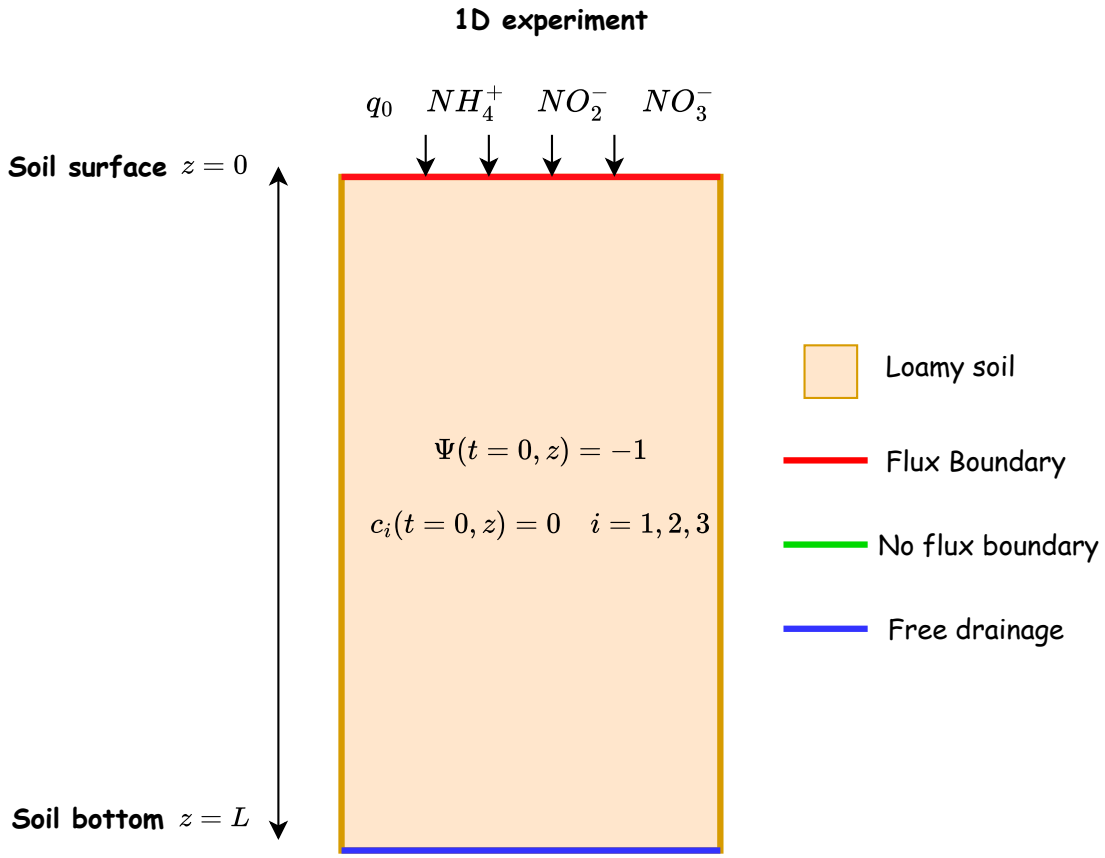


Figure: Transfer Learning PINN (Kamil et al., 2024).

Test 1

We consider a dry soil column with zero initial solute concentration. The TL-PINN is then applied to solve the full system.



Test 1

Results.

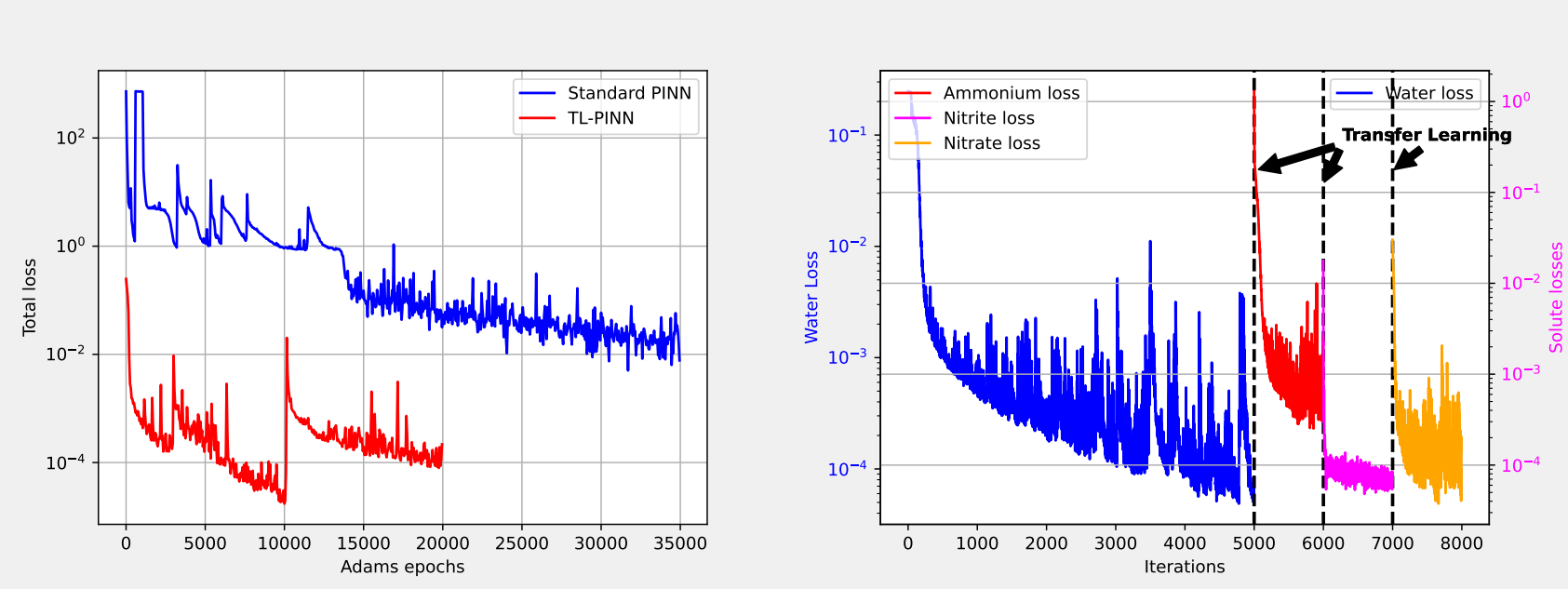
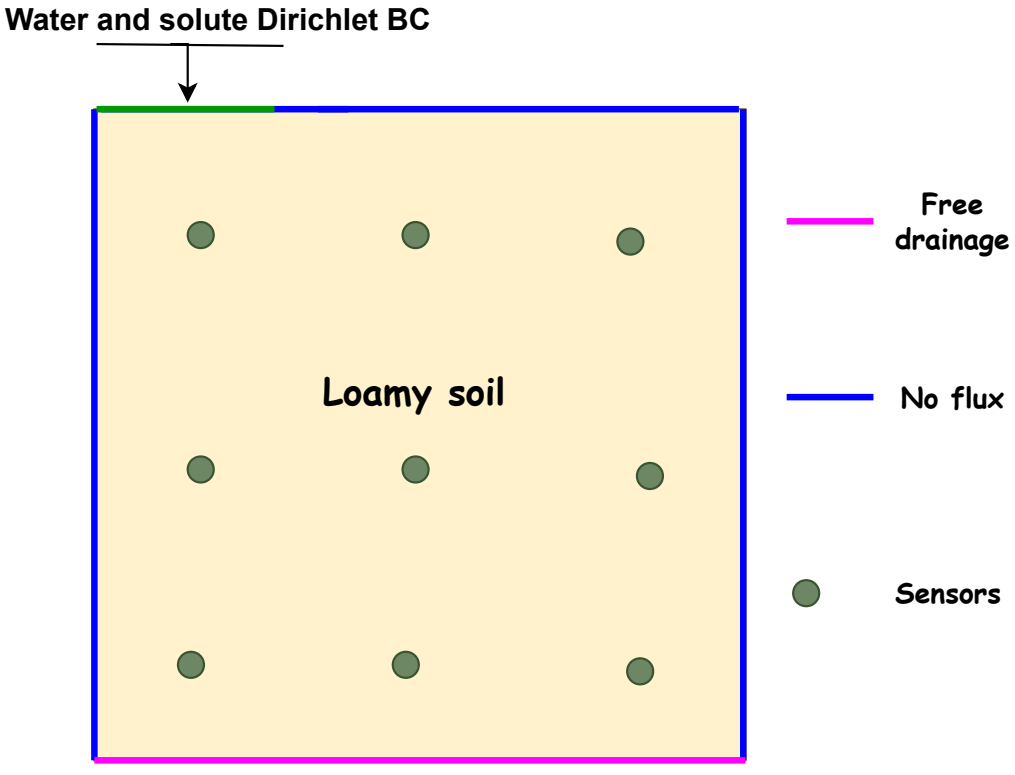


Figure: PINN vs TL-PINN (Kamil et al., 2024).

Test Case 2– TL-PINN Validation

We apply the TL-PINN approach to a real case by validating it against experimental data from (Kamil, Soulaïmani, and Beljadid, 2025). The following setup illustrates the 2D soil domain used in the simulation:



Test Case 2 – Results

Figure: TL-PINN compared with experimental data (Kamil, Soulaïmani, and Beljadid, 2025).

Remark

The TL-PINN shows good agreement with the reference data; however, calibrating the model parameters is necessary to achieve better alignment with the experimental results.

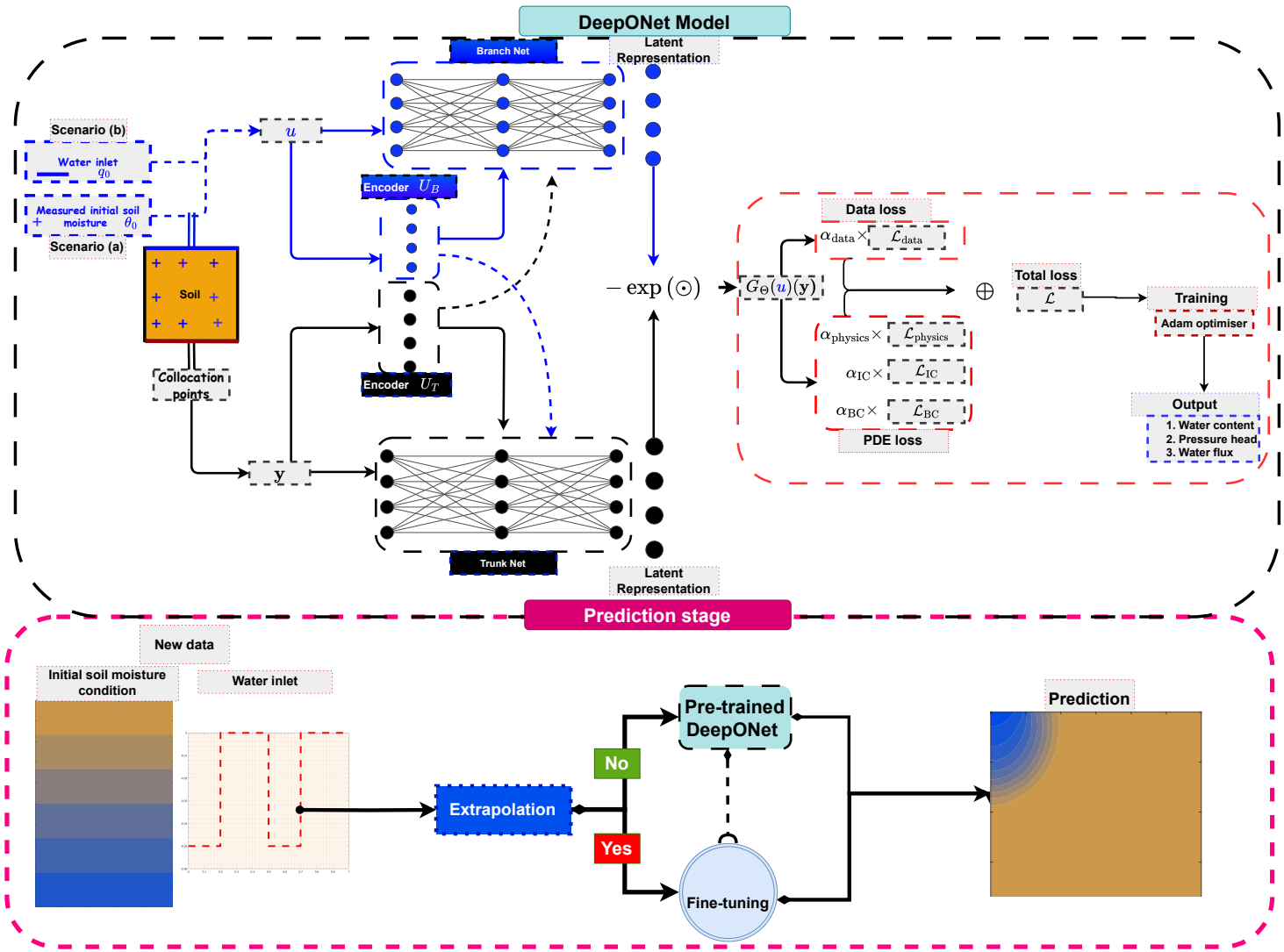
DeepONet for Multi-Query Tasks

Efficient irrigation design requires repeated solutions of the water flow equation under varying parameters. We propose the following approach (Kamil, Soulaımani, and Beljadid, 2025):

1. We develop a fast surrogate model to **multitask** water infiltration in soils.
2. The adopted approach is a Physics-Informed DeepONet that learns a mapping from boundary conditions to the spatiotemporal water distribution.
3. The model can also handle **extrapolation** scenarios. The core idea is to *fine-tune* the pretrained model using the available data when the inputs lie outside the training distribution.
4. The surrogate model is used alongside an optimizer to **calibrate** soil hydraulic parameters, with five parameters per soil layer.

Physics-informed DeepONet architecture

We employed the modified MLP architecture as the backbone of the PI-DeepONet framework.



Extrapolation

Test case. We train the DeepONet on initial water content conditions that are constant in space, sampled using Sobol sequences within the interval $\theta_0 \in (0.13, 0.43)$. We then test its performance on a linearly varying initial condition.

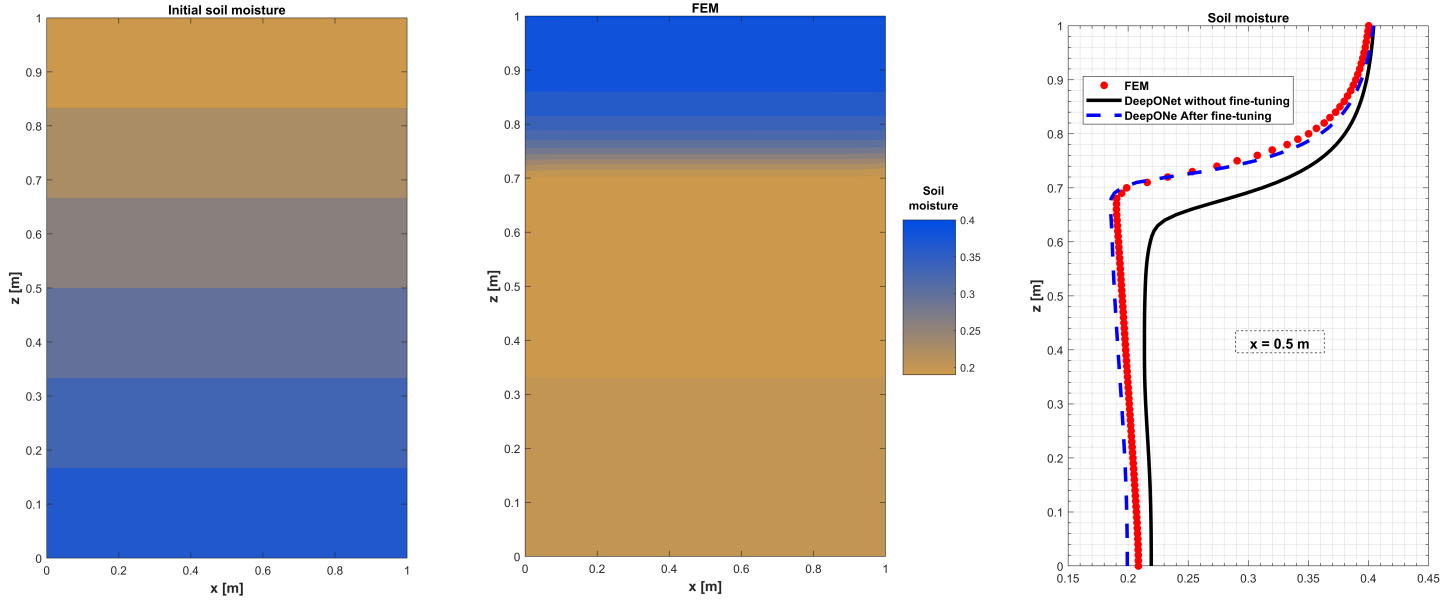
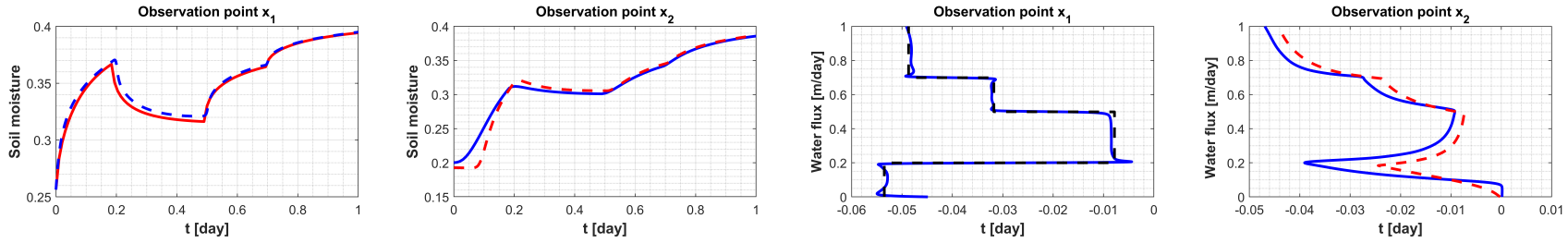


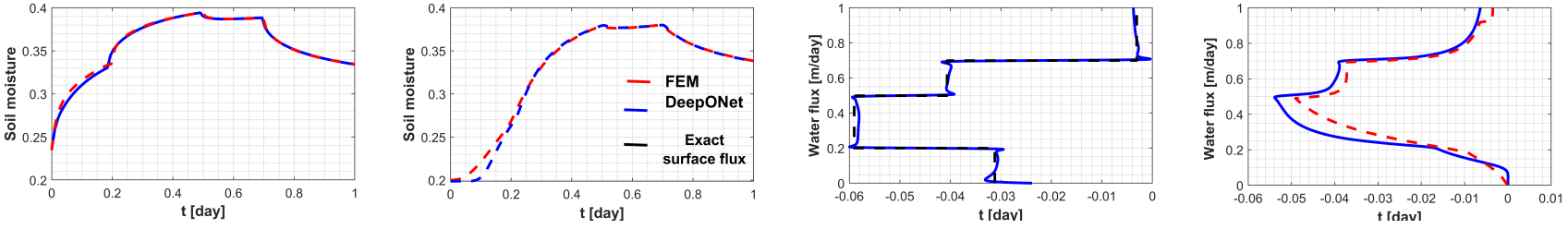
Figure: Results from DeepONet in an extrapolation scenario for loam soil, with and without fine-tuning. The initial soil moisture varies linearly from the surface to the bottom.

Extrapolation validation

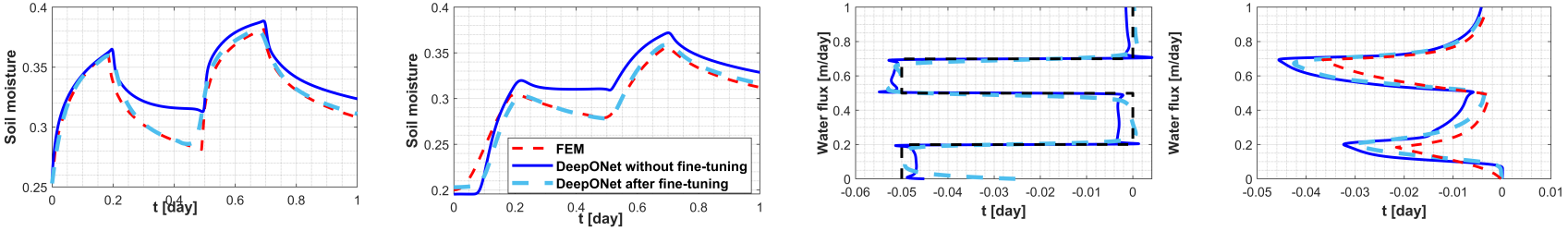
(a) Train sample



(b) Test sample



(c) Extrapolation sample



Parameters' calibration

Calibration of 5 soil parameters using 9 measurement sensors. Spatio-temporal data are obtained using a FEM solver. The DeepOnet is used as a surrogate by the optimizer.

Parameter	Exact Value	DeepONet	PINN	Data-driven MLP
θ_s	0.430	0.432	0.435	0.448
θ_r	0.078	0.079	0.082	0.090
K_s (m/day)	0.250	0.252	0.259	0.275
α_v (1/m)	3.60	3.58	3.62	3.75
n_v	1.560	1.558	1.562	1.585

Table: Comparison of predicted parameter values.

Approach	Fitness value	Absolute error
DeepONet	$\mathbf{1.2 \times 10^{-3}}$	**0.65%**
PINN	1.8×10^{-3}	2.11%
Data-driven MLP	3.5×10^{-3}	7.07%

Table: Fitness value and absolute error for each approach.

$$\text{Absolute error } (\%) = \frac{1}{5} \sum_{i=1}^{5} \left| \frac{\lambda_i - \lambda_i^{\text{exact}}}{\lambda_i^{\text{exact}}} \right| \times 100$$

Recommendations

Fourier embeddings and adaptive activations
These techniques significantly improve
training efficiency and model accuracy.

Optimizer has greater impact than architecture
Our experience shows optimization plays
a more critical role than network design.

SOAP optimizer outperforms Adam
The quasi-Hessian SOAP optimizer performed better on
fourth-order infiltration models (results not yet published).

Level set method

Temporal PDE :

$$\frac{\partial \phi}{\partial t} + \boldsymbol{u} \cdot \nabla \phi = 0,$$

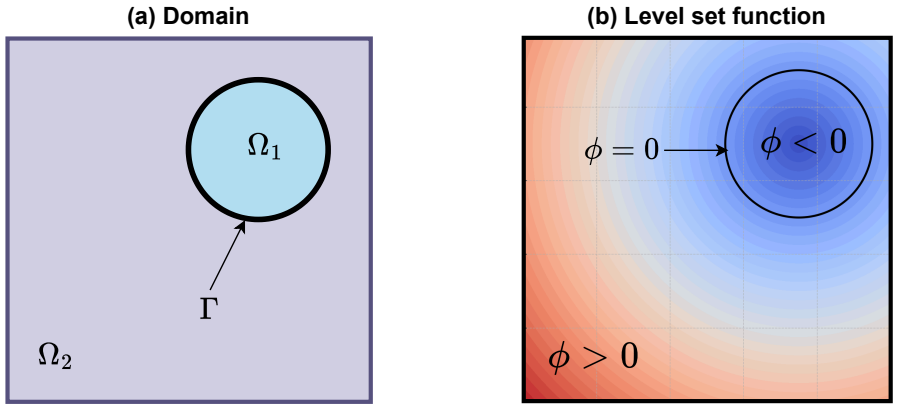


Figure: Two-phase domain

Initialisation :

$$d_{\text{signed}}(\boldsymbol{x}, t) = \begin{cases} -d(\boldsymbol{x}, t) & \text{if } \boldsymbol{x} \in \Omega_1, \\ 0 & \text{if } \boldsymbol{x} \in \Gamma, \\ +d(\boldsymbol{x}, t) & \text{if } \boldsymbol{x} \in \Omega_2, \end{cases}$$

with $d(\boldsymbol{x}, t)$ a signed distance function with regards to interface Γ :

$$d(\boldsymbol{x}, t) = \min_{\boldsymbol{x}_\Gamma \in \Gamma} \|\boldsymbol{x} - \boldsymbol{x}_\Gamma\|.$$

Ablation study: configurations

PINN frameworks tested

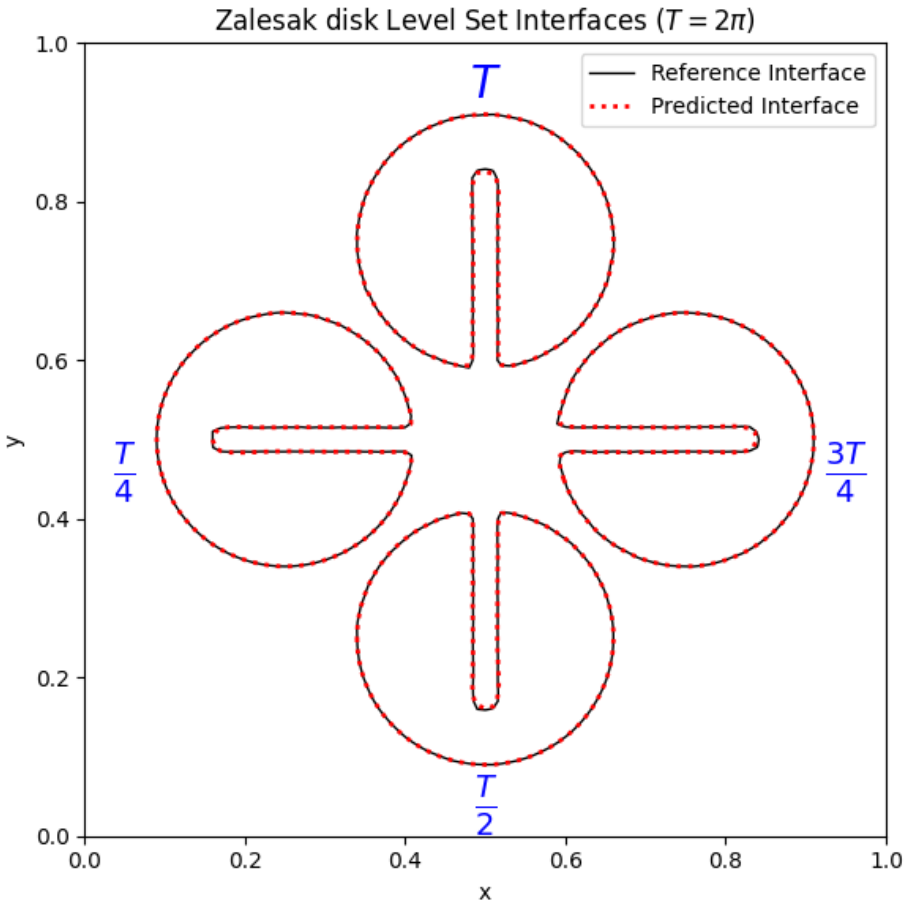
- **Plain** Raissi et al., 2019
- **Default** Wang et al., 2023
- **PirateNet** Wang et al., 2024
- **Sota**

Tested problems:

- **Zalesak's disk**
- **Time-reversed vortex flow**

Configuration	R. W. Factorization	Causal Training	Weights balancing	R. Fourier embending	S2S	Architecture
Plain						MLP
Default	x	x	x	x		Modified MLP
PirateNet	x	x	x	x		PirateNet
Sota*	x	x	x	x	x	PirateNet

Zalesak's disk



Time-reversed flow field:

$$u(x, y) = \frac{2\pi}{T}(0.5 - y)$$

$$v(x, y) = \frac{2\pi}{T}(x - 0.5)$$

Configuration	L^2 error (%)
Plain	4.18
Default	2.96
PirateNet	0.35
Sota	**0.14**

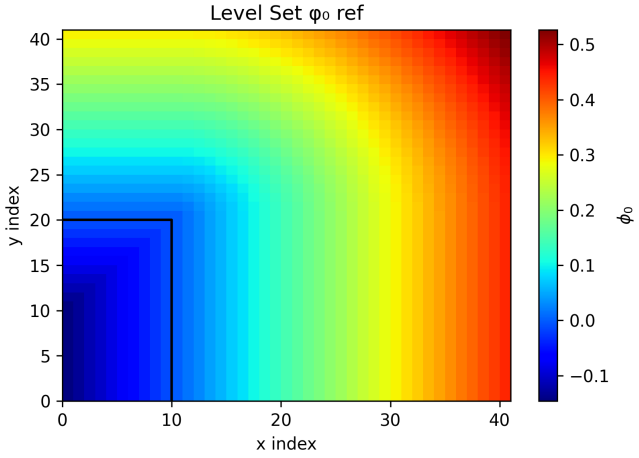
Figure: Zalesak's disk: Interface evolution for *reference* and *Sota* ($T = 2\pi$ s)

Coupled level set - Navier-Stokes

$$\mathcal{R}_\phi = \frac{\partial \phi^*}{\partial t^*} + u^* \frac{\partial \phi^*}{\partial x^*} + v^* \frac{\partial \phi^*}{\partial y^*}$$

$$\mathcal{R}_{\text{cont}} = \frac{\partial u^*}{\partial x^*} + \frac{\partial v^*}{\partial y^*}$$

$$\mathcal{R}_{\text{mom-x}} = \frac{\partial u^*}{\partial t^*} + u^* \frac{\partial u^*}{\partial x^*} + v^* \frac{\partial u^*}{\partial y^*} + \frac{1}{\rho^*(\phi)} \frac{\partial p^*}{\partial x^*} - \frac{1}{\text{Re}} \cdot \frac{\mu^*(\phi)}{\rho^*(\phi)} \left[\frac{\partial^2 u^*}{\partial (x^*)^2} + \frac{\partial^2 u^*}{\partial (y^*)^2} \right] - g_x^*$$



Non-dimensionalization variables:

$$x^* = \frac{x}{L}, \quad y^* = \frac{y}{L}, \quad t^* = \frac{t}{T} = t\sqrt{\frac{g}{L}}$$

$$u^* = \frac{u}{U} = \frac{u}{\sqrt{gL}}, \quad v^* = \frac{v}{U} = \frac{v}{\sqrt{gL}}$$

$$p^* = \frac{p}{P} = \frac{p}{\rho_{\text{ref}} gL}, \quad \phi^* = \frac{\phi}{L}$$

$$\rho^*(\phi) = \frac{\rho(\phi)}{\rho_{\text{ref}}}, \quad \mu^*(\phi) = \frac{\mu(\phi)}{\mu_{\text{ref}}}$$

SWE 1D: Flow over a bump

Continuity:

$$\frac{\partial h}{\partial t} + \bar{u}\frac{\partial h}{\partial x} + h\frac{\partial \bar{u}}{\partial x} = 0.$$

Momentum:

$$\frac{\partial u}{\partial t} + \bar{u}\frac{\partial \bar{u}}{\partial x} + \frac{1}{\mathrm{Fr}^2}\frac{\partial h}{\partial x} = -\frac{1}{\mathrm{Fr}^2}\frac{\partial b}{\partial x}$$

with:

- $x \in [-1, 1]$
- $t \in [0, 1]$
- Dimensional ref values: $g = 3.5$, $U = 3.5$, $L = 2$, $H = 1$
- Implies $Fr = 1.8$
- $s = h + b$

IC and BC:

$$b(x) = 0.8 \cdot \exp\left(\frac{-x^2}{0.2^2}\right) - 1$$

$$h(x, 0) = 0.2 \cdot \exp\left(\frac{-(x + 0.4)^2}{0.2^2}\right) - b(x)$$

$$u(x, 0) = 0$$

$$u(x_0, t) = u(x_1, t) = 0$$

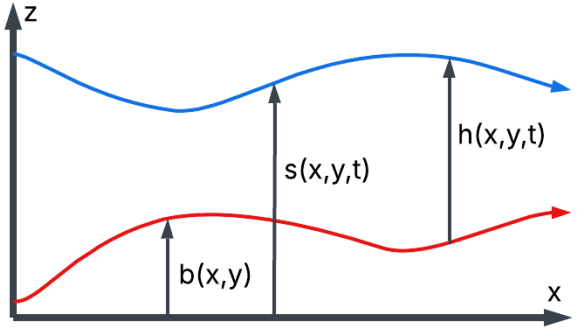


Figure: SWE notation

SWE 1D: Flow over a bump

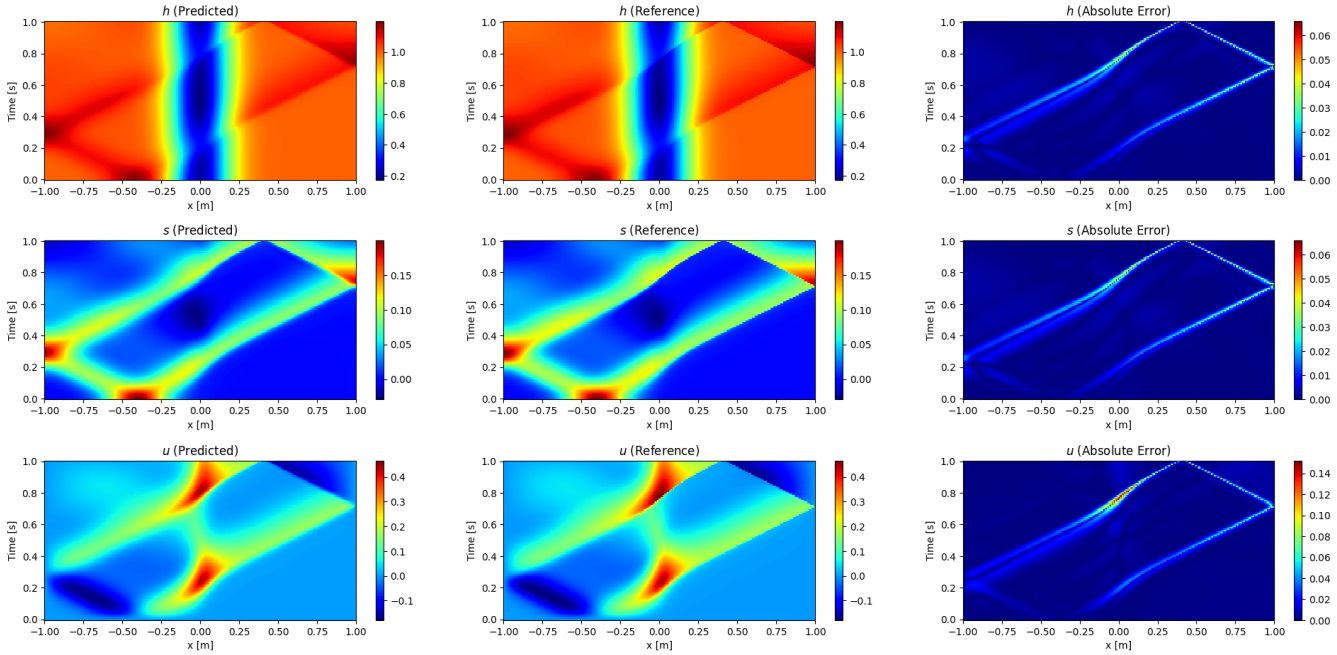


Figure: Prediction (left), reference (middle), absolute error (right) with *PirateNet*

Configuration name	$L_h^2(\%)$	$L_u^2(\%)$	$MAPE_{mass}$ (%)
Plain	0.65	13.33	0.027
Default	0.60	11.73	**0.010**
PirateNet	0.52	10.19	**0.013**
Plain*	0.52	10.85	0.024
Default*	**0.30**	**6.36**	0.025
PirateNet*	**0.30**	**6.53**	0.021

Hyperparameter	**Possible values**
Architecture	MMLP, PirateNet
Layers	8
Optimizer	Adam
Activation function	Tanh, Gelu, Swish
Fourier feature scale	1, 1.5, 2
RWF mean	0.5, 1
Causal tolerance	1, 1.5, 2

SWE 1D: Flow over a bump

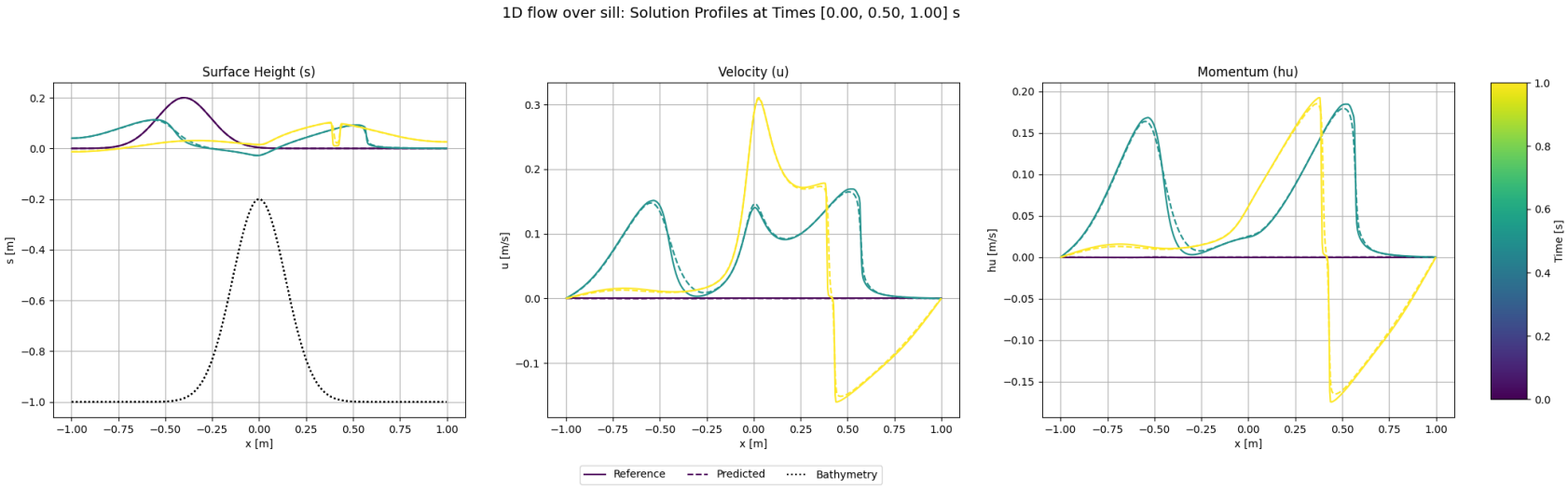


Figure: SWE 1D : Free surface elevation (left), velocity (mid) and momentum (right) for *PirateNet* at $t = [0, 0.5, 1.0]$s

SWE 2D: Radial dam break

Continuity:

$$\frac{\partial h}{\partial t} + \bar{u}\frac{\partial h}{\partial x} + \bar{v}\frac{\partial h}{\partial y} + h\left(\frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y}\right) = 0$$

Momentum:

$$\frac{\partial \bar{u}}{\partial t} + \bar{u}\frac{\partial \bar{u}}{\partial x} + \bar{v}\frac{\partial \bar{u}}{\partial y} + \frac{1}{\mathrm{Fr}^2}\frac{\partial h}{\partial x} = -\frac{1}{\mathrm{Fr}^2}\frac{\partial b}{\partial x}$$

$$\frac{\partial \bar{v}}{\partial t} + \bar{u}\frac{\partial \bar{v}}{\partial x} + \bar{v}\frac{\partial \bar{v}}{\partial y} + \frac{1}{\mathrm{Fr}^2}\frac{\partial h}{\partial y} = -\frac{1}{\mathrm{Fr}^2}\frac{\partial b}{\partial x}$$

with:

- $x, y \in [0, 40]$
- $t \in [0, 4]$
- Dimensional ref values: $g = 9.8$, $U = 5$, $L = 40$, $H = 2.5$
- Implies $Fr = 1.01$
- Flat bathymetry

IC:

At $t = 0$, water column height $h(x, y, 0)$ is given by:

$$h(x, y, 0) = \begin{cases} h_{\text{in}}, & \sqrt{(x-20)^2 + (y-20)^2} \leq r_0, \\ h_{\text{out}}, & \text{otherwise.} \end{cases}$$

where $r_0 = 5\,m$ is the dam initial radius, and:

$$h_{\text{in}} = 2.0\,m, \quad h_{\text{out}} = 0.5\,m.$$

The initial velocity field is set to zero :

$$u(x, y, 0) = v(x, y, 0) = 0.$$

SWE 2D: Radial dam break

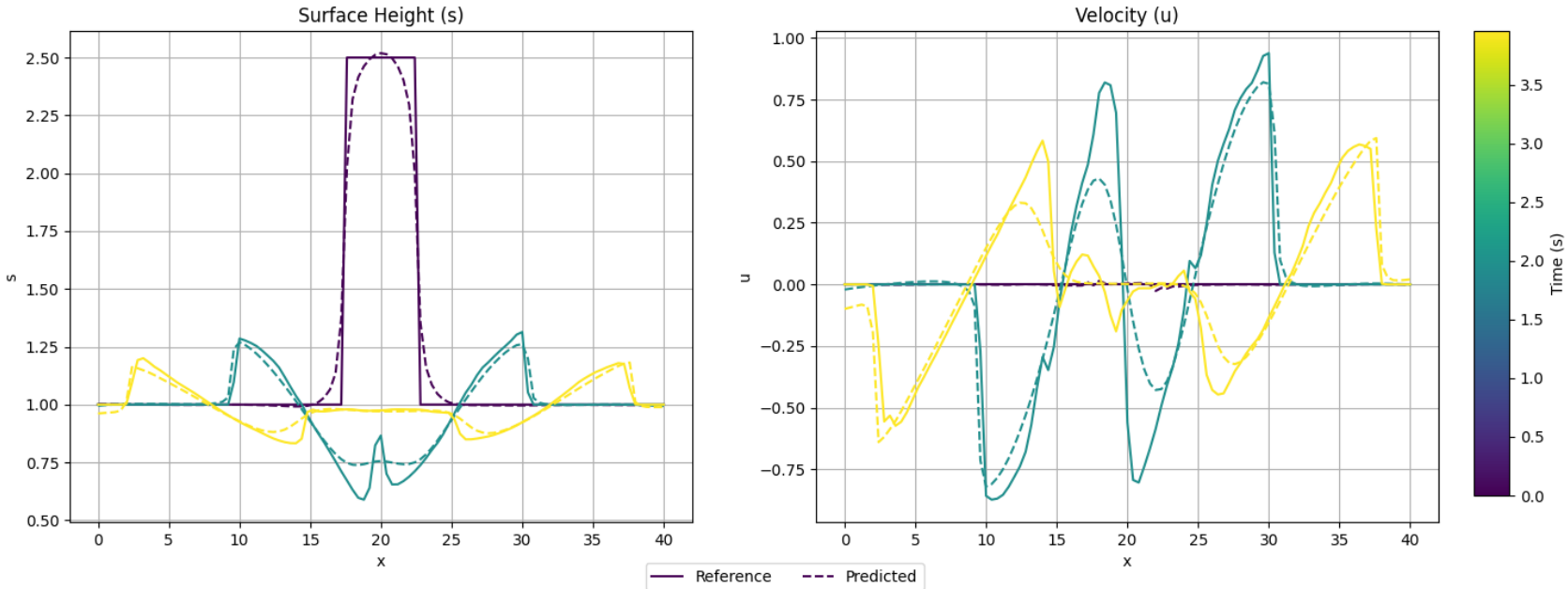


Figure: SWE 2D: Height (left), speed (right), *Sota* at $t = [0, 1, 2, 3, 4]$s.

Configuration	L_h^2 (%)	L_u^2 (%)	L_v^2 (%)	$MAPE_{mass}$ (%)	Time to converge(min)
Plain	3.1	33.0	34.5	0.12	
Default	8.2	86.9	87.4	0.98	
PirateNet	2.4	26.1	27.6	0.09	
Plain*	2.9	31.1	32.2	0.20	**106**
Default*	**2.0**	**23.5**	**25.2**	0.06	119
PirateNet*	**2.2**	**25.4**	**25.9**	**0.03**	155
Reference (FEM)					3

Hyperparameter	**Possible values**
Architecture	MMLP, PirateNet
Layers	8
Optimizer	Adam
Activation function	Tanh, Gelu, Swish
Fourier feature scale	1, 1.5, 2
RWF mean	0.5, 1
Causal tolerance	1, 1.5, 2

Conclusion: comments-recommendations

PINNs tend to smooth the solution
Integrate Riemann-like schemes for sharp gradients, Multi-?

Non-dimensionalization is essential for PINNs training
Analogous to how datasets are normalized in other deep learning fields.

Solution strategies are crucial for complex flows.

Future work: Integrate Finite-Volume and Discontinuous Galerkin (DG)
to handle complex geometries and discontinuities.

Questions?

https://www.researchgate.net/profile/Azzeddine-Soulaimani

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

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., & Ng, R. (2020).Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *Advances in Neural Information Processing Systems*, *33*, 7537–7547. Retrieved August 6, 2024, from https://proceedings.neurips.cc/paper/2020/hash/55053683268957697aa39fba6f231c68-Abstract.html

Krishnapriyan, A. S., Gholami, A., Zhe, S., Kirby, R. M., & Mahoney, M. W. (2021, November). Characterizing possible failure modes in physics-informed neural networks [arXiv:2109.01050 [physics]]. https://doi.org/10.48550/arXiv.2109.01050

Wang, S., Sankaran, S., & Perdikaris, P. (2022, March). Respecting causality is all you need for training physics-informed neural networks. Retrieved September 11, 2024, from https://arxiv.org/abs/2203.07404v1

Wang, S., Wang, H., Seidman, J. H., & Perdikaris, P. (2022, October). Random Weight Factorization Improves the Training of Continuous Neural Representations [arXiv:2210.01274 [cs]]. https://doi.org/10.48550/arXiv.2210.01274

Wang, S., Sankaran, S., Wang, H., & Perdikaris, P. (2023, August). An Expert's Guide to Training Physics-informed Neural Networks [arXiv:2308.08468 [physics]]. https://doi.org/10.48550/arXiv.2308.08468

Kamil, H., Soulaïmani, A., & Beljadid, A. (2024).A transfer learning physics-informed deep learning framework for modeling multiple solute dynamics in unsaturated soils. *Computer Methods in Applied Mechanics and Engineering*, *431*, 117276.

Rathore, P., Lei, W., Frangella, Z., Lu, L., & Udell, M. (2024).Challenges in training pinns: A loss landscape perspective. *arXiv preprint arXiv:2402.01868*.

- Wang, S., Li, B., Chen, Y., & Perdikaris, P. (2024, February). PirateNets: Physics-informed Deep Learning with Residual Adaptive Networks [arXiv:2402.00326 [cs, math]]. Retrieved July 8, 2024, from http://arxiv.org/abs/2402.00326
- Kamil, H., Soulaïmani, A., & Beljadid, A. (2025).A comparative study of physics-informed neural network strategies for modeling water and nitrogen transport in unsaturated soils. *Journal of Hydrology*.
- Kamil, H., Soulaımani, A., & Beljadid, A. (2025).Physics-informed neural operators for efficient modeling of infiltration in porous media. *Journal of Computational Physics*, 114156.
- Vyas, N., Morwani, D., Zhao, R., Kwun, M., Shapira, I., Brandfonbrener, D., Janson, L., & Kakade, S. (2025, February). SOAP: Improving and Stabilizing Shampoo using Adam [arXiv:2409.11321 [cs]]. https://doi.org/10.48550/arXiv.2409.11321