# Physics-Informed Neural Operators for One-Dimensional Turbulence: Learning Realizations or Statistics?

Babak Goshayeshi
*[Affiliation]*
*[City, Country]*
diphyx@gmail.com

***Abstract*—One-Dimensional Turbulence (ODT) is a stochastic turbulence model whose eddy events cannot be predicted from the flow state. Using one-dimensional Fourier Neural Operators (FNOs) on a temporal planar jet, we show that a deterministic surrogate trained on individual realizations converges to the ensemble mean; trained on ensemble averages, it interpolates the mean dynamics to unseen jet parameters at 356× lower cost; and a hybrid that learns only the continuous diffusion while replaying eddy events explicitly reconstructs individual realizations to a relative error of 0.028.**

***Index Terms*—one-dimensional turbulence, neural operators, physics-informed machine learning, surrogate modeling, stochastic simulation**

## I. Introduction

ODT [1], [2] represents a turbulent flow along a single line: molecular diffusion advances the velocity profiles continuously, while stochastic eddy events—triplet maps sampled at random times, locations, and sizes—stand in for turbulent advection. Its low cost makes it a practical sandbox for a central question of surrogate modeling: should a neural surrogate of a stochastic simulation reproduce a realization, or the statistics? Three scenarios of increasing physical structure answer it.

## II. Model and Methods

### A. ODT Temporal Jet

The streamwise and transverse velocities $u(y, t)$ and $v(y, t)$ evolve between eddy events by viscous diffusion only,

$$\partial u/\partial t = \nu\, \partial^2 u/\partial y^2, \quad \partial v/\partial t = \nu\, \partial^2 v/\partial y^2, \tag{1}$$

from a smoothed top-hat jet of velocity $U_j$ and width $D$ with ambient Dirichlet boundaries, on a 1001-point grid to $t = 0.1$ s. An eddy ($y_0$, $\ell$) applies the triplet map with velocity kernels that redistribute kinetic energy while conserving momentum and energy [2].

### B. Neural Operator

All scenarios use the same 1-D FNO [3] built with NVIDIA PhysicsNeMo [4]: four Fourier layers, 64 channels, 32 retained modes, and 1,071,778 parameters. It maps ($u$, $v$, $y$) to the one-step increment ($\Delta u$, $\Delta v$) and is rolled out autoregressively for 100 steps from the initial condition.

## III. Scenarios

### A. S1: Training on Realizations

The FNO is trained on one-step pairs from 32 ODT realizations with different seeds. Because future eddies are not determined by the current state, the MSE-optimal deterministic prediction is the conditional mean of the increment.

### B. S2: Training on Ensemble Averages

A 3×3 grid of jet velocities (6, 10, 14 m/s) and widths (6, 10, 14 mm) is simulated with 40 realizations per combination (360 runs), and each ensemble is reduced to its mean trajectory. The FNO trains on eight combinations; the combination $U_j = 10$ m/s, $D = 10$ mm is held out entirely.

### C. S3: Physics-Informed Advance with Explicit Eddies

Mirroring the operator splitting of the ODT solver, the FNO learns only the continuous diffusion. It is trained on 8,080 one-step states from the S2 realizations with exact-solver targets and the composite loss

$$L = \|\Delta\hat{u} - \Delta u\|^2 + 0.1\, \|\partial_t u - \nu\, \partial_y{}^2 u\|^2_{\mathrm{mid}}, \tag{2}$$

evaluated at the predicted midpoint, in the spirit of physics-informed learning [5], [6]. During rollout, the recorded eddy sequence of the realization (227 events) is replayed with the ODT triplet map, with kernel coefficients recomputed from the AI-predicted state.

## IV. Results

TABLE I
ROLLOUT RELATIVE L2 ERROR OF u (HELD-OUT JET FOR S2 AND S3)

| Scenario and reference | t = 25% | t = 50% | final |
|---|---|---|---|
| S1 vs. single realization | 0.254 | 0.194 | 0.253 |
| S1 vs. ensemble average | 0.165 | 0.084 | 0.140 |
| S2 vs. ensemble average | 0.117 | 0.102 | 0.083 |
| S3 vs. single realization | 0.071 | 0.037 | 0.028 |

Table I and Fig. 1 summarize the rollout errors. In S1, the relative $L_2$ error against the held-out realization is 0.19–0.25, but against the ensemble average only 0.08–0.17: trained on noise, the network learns the mean without ever seeing an averaged field. In S2, the held-out error stays near 0.1 (0.083 at the final time), so the operator interpolates the mean dynamics across jet parameters. In S3, the hybrid tracks the individual realization itself, reaching 0.028 on the held-out jet (0.041 over training combinations).

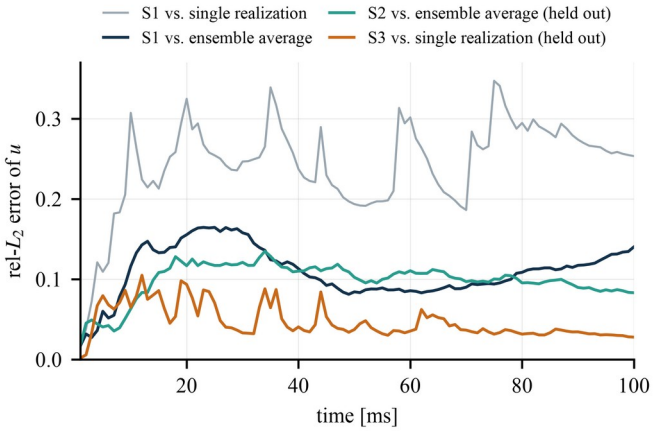



Fig. 1. Relative $L_2$ error of the streamwise velocity during 100-step autoregressive rollouts. S2 and S3 curves are for the held-out jet ($U_j$ = 10 m/s, $D$ = 10 mm).

One ODT realization takes 3.7 s on a CPU; an FNO rollout takes 0.42–0.44 s. A rollout in S1 or S2 replaces a 32- or 40-run ensemble (268× and 356× speedups); S3 replaces a single realization (9×). Training took 11–64 s on an Apple-silicon GPU, so generating the ODT data (up to 1,346 s) dominates the cost.

## V. Conclusions

Deterministic neural operators trained on stochastic simulations whose random events are unpredictable from the state learn conditional-mean dynamics. Ensemble-mean prediction is therefore the natural target for a deterministic surrogate, and it generalizes across jet parameters. Reconstructing individual realizations requires keeping the stochastic mechanism explicit: a physics-informed operator for the continuous dynamics plus explicit eddy events. Future work will replace the replayed eddies with a learned stochastic event model and extend the approach to reacting ODT. Code and data: github.com/bgoshayeshi/AI-ODT.

## References


[1] A. R. Kerstein, “One-dimensional turbulence: Model formulation and application to homogeneous turbulence, shear flows, and buoyant stratified flows,” *J. Fluid Mech.*, vol. 392, pp. 277–334, 1999.

[2] A. R. Kerstein, W. T. Ashurst, S. Wunsch, and V. Nilsen, “One-dimensional turbulence: Vector formulation and application to free shear flows,” *J. Fluid Mech.*, vol. 447, pp. 85–109, 2001.

[3] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, “Fourier neural operator for parametric partial differential equations,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.

[4] NVIDIA, “PhysicsNeMo: An open-source framework for physics-based deep learning.” [Online]. Available: https://github.com/NVIDIA/physicsnemo

[5] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *J. Comput. Phys.*, vol. 378, pp. 686–707, 2019.

[6] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, and A. Anandkumar, “Physics-informed neural operator for learning partial differential equations,” *ACM/IMS J. Data Sci.*, vol. 1, no. 3, 2024.