

Teaching Neural Operators the Local Physics of Hyperbolic Conservation Laws

Taeyoung Kim, KIAS

May 30, 2025

Table of Contents

- 1 Overview of Neural Operators
- 2 Flux Neural Operator
- 3 Flux Neural Operator for Ideal Magnetohydrodynamics

Table of Contents

1 Overview of Neural Operators

2 Flux Neural Operator

3 Flux Neural Operator for Ideal Magnetohydrodynamics

Motivation for Neural Operators

Consider the following linear PDE problem as an example:

$$\begin{aligned}Lu &= f \quad \text{on } D \\ u &= 0 \quad \text{on } \partial D.\end{aligned}$$

Where L is a linear differential operator. The solution to this problem can be expressed as the convolution of a kernel function $G(x, x')$ which depends on L and D with a function f .

Motivation for Neural Operators

Mathematically, this can be represented as follows:

$$u(x) = G * f(x) = \int_D G(x, x') f(x') dx.$$

In this Dirichlet problem, the solution can be regarded as an integral operator that maps the source term f to the solution u .

Definition of Neural Operators

Neural Operator is composed as following:

$$\mathcal{G}_\theta = \mathcal{N}_Q \circ \mathcal{A}_L \circ \cdots \circ \mathcal{A}_1 \circ \mathcal{N}_P.$$

Where each \mathcal{N}_Q , \mathcal{N}_P , \mathcal{A}_i are projection layer, lifting layer, and kernel integration layers respectively.

Definition of Neural Operators

Here, each kernel integration layer \mathcal{A}_i is defined as follows:

$$\mathcal{A}_i(v) := \sigma(W_{loc,i}(v) + \mathcal{K}_i(v)).$$

Where $W_{loc,i}$ is a sub-network that acts locally, and \mathcal{K}_i is an integral kernel operator represented as follows:

$$\mathcal{K}_i(v)(x) = \int_{D_{i-1}} k^i(x, y) v(y) d\mu(y).$$

Where $k^i \in C(D_i \times D_{i-1}; \mathbb{R}^{d_{v_i} \times d_{v_{i-1}}})$ and μ is a measure on domain D_{i-1} .

Discretization of functional Data

Neural Operators process functional data, but to numerically handle functions, it is necessary to convert them into finite-sized data. To achieve this, we select specific points $\tilde{D} = \{x_1, \dots, x_n\} \subset D$ in the domain D on which functions will be evaluated.

As a result, the input function a and target function u can be represented as vectors $a|_{\tilde{D}} \in \mathbb{R}^{n \times d_a}$ and $u|_{\tilde{D}} \in \mathbb{R}^{n \times d_a}$.

Problem Setting

The main task of Neural Operators is approximate operator $\mathcal{G}(a) = u$. To achieve this, we solve following minimization problem:

$$\min_{\theta} \mathbb{E}_{a \sim \mu} C(\mathcal{G}(a), \mathcal{G}_{\theta}(a))$$

For this, we aim to minimize the empirical loss function, which measures the difference between $\mathcal{G}_{\theta}(a_i)$ and $u_i = \mathcal{G}(a_i)$ on the training dataset.

The **Fourier Neural Operator** is implemented via treating convolution operator in kernel integration layer by Fourier transform:

$$\mathcal{K}(v)(x) = \int_D \mathbf{k}(x - y)v(y)d\mu(y) = \mathcal{F}^{-1}\left(\mathbf{R}_k \cdot (\mathcal{F}v)\right)(x).$$

Since we treat discretized objects, the kernel integration can be rewritten as follows:

$$\hat{\mathcal{F}}\left(\mathcal{K}(v|\tilde{D})\right)_{k,l} = \sum_{j=1}^{d_v} R_{k,l,j} \cdot (\hat{\mathcal{F}}v)_{k,j}$$

where $\hat{\mathcal{F}}$ denotes the Fast Fourier Transform (FFT). And we only use truncated frequency components. The truncation of frequencies is expressed by the maximal number of modes:

$$k_{\max} = |\{k \in \mathbb{Z}^n : |k_j| \leq k_{\max,j} \text{ for } j = 1, \dots, n\}|$$

Table of Contents

- 1 Overview of Neural Operators
- 2 Flux Neural Operator**
- 3 Flux Neural Operator for Ideal Magnetohydrodynamics

Hyperbolic Conservation Laws

The conservation law can be rewritten in quasi-linear form as follows:

$$U_t + \frac{\partial F(U)}{\partial U} \frac{\partial U}{\partial x} = 0.$$

If the Jacobian $\frac{\partial F(U)}{\partial U}$ has m real eigenvalues and is diagonalizable, we say that the above equation is hyperbolic.

A theorem exists regarding certain types of numerical methods that guarantee good quality. These methods are called "conservative methods" and have the following form:

$$U_j^{n+1} = U_j^n - \frac{k}{h} [\hat{F}(U_{j-p}^n, \dots, U_{j+q}^n) - \hat{F}(U_{j-p-1}^n, \dots, U_{j+q-1}^n)] \quad (1)$$

We define the numerical flux as consistent when the following conditions are satisfied:

$$\begin{aligned} \hat{F}(u, \dots, u) &= F(u), \quad \forall u \in \mathbb{R} \\ |\hat{F}(U_{j-p}, \dots, U_{j+q}) - F(u)| &\leq K \max_{-p \leq i \leq q} |U_{j+i} - u|. \end{aligned} \quad (2)$$

Loss for Flux NO

Motivated by equations (1) and (2), we design the loss function for our Flux NO model as follows:

$$\mathcal{L}_{tm}(U) = \sum_{n=0}^N \|U^{n+1} - U^n + \frac{t_n}{k}[G(U_{-p}^n, \dots, U_{+q}^n; \theta) - G(U_{-p-1}^n, \dots, U_{+q-1}^n; \theta)]\|_2^2.$$

$$\mathcal{L}_{consi}(U) = \sum_{n=0}^N \|G(U^n, \dots, U^n; \theta) - F(U^n)\|_2^2.$$

$$\mathcal{L}(\{U_i\}, G(\cdot; \theta)) = \sum_{i=1}^m (\mathcal{L}_{tm}(U_i) + \lambda \mathcal{L}_{consi}(U_i)), \quad 0 \leq \lambda.$$

Here, U corresponds to the vectorized functional data, and the data structure follows the format $[\text{batch size}, N_t, N_x, N_p]$. G represents the Neural Operator model, which takes as input the function concatenated along the last index. In our case, we used FNO for G .

Algorithm for Flux NO

Algorithm 3 An algorithm for training (Combined with TVD-RK)

Input: Dataset $\mathcal{U} = ((U_{b,i,j,1}), (\Delta t_{b,i}))$

Output: trained FNO model $G(\cdot; \theta)$

for epoch = 1, ..., E **do**

for Batch \in Train loader **do**

$\tilde{U}_{-j}^n \leftarrow$ roll $U_{b,n,\cdot,1}$ by \tilde{j} in the third index for $\tilde{j} = -q, \dots, p+1$

$U^l \leftarrow$ concatenate $(\tilde{U}_{-p}^n, \dots, \tilde{U}_q^n)$ along fourth index

$U^r \leftarrow$ concatenate $(\tilde{U}_{-p-1}^n, \dots, \tilde{U}_{q-1}^n)$ along fourth index \triangleright Thus, the concatenated function is now a $p+q$ dimension vector-valued

$U^0 \leftarrow U$

for $\tilde{k} = 1, \dots, l$ **do**

$\hat{U}^{\tilde{k}} \leftarrow \frac{1}{k} [G(U^{\tilde{k},l}, \theta) - G(U^{\tilde{k},r}; \theta)]$

$U^{\tilde{k},\cdot} \leftarrow \sum_{s=0}^{\tilde{k}-1} (\alpha_{\tilde{k}s} U^{s,\cdot} + \Delta t_{b,n} \beta_{\tilde{k}s} \hat{U}^{s,\cdot}) \triangleright$ Each α and β are selected to satisfy the CFL condition

end for

$\mathcal{L}_{tm}(\text{Batch}) \leftarrow \sum_{b=1}^B \sum_{n=0}^{N_t-1} \|U_{b,n+1,\cdot,\cdot} - U_{b,n,\cdot,\cdot}^l\|_2^2$

$V^{p+q} \leftarrow$ concatenate U $p+q$ times.

$\mathcal{L}_{consi}(\text{Batch}) \leftarrow \sum_{b=1}^B \sum_{n=0}^{N_t-1} \|G(V_{b,n,\cdot,\cdot}^{p+q}; \theta) - F(U_{b,n,\cdot,\cdot})\|_2^2$

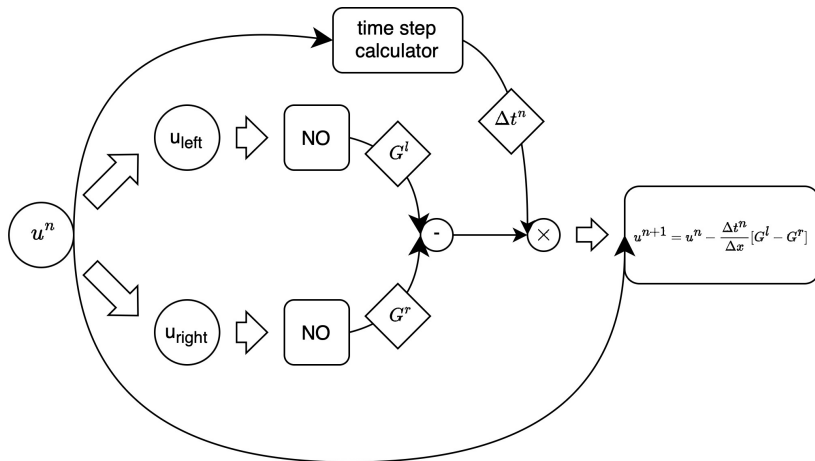
 Calculate backpropagation for $\mathcal{L}_{tm}(U) + \lambda \mathcal{L}_{consi}(U)$

 Take an optimization step.

end for

end for

Algorithm for Flux NO



Experiments (1D Burgers)

We conducted experiments on 1D linear advection and the Burgers' equation, which are simple types of HCLs.

$$\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0$$

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

Experiments (1D Burgers)

	$(\Delta t, \Delta x)$	Number of functions	Domain of function	Overall shape of dataset
Training for B.E	$(10^{-2}2^{-8}, 2^{-8})$	10	$[0, 0.3] \times [0, 1]$	$[760, 100, 256, 1]$
Test for B.E	$(10^{-2}2^{-8}, 2^{-8})$	10	$[0, 0.6] \times [0, 1]$	$[10, 1520, 256, 1]$

Table: Specifications of training and testing datasets

Experiments (1D Burgers)

	width	depth of Fourier layers	Number of modes	Batch size (Advection, Burgers)
Flux FNO	64	1	5	(1, 1)
1D FNO	64	1	5	(1, 1)
1D FNO(heavy)	32	3	20	(1, 1)
2D FNO	64	3	(10, 10)	(10, 10)

Table: Specification of architectures and hyperparameters

Experiments (1D Burgers)

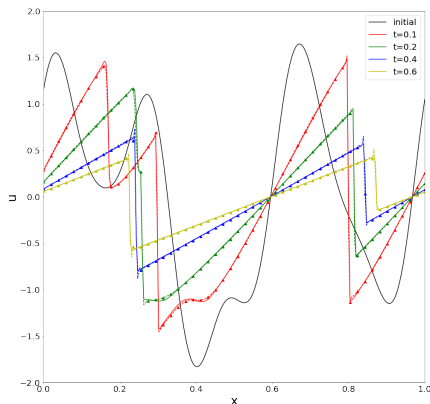


Figure: Output of Flux FNO (dashed line with triangle markers) compared with the exact solutions (solid line) for the 1D Burgers' equation problem.

Experiments (1D Burgers)

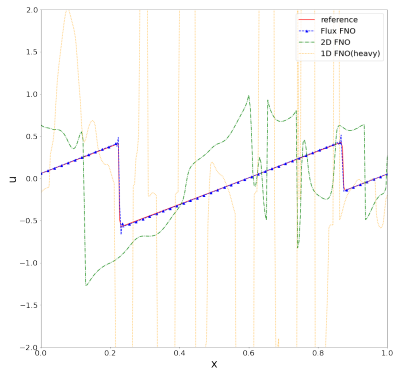
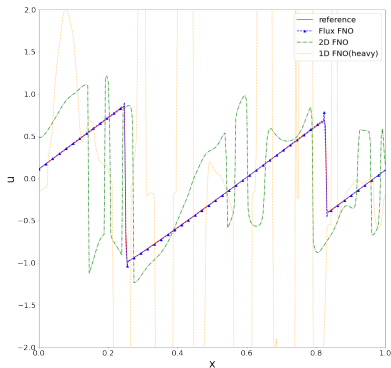


Figure: Comparison of Flux FNO output with the exact solutions and other FNO models at $t = 0.30$ (left), and $t = 0.60$ (right) for the 1D Burgers' equation problem.

Experiments (1D Burgers)

(relative L^2 , L^∞)	t=0.30	t=0.60
Flux FNO	(0.049, 0.21)	(0.052, 0.13)
1D FNO(heavy)	(6.55, 7.53)	(11.20, 7.32)
1D FNO	10.08, 4.94)	(19.09, 4.81)
2D FNO	(1.36, 1.65)	(2.21, 1.26)

Table: Quantitative results of each model for the 1D Burgers' equation problem. Each value represents the mean over the test dataset.

Experiments (1D Burgers)

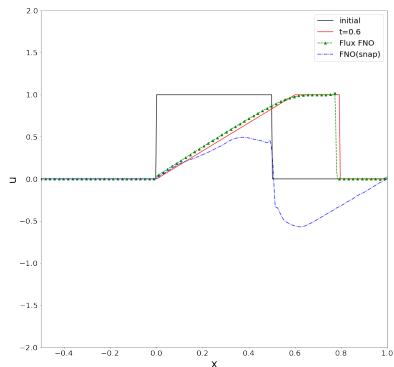
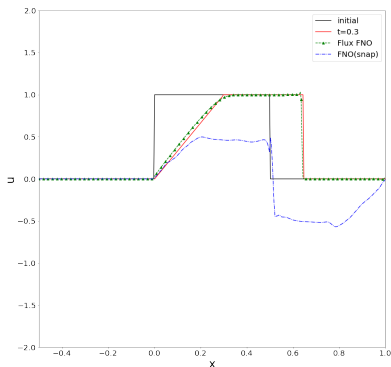


Figure: Inference of Flux FNO on out-of-distribution samples for the 1D Burgers equation problem: square wave.

Experiments (1D Shallow water)

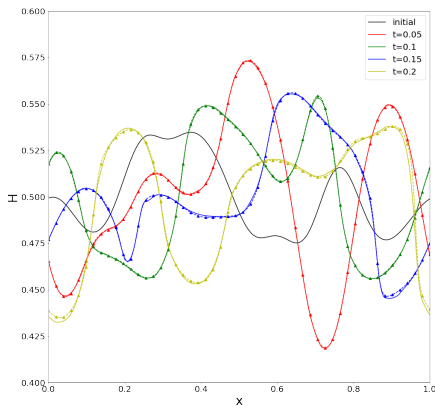


Figure: Output (H) of Flux FNO (dashed line with triangle markers) compared with the exact solutions (solid line) for the 1D Shallow water equation problem.

Experiments (1D Shallow water)

(relative L^2 , L^∞)	t=0.05	t=0.15
Flux FNO	(3.58e-3, 4.57e-3)	(9.56e-3, 6.26e-3)
2D FNO	(7.74e-3, 9.80e-3)	(1.59e-2, 1.55e-2)

Table: Quantitative results of each model for the 1D Shallow water equation problem. Each value represents the mean over the test dataset.

Experiments (1D Shallow water)

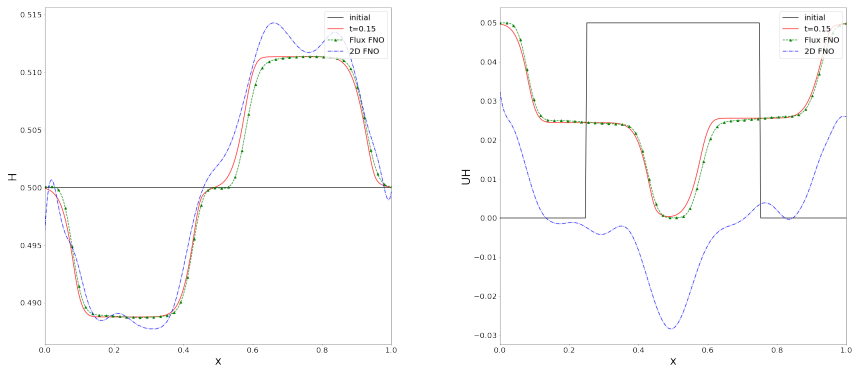


Figure: Inference of Flux FNO on out-of-distribution samples for the 1D Shallow water equation problem with initial condition is square pulse: U (left), UH (right).

Table of Contents

- 1 Overview of Neural Operators
- 2 Flux Neural Operator
- 3 Flux Neural Operator for Ideal Magnetohydrodynamics

Ideal Magnetohydrodynamics (Ideal MHD)

$$\begin{aligned}\rho_t + \nabla \cdot (\rho \mathbf{u}) &= 0, \\ (\rho \mathbf{u})_t + \nabla \cdot \left[\rho \mathbf{u} \otimes \mathbf{u} + \left(p + \frac{1}{2} \|\mathbf{B}\|^2 \right) \mathbf{I} - \mathbf{B} \otimes \mathbf{B} \right] &= 0, \\ \mathbf{B}_t + \nabla \cdot (\mathbf{u} \otimes \mathbf{B} - \mathbf{B} \otimes \mathbf{u}) &= 0, \\ \mathcal{E}_t + \nabla \cdot \left[\left(\mathcal{E} + p + \frac{1}{2} \|\mathbf{B}\|^2 \right) \mathbf{u} - \mathbf{B}(\mathbf{u} \cdot \mathbf{B}) \right] &= 0.\end{aligned}$$

where \mathbf{u} represents velocity, \mathbf{B} the magnetic field, ρ density, E energy, and p pressure. Additionally, according to Maxwell's equations, the magnetic field must satisfy the divergence-free condition, which is expressed as:

$$\nabla \cdot \mathbf{B} = 0.$$

Enhanced Loss Function for Ideal MHD

Motivated from TVD property of numerical scheme, we devise following loss:

$$\mathcal{L}_{TVD}(\{\mathbf{u}_i\}_{i=1}^B) := \sum_{i=1}^B \sum_{n=1}^{N_t-1} [TV(\tilde{\mathbf{u}}_{i,n+1}) - TV(\mathbf{u}_{i,n})]_+^2.$$

where $[\cdot]_+ := \max(0, x)$ and $\tilde{\mathbf{u}}$ is an output of Flux NO.

Enhanced Loss Function for Ideal MHD

To obtain divergence freeness of magnetic field, we consider following equation:

$$\begin{aligned}\nabla \cdot \frac{\partial \mathbf{U}}{\partial t} &= \frac{\partial \nabla \cdot \mathbf{U}}{\partial t} = \nabla \cdot \left(-\frac{\partial \mathbf{G}}{\partial x} - \frac{\partial \mathbf{F}}{\partial y} \right) \\ \Rightarrow 0 &= \frac{\partial \nabla \cdot \mathbf{B}}{\partial t} = \nabla \cdot \left(-\frac{\partial \mathbf{G}}{\partial x} - \frac{\partial \mathbf{F}}{\partial y} \right)_{\mathbf{B}}.\end{aligned}$$

From above equation, we get following:

$$\begin{aligned}\mathcal{L}_{div}(\{\mathbf{U}_i\}_{i=1}^B) &:= \\ \sum_{i=1}^B \sum_{j=1}^{N_t-1} &\left[\frac{\|\nabla \cdot \Delta \mathbf{R}(\mathbf{U}_{i,j}; \theta)_{\mathbf{B}}\|_2^2 - \theta_{div}}{\|\nabla \cdot \Delta \mathbf{R}(\mathbf{U}_{i,j}; \theta)_{\mathbf{B}}\|_2^2 - \theta_{div}} \right]_+ \|\nabla \cdot \Delta \mathbf{R}(\mathbf{U}_{i,j}; \theta)_{\mathbf{B}}\|_2^2.\end{aligned}$$

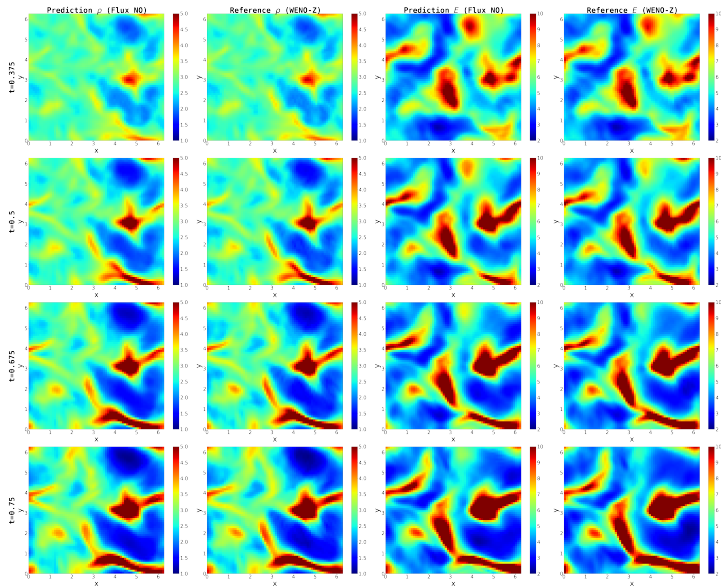
where θ_{div} is the threshold value.

Enhanced Loss Function for Ideal MHD

Utilizing the l^∞ norm allows for the expectation of pointwise convergence, thereby enabling a more accurate approximation of the flux embedded in the training dataset.

$$\mathcal{L}_\infty(\{\mathbf{U}_i\}_{i=1}^B) := \sum_{i=1}^B \sum_{n=1}^{N_t-1} \sum_{l=1}^{N_u} \sup_{j,k} \left((\mathbf{U}_{i,n+1,j,k,l} - \mathbf{U}_{i,n,j,k,l}) - \Delta \mathbf{R}(\mathbf{U}_{i,n,j,k,l}; \theta) \right).$$

Experiments (2D Ideal MHD)

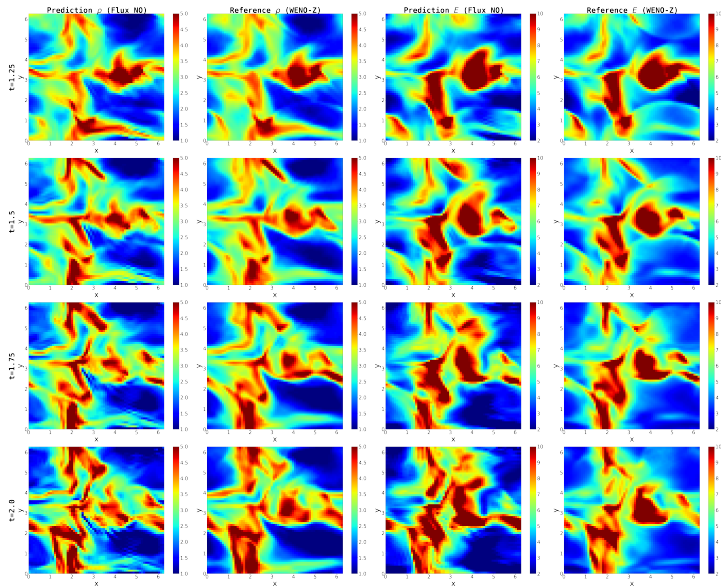


Experiments (2D Ideal MHD)

(M, SD)	t=0.5	t=0.75
rel l_ρ^2	(2.5e-3, 1.11e-1)	(6.7e-3, 1.11e-1)
rel l_ρ^∞	(4.24e-7, 1.09e-1)	(5.51e-6, 1.09e-1)
rel l_u^2	(4.7e-3, 1.11e-1)	(1.02e-2, 1.11e-1)
rel l_u^∞	(3e-4, 1.10e-1)	(1e-3, 1.10e-1)
rel l_B^2	(3.5e-3, 1.11e-1)	(7.8e-3, 1.11e-1)
rel l_B^∞	(2e-4, 1.10e-1)	(7e-4, 1.10e-1)
rel l_E^2	(3e-3, 1.11e-1)	(8.1e-3, 1.11e-1)
rel l_E^∞	(1.6e-3, 1.10e-1)	(2.7e-3, 1.11e-1)

Table: Means (M) and Standard Deviations (SD) of Relative l^2 and l^∞ Norms Between the Output of Flux NO and Reference for Each Component at Short Term Times in the Two-Dimensional Case, Across 10 Test Samples.

Experiments (2D Ideal MHD)



Experiments (2D Ideal MHD)

(M, SD)	t=1.5	t=2.0
rel l_ρ^2	(1.62e-2, 1.11e-1)	(2.45e-2, 1.11e-1)
rel l_ρ^∞	(2.78e-5, 1.10e-1)	(4.90e-5, 1.10e-1)
rel l_u^2	(2.91e-2, 1.11e-1)	(4.68e-2, 1.11e-1)
rel l_u^∞	(6.6e-3, 1.11e-1)	(1.98e-2, 1.11e-1)
rel l_B^2	(2.39e-2, 1.11e-1)	(3.59e-2, 1.11e-1)
rel l_B^∞	(9.2e-3, 1.11e-1)	(1.63e-2, 1.11e-1)
rel l_E^2	(1.69e-2, 1.11e-1)	(3.16e-2, 1.11e-1)
rel l_E^∞	(8.6e-3, 1.11e-1)	(4.4e-2, 1.11e-1)

Table: Means (M) and Standard Deviations (SD) of Relative l^2 and l^∞ Norms Between the Output of Flux NO and Reference for Each Component at Long Term Times in the Two-Dimensional Case, Across 10 Test Samples.

Experiments (2D Ideal MHD)

To test our model on OOD sample, we addressed the Orszag-Tang problem. The initial conditions for the Orszag-Tang problem are described as follows:

$$\begin{aligned}\rho(x, y, 0) &= \gamma^2, & v_x(x, y, 0) &= -\sin y, & v_y(x, y, 0) &= \sin x, \\ \rho(x, y, 0) &= \gamma, & B_x(x, y, 0) &= -\sin y, & B_y(x, y, 0) &= \sin 2x, \\ u_z(x, y, 0) &= B_z(x, y, 0) = 0.\end{aligned}$$

When given the initial conditions of the Orszag-Tang problem, the solution at $t = 0.25$ was used as the initial condition.

Experiments (2D Ideal MHD)

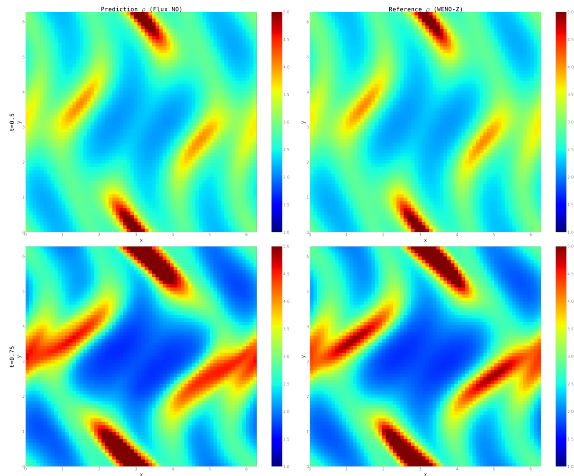


Figure: Snapshot of the Output from Flux NO and Reference Data at $t = 0.5$ and $t = 0.75$ for the Orszag-Tang Vortex Problem.

Experiments (2D Ideal MHD)

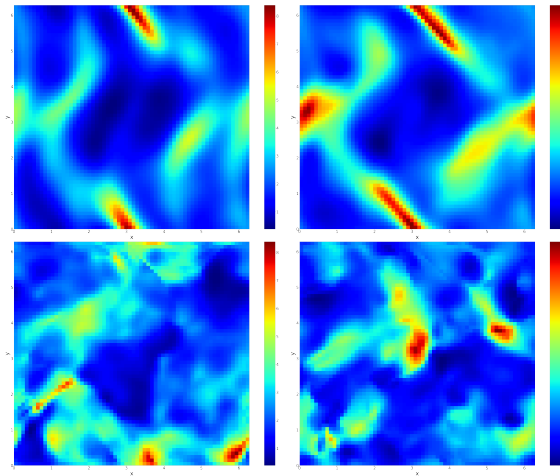


Figure: Snapshot of the Output from 3D FNO at $t = 0.5$ (top left), $t = 0.75$ (top right), $t = 2.0$ (bottom left) and $t = 3.0$ (bottom right) for the Orszag-Tang Vortex Problem.

Experiments (2D Ideal MHD)

Models	Inference Time for $\Delta t = 0.5$	Number of Parameters
Flux NO	4.16e-1s (4.16e-3s)	8,204,176
2D FNO	3.04e-3s	1,022,264
2D FNO (heavy)	4.74e-3s	8,355,064
3D FNO	1.81e-2s (9.05e-3s)	7,990,064
WENO-Z	1.05e+1s (1.05e-1s)	
WENO-Z (on PyTorch)	9.98e+0s (9.98e-2s)	

Table: Comparisons of Flux NO with Other Standard FNO Models: Memory Requirements and Inference Times. The time in parentheses represents the inference time for a single run.

- Kim, T., and M. Kang, 2024: Bounding the rademacher complexity of fourier neural operators. *Machine Learning*, 113, 2467–2498.
- Kim, T., and M. Kang, 2025: Approximating numerical fluxes using fourier neural operators for hyperbolic conservation laws. *Communications in Computational Physics*.
- T. Kim, Y. Ha, and M. Kang, 2024: Neural operators learn the local physics of magnetohydrodynamics. *arXiv*, arXiv:2404.16015.
- Kovachki, N., Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, 2021b: Neural operator: learning maps between function spaces. *arXiv*, arXiv:2108.08481.

Thank you for listening!