Can Physics-Informed Training Improve Neural-Operator Data Efficiency?

A Controlled FNO and PINO Comparison for PDE Surrogate Modelling

Samuel Campos Vilar[1]

Supervisors: Dr. Jing Sun[1]**, Dr. Tiexing Wang**[2]

[1]EEMCS, Delft University of Technology, The Netherlands
[2]AI for Engineering, AECOM

Name of the student: Samuel Campos Vilar
Final project course: CSE3000 Research Project
Thesis committee: Dr. Jing Sun, Dr. Tiexing Wang, Prof. Dr. Mathijs de Weerdt

Abstract

Numerical solvers for partial differential equations (PDEs) are accurate, but expensive when many related problem instances must be solved. Neural operators offer a faster alternative by learning mappings from input fields to solution fields. However, standard Fourier Neural Operator (FNO) training still depends on labelled simulation data, which must be generated by a solver. This study examines whether physics-informed training can improve data efficiency for neural-operator PDE surrogates. A data-driven FNO is compared with a Physics-Informed Neural Operator (PINO) using the same FNO backbone. This keeps the comparison centred on the training objective rather than the architecture. PINO adds a PDE-residual penalty to the supervised loss, encouraging predictions to satisfy the governing equation as well as match labelled data. The experiments use two PDEBench benchmarks: 2D Darcy flow and 1D Burgers' equation. Accuracy is measured with relative L_2 error, and physical consistency with PDE residual. PINO reached lower relative L_2 error than FNO on both equations at every tested fraction. It also surpassed the full-data FNO using 10% of the labels on Darcy flow and 50% on Burgers' equation. PINO reduced PDE residuals, especially for Darcy flow. However, the out-of-distribution tests showed that better in-distribution accuracy did not imply robust extrapolation. Overall, physics-informed training improved in-distribution data efficiency and physical consistency, but the gain depended on the equation, labelled-data budget, physics-loss weight and evaluation regime.

1 Introduction

Numerical simulations based on partial differential equations (PDEs) are widely used to model physical systems such as fluid flow, diffusion, and other scientific and engineering processes. Although these simulations can produce accurate solutions, they are often computationally expensive, especially when the same problem must be solved repeatedly for different input conditions or parameter settings. This motivates the use of machine learning surrogate models, which aim to approximate the behaviour of numerical solvers at a lower computational cost.

Neural operators are a promising class of surrogate models for PDE problems because they learn mappings between input functions and output solution functions, rather than mappings between fixed-dimensional vectors [1]. This is useful for parametric PDEs, where the target is a reusable solution operator rather than a single solution. DeepONet showed that neural networks can learn nonlinear operators from data [2]. Graph neural operators also formulated operator learning through kernel integral operators computed with graph message passing [3]. The Fourier Neural Operator (FNO) made this idea efficient on regular grids by parameterising global convolution in Fourier space, and showed strong performance on Darcy flow, Burgers' equation, and Navier-Stokes benchmarks [4]. FNO-style models have also been extended to more specialised PDE-surrogate settings, such as U-FNO for multiphase flow in porous media, where improved data utilisation was reported against a convolutional baseline [5]. However, these works still rely mainly on supervised solver data, so they do not show how neural-operator performance changes when labelled data are scarce.

This labelled-data dependence limits practical surrogate modelling, because each input-output pair normally has to be generated by a numerical solver. Recent data-efficient operator-learning work has addressed this issue through unsupervised pretraining and in-context learning [6], but without enforcing the governing equation during training. A complementary route is to include physical knowledge in the objective. Physics-constrained surrogate models showed that governing equations can be incorporated into deep-learning

losses with little or no labelled output data [7]. However, these approaches do not directly test whether a physics loss improves labelled-data efficiency for a modern neural operator under a matched data-only baseline.

Physics-Informed Neural Networks (PINNs) are the standard starting point for residual-based scientific machine learning, since they train neural networks to satisfy PDE constraints when solving forward and inverse PDE problems [8]. However, PINNs are usually optimised for individual PDE instances, whereas this study concerns a reusable solution operator. Physics-informed operator learning addresses this distinction by adding similar PDE constraints to neural operators. For example, physics-informed DeepONets extend DeepONet training with PDE residual and boundary-condition losses for parametric PDEs [9]. PINO takes the same general direction with an FNO-style architecture, combining supervised data loss with a PDE residual loss to learn a family of PDE solutions [10]. These studies show that physics constraints can improve operator learning, but leave open how much of the gain is due to improved labelled-data efficiency across systematically reduced data budgets.

What remains less clear is how the effect of physics-informed neural-operator training changes as the amount of labelled simulation data is systematically reduced. This paper therefore addresses the following research question:

> To what extent can physics-informed neural operator training improve data efficiency compared with purely data-driven neural operator training for PDE surrogate modelling?

To answer this question, this work compares a data-driven FNO-style neural operator with a physics-informed variant using the same architecture, so that the main difference is the training objective rather than the model structure. The experiments are conducted on two PDEBench benchmarks: 2D Darcy flow and 1D Burgers' equation. These benchmarks provide complementary test cases, since Darcy flow is a steady elliptic problem mapping a permeability field to a pressure field, while Burgers' equation is a time-dependent nonlinear problem mapping an initial condition to a full spatio-temporal trajectory. For both equations, the models are trained at several labelled-data fractions and evaluated on held-out test data using relative L_2 error as the primary accuracy metric. The comparison also includes PDE residual error to assess whether physics-informed training improves physical consistency as well as predictive accuracy.

The main contribution of this research is a controlled empirical comparison of FNO and PINO under different labelled-data budgets on both benchmarks. The results show how the benefit of the physics-informed loss changes across the data-efficiency curve, and whether PINO can match or improve the accuracy of a purely data-driven neural operator with fewer labelled simulations. This is relevant for PDE surrogate modelling settings where generating large labelled datasets is expensive, but the governing equation is known.

The remainder of the paper follows the experimental pipeline used to answer the research question. First, Section 2 defines the shared FNO backbone and explains how the physics-informed objective turns the baseline into the PINO variant. The datasets, labelled-data protocol, training setup, and evaluation metrics are then described in Section 3. The empirical results for Darcy flow and Burgers' equation are presented in Section 4, after which Section 5 interprets the findings and discusses the main limitations of the study. The paper concludes in Section 6 by answering the research question and outlining future work. Finally, Section 7 reflects on reproducibility, integrity and ethical considerations.

2 Fourier and Physics-Informed Neural Operators

This study compares two neural-operator training objectives under a shared architecture. The baseline model is a data-driven Fourier Neural Operator (FNO), trained only to match labelled PDE solutions. The physics-informed model is a Physics-Informed Neural Operator (PINO), which uses the same FNO backbone but adds a soft penalty on the governing PDE residual. Keeping the architecture, optimiser and training hyperparameters fixed across the two models keeps the experimental comparison centred on the effect of the physics-informed objective rather than on differences in model capacity or training protocol. This section defines the shared backbone, the physics-informed loss, the equation-specific residuals, and the procedure used to choose the physics-loss weight. The datasets, training protocol and evaluation metrics are described in Section 3.

2.1 Fourier neural operator backbone

A neural operator learns a mapping between function spaces rather than between fixed-dimensional vectors. In this study, the learnt map

$$\mathcal{G}_\theta : \mathcal{A} \to \mathcal{U}$$

takes an input field, such as a coefficient field or an initial condition, and predicts the corresponding PDE solution field [1]. This makes neural operators suitable for surrogate modelling of PDEs, where the target is the solution operator itself rather than a regression model tied to a single discretised grid.

The FNO was used as the backbone because it is an established neural-operator architecture for PDE surrogate modelling [4]. Its Fourier layers mix information globally through spectral convolution, which is useful for PDEs whose solutions depend on non-local interactions. This makes it a more appropriate baseline than a plain convolutional network, which mainly captures local features and is typically tied to a single training resolution, or a multilayer perceptron, which treats the discretised field as a fixed-dimensional vector rather than as a sampled function [11, 1]. The FNO is therefore a suitable architecture for testing whether a physics-informed loss improves data efficiency, because the baseline is already a competitive operator-learning model.

The same backbone was used for the data-driven FNO and the PINO model. The input field was first lifted to a latent representation, passed through four Fourier layers, and then projected back to the output field. Only a fixed number of low-frequency Fourier modes was retained in each layer, which keeps the global convolution efficient and introduces a smoothness bias. Normalised coordinate channels were concatenated to the input so that the model could access absolute position. For Darcy flow, the model maps a permeability field to a pressure field on the spatial grid (x, y). For Burgers' equation, which is one-dimensional in space, the model predicts the full time evolution: it maps the initial condition to the solution trajectory on the space-time grid (t, x). Table 1 summarises the shared FNO backbone used for both the data-driven FNO and PINO models.

2.2 Physics-informed objective

PINO uses the same FNO backbone as the data-driven baseline, but changes the training objective by adding a penalty on the PDE residual [10]. The comparison is therefore focused on the effect of the physics-informed loss rather than on a change in architecture. This is

	Darcy	Burgers
Spatial dimensions	2 (x, y)	1 (x)
Temporal axis	no	yes, $t \in [0, 2]$
Input / output channels	1 / 1	1 / 1
Latent width d_v	64	64
Fourier layers L	4	4
Retained modes $k_{\max}$	20	16
Domain padding	0	8
Decoder	1 layer, 128, GELU	1 layer, 64, GELU
Coordinate features	yes	yes
Trainable parameters	$\approx 26.2\,\mathrm{M}$	$\approx 16.8\,\mathrm{M}$

Table 1: Shared FNO backbone configuration. The same architecture is used for the data-driven FNO and the physics-informed PINO model; only the loss objective differs.

also why a PINN is not the main baseline in this study. PINNs are designed mainly to solve individual PDE instances [8], whereas the research question concerns whether physics-informed training improves the data efficiency of a neural operator that learns a whole family of input-output maps.

The training loss combines a supervised data term with a physics term:

$$\mathcal{L}(\theta) = \lambda_{\text{data}}\,\mathrm{MSE}(\hat{u}, u) + \lambda_{\text{pde}}\,\mathrm{MSE}(s\,\mathcal{R}[\hat{u}], 0). \tag{1}$$

Here, $\hat{u}$ is the predicted field, u is the reference solution, $\mathcal{R}$ is the PDE residual evaluated on the prediction, and s is a grid-dependent scaling factor. The data term was computed on raw, unnormalised fields so that the physics residual remained meaningful in physical units. The scaling factor was used to keep the residual loss on a comparable numerical scale to the supervised loss.

For Darcy flow, the residual corresponds to the steady diffusion equation,

$$\mathcal{R}[\hat{u}] = -\nabla\cdot(k\nabla\hat{u}) - f, \qquad f = 1, \tag{2}$$

where k is the permeability field and $\hat{u}$ is the predicted pressure. The residual was discretised with the same cell-centred finite-volume stencil and ghost-cell boundary treatment as the PDEBench Darcy generator [12]. It was scaled by one grid spacing, $s = \Delta x = 1/128$, before squaring.

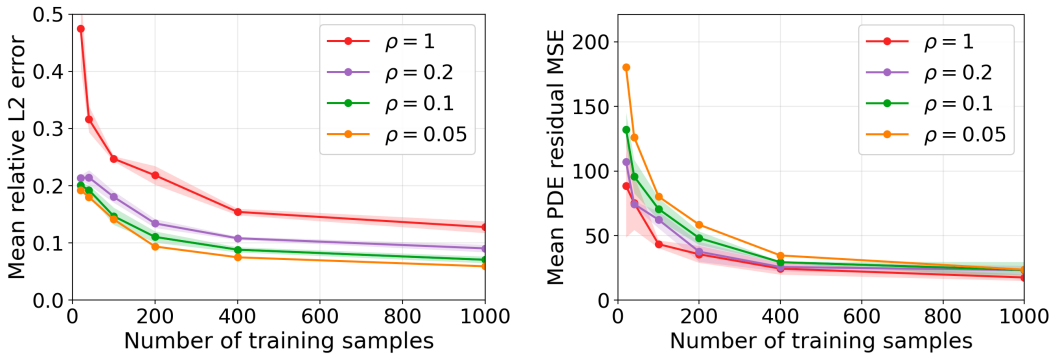
For Burgers' equation, the residual corresponds to

$$\mathcal{R}[\hat{u}] = \hat{u}_t + \hat{u}\hat{u}_x - \frac{\nu}{\pi}\hat{u}_{xx}. \tag{3}$$

It was evaluated with central finite differences and periodic boundaries in space, then scaled by the time step $s = \Delta t = 0.02$, using the PDEBench ν/π diffusion coefficient and $t \in [0, 2]$ domain [12]. A soft initial-condition term ($\lambda_{\text{ic}} = \lambda_{\text{pde}} = 1$) anchored the prediction at $t = 0$ to the initial condition u_0, following PINO [10], and no boundary term was needed because the residual already enforces spatial periodicity ($\lambda_{\text{bc}} = 0$).

The physics weight is controlled by the ratio

$$\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}.$$



(a) Relative L_2 error.

(b) PDE residual.

Figure 1: Physics-loss weight sensitivity on Darcy flow. Each curve shows a PINO model trained with a fixed ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}$.

In the implementation, λ_{pde} was fixed to 1 and λ_{data} was varied to set the desired ratio. The data-only FNO baseline is recovered by setting $\lambda_{\text{pde}} = 0$. Thus, FNO and PINO use the same architecture and differ only in the additional physics-loss terms.

2.3 Setting the physics-loss weight

The ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}$ is the only extra hyperparameter in the physics-informed objective, so the main comparison used a single fixed value rather than tuning this ratio separately for each training-set size. For Darcy flow, a small sensitivity sweep used $\rho \in \{0.05, 0.1, 0.2, 1.0\}$, together with a physics-only ablation where $\lambda_{\text{data}} = 0$. These settings were chosen to cover interpretable regimes without turning the sweep into a large hyperparameter search: $\rho = 0.05$ and $\rho = 0.1$ test physics-light regularisation, $\rho = 0.2$ matches data-to-physics weighting used in the PINO paper [10] for time-dependent flow, $\rho = 1.0$ tests equal weighting of data and residual losses, and the physics-only ablation checks whether the residual alone can identify a useful solution operator. Figure 1 reports the two quantities used to compare these settings: relative L_2 error as the primary accuracy metric and PDE residual MSE as a physical-consistency check.

The sweep was used to choose one representative physics-light setting for the controlled FNO-PINO comparison. The trade-off prioritised predictive accuracy while avoiding settings whose residual MSE indicated that the physics term had become too weak. Under this trade-off, $\rho = 0.1$ was used for the main experiments. Although $\rho = 0.05$ gave a very small improvement in relative L_2 error, this gain was minor compared with its increase in PDE residual. In contrast, $\rho = 0.1$ gave nearly the same predictive accuracy while preserving substantially better physical consistency. Larger weights, especially $\rho = 1.0$, shifted too much emphasis towards residual reduction and degraded accuracy. The remaining weights are therefore retained as a sensitivity analysis in Section 4. The same comparison was repeated for Burgers' equation, as shown in Appendix B, where $\rho = 0.1$ again gave a strong accuracy-consistency trade-off.

Both models were implemented with NVIDIA PhysicsNeMo [13]. Full software versions, hardware details and training hyperparameters are reported in Section 3.

3 Experimental setup

This section defines the pipeline used to compare data-driven FNO training with physics-informed PINO training. For each equation, a fresh model with the backbone of Section 2 was trained from a nested subset for each fraction $f \in \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0\}$ and seed $s \in \{0, 1, 2\}$, then evaluated on held-out data. Within each equation, the architecture and training hyperparameters were fixed across all fractions and seeds, and the models differed only in the objective of Equation (1): FNO used $\lambda_{\text{pde}} = 0$, while PINO used a non-zero physics weight. Results are averaged over three seeds. Figures show seed means with shaded bands denoting ± 1 standard deviation, while tables report the corresponding seed means. The main comparison contains $7 \times 3 \times 2 = 42$ runs per equation, with additional weight-sensitivity runs described in Section 3.2.

3.1 Data

Both benchmarks were taken from PDEBench [12] using the public DaRUS dataset release [14], a peer-reviewed collection of scientific machine-learning datasets distributed under a CC BY 4.0 licence. PDEBench provides reference solutions from validated numerical solvers, so no solver time was spent generating data and the results remain comparable to a recognised benchmark suite. The two equations are complementary. Darcy flow is a steady elliptic problem, while Burgers' equation is a time-dependent nonlinear problem. Table 2 summarises the dataset configuration, while representative input-output examples are provided in Appendix C.

	Darcy flow	Burgers' equation
PDEBench file	`...beta1.0`	`...Nu0.01`
Key parameter	$\beta = 1.0,\ f = 1$	$\nu = 0.01$
Grid	128×128	$201 \times 1024 \to 101 \times 512$
Domain	$[0, 1]^2$	$t \in [0, 2],\ x \in [0, 1]$
Boundary	Dirichlet 0	periodic in x
Input $\to$ output	$k \to u$	u_0 tiled $\to u(t, x)$
Normalisation	none (raw)	none (raw)
Samples $N_{\max}$	2500	1250
Train / val / test	2000/250/250	1000/125/125
Fractions	7 (0.01 to 1.0)	7 (0.01 to 1.0)
OOD test	$\beta = 0.1$ (256)	$\nu = 0.001$

Table 2: Dataset configuration for both benchmarks. Training subsets at each fraction are nested within a seed, and validation and test sets are held out from the training pool for every fraction.

For **Darcy flow**, the file `2D_DarcyFlow_beta1.0_Train.hdf5` was used. Each sample is a 128×128 cell-centred field on $[0, 1]^2$, with homogeneous Dirichlet-zero boundaries and constant forcing $f = 1$. The model maps the permeability field k to the pressure field u. Fields were kept in raw physical units, without normalisation, so that the residual of Equation (2) remains consistent with the loss definition in Section 2.2.

For **Burgers' equation**, the file `1D_Burgers_Sols_Nu0.01.hdf5` was used, with viscosity $\nu = 0.01$. The stored solutions lie on a 201×1024 space-time grid over $t \in [0, 2]$ and $x \in [0, 1]$, with periodic spatial boundaries. To fit the available memory and wall-clock budget, every second time step and spatial point was retained, reducing the grid to 101×512. The supervised loss and PDE residual were both computed on this coarser grid, with effective temporal spacing $\Delta t = 0.02$. The model maps the initial condition u_0, tiled along the

time axis, to the full trajectory $u(t, x)$. Fields were left unnormalised. The residual also follows the PDEBench conventions described in Section 2.2, where the stored time interval is $t \in [0, 2]$ and the diffusion coefficient is ν/π.

Splits, fractions and OOD tests. Each dataset was capped at $N_{\max}$ samples and split 80/10/10 into a training pool, validation set and test set. This gives 2000/250/250 samples for Darcy flow and 1000/125/125 for Burgers' equation. The split permutation is seeded by the training seed, so within a seed the validation and test partitions are fixed across all fractions and shared by FNO and PINO, while across seeds the partitions are reshuffled. Training subsets are nested. For a fixed seed, the 5% set is contained in the 10% set, which is contained in the 20% set, and so on. No validation or test sample appears in any training subset. The approximately logarithmic fraction sweep gives finer resolution in the scarce-data regime while still covering the full-data setting.

OOD evaluation used parameter shifts with the same input-output format as the in-distribution data. Darcy flow used $\beta = 0.1$ instead of $\beta = 1.0$, and Burgers' equation used $\nu = 0.001$ instead of $\nu = 0.01$, with 256 samples in each OOD test set. The Darcy shift mainly probes amplitude extrapolation because the forcing parameter is not a model input, while the Burgers shift reduces diffusion and sharpens the dynamics. In both cases the OOD residual was evaluated against the shifted parameter, with the Darcy residual using the shifted forcing $f = 0.1$ and the Burgers residual using the shifted viscosity $\nu = 0.001$ under the same ν/π convention as in Section 2.2.

3.2 Training setup

For every fraction and seed, a fresh model was trained from scratch with the same architecture and the same training hyperparameters. The same training settings were used for every fraction so that changes in performance can be attributed to the amount of labelled data and the training objective, rather than to fraction-specific tuning. The full training configuration is given in Table 3.

	Darcy flow	Burgers' equation
Max epochs	500	400
Early stopping	none	patience 60 (val. rel. L_2)
Batch size	20	20
Optimiser	Adam [15], $\beta_1 = 0.9, \beta_2 = 0.999$	Adam [15], $\beta_1 = 0.9, \beta_2 = 0.999$
Learning rate	10^{-3}	10^{-3}
LR schedule	$\times 0.5$ every 100 ep.	$\times 0.5$ every 100 ep.
Weight decay	10^{-4}	10^{-4}
Gradient clipping	none	1.0
Seeds	$\{0, 1, 2\}$	$\{0, 1, 2\}$
Hardware	A100 (`gpu-a100-small`)	A100 (`gpu-a100-small`)
Wall-clock	$\approx 2\,\text{h}\,17\,\text{min}$ (full data)	$\approx 3\,\text{h}\,40\,\text{min}$ (full data)
GPU peak memory	$\approx 1.9\,\text{GB}$	$\approx 4.8\,\text{GB}$

Table 3: Training configuration for both benchmarks. The configuration was kept fixed across fractions and seeds within each equation.

Loss configuration. The two models differ only in their training objective. In every PINO configuration, λ_{pde} was fixed to 1 and the ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}$ was set by scaling λ_{data}. The data-driven FNO baseline uses $\lambda_{\text{pde}} = 0$, so $\rho = 0$. For Burgers' equation, the PINO

objective additionally included the initial-condition penalty described in Section 2.2. The final PINO results use $\rho = 0.1$, selected as described in Section 2.3. Ratios $\rho \in \{0.05, 0.2, 1.0\}$ and a physics-only configuration with $\lambda_{\text{data}} = 0$ are reported as a sensitivity analysis in Section 4.

Seeds, runs and reproducibility. The results are averaged over three seeds, 0, 1 and 2, which controlled both the model initialisation and the data split permutation described in Section 3.1. The reported mean ± standard deviation therefore summarises variability across the full seeded experimental pipeline, not pure initialisation variance. All models were trained with the NVIDIA PhysicsNeMo FNO implementation [13] inside a fixed Apptainer container. The main software and hardware settings are summarised in Table 3, with the full environment listed in Appendix A.

3.3 Evaluation

The primary accuracy metric is the relative L_2 error, averaged over the test set:

$$\text{relL2} = \frac{1}{N} \sum_{i=1}^{N} \frac{\|\hat{u}_i - u_i\|_2}{\|u_i\|_2}, \tag{4}$$

where $\hat{u}_i$ and u_i are the predicted and reference fields for test sample i, $\|\cdot\|_2$ is the Euclidean norm over all grid points, and N is the test-set size. Relative L_2 is scale-invariant per sample, which makes it comparable across the two benchmarks and suitable as the main data-efficiency metric.

The primary physical-consistency metric is the PDE residual MSE, defined as the mean squared scaled residual evaluated on the model's own test predictions:

$$\text{PDEres} = \frac{1}{N} \sum_{i=1}^{N} \text{MSE}\big(s\,\mathcal{R}[\hat{u}_i],\, 0\big). \tag{5}$$

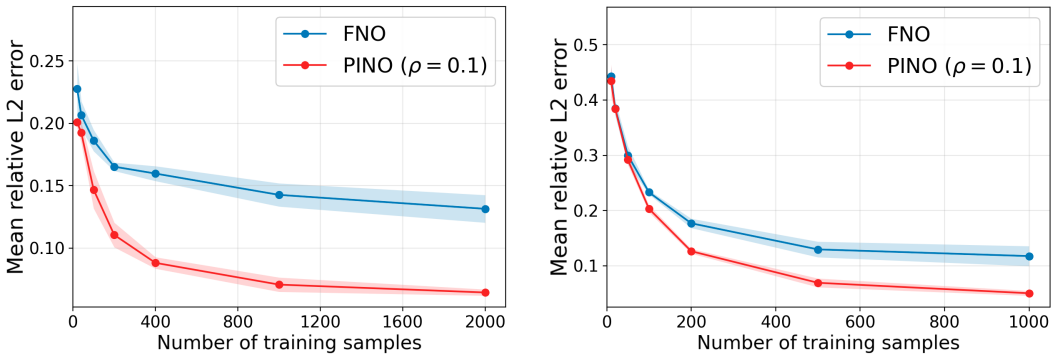
Here, $\mathcal{R}[\hat{u}_i]$ is the discrete PDE residual of the prediction and s is the residual scaling used by the corresponding physics loss. For Darcy flow, this is the finite-volume residual of Equation (2) scaled by Δx. For Burgers' equation, it is the residual of Equation (3) scaled by Δt. Computing this metric on held-out predictions separates physical consistency from raw accuracy: a model can be accurate while violating the equation, or satisfy the equation while still predicting the wrong solution.

Two error-localisation metrics are also reported where they add information: the boundary MSE on the four Dirichlet edges for Darcy flow, and the MSE at $t = 0$ against the prescribed initial condition for Burgers' equation. Training cost is recorded as wall-clock time and GPU peak memory.

The data-driven FNO with $\lambda_{\text{pde}} = 0$ served as the baseline. Because the architecture, data splits, training budget and evaluation metrics were kept fixed, the added physics-informed term was the main experimental difference between FNO and PINO. The reference fields were the held-out PDEBench solutions, which were generated by the benchmark's numerical solvers, so no additional solver was run during evaluation. The same metrics were computed on the in-distribution test sets and on the OOD test set defined in Section 3.1. This evaluation protocol leads directly to the main data-efficiency comparison in Section 4, where relative L_2 error is plotted against the number of training samples for FNO and PINO, with one panel per equation and seed-variance bands.

4 Results

This section reports the comparison between the data-only FNO and the physics-informed PINO under the experimental protocol in Section 3. Figure 2 gives the headline accuracy result for both benchmark equations as relative L_2 test error against the number of labelled training samples, and Tables 4a and 4b give the corresponding values, including the PDE residual MSE used to measure physical consistency.



(a) Darcy flow, $\rho = 0.1$, three seeds.

(b) Burgers' equation, $\rho = 0.1$, three seeds.

Figure 2: Data efficiency on both benchmark equations. Relative L_2 test error is shown against the number of labelled training samples for the data-only FNO and the physics-informed PINO. Curves show the mean over three seeds, and shaded bands show ± 1 standard deviation across seeds.

		Rel. L_2			Residual MSE	
f	n	FNO	PINO	Gain	FNO	PINO
0.01	20	0.228	0.201	12%	5334	132
0.02	40	0.207	0.193	7%	4243	96
0.05	100	0.186	0.147	21%	2788	71
0.10	200	0.165	0.111	33%	2586	48
0.20	400	0.160	0.088	45%	2849	29
0.50	1000	0.143	0.071	50%	2680	24
1.00	2000	0.132	0.065	51%	2841	20

(a) Darcy flow.

		Rel. L_2			Residual MSE	
f	n	FNO	PINO	Gain	FNO	PINO
0.01	10	0.443	0.435	2%	7.2	7.1
0.02	20	0.385	0.384	0%	14.7	12.6
0.05	50	0.300	0.292	3%	10.4	7.7
0.10	100	0.233	0.203	13%	5.0	5.1
0.20	200	0.177	0.126	29%	5.9	3.5
0.50	500	0.130	0.069	47%	5.2	1.8
1.00	1000	0.118	0.050	57%	4.8	1.2

(b) Burgers' equation.

Table 4: In-distribution results for both benchmark equations against labelled-data fraction f and number of labelled training samples n. Accuracy is measured by relative L_2 error, and physical consistency is measured by PDE residual MSE. Values are averaged over three seeds. PINO uses $\rho = 0.1$. *Gain* is the relative reduction in relative L_2 error of PINO over FNO.

4.1 Data efficiency

Darcy flow. For Darcy flow, PINO reached a lower relative L_2 error than the data-only FNO at every labelled-data fraction, as shown in Figure 2a and Table 4a. The relative gain grew with labelled-set size, from 12% at the smallest fraction to 51% at the full 2000-sample pool. Read horizontally, PINO trained on 100 samples reached a relative L_2 error close to

the FNO trained on 1000 samples. PINO trained on 200 samples already improved on the FNO trained on the full pool, using 1800 fewer labelled samples, a 90% reduction.

Burgers' equation. For Burgers' equation, the same accuracy ordering held, with PINO reaching a lower relative L_2 error than FNO at every tested fraction, as shown in Figure 2b and Table 4b. The gain was negligible in the scarcest-data regime, with a near-tie at 20 samples, and grew to 57% at the full 1000-sample pool. Read horizontally, PINO trained on 200 samples reached an error close to the FNO trained on all 1000 samples, and PINO trained on 500 samples already improved on the FNO trained on the full pool, using 500 fewer labelled samples, a 50% reduction.

4.2 Physical consistency

Darcy flow. The physical-consistency results for Darcy flow are shown numerically in Table 4a. The PINO PDE residual was one to two orders of magnitude below the FNO residual at every training-set size, from about 40 times lower at the smallest fraction to about 142 times lower at the full pool. The boundary metric showed the same ordering, with PINO producing roughly 12 to 42 times lower error on the four Dirichlet edges than FNO. The per-fraction boundary values can be found in Appendix E.

Burgers' equation. For Burgers' equation, the physical-consistency gap was much narrower than for Darcy flow. PINO produced a lower residual at most fractions, and the reduction widened with labelled data to about a factor of 4 at the full-data point. In the scarce-data regime the two models were close, and the data-only FNO was practically tied at 100 samples. PINO also matched the $t = 0$ initial condition more closely, by about 3 to 24 times (Appendix E).

4.3 Sensitivity to the physics weight

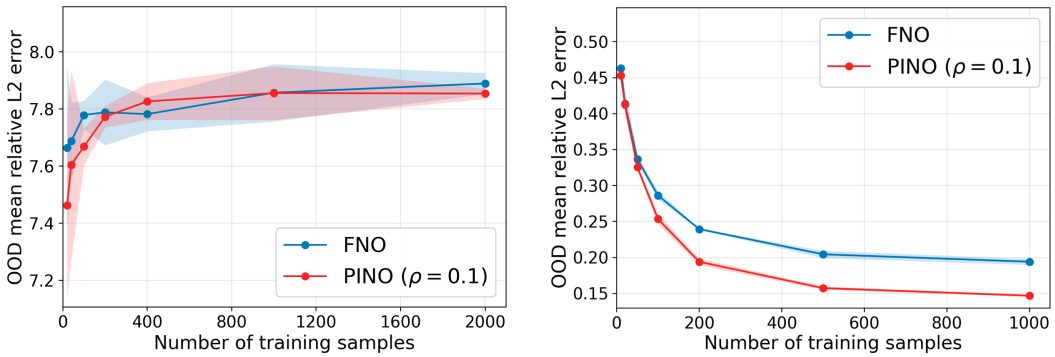
The physics-weight sweep in Figure 1 shows that the size of the PINO gain depended strongly on the ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}$.

Darcy flow. For Darcy flow, $\rho = 0.05$ and $\rho = 0.1$ gave the strongest accuracy, while $\rho = 0.1$ had the better combined behaviour across relative L_2 error and PDE residual MSE. Larger physics weights reduced accuracy. At the smallest fraction, $\rho = 1.0$ reached a relative L_2 error of 0.475, compared with 0.201 for $\rho = 0.1$ and 0.228 for the data-only FNO. The physics-only ablation, where $\lambda_{\text{data}} = 0$, reached relative L_2 errors of 0.72 at the smallest fraction and 0.98 at the full-data fraction.

Burgers' equation. The same behaviour held for Burgers over the weights tested: $\rho = 0.05$ and $\rho = 0.1$ were at least as accurate as $\rho = 0.2$ at most fractions, and the physics-only ablation ($\lambda_{\text{data}} = 0$) reached a relative L_2 error of 0.77 at the full-data point, far above either operator.

4.4 Out-of-distribution generalisation

Darcy flow. The Darcy OOD results for the $\beta = 0.1$ test set are shown in Figure 3a. Both models had relative L_2 errors far above their in-distribution values, and the OOD



(a) Darcy flow, $\beta = 0.1$.

(b) Burgers' equation, $\nu = 0.001$.

Figure 3: Out-of-distribution accuracy for Darcy flow and Burgers' equation. Mean relative L_2 error is plotted against the number of labelled training samples for the data-only FNO and PINO ($\rho = 0.1$), with seed-averaged curves and shaded bands showing ± 1 standard deviation across seeds.

error increased with training-set size rather than decreasing, opposite to the in-distribution trend in Figure 2a. PINO had the lower OOD error at the smaller training sizes, after which the two models were comparable within overlapping seed bands. The OOD residual kept the same ordering as in distribution, with the PINO residual remaining roughly 40 to 130 times below the FNO residual.

Burgers' equation. The Burgers $\nu = 0.001$ OOD shift was far less damaging than the Darcy amplitude shift, as shown in Figure 3b. Unlike Darcy flow, the OOD error decreased as the labelled set grew. The two models were almost tied in the scarce-data regime, after which PINO's accuracy advantage gradually widened with more labelled data. However, the in-distribution residual advantage of PINO did not carry over to the shifted viscosity, where the two models had comparable OOD residuals.

The exact per-fraction values for both equations, including the OOD residual comparison, are given in Table 5 in Appendix D.

5 Discussion

This study examined whether a physics-informed neural-operator objective improves data efficiency relative to purely data-driven FNO training. The results give a positive but conditional answer. PINO achieved lower relative L_2 error than the data-only FNO at every tested labelled-data fraction for both Darcy flow and Burgers' equation, while also generally reducing the PDE residual on held-out predictions. However, the gain depended on the equation, the labelled-data regime and the physics-loss weight. The residual term was therefore useful for refining a model that had already learnt the main solution pattern from labelled data, but it could not replace labelled simulation data.

5.1 Data efficiency and labelled data

The clearest data-efficiency evidence comes from the Darcy comparison in Figure 2. With $\rho = 0.1$, PINO reduced the relative L_2 error at every training-set size, with the reduction growing from 12% in the smallest-data regime to 51% at the full training pool. More importantly for the research question, the Darcy curve in Figure 2a is shifted left relative

to the FNO curve. PINO reached the same error range as much larger FNO training sets, and even surpassed the full-data FNO after only 200 labelled samples. This indicates that the physics-informed objective improved data efficiency, not only final accuracy. Since the architecture was shared, this improvement is consistent with the training objective being the relevant difference rather than model capacity.

The Burgers results in Figure 2b show the same qualitative pattern on a time-dependent nonlinear equation. PINO with $\rho = 0.1$ improved on FNO at every tested fraction, with the relative gain increasing from almost no advantage in the smallest-data regime to 57% at the full 1000-sample training pool. The leftward shift was smaller than for Darcy, but still showed that PINO approached or improved on the strongest FNO accuracy with a reduced labelled subset. Thus, the data-efficiency pattern was not specific to the steady Darcy problem, but also appeared in the nonlinear Burgers benchmark.

A notable result is that the largest gains did not occur where labelled data were scarcest. Instead, the advantage grew as more labels were added. This suggests that the supervised term first teaches the model the correct scale, structure and solution family, after which the residual term can refine the prediction and suppress physically inconsistent solutions. The ablations support this interpretation. The physics-only models produced poor relative L_2 errors, and the largest tested weight, $\rho = 1.0$, was worse than the data-only FNO at the smallest Darcy fraction. Physics-informed training therefore improved data efficiency, but only when balanced with enough labelled information.

5.2 Accuracy, physical consistency and loss balance

The residual results show that the physics-informed objective improved physical consistency more strongly than predictive accuracy, especially for Darcy flow. For Darcy flow, PINO reduced the PDE residual by roughly 40 to 142 times across the data sweep, as shown in Table 4a. This agrees with the motivation of PINO, which uses the PDE residual to constrain predictions towards physically plausible fields [10].

For Burgers' equation, the residual gap was much smaller and less uniform. PINO had a lower residual at most fractions, with a near tie in the smallest-data regime and one marginal reversal at 100 samples. The largest reduction was about $4\times$ at the full-data point. This difference indicates that physical-consistency gains are equation-dependent. A data-only FNO trained on smooth, subsampled Burgers trajectories may already satisfy the residual reasonably well on the evaluated grid, whereas the unconstrained Darcy FNO leaves a much larger residual for the physics term to correct. Reporting both relative L_2 error and PDE residual was therefore necessary: a low residual does not guarantee an accurate surrogate, as the physics-only ablation showed, and a data-only model can be reasonably accurate while still violating the discrete PDE.

The weight sweep in Figure 1 further shows that the PINO advantage is not automatic. A residual term that is too strong can bias optimisation towards smooth, residual-reducing fields that match the reference solution poorly. This agrees with broader observations in physics-informed learning, where different loss components can converge at different rates and make loss balancing a central training issue [16]. The selected physics weight is therefore part of the empirical finding, not an incidental implementation detail.

5.3 Out-of-distribution behaviour

The out-of-distribution tests show that the benefit of PINO did not translate into robust extrapolation in a simple way. For Darcy flow, Figure 3a gives the main negative result.

Both operators performed poorly on the $\beta = 0.1$ test set, with relative L_2 errors of roughly 7.5 to 7.9, far above the in-distribution values. PINO was lower in the smallest-data regime, but the seed bands largely overlapped. The OOD error also increased with training-set size, opposite to the in-distribution trend in Figure 2. Since the models were trained on $\beta = 1.0$ and β was not supplied as an input, neither operator had direct information about the shifted solution scale. This test therefore mainly probes amplitude extrapolation. PINO retained a much lower OOD residual, but this confirms that residual consistency alone is not sufficient for robust extrapolation.

The Burgers OOD shift in Figure 3b gives a different interpretation. Lowering the viscosity from $\nu = 0.01$ to $\nu = 0.001$ makes the dynamics less diffusive and sharpens the solution features, but the shift did not cause the same collapse as in Darcy flow. Instead, additional labelled data still improved both operators, and PINO retained an accuracy advantage that became clearer at larger training-set sizes. This suggests that the Burgers shift remained a regime where better in-distribution learning could still transfer, whereas the Darcy shift mainly exposed a missing conditioning variable. The comparable Burgers OOD residuals also show that the in-distribution residual advantage of PINO does not necessarily persist when the physical parameter in the equation changes. Overall, PINO improved interpolation within the tested regimes more reliably than extrapolation beyond them, which is consistent with work on deep neural operators showing that purely data-driven operators can be unreliable outside the support of the training distribution, and that extrapolation may require additional PDE information, sparse observations or explicit conditioning on the shifted parameter [17]. PINO should therefore not be interpreted as a general solution to out-of-distribution generalisation.

5.4 Limitations

Three limitations qualify these conclusions. First, the study used capped PDEBench subsets rather than the full datasets, and the Burgers grid was subsampled from 201×1024 to 101×512 due to computational constraints. This kept the comparison controlled, but it may have affected the absolute errors. As a check that this capping did not leave an under-tuned baseline, the full-data Darcy FNO trained in this study reached an error of 0.132, higher than the 0.064 reported for the PDEBench FNO on the same $\beta = 1.0$ benchmark and error metric, but with 2000 training samples rather than roughly 9000 [12]. The gap is therefore consistent with the smaller labelled budget rather than weak optimisation. For Burgers in particular, the coarser grid removes high-frequency information and may partly explain the smaller residual gap. Repeating the Burgers experiments at the original resolution and on the full dataset would show whether this smaller gap comes from the equation itself, or partly from evaluating both models on a coarser grid.

Second, the conclusions are based on two PDE families: steady elliptic Darcy flow and time-dependent nonlinear Burgers' equation. The two benchmarks already show that the size of the residual benefit varies substantially with the equation. The observed data-efficiency trend should therefore be read as evidence for these PDEBench benchmarks, not as a universal property of all physics-informed neural operators. Broader testing on higher-dimensional equations and problems with discontinuous solutions would show whether the same pattern holds in more demanding settings. A particularly relevant extension could be advection-type hyperbolic conservation laws, where shocks and discontinuities require careful treatment [18]. This would clarify whether the PINO gain observed here extends beyond the two equation families tested in this study.

Third, neither OOD test included the shifted parameter as a model input. The Darcy test changed β from 1.0 to 0.1, a tenfold amplitude shift that caused large errors in both models. The absence of β from the input is a likely cause, because the model cannot compensate for a forcing parameter it never observes. The Burgers test changed the viscosity from $\nu = 0.01$ to $\nu = 0.001$ and showed a different pattern: accuracy still improved with labelled-data size, but the in-distribution residual advantage of PINO did not carry over. The two tests therefore probe two specific shifts rather than general robustness. Supplying the shifted parameter as an input, or testing smaller shifts in amplitude, resolution or initial conditions, would give a fuller picture of out-of-distribution generalisation.

Overall, physics-informed training improved the data efficiency of neural-operator surrogates on the tested benchmarks. The residual term reliably improved in-distribution physical consistency and improved accuracy when properly balanced with labelled data, but it did not remove the need for labels and did not guarantee OOD robustness. The most defensible answer to the research question is therefore that PINO improved data efficiency over a matched data-only FNO on the tested PDEBench benchmarks, with the size and reliability of the gain depending on the equation, the labelled-data regime and the physics-loss weight.

6 Conclusions and Future Work

This study examined whether physics-informed neural-operator training improves data efficiency relative to purely data-driven FNO training for PDE surrogate modelling. The comparison kept the FNO architecture, data splits and training protocol fixed, so the main difference was the training objective, with the baseline using labelled solver data alone and PINO additionally penalising the PDE residual. Across the two PDEBench benchmarks, steady Darcy flow and time-dependent Burgers' equation, PINO achieved lower relative L_2 error than the data-only FNO at every tested labelled-data fraction and generally reduced the PDE residual on held-out predictions.

The answer to the research question is therefore positive, but conditional. Physics-informed training helped when enough labelled data anchored the operator. The gains were smallest where labels were scarcest and grew as labels were added, and at matched accuracy PINO needed roughly two to ten times fewer labelled samples than the FNO. It helped when the residual term stayed subordinate to the data term, since a small physics weight performed best but a large one was harmful and physics-only training failed outright. Its physical-consistency benefit was largest where the data-only baseline left the largest residual, spanning one to two orders of magnitude on Darcy flow against a factor of about four on Burgers' equation. It did not help as a substitute for labelled simulation data, and it did not provide uniform robustness under distribution shift: the Darcy shift largely removed the accuracy benefit, while the Burgers shift retained an accuracy advantage but not the in-distribution residual advantage.

Future work should first test whether this conclusion holds under a wider and more precise experimental protocol. A finer physics-weight sweep, possibly with weights adapted to the equation or labelled-data budget, would clarify how much performance is left untuned by using a single fixed value. Generalisation should also be tested with shifts in quantities that the operator observes, such as viscosity supplied as an input channel, or through changes in resolution and initial-condition distribution. Finally, repeating the comparison at native resolution on the full PDEBench datasets and extending it to harder equation families would show how broadly physics-informed training can serve as a data-efficiency tool for neural-operator surrogates.

7 Responsible Research

This section reflects on the research process behind the experimental comparison. Section 7.1 addresses reproducibility and integrity, including what a reader needs in order to regenerate the experiments, and which parts of the outcome cannot be guaranteed to repeat exactly. Section 7.2 addresses the ethical footprint of the work, including the origin and licensing of the data, the risk of misplaced trust in surrogate models, the environmental cost of training, and the use of generative AI tools.

7.1 Reproducibility and integrity

Reproducibility was treated as part of the experimental design rather than as an afterthought. Every run was executed inside the same Apptainer container, `physicsnemo_26.03.sif`, which fixes the software stack, and the exact package versions are listed in Appendix A. The random seeds fixed both the data split and the weight initialisation, and results are reported as mean ± standard deviation across seeds rather than as single-run outcomes. A configuration snapshot was saved for each run, so the exact training settings are documented and the sweep can be regenerated. The datasets are publicly archived, so the data side of the pipeline is available to any reader. The physics-weight ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}}$ is stated explicitly, together with the two PDEBench conventions that affect the Burgers residual, namely the ν/π diffusion factor and the $t \in [0, 2]$ time domain. These conventions were the main reproducibility pitfalls encountered in this project. Misapplying either one does not interrupt training but silently rescales the physics loss, which would alter every PINO result while leaving the FNO baseline untouched, so stating them explicitly is essential for anyone attempting replication.

The main caveat is that runs are not expected to be bitwise identical. cuDNN benchmarking was enabled and deterministic mode was disabled, so low-level GPU kernel selection can vary between executions, and HPC scheduler variability can also affect wall-clock measurements. The seeded data splits, the fixed container, the stored configurations and the evaluation protocol are therefore reproducible, while the exact GPU execution order and timing are not. This distinction is why the conclusions rest on averages across seeds rather than on any single run. The reporting also follows this principle of transparency. Unfavourable findings, including the Darcy out-of-distribution failure and the physics-weight settings that degraded accuracy, are presented alongside the favourable ones, so that the reported evidence is complete rather than selectively positive.

7.2 Ethical footprint

The direct ethical footprint of the work is small. Both benchmarks come from PDEBench [12], a public collection of simulated-physics datasets distributed through DaRUS under a CC BY 4.0 licence, with the accompanying PDEBench code released under the MIT licence. Attribution is given through citation, and the study involves no personal, human-subject or otherwise sensitive data. PDE surrogate models are general scientific-computing tools with limited direct misuse potential. The main practical risk is instead over-trust, since a surrogate can appear accurate on average while violating the governing equation, or while generalising poorly outside the tested distribution. The Darcy out-of-distribution results show that this failure mode is concrete rather than hypothetical. To reduce this risk, predictive accuracy is reported alongside a physical-consistency metric, the PDE residual, and the limitations of the out-of-distribution tests are made explicit in the Discussion.

A further consideration is the environmental cost of training. Energy use and the associated carbon emissions are an increasingly recognised cost of deep learning experiments [19], and a controlled sweep multiplies that cost by design. The main comparison alone required 42 seeded training runs per equation, with further runs for the physics-weight sensitivity analysis. The logged wall-clock times of the training runs behind this study sum to roughly 160 hours on a single A100 GPU. Assuming the 400 W rated board power of an A100, this corresponds to about 64 kWh of GPU energy, roughly one to two weeks of electricity use for an average Dutch household, based on the 2024 CBS average of 2,550 kWh per year for private dwellings [20], excluding node and cooling overhead. The footprint was kept modest by deliberate choices. Reusing a published benchmark avoided spending any solver time on data generation, the backbones are small by modern deep-learning standards, and every run fitted within a four-hour cap. The remaining cost was treated as a trade-off. Averaging across three seeds and reporting a sensitivity sweep multiplied the number of runs, but these repetitions are what make the uncertainty bands and the loss-balance conclusions trustworthy. Spending compute on reliability was therefore judged more responsible than reporting cheaper single-run results.

The data-efficiency findings themselves also carry an environmental implication. Outside benchmark settings, labelled training pairs must be produced by numerical solvers, so solver time is part of the energy budget of building a surrogate. On Darcy flow, PINO reached the accuracy of the data-only FNO with roughly ten times fewer labelled samples, and on Burgers' equation it clearly improved on the full-data FNO with half the labels. Physics-informed training can therefore reduce the energy cost of surrogate construction precisely where the surrogate is intended to amortise the cost of repeated simulation. This saving was not realised within this study, since the PDEBench data were generated once and published for reuse, but it strengthens the practical case for the method wherever new training data would otherwise have to be simulated.

Finally, generative AI tools, primarily ChatGPT and Claude, were used during the writing and development process, in line with the TU Delft guidelines on the use of generative AI in end projects. Their use was limited to proof-reading, clarity improvements, assistance with LaTeX issues, code bug fixing, git repository setup, brainstorming, and support in understanding the initial literature during the early reading stage. The tools were not used to accept claims uncritically or to replace scientific judgement. All generated content, code suggestions, interpretations and wording were reviewed and verified before inclusion. The experimental design, implementation, results, interpretation and final writing decisions are the author's own.

Acknowledgements

I would like to thank my supervisors, Dr. Jing Sun and Dr. Tiexing Wang, for their guidance, feedback and support throughout the Research Project. I am also grateful to Prof. Dr. Mathijs de Weerdt for taking on the role of examiner for this work. I would like to thank Christopher Wilczewski, Filip Nedić and Joris Timmermans for their collaboration, discussions and support during the project, as well as the peers who provided useful feedback through the university platform.

I would also like to thank the developers and maintainers of NVIDIA PhysicsNeMo, whose framework provided the neural-operator implementation used in this study. I am grateful to the PDEBench contributors for making the benchmark datasets and reference solutions publicly available, which made the controlled comparison possible. I also thank DelftBlue for providing the computing resources used to train and evaluate the models.

A Reproducibility details

Repository availability. The full implementation is openly available at `https://github.com/samuekisde/fno-pino-data-efficiency`. The repository contains the code, experiment configuration files, result-aggregation scripts, plotting scripts and a README. The README documents how to reproduce the runs and regenerate the figures and tables, so those instructions are not repeated here.

Software environment. All runs used the `physicsnemo_26.03.sif` Apptainer container with PhysicsNeMo 2.0.0, PyTorch `2.10.0a0` (NVIDIA build), CUDA 13.1, cuDNN 9.17.1, Python 3.12.3, NumPy 1.26.4, h5py 3.15.1, pandas 2.3.3, matplotlib 3.10.8 and PyYAML 6.0.3. The FNO implementation was provided by PhysicsNeMo [13], with no `neuraloperator` dependency.

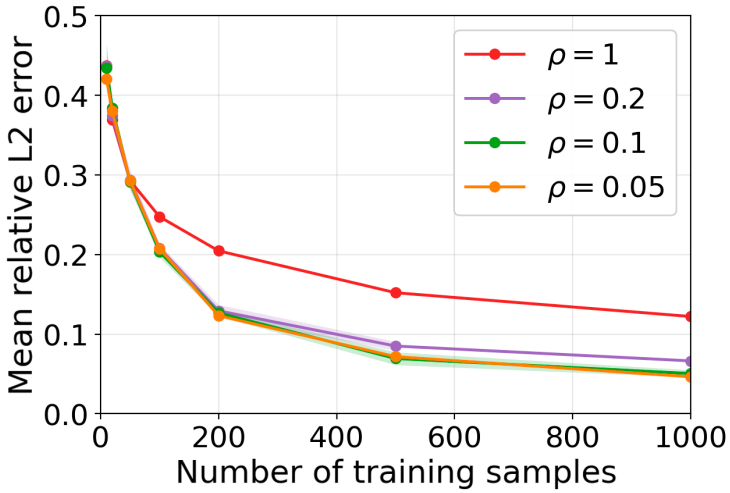
Data. Both benchmarks were taken from PDEBench [12] through its public DaRUS release. The in-distribution experiments used `2D_DarcyFlow_beta1.0_Train.hdf5` for Darcy flow and `1D_Burgers_Sols_Nu0.01.hdf5` for Burgers' equation. The out-of-distribution tests used the shifted settings $\beta = 0.1$ for Darcy flow and $\nu = 0.001$ for Burgers' equation, drawn from `2D_DarcyFlow_beta0.1_Train.hdf5` and `1D_Burgers_Sols_Nu0.001.hdf5`. All fields were kept in raw physical units with no normalisation, so that the PDE residual stays physically meaningful. Two PDEBench conventions are load-bearing for the Burgers residual and were applied throughout: the diffusion coefficient is ν/π rather than the bare viscosity, and the stored time domain is $t \in [0, 2]$ rather than $[0, 1]$. Misapplying either convention does not interrupt training but silently rescales the physics loss.

Experimental protocol. For each equation the labelled-data fraction was varied over $f \in \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0\}$ and the random seed over $s \in \{0, 1, 2\}$. Within each seed the training subsets were nested, so increasing f only added samples, and the validation and test partitions were fixed and shared by the FNO and PINO models. The two models differed only in the training objective, isolating the effect of the physics term. The main comparison therefore comprised 7 fractions $\times$ 3 seeds $\times$ 2 models = 42 runs per equation. The physics-informed runs used the headline physics-weight ratio $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}} = 0.1$, while further runs, not included in the count above, covered the larger ratios $\rho \in \{0.2, 1.0\}$ for the physics-weight sensitivity analysis and a physics-only ablation. For Burgers' equation the physics-informed objective additionally included a soft initial-condition penalty $\lambda_{\text{ic}} = \text{MSE}\big(\hat{u}(0, \cdot), u_0\big)$ with $\lambda_{\text{ic}} = \lambda_{\text{pde}} = 1$, following the initial-condition term of the PINO objective [10], which anchors the predicted $t = 0$ slice to the prescribed initial condition. The spatial periodicity was enforced structurally, so $\lambda_{\text{bc}} = 0$, and Darcy flow used neither additional term. All metrics are reported as mean $\pm$ standard deviation across the three seeds.

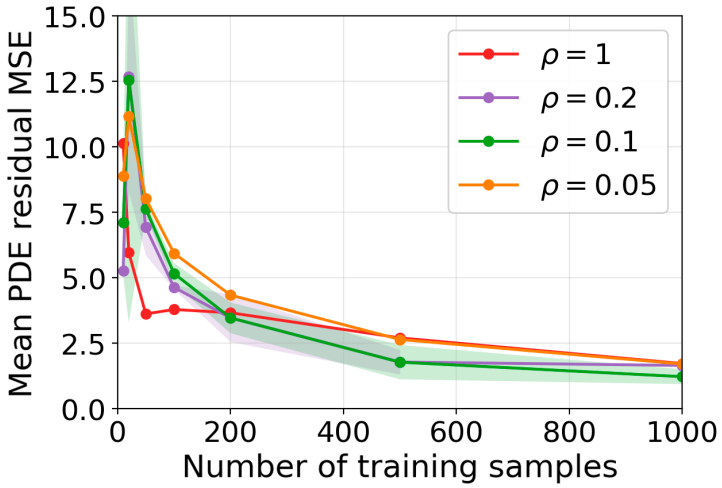
Reproducibility limitations. cuDNN benchmarking was enabled and deterministic mode was disabled, so low-level GPU kernel selection can vary between executions, as documented in PyTorch's reproducibility guidance [21]. The seeded data splits, fixed container, stored configurations and evaluation protocol are reproducible, and together determine the reported quantities. Exact bitwise GPU trajectories and wall-clock times are not guaranteed to repeat because of kernel selection and HPC scheduler variability. For this reason, the conclusions rest on seed-averaged results rather than any single run.

B Burgers physics-weight sensitivity

Figure 4 shows the Burgers' equation PINO weight sweep. It compares $\rho \in \{0.05, 0.1, 0.2, 1.0\}$ using the same labelled-data fractions as the main Burgers experiment.



(a) Relative L_2 error.

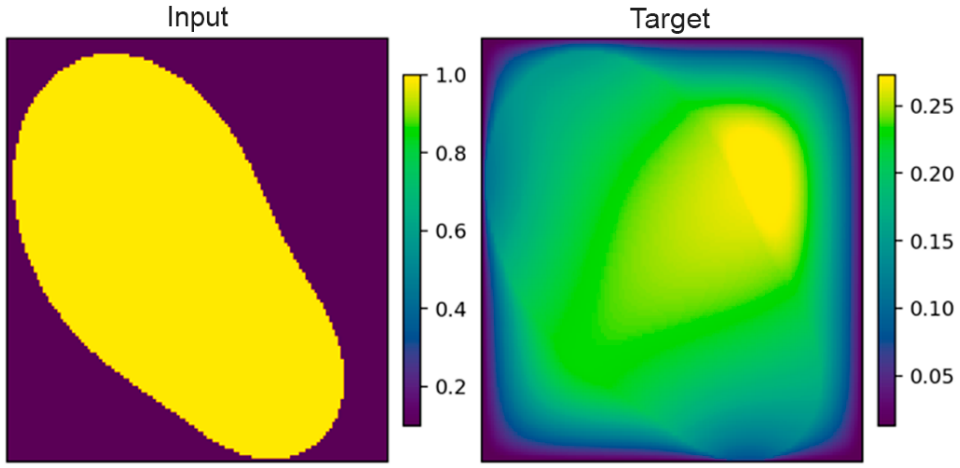


(b) PDE residual MSE.

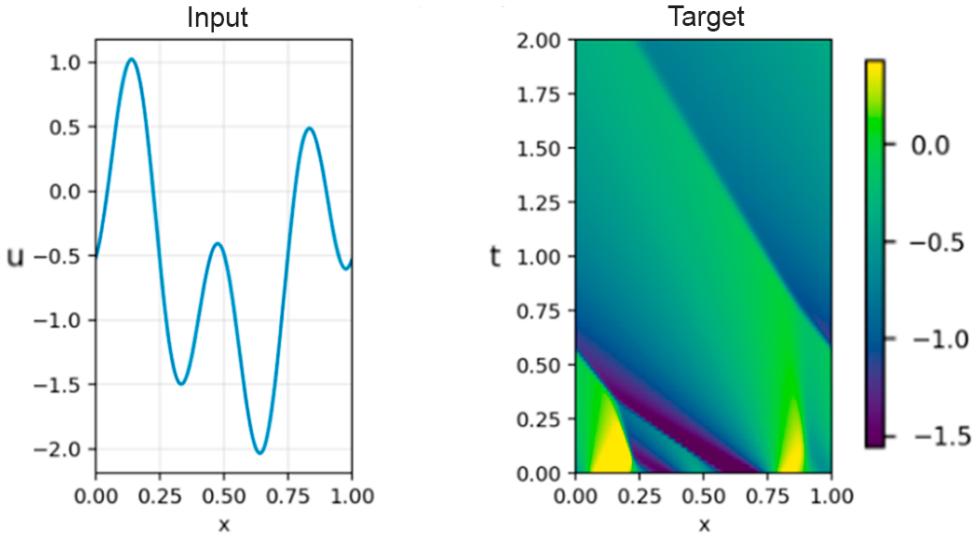
Figure 4: Physics-loss weight sensitivity for Burgers' equation. Mean relative L_2 error and mean PDE residual MSE are plotted against the number of labelled training samples for PINO models trained with $\rho = \lambda_{\text{pde}}/\lambda_{\text{data}} \in \{0.05, 0.1, 0.2, 1.0\}$.

C Dataset examples

Figure 5 shows representative input-target pairs from the two PDEBench benchmarks used in this study. Figure 5a shows the Darcy flow map from permeability to pressure, while Figure 5b shows the Burgers' equation map from an initial condition to a space-time trajectory.



(a) Darcy flow sample. The input is the permeability field k, and the target is the corresponding pressure field u.



(b) Burgers' equation sample. The input is the initial condition $u_0(x)$, and the target is the resulting space-time trajectory $u(t, x)$.

Figure 5: Example PDEBench samples used in this study. Panel (a) illustrates the Darcy flow input-to-target map, and panel (b) illustrates the Burgers' equation input-to-target map learnt by the neural operators.

D Out-of-distribution results

Table 5 gives the per-fraction out-of-distribution values behind Figure 3, for both benchmark equations. The Darcy test uses the shifted $\beta = 0.1$ setting and the Burgers test uses the shifted $\nu = 0.001$ setting.

		Rel. L_2			Residual MSE	
f	n	FNO	PINO	Gain	FNO	PINO
0.01	20	7.665 ± 0.288	7.464 ± 0.317	3%	5282	133
0.02	40	7.688 ± 0.136	7.605 ± 0.329	1%	4307	98
0.05	100	7.779 ± 0.049	7.668 ± 0.063	1%	2735	72
0.10	200	7.789 ± 0.115	7.772 ± 0.039	0%	2586	48
0.20	400	7.782 ± 0.060	7.834 ± 0.053	−1%	2782	32
0.50	1000	7.857 ± 0.100	7.864 ± 0.085	0%	2588	25
1.00	2000	7.890 ± 0.037	7.837 ± 0.013	1%	2786	22

(a) Darcy flow, $\beta = 0.1$.

		Rel. L_2			Residual MSE	
f	n	FNO	PINO	Gain	FNO	PINO
0.01	10	0.463 ± 0.004	0.453 ± 0.011	2%	2.1	2.8
0.02	20	0.414 ± 0.019	0.412 ± 0.011	0%	2.9	3.8
0.05	50	0.337 ± 0.006	0.326 ± 0.004	3%	2.1	2.2
0.10	100	0.286 ± 0.006	0.254 ± 0.007	11%	1.5	1.7
0.20	200	0.239 ± 0.001	0.194 ± 0.005	19%	1.5	1.5
0.50	500	0.204 ± 0.005	0.158 ± 0.002	23%	1.4	1.4
1.00	1000	0.194 ± 0.004	0.147 ± 0.001	24%	1.4	1.4

(b) Burgers' equation, $\nu = 0.001$.

Table 5: Out-of-distribution results for both benchmark equations against labelled-data fraction f and number of labelled training samples n. Accuracy is measured by relative L_2 error on the shifted test set, and physical consistency by PDE residual MSE. Relative L_2 values are the seed mean ± one standard deviation across the three seeds, and residual MSE is the seed mean. PINO uses $\rho = 0.1$. *Gain* is the relative reduction in relative L_2 error of PINO over FNO.

E Error-localisation metrics

This appendix reports the two error-localisation metrics introduced in Section 3. For Darcy flow, the metric is the boundary MSE on the four Dirichlet edges, where the prescribed value is $u = 0$. For Burgers' equation, the metric is the MSE at the initial time slice, where the prediction should match the prescribed initial condition $u_0(x)$. For Burgers' equation this $t = 0$ slice was also an explicit PINO training target, through the initial-condition penalty in Appendix A, so part of the PINO advantage in Table 6b is encouraged directly by the objective rather than being purely emergent. The Darcy boundary was not trained directly ($\lambda_{\text{bc}} = 0$), so Table 6a reflects an emergent effect of the residual. These metrics measure errors on regions with known values, and therefore expose constraint violations that can be hidden by global averages. Table 6 reports both metrics on the in-distribution test sets.

f	n	FNO	PINO	Factor
0.01	20	1.13×10^{-3}	9.19×10^{-5}	12.2
0.02	40	8.98×10^{-4}	6.49×10^{-5}	13.8
0.05	100	8.32×10^{-4}	6.39×10^{-5}	13.0
0.10	200	8.91×10^{-4}	3.74×10^{-5}	23.8
0.20	400	8.54×10^{-4}	2.09×10^{-5}	40.9
0.50	1000	5.90×10^{-4}	1.46×10^{-5}	40.3
1.00	2000	5.07×10^{-4}	1.21×10^{-5}	42.0

(a) Darcy flow, boundary MSE on the four Dirichlet edges, where $u = 0$.

f	n	FNO	PINO	Factor
0.01	10	8.51×10^{-2}	2.78×10^{-2}	3.1
0.02	20	6.37×10^{-2}	2.17×10^{-2}	2.9
0.05	50	2.42×10^{-2}	8.68×10^{-3}	2.8
0.10	100	1.54×10^{-2}	3.35×10^{-3}	4.6
0.20	200	8.37×10^{-3}	1.31×10^{-3}	6.4
0.50	500	4.68×10^{-3}	3.66×10^{-4}	12.8
1.00	1000	4.21×10^{-3}	1.78×10^{-4}	23.7

(b) Burgers' equation, MSE at $t = 0$ against the prescribed initial condition.

Table 6: Error-localisation metrics on the in-distribution test sets, averaged over three seeds, for the data-only FNO and PINO with $\rho = 0.1$. Panel (a) reports the boundary MSE on the four Dirichlet edges for Darcy flow, and panel (b) reports the MSE at $t = 0$ for Burgers' equation. Both metrics are computed on raw, unnormalised fields. The Factor column is the ratio of the FNO value to the PINO value, so a value above one means that PINO matches the known constraint more closely. Here f is the labelled-data fraction and n is the number of labelled training samples.

References

[1] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[2] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[3] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020. arXiv:2003.03485.

[4] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations. In *International Conference on Learning Representations*, 2021.

[5] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. U-FNO: An enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022.

[6] Wuyang Chen, Jialin Song, Pu Ren, Shashank Subramanian, Dmitriy Morozov, and Michael W. Mahoney. Data-efficient operator learning via unsupervised pretraining and in-context learning. In *Advances in Neural Information Processing Systems*, volume 37, pages 6213–6245, 2024.

[7] Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, 2019.

[8] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. 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, 2019.

[9] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021.

[10] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-Informed Neural Operator for Learning Partial Differential Equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024.

[11] Miguel Liu-Schiaffini, Julius Berner, Boris Bonev, Thorsten Kurth, Kamyar Azizzadenesheli, and Anima Anandkumar. Neural operators with localized integral and differential kernels. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32576–32594. PMLR, 2024.

[12] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 1596–1611, 2022.

[13] NVIDIA PhysicsNeMo Contributors. NVIDIA PhysicsNeMo: An open-source framework for physics-based deep learning in science and engineering. `https://github.com/NVIDIA/physicsnemo`, 2023. Accessed: 21 June 2026.

[14] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench Datasets. DaRUS, Version 8.0, 2022. doi:10.18419/DARUS-2986.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[16] Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.

[17] Min Zhu, Handi Zhang, Anran Jiao, George Em Karniadakis, and Lu Lu. Reliable extrapolation of deep neural operators informed by physics or sparse observations. *Computer Methods in Applied Mechanics and Engineering*, 412:116064, 2023.

[18] Taeyoung Kim and Myungjoo Kang. Approximating numerical fluxes using fourier neural operators for hyperbolic conservation laws. *Communications in Computational Physics*, 37(2):420–456, 2025.

[19] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics.

[20] Statistics Netherlands (CBS). Energy consumption private dwellings; type of dwelling and regions. CBS StatLine, `https://www.cbs.nl/en-gb/figures/detail/81528ENG`, 2025. Changed on 17 September 2025. Accessed: 21 June 2026.

[21] PyTorch Contributors. Reproducibility. PyTorch documentation, `https://docs.pytorch.org/docs/stable/notes/randomness.html`, 2025. Accessed: 21 June 2026.