arXiv:2608.09764v1 [cs.LG] 10 Aug 2026

MoNo: Multiscale Optimal Transport Neural Operator for Solving PDEs on General Geometries

Zijiang Yang[1], Xiaomeng Wu[1], Dongmei Fu[12]*

[1]School of Automation and Electrical Engineering, University of Science and Technology Beijing
[2]Beijing Engineering Research Center of Industrial Spectrum Imaging
zijiangyang@xs.ustb.edu.cn, fdm_ustb@ustb.edu.cn

Abstract

Transformer-based neural operators have achieved substantial progress in solving Partial Differential Equations (PDEs) by projecting spatial observations into compact latent tokens and learning physical interactions in latent spaces. However, we reveal that existing learnable projection mechanisms cannot ensure stable and balanced assignments from observation points to latent tokens, causing some latent tokens to be overassigned while others remain underutilized. This limitation further restricts the design of hierarchical architectures, as assignment imbalance is continuously inherited and amplified across latent spaces, eventually causing severe token collapse in deeper spaces. To address these issues, we propose **MoNo** (**M**ultiscale **O**ptimal Transport **N**eural **O**perator), a progressive multiscale neural operator that efficiently solves PDEs on general geometries through stable latent-space construction. At its core is **CoTAP** (**C**ross-scale **O**ptimal **T**ransport **A**ssignment and **P**rojection), a novel latent-space construction method that formulates cross-space assignment between adjacent spaces as an entropy-regularized optimal transport problem, thereby constructing balanced bidirectional projections and stable latent spaces. CoTAP also ensures stable information transfer across multiple latent spaces, further enabling multiscale architectures on general geometries, which in turn support more efficient learning of long-range physical interactions. Extensive experiments demonstrate that MoNo outperforms existing state-of-the-art neural operators in both prediction performance and computational efficiency. Code is available at https://github.com/ZijiangY1116/MoNo.

Introduction

Partial Differential Equations (PDEs) are fundamental tools for describing the evolution of physical systems (Raissi, Yazdani, and Karniadakis 2020; Karniadakis et al. 2021; Wang et al. 2025). Neural operators (Li et al. 2020a), particularly Transformer-based variants (Kovachki et al. 2023; Luo et al. 2025a), have substantially advanced efficient PDE solving by learning operator mappings from input observations to target physical fields through sequence-based representations. As standard self-attention scales quadratically with the number of observation points (Vaswani et al. 2017), vanilla Transformer-based neural operators are computationally prohibitive for large-scale PDE problems (Wu et al. 2024), motivating extensive research on performing attention in compact latent spaces (Wang and Wang 2024; Alkin et al. 2024; Hu et al. 2026). However, the mechanisms for constructing latent spaces and the stability of cross-space mappings remain underexplored, limiting efficient compression of observation sequences and constraining flexibility in architectural design. As illustrated in Figure 1, SOTA methods not only exhibit limited prediction performance but also remain computationally expensive despite employing lower-complexity attention mechanisms (Hu et al. 2026).

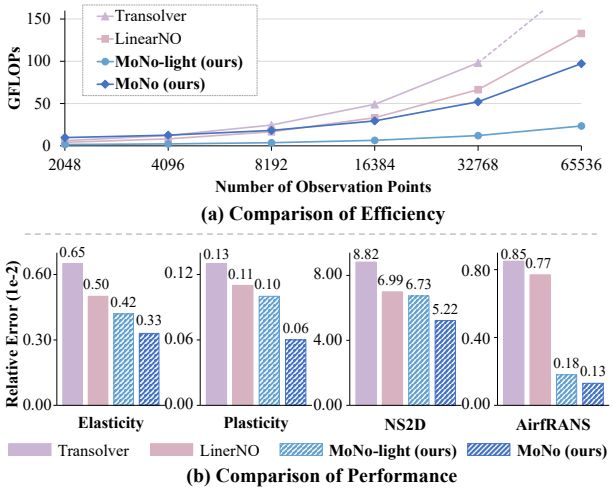
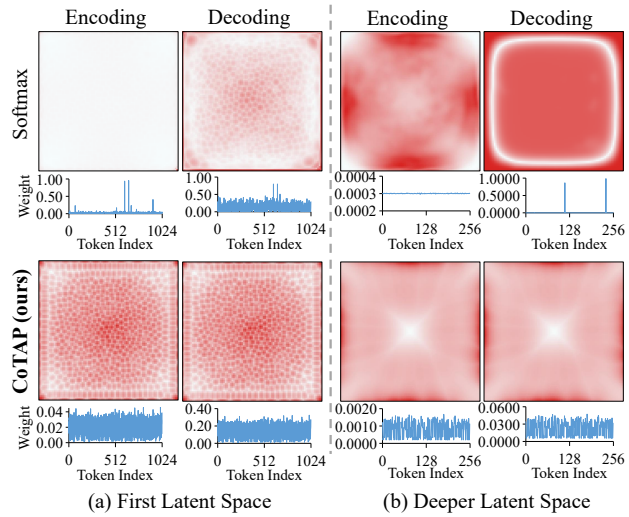


Figure 1: Comparison of MoNo with State-Of-The-Art (SOTA) methods. (a) MoNo achieves higher computational efficiency. (b) MoNo achieves lower prediction errors.

To construct latent spaces, softmax-based projection is widely adopted, where the assignments from each observation point in the observation space to latent tokens in the latent space are represented by an unconstrained learnable projection matrix and normalized only afterward through independent row-wise or column-wise softmax operations (Wang and Wang 2024; Hu et al. 2026). In this work, we reveal that this latent-space construction lacks joint constraints on the assignment distribution, resulting in unstable and imbalanced assignments. As shown in Figure 2(a), softmax-based projection concentrates dominant assignment weights on a small subset of latent tokens and produces substantially inconsis-

*Corresponding author.



(a) First Latent Space (b) Deeper Latent Space

Figure 2: Comparison of assignment patterns on Darcy at a resolution of 85×85. The heatmaps visualize the maximum assignment of each observation point across latent tokens. In the heatmaps, colors transition from white to red as the assignment weights increase. The line plots show the maximum assignment weight of each latent token across observation points. Softmax-based projection not only fails to preserve consistency between encoding and decoding mappings, but also exhibits (a) imbalanced token assignments in the first latent space and (b) severe token collapse in the deeper latent space.

tent assignment patterns between encoding and decoding. Furthermore, when the model is extended into a multiscale latent-space architecture built with softmax-based projections, assignment imbalance is continuously inherited and amplified across latent spaces, eventually causing severe token collapse in deeper latent spaces (Figure 2(b)). These issues reduce the effective capacity of latent representations, undermine stable cross-space information transfer, and hinder the design of multiscale architectures.

To address these issues, we propose **MoNo** (**Mo**ltiscale **O**ptimal Transport **N**eural **O**perator), an efficient framework for solving PDEs on general geometries by combining stable cross-space projections with a progressive multiscale hierarchy. At its core is **CoTAP** (**C**ross-scale **O**ptimal **T**ransport **A**ssignment and **P**rojection), a novel latent-space construction method that formulates assignment learning between adjacent spaces as an entropy-regularized optimal transport problem, further constructing balanced bidirectional projections from a shared transport plan with uniform marginal constraints (Figure 2). CoTAP further ensures stable information transfer across multiple latent spaces, thereby enabling efficient multiscale architectures on general geometries beyond standard grids (Wen et al. 2022). Building on this hierarchy, MoNo progressively aggregates physical states into more compact latent spaces to model long-range physical interactions at substantially lower computational cost than repeatedly stacking single-scale blocks (Wu et al. 2024; Hu et al. 2026), while fusing multiscale physical state tokens during decoding to predict the physical field. Extensive experiments on seven widely used PDE benchmarks demonstrate that MoNo outperforms SOTA methods while maintaining superior computational efficiency. Our contributions are summarized as follows:

- We propose MoNo, a progressive multiscale neural operator for solving PDEs on general geometries by constructing a stable hierarchy of latent spaces at multiple compression scales to efficiently learn physical interactions across scales.
- We propose CoTAP, which formulates cross-space assignment as an entropy-regularized optimal transport problem and constructs balanced bidirectional projections, thereby enabling the stable construction of latent spaces.
- Extensive experiments on seven PDE benchmarks show that MoNo outperforms SOTA methods in both prediction performance and computational efficiency.

Related Work

Learnable PDE Solver

Physics-Informed Neural Networks. Physics-Informed Neural Networks (PINNs) incorporate governing PDE constraints, initial conditions, and boundary conditions into the training objective to optimize a neural approximation of the target physical field (Niaki et al. 2021; Raissi, Yazdani, and Karniadakis 2020; Wandel et al. 2022; Yang, Qiu, and Fu 2023; Luo et al. 2025b; Wu and Wu 2026). However, PINNs generally require complete and explicit formulations of the governing PDEs and associated solution conditions, which are difficult to obtain in real-world problems. In this work, we focus on observation-driven PDE solving without requiring complete prior definition of the physical system.

Neural Operators. Neural operators have shown great potential for solving PDEs by learning operator mappings from input observations to target fields (Lu, Jin, and Karniadakis 2019; Li et al. 2020b,a; Herde et al. 2024; McCabe et al. 2025; Zhou et al. 2024). Recently, Transformer-based neural operators represent discrete spatial observations and target physical fields as sequences, further formulating PDE solving as a sequence-to-sequence mapping (Li, Meidani, and Farimani 2022), which provides substantial flexibility in handling general geometries (Wu et al. 2024; Luo et al. 2025a; Xiao et al. 2023). To efficiently process large-scale spatial observations, existing methods mainly introduce linear attention (Li, Meidani, and Farimani 2022; Li, Shu, and Barati Farimani 2023; Hao et al. 2023; Hu et al. 2026; Luo et al. 2025a) to reduce the computational complexity and construct compact latent spaces to compress long observation sequences (Wang and Wang 2024; Serrano et al. 2024; Alkin et al. 2024; Liu et al. 2025; Hu et al. 2026; Wen et al. 2026).

Optimal Transport

Optimal transport (OT) provides a unified mathematical framework for distribution matching across spaces (Villani et al. 2009). In neural operator research, OT has primarily been employed as a regularization constraint to improve

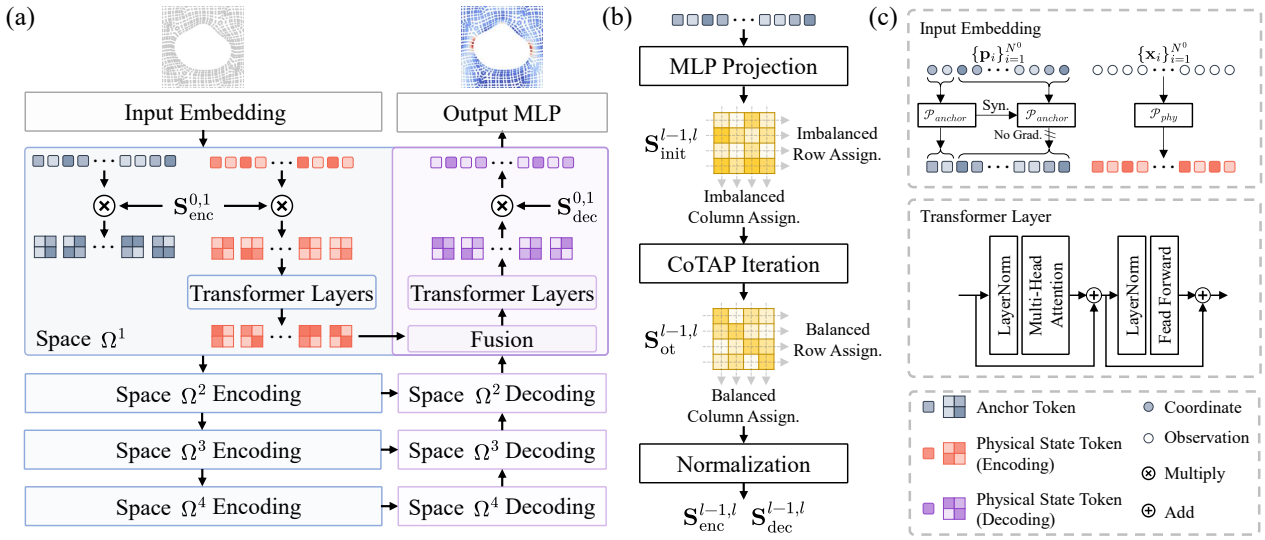


Figure 3: Illustration of MoNo. (a) Given a physical system, MoNo first embeds the spatial positions and input observations into high-dimensional tokens, and then efficiently learns physical interaction with a hierarchy of latent spaces. (b) CoTAP constructs an initial assignment matrix, iteratively solves for a balanced assignment matrix, and normalizes it to obtain the encoding matrix and decoding matrix. (c) Illustration of the input embedding and Transformer layer.

transfer learning (Yang and Ren 2025) and dense reconstruction (Ma et al. 2026). In addition, some studies use OT to align arbitrary input geometries with predefined grids (Li et al. 2025; Qi et al. 2026). In this work, we propose CoTAP, which directly predicts assignment relations between adjacent spaces and further constructs balanced bidirectional projections through OT, enabling stable cross-scale information transfer without predefined grids.

Preliminaries

We consider PDE-solving problems defined on a spatial domain $\Omega^0 \subset \mathbb{R}^{D_s}$, where D_s denotes the spatial dimension. Given input observations determined by the governing PDE, solution conditions, and external parameters, the goal is to estimate the corresponding target physical field. Beyond settings restricted to regular grids, we focus on a general observation setting. Let $P = \{\mathbf{p}_i\}_{i=1}^{N^0}$ denote a set of N^0 discrete spatial observation points in Ω^0. Given observations $X = \{\mathbf{x}_i\}_{i=1}^{N^0}$ at P, the model aims to predict the target physical field $Y = \{\mathbf{y}_i\}_{i=1}^{N^0}$ on the same point set, where $\mathbf{x}_i \in \mathbb{R}^{D_x}$ and $\mathbf{y}_i \in \mathbb{R}^{D_y}$, with D_x and D_y denoting the input observation and target physical field dimensions, respectively. Each $\mathbf{x}_i$ is a tensor composed of task-specific physical quantities, such as material parameters.

Method

The overall framework of MoNo is illustrated in Figure 3. Given the observation point set P and the corresponding input observations X, MoNo first embeds spatial positions and input observations into an anchor token set $G^0 = \{\mathbf{g}_i^0\}_{i=1}^{N^0}$ and a physical state token set $F_{\text{enc}}^0 = \{\mathbf{f}_i^0\}_{i=1}^{N^0}$ by $\mathcal{P}_{\text{anchor}}$ and $\mathcal{P}_{\text{phy}}$, respectively, where $\mathbf{g}_i^0 \in \mathbb{R}^{D_g}$ and $\mathbf{f}_i^0 \in \mathbb{R}^{D_f}$ denote the anchor token and physical state token of the i-th observation point, and D_g and D_f denote dimensions. As the mapping from P to G^0 is smooth, gradients through $\mathcal{P}_{\text{anchor}}$ are retained for only 25% of randomly sampled P, while the remaining points are processed by a gradient-free copy, providing performance comparable to full-point backpropagation with lower training memory (Figure 3(c)). Furthermore, MoNo progressively builds stable latent spaces at multiple compressed scales to learn physical interactions. Finally, the decoder maps multiscale latent tokens back to the original observation points and outputs the predicted target field $\hat{Y} = \{\hat{\mathbf{y}}_i\}_{i=1}^{N^0}$.

CoTAP

Cross-space Assignment and Projection. For any l-th latent space Ω^l, where $l \in \{1, \ldots, L\}$, we use CoTAP to construct the cross-space assignment and projection between the preceding space Ω^{l-1} and the current latent space Ω^l. Ω^0 denotes the original observation space, while Ω^l for $l \geq 1$ denotes the l-th latent space. For the preceding space Ω^{l-1}, we are given the anchor token set $G^{l-1} = \{\mathbf{g}_i^{l-1}\}_{i=1}^{N^{l-1}}$, where N^{l-1} denotes the number of elements in Ω^{l-1} and $\mathbf{g}_i^{l-1}$ denotes the anchor token of the i-th element.

CoTAP first represents G^{l-1} in matrix form $G^{l-1} \in \mathbb{R}^{N^{l-1} \times D_g}$ and constructs an initial assignment matrix $\mathbf{S}_{\text{init}}^{l-1,l}$ through an MLP projector $\mathcal{P}^l(\cdot)$:

$$\mathbf{S}_{\text{init}}^{l-1,l} = \mathcal{P}^l(G^{l-1}) \in \mathbb{R}^{N^{l-1} \times N^l}, \tag{1}$$

where $\mathbf{S}_{\text{init},ij}^{l-1,l}$ denotes the initial assignment score between the i-th element in Ω^{l-1} and the j-th latent token in Ω^l.

CoTAP further treats $\mathbf{S}_{\text{init}}^{l-1,l}$ as the transport reward and solves the entropy-regularized optimal transport problem:

$$\mathbf{S}_{\text{ot}}^{l-1,l} = \arg\max_{\mathbf{S}\in\mathcal{C}^{l-1,l}} \left\langle \mathbf{S}, \mathbf{S}_{\text{init}}^{l-1,l} \right\rangle + \tau\mathcal{H}(\mathbf{S}), \tag{2}$$

where $\mathbf{S}_{\text{ot}}^{l-1,l} \in \mathbb{R}^{N^{l-1}\times N^l}$ denotes the OT-normalized assignment matrix, $\tau = 1$ denotes the temperature coefficient, $\mathcal{H}(\mathbf{S}) = -\sum_{i=1}^{N^{l-1}}\sum_{j=1}^{N^l} S_{ij}\log S_{ij}$ denotes the entropy regularization term, and $\mathcal{C}^{l-1,l}$ is the transport polytope with uniform marginal constraints. These marginal constraints ensure that each input element sends the same total mass and each latent token receives the same total mass.

Based on $\mathbf{S}_{\text{ot}}^{l-1,l}$, CoTAP constructs the bidirectional projection matrices between Ω^{l-1} and Ω^l. Since each column and row of $\mathbf{S}_{\text{ot}}^{l-1,l}$ sums to $1/N^l$ and $1/N^{l-1}$, respectively, CoTAP further constructs projections through normalization:

$$\begin{aligned} \mathbf{S}_{\text{enc}}^{l-1,l} &= N^l\mathbf{S}_{\text{ot}}^{l-1,l}, \\ \mathbf{S}_{\text{dec}}^{l-1,l} &= N^{l-1}\mathbf{S}_{\text{ot}}^{l-1,l}, \end{aligned} \tag{3}$$

where $\mathbf{S}_{\text{enc}}^{l-1,l}$ denotes the projection from Ω^{l-1} to Ω^l and $\mathbf{S}_{\text{dec}}^{l-1,l}$ denotes the projection from Ω^l to Ω^{l-1}. Since both projections are constructed from $\mathbf{S}_{\text{ot}}^{l-1,l}$, CoTAP maintains a consistent mass-transfer relation in the encoding and decoding, rather than arbitrary learnable token mixing.

CoTAP solves Equation (2) with log-domain Sinkhorn iterations (Sinkhorn 1967). To reduce the computational cost for large-scale tasks, we further implement a Triton-based fused Sinkhorn solver, termed CoTAP Iteration. Specifically, CoTAP Iteration directly fuses the log-sum-exp reductions into GPU kernels, thereby avoiding the materialization of large intermediate tensors. For the backward pass, we derive and implement gradient computation equivalent to backpropagation through standard Sinkhorn iterations. The details of CoTAP Iteration are provided in the Appendix.

We further demonstrate that the composition of CoTAP and a single latent self-attention layer can be represented as a learnable integral operator on the original physical domain Ω^0, consistent with the operator interpretation of other methods (Wu et al. 2024; Hu et al. 2026). The detailed proof of Theorem 1 is provided in the Appendix.

Theorem 1 (**CoTAP with latent self-attention as an integral operator on** Ω^0)**.** *Let Ω^0 be the original physical domain endowed with a probability measure μ^0. For any bounded measurable physical state token field $\mathcal{F}_{\text{enc}}^0 : \Omega^0 \to \mathbb{R}^{D_f}$ and any continuous anchor token field $\mathcal{G}^0 : \Omega^0 \to \mathbb{R}^{D_g}$, the composition of CoTAP and a single latent self-attention layer can be represented as an input-dependent integral kernel operator on Ω^0. Specifically, for μ^0-almost every $\mathbf{p} \in \Omega^0$,*

$$\mathcal{T}\big(\mathcal{G}^0, \mathcal{F}_{\text{enc}}^0\big)(\mathbf{p}) = \int_{\Omega^0} \kappa\big(\mathbf{p}, \boldsymbol{\xi}\big)\mathcal{F}_{\text{enc}}^0(\boldsymbol{\xi})\mathbf{W}_v\, d\mu^0(\boldsymbol{\xi}), \tag{4}$$

where $\mathcal{T}$ denotes the composite operator, $\mathbf{W}_v$ denotes the value projection in latent self-attention, and κ denotes the input-dependent kernel.

Progressive Multiscale Modeling

Based on CoTAP, MoNo builds an encoding-decoding hierarchy with L latent spaces to learn physical interactions at progressively compressed scales.

Encoding. For each latent space Ω^l, where $l \in \{1, \ldots, L\}$, the encoding stage projects anchor tokens and physical state tokens from Ω^{l-1} to Ω^l and learns physical interactions:

$$\begin{aligned} F_{\text{enc}}^l &= \mathcal{M}_{\text{enc}}^l\left(\left(\mathbf{S}_{\text{enc}}^{l-1,l}\right)^\top F_{\text{enc}}^{l-1}\right), \\ G^l &= \left(\mathbf{S}_{\text{enc}}^{l-1,l}\right)^\top G^{l-1}, \end{aligned} \tag{5}$$

where $\mathcal{M}_{\text{enc}}^l$, F_{enc}^l, and G^l denote the encoding transformer block, the updated physical state token set, and the projected anchor token set in Ω^l, respectively. By repeating Equation (5) progressively, MoNo preserves local geometry and fine-grained physical information in shallow latent spaces while forming more compressed and global physical state representations in deeper latent spaces.

Decoding. Starting from the deepest latent space Ω^L, the decoding stage projects physical state tokens back through the hierarchy and fuses them with the encoded tokens:

$$F_{\text{dec}}^l = \begin{cases} \mathcal{M}_{\text{dec}}^l(F_{\text{enc}}^l), & l = L, \\ \mathcal{M}_{\text{dec}}^l\left(F_{\text{enc}}^l + \mathbf{S}_{\text{dec}}^{l,l+1}F_{\text{dec}}^{l+1}\right), & 1 \le l < L, \end{cases} \tag{6}$$

where $\mathcal{M}_{\text{dec}}^l$ and F_{dec}^l denote the decoding transformer block and the decoded physical state token set in Ω^l, respectively. Finally, F_{dec}^1 is projected back to the original observation space as $F_{\text{dec}}^0 = \mathbf{S}_{\text{dec}}^{0,1}F_{\text{dec}}^1$. An MLP output head then maps F_{dec}^0 to the target physical field prediction $\hat{Y}$.

Furthermore, we demonstrate that MoNo defines a neural operator that maps input functions to target physical-field functions. The detailed proof of Theorem 2 is provided in the Appendix.

Theorem 2 (**MoNo as a neural operator**)**.** *Given a continuous anchor token field $\mathcal{G}^0 : \Omega^0 \to \mathbb{R}^{D_g}$ and a bounded measurable physical state token field $\mathcal{F}_{\text{enc}}^0 : \Omega^0 \to \mathbb{R}^{D_f}$, MoNo with L latent spaces defines the following learnable function-to-function mapping:*

$$\widehat{\mathcal{Y}} = \mathcal{N}_\theta\big(\mathcal{G}^0, \mathcal{F}_{\text{enc}}^0\big), \qquad \widehat{\mathcal{Y}} : \Omega^0 \to \mathbb{R}^{D_y}, \tag{7}$$

where $\mathcal{N}_\theta$ denotes the MoNo neural operator parameterized by the learnable parameters θ, and $\widehat{\mathcal{Y}}$ denotes the predicted physical-field function defined on Ω^0.

MoNo Variants

We define two variants and adjust their input and output dimensions according to each task. Both variants contain four latent spaces, with the numbers of encoding and decoding Transformer layers set to $\{3, 1, 1, 1\}$, and differ as follows:

- **MoNo-light**: $D_g = 96$, $D_f = 96$, with latent token numbers of $\{512, 256, 128, 64\}$.
- **MoNo**: $D_g = 192$, $D_f = 192$, with latent token numbers of $\{1024, 512, 256, 128\}$.

Method	Structured Mesh			Standard Grid	Point Cloud
	Airfoil	Pipe	Plasticity	Navier-Stokes	Elasticity
FNO (Li et al. 2020a)	-	-	-	0.1556	-
GEO-FNO (Li et al. 2023a)	0.0138	0.0067	0.0074	0.1556	0.0229
F-FNO (Tran et al. 2021)	0.0078	0.0070	0.0047	0.2322	0.0263
Galerkin (Cao 2021)	0.0118	0.0098	0.0120	0.1401	0.0240
OFormer (Li et al. 2023)	0.0183	0.0168	0.0017	0.1705	0.0183
GNOT (Hao et al. 2023)	0.0076	0.0047	0.0336	0.1380	0.0086
FactFormer (Li et al. 2023)	0.0071	0.0060	0.0312	0.1214	-
ONO (Xiao et al. 2023)	0.0061	0.0052	0.0048	0.1195	0.0118
LSM (Wu et al. 2023)	0.0059	0.0050	0.0025	0.1535	0.0218
LNO (Wang and Wang 2024)	0.0053	0.0031	0.0028	0.0830	0.0066
Transolver (Wu et al. 2024)	0.0053	0.0030	0.0013	0.0882	0.0065
Transolver++ (Luo et al. 2025a)	0.0051	0.0027	0.0014	0.1010	0.0064
LinearNO (Hu et al. 2026)	<u>0.0049</u>	<u>0.0024</u>	0.0011	0.0699	0.0050
MoNo-Light (ours)	**0.0048**	0.0027	<u>0.0010</u>	<u>0.0673</u>	<u>0.0042</u>
MoNo (ours)	**0.0048**	**0.0021**	**0.0006**	**0.0522**	**0.0033**

Table 1: Comparison on standard benchmarks in relative L2 (↓). "-" denotes that the method is not applicable to the corresponding task. The best results are highlighted in bold, and the second-best results are underlined. MoNo outperforms other methods.

MoNo-light has a parameter count comparable to existing methods, whereas MoNo uses a larger configuration to increase model capacity. Notably, as shown in the Experiments section, both variants achieve higher computational efficiency than existing SOTA methods on large-scale tasks.

Experiments

Experiment Settings

Benchmarks. To comprehensively evaluate MoNo across diverse geometries and physical systems, we conduct experiments on six widely used standard benchmarks (Li et al. 2020a; Hu et al. 2026; Wang and Wang 2024), including Airfoil, Pipe, Plasticity, Navier-Stokes (NS2D), Elasticity, and Darcy. Following recent studies (Wu et al. 2024; Hu et al. 2026), we further evaluate MoNo on AirfRANS (Bonnet et al. 2022), a real-world advanced benchmark.

Baselines. We compare MoNo with SOTA neural operator methods, including FNO (Li et al. 2020a), GEO-FNO (Li et al. 2023a), F-FNO (Tran et al. 2021), Galerkin Transformer (Cao 2021), OFormer (Li, Meidani, and Farimani 2022), GNOT (Hao et al. 2023), FactFormer (Li, Shu, and Barati Farimani 2023), ONO (Xiao et al. 2023), LSM (Wu et al. 2023), LNO (Wang and Wang 2024), Transolver (Wu et al. 2024), Transolver++ (Luo et al. 2025a), and LinearNO (Hu et al. 2026). For AirfRANS, we additionally include MLP, PointNet (Qi et al. 2017), Graph U-Net (Bronstein et al. 2016), and MeshGraphNet (Pfaff et al. 2020).

Implementations. Please refer to the Appendix for detailed hyper-parameters and training configurations.

Main Results

Standard Benchmarks. Table 1 reports the results on standard benchmarks. MoNo achieves the best performance across all tasks. For the three structured-mesh tasks, including Airfoil, Pipe, and Plasticity, MoNo outperforms existing methods by at least 2.0%, 12.5%, and 45.5% in terms of relative L2 error, respectively. MoNo also achieves the best performance for complex time-dependent physical systems and point-cloud representations.

Method	Vol. (↓)	Surf. (↓)	C_L (↓)	ρ_L (↑)
MLP	0.0081	0.0200	0.2108	0.9932
PointNet (Qi et al. 2017)	0.0253	0.0996	0.1973	0.9919
Graph U-Net (2017)	0.0076	0.0144	0.1677	0.9949
MeshGraphNet (2020)	0.0214	0.0387	0.2252	0.9945
GNO (Li et al. 2020b)	0.0269	0.0405	0.2016	0.9938
GEO-FNO (Li et al. 2023a)	0.0361	0.0301	0.6161	0.9257
GALERKIN (Cao 2021)	0.0074	0.0159	0.2336	0.9951
GNOT (Hao et al. 2023)	0.0049	0.0152	0.1992	0.9942
GINO (Li et al. 2023b)	0.0297	0.0482	0.1821	0.9958
LNO (Wang and Wang 2024)	0.0214	0.0268	0.1480	0.9744
Transolver (Wu et al. 2024)	0.0023	0.0085	0.1230	0.9978
Transolver++ (2025)	0.0068	0.0159	0.1880	0.9910
LinearNO (Hu et al. 2026)	<u>0.0011</u>	0.0077	<u>0.0491</u>	**0.9992**
MoNo-Light (ours)	0.0025	<u>0.0018</u>	0.0616	<u>0.9986</u>
MoNo (ours)	**0.0009**	**0.0013**	**0.0415**	**0.9992**

Table 2: Comparison on the AirfRANS benchmark. Following previous works (Wu et al. 2024; Hu et al. 2026), we evaluate the prediction error of surrounding (Vol.) and surface (Surf.) physical fields, as well as the lift coefficient (C_L). ρ_L denotes Spearman's rank correlation for C_L. MoNo outperforms other methods.

For fair comparisons, we also report the performance of MoNo-light, whose parameter count is comparable to that of SOTA methods. MoNo-light still shows competitive and overall superior solving accuracy. Specifically, MoNo-light outperforms SOTA methods by at least 2.0%, 9.1%, 3.7%, and 16.0% on Airfoil, Plasticity, NS2D, and Elasticity in

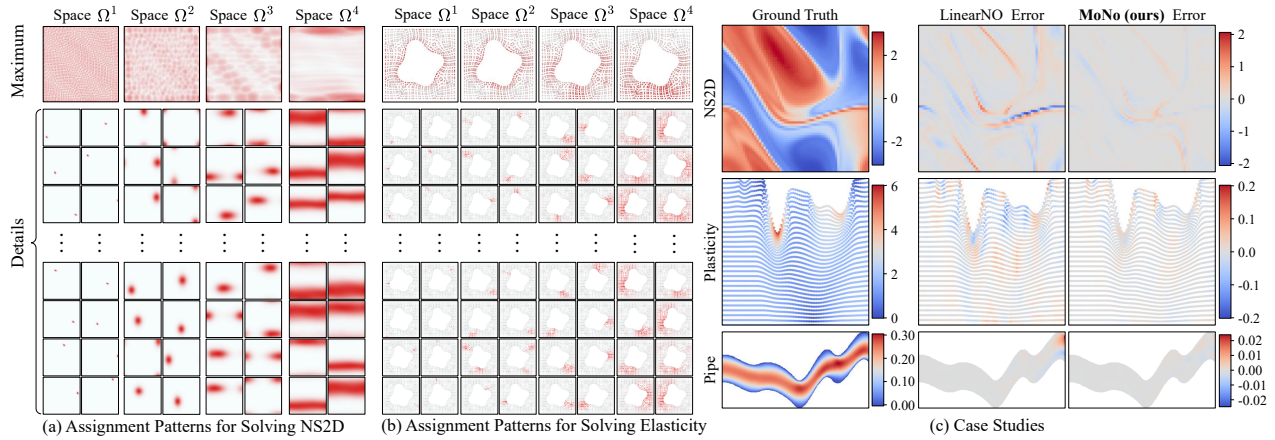


Figure 4: Visualization of assignment patterns and prediction errors. (a) The assignment patterns learned by MoNo on NS2D. (b) The assignment patterns learned by MoNo-light on Elasticity. (c) Case studies. MoNo constructs assignment relations with clear multiscale characteristics and achieves substantially lower prediction errors.

Method	85×85	141×141	211×211
LNO	0.0081	0.0074	0.0073
Transolver	0.0055	0.0064	0.0063
LinearNO	**0.0050**	0.0053	0.0053
MoNo-Light (ours)	0.0070	0.0057	0.0059
MoNo (ours)	0.0059	**0.0049**	**0.0046**

Table 3: Comparison on Darcy at multiple resolutions in relative L2 error (↓). MoNo effectively reduces prediction errors as the spatial resolution increases.

terms of relative L2 error, respectively. These results demonstrate that the stable cross-space projections constructed by CoTAP and the progressive multiscale hierarchy enable effective learning of multiscale physical states across diverse geometries and physical systems.

Advanced Benchmarks. Table 2 reports the results on the AirfRANS advanced benchmark. MoNo achieves the best performance in predicting the surrounding physical fields, surface physical fields, and lift coefficient. Specifically, MoNo outperforms the existing SOTA methods by at least 18.2% and 83.1% in predicting the surrounding physical fields and surface physical fields, respectively. For aerodynamic performance evaluation, MoNo achieves an error of 0.0415 in predicting the lift coefficient, outperforming all existing methods by at least 15.5%. MoNo-light also outperforms all baselines by at least 76.6% in predicting the surface physical fields. Moreover, MoNo-light surpasses most existing SOTA methods on the remaining metrics. These results further demonstrate that MoNo can handle complex geometries and real-world problems.

Evaluation at Multiple Resolutions. To further evaluate the ability of models to utilize spatial observations of varying densities, we conduct experiments on Darcy at three spatial resolutions: 85×85, 141×141, and 211×211. The results are reported in Table 3. Both MoNo and MoNo-light effectively utilize the additional information provided by higher-resolution observations, achieving more accurate physical field predictions overall as the number of observation points increases. In contrast, both Transolver and LinearNO exhibit performance degradation at higher resolutions. In particular, at the resolution of 211×211, MoNo outperforms all other methods by at least 13.2% in terms of relative L2 error. These results further demonstrate that MoNo can effectively aggregate and propagate physical information across different observation resolutions, thereby improving solving performance as the observation density increases.

Method	OOD Reynolds		OOD Angles	
	Vol. (↓)	Surf. (↓)	Vol. (↓)	Surf. (↓)
MLP	0.0669	0.1153	0.1309	0.3311
PointNet	0.0838	0.1403	0.2021	0.4649
Graph U-Net	0.0538	0.1168	0.0979	0.2391
MeshGraphNet	0.2789	0.2382	0.4902	1.1071
GNO	0.0833	0.1562	0.1626	0.2359
Galerkin	0.0330	0.0972	0.0577	0.2773
GNOT	0.0305	0.0959	0.0471	0.3466
GINO	0.0839	0.1825	0.1589	0.2469
LNO	0.0825	0.1762	0.0346	0.0790
Transolver	0.0122	0.0550	0.0480	0.2335
LinearNO	0.0112	0.0372	0.0464	0.2500
MoNo-light (ours)	0.0090	0.0146	0.0222	0.0553
MoNo (ours)	**0.0066**	**0.0121**	**0.0097**	**0.0248**

Table 4: Comparison on the AirfRANS OOD benchmarks.

Out-of-Distribution Evaluation. We further introduce out-of-distribution (OOD) experiments on the AirfRANS benchmark. Specifically, following previous work (Wu et al. 2024), we consider two OOD settings on AirfRANS: Reynolds-number extrapolation (OOD Reynolds) and angle-of-attack extrapolation (OOD Angles). As shown in Table 4, MoNo

Model	Params. (M)	GFLOPs (↓)
GNOT (Hao et al. 2023)	6.73	444.18
Transolver (Wu et al. 2024)	2.81	196.19
Transolver++ (Luo et al. 2025a)	<u>1.74</u>	121.17
LinearNO (Hu et al. 2026)	1.77	132.83
MoNo-light (ours)	**1.73**	**23.46**
MoNo (ours)	6.90	<u>97.26</u>

Table 5: Efficiency comparison on parameter count (Params.) and GFLOPs (↓). MoNo shows substantially higher computational efficiency than existing methods.

MultiScale	CoTAP	Airfoil	Elasticity	Pipe
✗	✗	0.0060	0.0077	0.0036
✓	✗	0.0056	0.0072	0.0036
✗	✓	<u>0.0051</u>	<u>0.0070</u>	<u>0.0028</u>
✓	✓	**0.0048**	**0.0042**	**0.0027**

Table 6: Ablation studies of CoTAP and progressive multiscale modeling in relative L2 error (↓).

and MoNo-light outperform existing methods. MoNo outperforms existing methods by at least 41.1% and 67.5% in predicting the surrounding physical fields and surface physical fields on OOD Reynolds, respectively. In addition, MoNo also outperforms existing methods by at least 72.0% and 68.6% in predicting the surrounding physical fields and surface physical fields on OOD Angles, respectively. Table 4 demonstrates the strong extrapolation capability of MoNo to unseen flow regimes and aerodynamic configurations.

Efficiency. Table 5 compares the parameter counts and GFLOPs of models when processing 65,536 spatial observation points. MoNo-light contains only 1.73M parameters and achieves the lowest computational cost. Despite using a larger configuration with 6.90M parameters, MoNo requires only 97.26 GFLOPs, which is 19.7% lower than that of Transolver++. These results demonstrate that MoNo-light provides substantial advantages in both model size and computational cost, while MoNo increases representation capacity and solving performance while remaining more computationally efficient than existing methods.

Ablation Studies

Unless otherwise specified, ablation studies are conducted using MoNo-light.

CoTAP. Table 6 verifies the critical role of CoTAP. Building upon the baseline with softmax-based projection, introducing CoTAP reduces the errors on Airfoil, Elasticity, and Pipe by 15.0%, 9.1%, and 22.2%, respectively. These results demonstrate that CoTAP effectively improves the learning of physical interactions in latent spaces by constructing balanced cross-space projections.

Progressive Multiscale Modeling. Table 6 also verifies the importance of progressive multiscale modeling in improving prediction performance. Although introducing multiscale modeling alone with softmax-based projection provides only limited improvements, further incorporating progressive multiscale modeling upon CoTAP reduces the errors on Airfoil, Elasticity, and Pipe by 5.9%, 40.0%, and 3.6%, respectively. This demonstrates that the effectiveness of multiscale modeling highly depends on the stable construction of latent spaces.

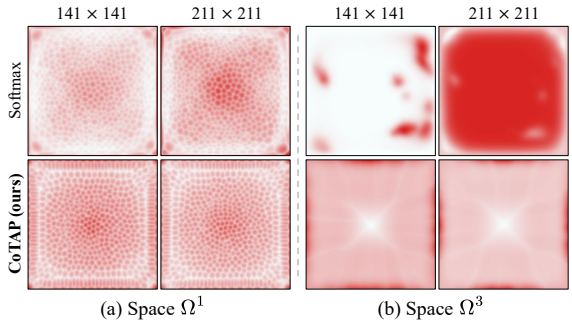


Figure 5: Visualization of assignments across different spatial resolutions. (a) The assignments in the first latent space (Space Ω^1). (b) The assignments in the deeper latent space (Space Ω^3). CoTAP constructs balanced assignment relations that remain consistent across spatial resolutions.

Visualizations

Visualization of CoTAP. Figures 4(a) and (b) visualize the assignment patterns learned by MoNo on NS2D and by MoNo-light on Elasticity, respectively. Across both tasks with distinct geometries, CoTAP constructs assignment relations with clear multiscale characteristics. Latent tokens in shallow spaces provide fine-grained partitions of the original physical domain, while each latent token progressively covers a larger spatial region and forms a more global representation as the hierarchy deepens.

Figure 5 further compares the assignment patterns generated by CoTAP and softmax-based projection on Darcy at resolutions of 141×141 and 211×211. Across multiple latent spaces, CoTAP constructs balanced assignment patterns that remain consistent between the two spatial resolutions. In contrast, softmax-based projection fails to preserve assignment consistency across resolutions, exhibits imbalanced assignments, and suffers from severe token collapse in deeper latent spaces. These results validate the critical role of CoTAP in balancing token assignments, mitigating token collapse in deeper latent spaces, and preserving assignment stability across spatial resolutions.

Case Studies. Figure 4(c) presents qualitative prediction results. Compared with LinearNO, MoNo exhibits substantially lower prediction errors throughout the physical domain, particularly in high-gradient regions. These results show that MoNo can accurately predict fine-grained physical behaviors and complex local variations.

Conclusion

In this work, we proposed MoNo, a progressive multiscale neural operator for solving PDEs on general geometries. At

its core, we propose CoTAP to formulate cross-space assignment as an entropy-regularized optimal transport problem and construct stable latent spaces. CoTAP further unlocks stable progressive multiscale architectures on general geometries, thereby enabling more efficient learning of physical interactions. Extensive experiments demonstrate that MoNo outperforms SOTA methods in both prediction performance and computational efficiency. This work highlights the critical role of stable latent-space construction in developing Transformer-based neural operators with stronger representational capacity and higher computational efficiency.

Appendix A: Notation Summary

For clarity, Table 7 summarizes the main notation used throughout this paper. Symbols that are only used in Appendix E are defined upon their first occurrence.

Appendix B: Proof of Theorem 1

Proof. Assume that the continuous CoTAP coupling between (Ω^0, μ^0) and (Ω^1, μ^1) admits a nonnegative measurable density with respect to $\mu^0 \otimes \mu^1$, denoted as:

$$r(\mathbf{p}, \boldsymbol{\zeta}) := r^{0,1}_{\mathcal{G}^0}(\mathbf{p}, \boldsymbol{\zeta}), \tag{8}$$

where μ^1 denotes the probability measure on the latent space Ω^1, r denotes the CoTAP coupling density induced by the anchor token field $\mathcal{G}^0$, and $\mathbf{p}$ denotes the target point in Ω^0. The marginal constraints are as follows:

$$\begin{aligned} \int_{\Omega^1} r(\mathbf{p}, \boldsymbol{\zeta})\, d\mu^1(\boldsymbol{\zeta}) = 1, \quad \mu^0\text{-a.e. } \mathbf{p}, \\ \int_{\Omega^0} r(\boldsymbol{\xi}, \boldsymbol{\zeta})\, d\mu^0(\boldsymbol{\xi}) = 1, \quad \mu^1\text{-a.e. } \boldsymbol{\zeta}. \end{aligned} \tag{9}$$

Continuous CoTAP. The continuous CoTAP encoding and decoding operators induced by this coupling are as follows:

$$\begin{aligned} \mathcal{F}^1(\boldsymbol{\zeta}) &:= \left(\mathcal{S}^{0,1}_{\text{enc}} \mathcal{F}^0_{\text{enc}}\right)(\boldsymbol{\zeta}) \\ &= \int_{\Omega^0} r(\boldsymbol{\xi}, \boldsymbol{\zeta}) \mathcal{F}^0_{\text{enc}}(\boldsymbol{\xi})\, d\mu^0(\boldsymbol{\xi}), \\ \left(\mathcal{S}^{0,1}_{\text{dec}} \Phi^1\right)(\mathbf{p}) &:= \int_{\Omega^1} r(\mathbf{p}, \boldsymbol{\zeta}) \Phi^1(\boldsymbol{\zeta})\, d\mu^1(\boldsymbol{\zeta}), \end{aligned} \tag{10}$$

where $\mathcal{F}^1$ denotes the physical state field projected into Ω^1, and Φ^1 denotes a measurable latent feature field on Ω^1.

Latent self-attention. As established in prior work (Kovachki et al. 2023), standard self-attention admits an integral-operator representation. Denoting its input-dependent attention kernel by $\alpha(\boldsymbol{\zeta}, \boldsymbol{\eta})$, the latent self-attention operator in the present setting is:

$$\left(\mathcal{A}^1 \mathcal{F}^1\right)(\boldsymbol{\zeta}) = \int_{\Omega^1} \alpha(\boldsymbol{\zeta}, \boldsymbol{\eta}) \mathcal{F}^1(\boldsymbol{\eta}) \mathbf{W}_v\, d\mu^1(\boldsymbol{\eta}), \tag{11}$$

where $\mathcal{A}^1$ denotes the latent self-attention operator on Ω^1. The attention kernel satisfies:

$$\alpha(\boldsymbol{\zeta}, \boldsymbol{\eta}) \geq 0, \qquad \int_{\Omega^1} \alpha(\boldsymbol{\zeta}, \boldsymbol{\eta})\, d\mu^1(\boldsymbol{\eta}) = 1. \tag{12}$$

Composition on the original space. To represent the composition of latent self-attention and CoTAP decoding, we first define an intermediate kernel. Specifically, $\alpha(\boldsymbol{\zeta}, \boldsymbol{\eta})$ transfers the physical state at the latent point $\boldsymbol{\eta}$ to the attention output at $\boldsymbol{\zeta}$, while $r(\mathbf{p}, \boldsymbol{\zeta})$ further projects this output from $\boldsymbol{\zeta}$ to the original-space point $\mathbf{p}$. Marginalizing over the intermediate latent point $\boldsymbol{\zeta}$ yields the effective transfer kernel from $\boldsymbol{\eta}$ to $\mathbf{p}$:

$$\beta(\mathbf{p}, \boldsymbol{\eta}) := \int_{\Omega^1} r(\mathbf{p}, \boldsymbol{\zeta}) \alpha(\boldsymbol{\zeta}, \boldsymbol{\eta})\, d\mu^1(\boldsymbol{\zeta}). \tag{13}$$

β is the intermediate kernel induced by the composition of latent self-attention and CoTAP decoding.

Substituting the intermediate kernel β into the composite operator and then applying the continuous CoTAP encoding gives, for μ^0-almost every $\mathbf{p} \in \Omega^0$:

$$\begin{aligned} &\mathcal{T}\big(\mathcal{G}^0, \mathcal{F}^0_{\text{enc}}\big)(\mathbf{p}) \\ &= \int_{\Omega^1} \beta(\mathbf{p}, \boldsymbol{\eta}) \mathcal{F}^1(\boldsymbol{\eta}) \mathbf{W}_v\, d\mu^1(\boldsymbol{\eta}) \\ &= \int_{\Omega^1} \int_{\Omega^0} \beta(\mathbf{p}, \boldsymbol{\eta}) r(\boldsymbol{\xi}, \boldsymbol{\eta}) \mathcal{F}^0_{\text{enc}}(\boldsymbol{\xi}) \mathbf{W}_v\, d\mu^0(\boldsymbol{\xi})\, d\mu^1(\boldsymbol{\eta}). \end{aligned} \tag{14}$$

To express this double integral as an integral solely over the original space Ω^0, we first verify the absolute integrability required by Fubini's theorem:

$$\begin{aligned} &\int_{\Omega^1} \int_{\Omega^0} \beta(\mathbf{p}, \boldsymbol{\eta}) r(\boldsymbol{\xi}, \boldsymbol{\eta}) \\ &\qquad \cdot \left\| \mathcal{F}^0_{\text{enc}}(\boldsymbol{\xi}) \mathbf{W}_v \right\|\, d\mu^0(\boldsymbol{\xi})\, d\mu^1(\boldsymbol{\eta}) \\ &\leq \left\| \mathcal{F}^0_{\text{enc}} \right\|_{L^\infty(\mu^0)} \|\mathbf{W}_v\| \\ &\qquad \cdot \int_{\Omega^1} \beta(\mathbf{p}, \boldsymbol{\eta}) \left[\int_{\Omega^0} r(\boldsymbol{\xi}, \boldsymbol{\eta})\, d\mu^0(\boldsymbol{\xi}) \right] d\mu^1(\boldsymbol{\eta}) \\ &= \left\| \mathcal{F}^0_{\text{enc}} \right\|_{L^\infty(\mu^0)} \|\mathbf{W}_v\| < \infty. \end{aligned} \tag{15}$$

The equality follows from Equations (9), (12), and the definition of β in Equation (13). Therefore, Fubini's theorem allows us to exchange the order of integration:

$$\begin{aligned} &\mathcal{T}\big(\mathcal{G}^0, \mathcal{F}^0_{\text{enc}}\big)(\mathbf{p}) \\ &= \int_{\Omega^0} \left[\int_{\Omega^1} \beta(\mathbf{p}, \boldsymbol{\eta}) r(\boldsymbol{\xi}, \boldsymbol{\eta})\, d\mu^1(\boldsymbol{\eta}) \right] \\ &\qquad \cdot \mathcal{F}^0_{\text{enc}}(\boldsymbol{\xi}) \mathbf{W}_v\, d\mu^0(\boldsymbol{\xi}). \end{aligned} \tag{16}$$

The inner integral in Equation (16) defines the effective original-space kernel:

$$\kappa(\mathbf{p}, \boldsymbol{\xi}) := \int_{\Omega^1} \beta(\mathbf{p}, \boldsymbol{\eta}) r(\boldsymbol{\xi}, \boldsymbol{\eta})\, d\mu^1(\boldsymbol{\eta}), \tag{17}$$

It follows that, for μ^0-almost every $\mathbf{p} \in \Omega^0$, we can get:

$$\mathcal{T}\big(\mathcal{G}^0, \mathcal{F}^0_{\text{enc}}\big)(\mathbf{p}) = \int_{\Omega^0} \kappa(\mathbf{p}, \boldsymbol{\xi}) \mathcal{F}^0_{\text{enc}}(\boldsymbol{\xi}) \mathbf{W}_v\, d\mu^0(\boldsymbol{\xi}), \tag{18}$$

which proves the integral-operator representation on Ω^0. □

Notation	Description
Dimensions and indices	
D_s	Spatial dimension of the original physical domain.
D_x	Dimension of the input observation at each observation point.
D_y	Dimension of the target physical field at each observation point.
D_g	Dimension of the anchor tokens.
D_f	Dimension of the physical state tokens.
N^0	Number of observation points in the original observation space.
N^l	Number of elements in Ω^l. For $l \geq 1$, the number of latent tokens in the l-th latent space.
L	Total number of latent spaces in the progressive hierarchy.
l	Space index, where $l = 0$ denotes the original observation space and $l \in \{1, \dots, L\}$ denotes a latent space.
Spaces, observations, and tokens	
Ω^l	The l-th space in the hierarchy, where $\Omega^0 \subset \mathbb{R}^{D_s}$ is the original physical domain and Ω^l for $l \geq 1$ is the l-th latent space.
$P = \{\mathbf{p}_i\}_{i=1}^{N^0}$	Spatial observation point set, where $\mathbf{p}_i \in \Omega^0$.
$X = \{\mathbf{x}_i\}_{i=1}^{N^0}$	Input observation set, where $\mathbf{x}_i \in \mathbb{R}^{D_x}$.
$Y = \{\mathbf{y}_i\}_{i=1}^{N^0}$	Ground-truth target physical field, where $\mathbf{y}_i \in \mathbb{R}^{D_y}$.
$\hat{Y} = \{\hat{\mathbf{y}}_i\}_{i=1}^{N^0}$	Discrete target physical field predicted by MoNo.
$G^l = \{\mathbf{g}_i^l\}_{i=1}^{N^l}$	Anchor token set in Ω^l, where $\mathbf{g}_i^l \in \mathbb{R}^{D_g}$.
F_{enc}^l	Encoded physical state token set in Ω^l, with $F_{\text{enc}}^l \in \mathbb{R}^{N^l \times D_f}$.
F_{dec}^l	Decoded physical state token set in Ω^l, with $F_{\text{dec}}^l \in \mathbb{R}^{N^l \times D_f}$.
Trainable modules	
$\mathcal{P}_{\text{anchor}}$	Anchor embedding module that maps observation positions to the initial anchor tokens.
$\mathcal{P}_{\text{phy}}$	Physical state embedding module that maps input observations to the initial physical state tokens.
$\mathcal{P}^l$	MLP assignment projector that maps G^{l-1} to the initial assignment scores for Ω^l.
$\mathcal{M}_{\text{enc}}^l$	Encoding Transformer block that learns physical interactions in Ω^l.
$\mathcal{M}_{\text{dec}}^l$	Decoding Transformer block that updates and fuses decoded physical state tokens in Ω^l.
CoTAP	
$\mathbf{S}_{\text{init}}^{l-1,l}$	Initial assignment matrix generated by $\mathcal{P}^l$, with $\mathbf{S}_{\text{init}}^{l-1,l} \in \mathbb{R}^{N^{l-1} \times N^l}$.
$\mathbf{S}_{\text{init},ij}^{l-1,l}$	Initial assignment score between the i-th element in Ω^{l-1} and the j-th latent token in Ω^l.
$\mathbf{S}_{\text{ot}}^{l-1,l}$	OT-normalized assignment matrix between Ω^{l-1} and Ω^l with uniform marginal constraints.
$\mathbf{S}_{\text{enc}}^{l-1,l}$	Encoding projection matrix from Ω^{l-1} to Ω^l.
$\mathbf{S}_{\text{dec}}^{l-1,l}$	Decoding projection matrix from Ω^l to Ω^{l-1}.
$\mathbf{S}$	Candidate transport matrix in the entropy-regularized optimal transport problem.
$\mathcal{C}^{l-1,l}$	Transport polytope with uniform marginal constraints between Ω^{l-1} and Ω^l.
T	Number of Sinkhorn iterations used to solve the entropy-regularized optimal transport problem.
τ	Temperature coefficient of the entropy-regularized optimal transport problem.
$\mathcal{H}(\mathbf{S})$	Entropy regularization term of the candidate transport matrix $\mathbf{S}$.
Continuous operator analysis	
μ^l	Probability measure defined on Ω^l.
$\mathcal{G}^l$	Continuous anchor token field associated with the discrete anchor token set G^l.
$\mathcal{F}_{\text{enc}}^l, \mathcal{F}_{\text{dec}}^l$	Continuous encoded and decoded physical state token fields on Ω^l.
$\mathcal{S}_{\text{enc}}^{l-1,l}, \mathcal{S}_{\text{dec}}^{l-1,l}$	Continuous CoTAP encoding and decoding operators between adjacent spaces.
$r_{\mathcal{G}^{l-1}}^{l-1,l}$	CoTAP coupling density induced by the anchor token field $\mathcal{G}^{l-1}$.
$\mathcal{A}^1, \alpha$	Latent self-attention operator on Ω^1 and its input-dependent attention kernel.
$\mathcal{T}$	Composite operator formed by CoTAP and a single latent self-attention layer.
κ	Input-dependent integral kernel induced by CoTAP and latent self-attention.
$\mathbf{W}_v$	Value projection matrix in the integral representation of latent self-attention.
$\mathcal{N}_\theta$	MoNo neural operator parameterized by learnable parameters θ.
$\widehat{\mathcal{Y}}$	Continuous target physical-field function predicted by MoNo on Ω^0.

Table 7: Summary of the main notation used in MoNo.

Appendix C: Discrete Kernel Representation of Theorem 1

We further provide the finite-dimensional counterpart of the operator considered in Theorem 1. Using the CoTAP projection matrices defined in the main paper, we show that the composition of CoTAP encoding, one pure self-attention layer, and CoTAP decoding admits an exact input-dependent kernel-matrix representation on the discrete observation space. The encoded latent features are:

$$F^1 = \left(\mathbf{S}_{\text{enc}}^{0,1}\right)^\top F_{\text{enc}}^0. \tag{19}$$

In addition, the latent self-attention is defined as:

$$\begin{aligned} \mathbf{Q} &= F^1\mathbf{W}_q, \\ \mathbf{K} &= F^1\mathbf{W}_k, \\ \mathbf{A}(F^1) &= \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_a}}\right), \end{aligned} \tag{20}$$

where $\mathbf{Q}$ and $\mathbf{K}$ denote the query and key feature matrices, $\mathbf{A}(\cdot)$ denotes the row-normalized attention matrix based on input, $\mathbf{W}_q$ and $\mathbf{W}_k$ denote the query and key projections, and D_a denotes the query/key dimension. The corresponding latent attention output is $\mathbf{A}(F^1)F^1\mathbf{W}_v$. The projection matrices and the resulting CoTAP decoding are as follows:

$$\begin{aligned} \mathbf{S}_{\text{enc}}^{0,1} &= N^1\mathbf{S}_{\text{ot}}^{0,1}, \\ \mathbf{S}_{\text{dec}}^{0,1} &= N^0\mathbf{S}_{\text{ot}}^{0,1}, \\ F_{\text{dec}}^0 &= \mathbf{S}_{\text{dec}}^{0,1}\mathbf{A}(F^1)\cdot\left(\mathbf{S}_{\text{enc}}^{0,1}\right)^\top F_{\text{enc}}^0\mathbf{W}_v, \end{aligned} \tag{21}$$

where F_{dec}^0 denotes the physical state feature matrix projected back to the original observation space after the composition of CoTAP encoding, latent self-attention, and CoTAP decoding. Therefore, the effective discrete kernel is:

$$\begin{aligned} &\mathbf{K}_{\text{eff}}^0\left(G^0, F_{\text{enc}}^0\right) \\ &:= \mathbf{S}_{\text{dec}}^{0,1}\mathbf{A}(F^1)\left(\mathbf{S}_{\text{enc}}^{0,1}\right)^\top \\ &= \left(N^0\mathbf{S}_{\text{ot}}^{0,1}\right)\mathbf{A}(F^1)\left(N^1\mathbf{S}_{\text{ot}}^{0,1}\right)^\top \\ &= N^0N^1\mathbf{S}_{\text{ot}}^{0,1}\mathbf{A}\left(N^1\left(\mathbf{S}_{\text{ot}}^{0,1}\right)^\top F_{\text{enc}}^0\right)\cdot\left(\mathbf{S}_{\text{ot}}^{0,1}\right)^\top, \end{aligned} \tag{22}$$

where $\mathbf{K}_{\text{eff}}^0 \in \mathbb{R}^{N^0\times N^0}$ acts directly on the original observation space. The complete discrete composition can therefore be written as:

$$F_{\text{dec}}^0 = \mathbf{K}_{\text{eff}}^0\left(G^0, F_{\text{enc}}^0\right)F_{\text{enc}}^0\mathbf{W}_v. \tag{23}$$

This is an exact algebraic identity for finite N^0 and N^1. The kernel is input-dependent because $\mathbf{S}_{\text{ot}}^{0,1}$ is induced by G^0, while $\mathbf{A}(\cdot)$ depends on F^1, which is jointly determined by G^0 and F_{enc}^0.

Appendix D: Proof of Theorem 2

Proof. We consider the deterministic inference map of MoNo. For every $l = 1,\dots,L$, assume that the CoTAP coupling between (Ω^{l-1},μ^{l-1}) and (Ω^l,μ^l) admits a non-negative measurable density with respect to $\mu^{l-1}\otimes\mu^l$. By the continuous CoTAP construction in Equation (10), $\mathcal{S}_{\text{enc}}^{l-1,l}$ and $\mathcal{S}_{\text{dec}}^{l-1,l}$ are cross-space integral operators from Ω^{l-1} to Ω^l and from Ω^l to Ω^{l-1}, respectively. Both operators depend on the current anchor token field $\mathcal{G}^{l-1}$.

As shown in Equation (11), latent self-attention is a non-local integral operator on the corresponding latent space. Multi-head concatenation and output projection act only along the feature dimension, while LayerNorm, feed-forward networks, and activation functions act point-wise on each feature field. Moreover, residual and skip addition can be written as the point-wise local operators. Thus, after lifting branched features to a product function space, every standard Transformer block is a finite composition of latent-space nonlocal integral operators and point-wise local operators. Specifically, each $\mathcal{M}_{\text{enc}}^l$ and $\mathcal{M}_{\text{dec}}^l$ has the same operator structure.

Single latent space. For $L = 1$, the complete model is as follows:

$$\begin{aligned} \mathcal{F}_{\text{enc}}^1 &= \mathcal{M}_{\text{enc}}^1\left(\mathcal{S}_{\text{enc}}^{0,1}\mathcal{F}_{\text{enc}}^0\right), \\ \mathcal{F}_{\text{dec}}^1 &= \mathcal{M}_{\text{dec}}^1\left(\mathcal{F}_{\text{enc}}^1\right), \\ \mathcal{F}_{\text{dec}}^0 &= \mathcal{S}_{\text{dec}}^{0,1}\mathcal{F}_{\text{dec}}^1, \\ \widehat{\mathcal{Y}} &= \text{MLP}_{\text{out}}\left(\mathcal{F}_{\text{dec}}^0\right). \end{aligned} \tag{24}$$

The CoTAP operators in Equation (24) are induced by $\mathcal{G}^0$. Hence, these equations define a learnable mapping $(\mathcal{G}^0,\mathcal{F}_{\text{enc}}^0)\mapsto\widehat{\mathcal{Y}}$ from input fields on Ω^0 to an output field on Ω^0. By the operator characterization above, this mapping is a finite composition of cross-space integral, latent-space nonlocal, and point-wise local operators. The claim therefore holds for a single latent space.

Multiple latent spaces. For $l = 1,\dots,L$, the progressive encoding recursion is:

$$\begin{aligned} \mathcal{F}_{\text{enc}}^l &= \mathcal{M}_{\text{enc}}^l\left(\mathcal{S}_{\text{enc}}^{l-1,l}\mathcal{F}_{\text{enc}}^{l-1}\right), \\ \mathcal{G}^l &= \mathcal{S}_{\text{enc}}^{l-1,l}\mathcal{G}^{l-1}. \end{aligned} \tag{25}$$

The base case $l = 1$ follows from the single-space construction. If $(\mathcal{G}^{l-1},\mathcal{F}_{\text{enc}}^{l-1})$ is obtained from the original input through the permitted operators, then Equation (25) applies one cross-space integral operator followed by the encoder model in Ω^l. Therefore, by forward induction, every $(\mathcal{G}^l,\mathcal{F}_{\text{enc}}^l)$ is a function field on Ω^l produced by a finite composition of the same operators. The progressive decoder is:

$$\mathcal{F}_{\text{dec}}^l = \begin{cases} \mathcal{M}_{\text{dec}}^L\left(\mathcal{F}_{\text{enc}}^L\right), & l = L, \\ \mathcal{M}_{\text{dec}}^l(\mathcal{F}_{\text{enc}}^l + \mathcal{S}_{\text{dec}}^{l,l+1}\mathcal{F}_{\text{dec}}^{l+1}), & 1\le l < L. \end{cases} \tag{26}$$

At the deepest latent space, $\mathcal{F}_{\text{dec}}^L$ is obtained by applying the decoding Transformer model $\mathcal{M}_{\text{dec}}^L$ to $\mathcal{F}_{\text{enc}}^L$. Therefore, $\mathcal{F}_{\text{dec}}^L$ has the operator structure characterized above. The projected field is then combined with $\mathcal{F}_{\text{enc}}^l$ through point-wise skip addition and subsequently updated by $\mathcal{M}_{\text{dec}}^l$, which is a finite composition of latent-space nonlocal and point-wise local operators. Therefore, backward induction from $l = L$ to $l = 1$ shows that every decoded field $\mathcal{F}_{\text{dec}}^l$ is produced by a finite composition of these operators. Then, the predicted physical-field function $\widehat{\mathcal{Y}}$ is obtained as:

$$\begin{aligned} \mathcal{F}_{\text{dec}}^0 &= \mathcal{S}_{\text{dec}}^{0,1}\mathcal{F}_{\text{dec}}^1, \\ \widehat{\mathcal{Y}} &= \text{MLP}_{\text{out}}\left(\mathcal{F}_{\text{dec}}^0\right). \end{aligned} \tag{27}$$

$\mathcal{S}^{0,1}_{\mathrm{dec}}$ maps $\mathcal{F}^{1}_{\mathrm{dec}}$ back to the original space Ω^0 through a cross-space integral operator, while $\mathrm{MLP}_{\mathrm{out}}$ transforms the resulting field into the target physical field through point-wise linear mappings and nonlinear activations. Consequently, $\widehat{\mathcal{Y}}$ is a function on Ω^0 obtained through a finite composition of cross-space integral, latent-space nonlocal, and point-wise local operators. Combining the forward and backward inductions, MoNo defines:

$$\mathcal{N}_\theta : \left(\mathcal{G}^0, \mathcal{F}^0_{\mathrm{enc}}\right) \longmapsto \widehat{\mathcal{Y}}. \tag{28}$$

Combining the encoding and decoding processes, $\mathcal{N}_\theta$ is a finite composition of cross-space integral, latent-space nonlocal, and point-wise local operators. Therefore, MoNo constitutes a neural operator that maps the original input functions to the target physical-field function on Ω^0. □

Appendix E: CoTAP Iteration

Standard Log-domain Sinkhorn Iteration

We first describe the standard log-domain Sinkhorn iteration (Sinkhorn 1967) for solving the entropy-regularized optimal transport problem in CoTAP. Let $\overline{\mathbf{S}}^{l-1,l}_{\mathrm{init}} = \mathbf{S}^{l-1,l}_{\mathrm{init}}/\tau$, and initialize the row and column dual variables $\mathbf{u}^0 \in \mathbb{R}^{N^{l-1}}$ and $\mathbf{v}^0 \in \mathbb{R}^{N^l}$ as zero vectors. Since CoTAP adopts uniform marginal constraints, the row and column marginals in the log domain are $-\log N^{l-1}$ and $-\log N^l$, respectively. Therefore, at the t-th iteration, $\mathbf{u}^{t+1}$ and $\mathbf{v}^{t+1}$ are updated as:

$$\begin{aligned} u_i^{t+1} &= -\log N^{l-1} - \log \sum_{j=1}^{N^l} \exp\left(\overline{S}^{l-1,l}_{\mathrm{init},ij} + v_j^t\right), \\ v_j^{t+1} &= -\log N^{l} - \log \sum_{i=1}^{N^{l-1}} \exp\left(\overline{S}^{l-1,l}_{\mathrm{init},ij} + u_i^{t+1}\right), \end{aligned} \tag{29}$$

where, $t \in \{0, \ldots, T-1\}$ denotes the iteration index, and T denotes the total number of iterations. After T iterations, CoTAP obtains the OT-normalized assignment matrix:

$$S^{l-1,l}_{\mathrm{ot},ij} = \exp\left(\overline{S}^{l-1,l}_{\mathrm{init},ij} + u_i^T + v_j^T\right). \tag{30}$$

Equation (29) alternately enforces the uniform row and column marginals in the log domain, yielding a balanced assignment. However, the standard implementation produces large intermediate tensors at each iteration and retains the complete computation graph for automatic differentiation during training, resulting in substantial GPU memory consumption for large-scale observations. To address this issue, we further develop a Triton-based fused Sinkhorn solver, termed CoTAP Iteration.

Fused Forward in CoTAP Iteration

CoTAP Iteration implements the row and column dual updates at each iteration of Equation (29) as fused Triton kernels. Each Triton program processes one row or column in parallel, performs assignment scaling, log-sum-exp reduction, and dual-variable update using on-chip memory, and writes only the updated dual variable back to GPU memory. After T iterations, a fused output kernel generates $\mathbf{S}^{l-1,l}_{\mathrm{enc}}$ and $\mathbf{S}^{l-1,l}_{\mathrm{dec}}$ without separately materializing $\mathbf{S}^{l-1,l}_{\mathrm{ot}}$.

The fused forward avoids materializing intermediate tensors of size $N^{l-1} \times N^l$ at each iteration and caches only the dual variables required for the backward pass.

Fused Backward in CoTAP Iteration

End-to-end training of the assignment projector $\mathcal{P}^l(\cdot)$ requires propagating the gradient from $\mathbf{S}^{l-1,l}_{\mathrm{enc}}$ and $\mathbf{S}^{l-1,l}_{\mathrm{dec}}$ back to the initial assignment matrix $\mathbf{S}^{l-1,l}_{\mathrm{init}}$. In the standard log-domain Sinkhorn implementation, automatic differentiation computes this gradient by retaining the complete computation graph of all Sinkhorn iterations. However, the fused forward incorporates assignment reconstruction, log-sum-exp reductions, and dual-variable updates into Triton kernels, whose internal operations are not recorded by the automatic differentiation system. Therefore, we explicitly reverse the finite Sinkhorn iterations and accumulate reverse-mode gradients using the dual-variable histories cached during the forward pass.

As $\mathbf{S}^{l-1,l}_{\mathrm{enc}}$ and $\mathbf{S}^{l-1,l}_{\mathrm{dec}}$ are outputs of the fused forward, their upstream gradients $\partial\mathcal{L}/\partial\mathbf{S}^{l-1,l}_{\mathrm{enc}}$ and $\partial\mathcal{L}/\partial\mathbf{S}^{l-1,l}_{\mathrm{dec}}$ are directly available at the backward pass, where $\mathcal{L}$ denotes the training loss. The fused backward first reverses the fused output operation and accumulates the gradients from both projection outputs into the final log-assignment. It then traverses the Sinkhorn iterations in reverse order, propagating gradients through each column dual update followed by the corresponding row dual update. This procedure yields the same gradient with respect to $\mathbf{S}^{l-1,l}_{\mathrm{init}}$ as automatic differentiation through the standard finite-step log-domain Sinkhorn iterations.

We first describe the reverse-mode gradient accumulation with zero and one Sinkhorn iteration and then generalize it to an arbitrary number of iterations T. For simplicity, we denote the final log-assignment after T iterations by $\mathbf{Z}^{l-1,l}$:

$$Z^{l-1,l}_{ij} = \overline{S}^{l-1,l}_{\mathrm{init},ij} + u_i^T + v_j^T. \tag{31}$$

Since $\mathbf{S}^{l-1,l}_{\mathrm{enc}} = N^l \mathbf{S}^{l-1,l}_{\mathrm{ot}}$ and $\mathbf{S}^{l-1,l}_{\mathrm{dec}} = N^{l-1}\mathbf{S}^{l-1,l}_{\mathrm{ot}}$, the upstream gradients from the two projection outputs are accumulated at $\mathbf{Z}^{l-1,l}$:

$$\begin{aligned} \frac{\partial\mathcal{L}}{\partial Z^{l-1,l}_{ij}} &= \frac{\partial\mathcal{L}}{\partial S^{l-1,l}_{\mathrm{enc},ij}}\frac{\partial S^{l-1,l}_{\mathrm{enc},ij}}{\partial Z^{l-1,l}_{ij}} + \frac{\partial\mathcal{L}}{\partial S^{l-1,l}_{\mathrm{dec},ij}}\frac{\partial S^{l-1,l}_{\mathrm{dec},ij}}{\partial Z^{l-1,l}_{ij}} \\ &= \left(N^l \frac{\partial\mathcal{L}}{\partial S^{l-1,l}_{\mathrm{enc},ij}} + N^{l-1}\frac{\partial\mathcal{L}}{\partial S^{l-1,l}_{\mathrm{dec},ij}}\right) S^{l-1,l}_{\mathrm{ot},ij}. \end{aligned} \tag{32}$$

In the fused backward, neither $\mathbf{Z}^{l-1,l}$ nor $\mathbf{S}^{l-1,l}_{\mathrm{ot}}$ is cached as a complete tensor during the forward pass. Instead, both quantities are reconstructed element-wise on demand from $\mathbf{S}^{l-1,l}_{\mathrm{init}}$ within the backward kernel and are discarded immediately after evaluating the local gradient in Equation (32).

Backward with Zero Sinkhorn Iteration. We first consider $T = 0$ as the base case for analyzing the fused output operation. When $T = 0$, no dual-variable update is performed. Therefore, the fused backward only needs to reverse the fused

output operation. In this case, the log-assignment and resulting assignment at each position are:

$$\begin{aligned} Z_{ij}^{l-1,l} &= \overline{S}_{\text{init},ij}^{l-1,l} = \frac{S_{\text{init},ij}^{l-1,l}}{\tau}, \\ S_{\text{ot},ij}^{l-1,l} &= \exp\left(Z_{ij}^{l-1,l}\right). \end{aligned} \tag{33}$$

Therefore, propagating the accumulated gradient in Equation (32) to the initial assignment gives:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial S_{\text{init},ij}^{l-1,l}} &= \frac{\partial \mathcal{L}}{\partial Z_{ij}^{l-1,l}} \frac{\partial Z_{ij}^{l-1,l}}{\partial S_{\text{init},ij}^{l-1,l}} \\ &= \frac{1}{\tau}\left(N^l \frac{\partial \mathcal{L}}{\partial S_{\text{enc},ij}^{l-1,l}} + N^{l-1} \frac{\partial \mathcal{L}}{\partial S_{\text{dec},ij}^{l-1,l}}\right) S_{\text{ot},ij}^{l-1,l}. \end{aligned} \tag{34}$$

Backward with One Sinkhorn Iteration. When $T = 1$, the forward pass performs one row dual update followed by one column dual update, and constructs the final log-assignment as:

$$Z_{ij}^{l-1,l} = \overline{S}_{\text{init},ij}^{l-1,l} + u_i^1 + v_j^1. \tag{35}$$

The fused backward first accumulates $\partial \mathcal{L}/\partial Z_{ij}^{l-1,l}$ from the two projection outputs according to Equation (32), and then reverses the column and row dual updates. The normalized weights associated with the row and column log-sum-exp reductions are:

$$\begin{aligned} h_{ij}^0 &= \frac{\exp\left(\overline{S}_{\text{init},ij}^{l-1,l} + v_j^0\right)}{\sum_{k=1}^{N^l} \exp\left(\overline{S}_{\text{init},ik}^{l-1,l} + v_k^0\right)}, \\ q_{ij}^1 &= \frac{\exp\left(\overline{S}_{\text{init},ij}^{l-1,l} + u_i^1\right)}{\sum_{k=1}^{N^{l-1}} \exp\left(\overline{S}_{\text{init},kj}^{l-1,l} + u_k^1\right)}. \end{aligned} \tag{36}$$

Directly incorporating local derivatives into the reverse-mode accumulation gives the complete gradients of the scaled initial assignment:

$$\frac{\partial \mathcal{L}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} = \underbrace{\frac{\partial \mathcal{L}}{\partial Z_{ij}^{l-1,l}}}_{\text{final log-assignment}} - \underbrace{q_{ij}^1 \frac{\partial \mathcal{L}}{\partial v_j^1}}_{\text{column dual update}} - \underbrace{h_{ij}^0 \frac{\partial \mathcal{L}}{\partial u_i^1}}_{\text{row dual update}}. \tag{37}$$

Finally, propagating through the temperature scaling gives the gradient with respect to the initial assignment matrix:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial S_{\text{init},ij}^{l-1,l}} &= \frac{1}{\tau} \frac{\partial \mathcal{L}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} \\ &= \frac{1}{\tau}\left(\frac{\partial \mathcal{L}}{\partial Z_{ij}^{l-1,l}} - q_{ij}^1 \frac{\partial \mathcal{L}}{\partial v_j^1} - h_{ij}^0 \frac{\partial \mathcal{L}}{\partial u_i^1}\right). \end{aligned} \tag{38}$$

Since the dual variables $\mathbf{u}^1$ and $\mathbf{v}^1$ are cached during the fused forward, the backward kernels can explicitly accumulate the corresponding gradients in reverse-mode order. The normalized weights $\mathbf{h}^0$ and $\mathbf{q}^1$ are reconstructed on demand from the initial assignment and cached dual variables, without being stored as complete assignment-sized tensors.

Backward with an Arbitrary Number of Sinkhorn Iterations. For an arbitrary number of iterations T, the fused backward starts from the final log-assignment and reverses the column and row dual updates of each iteration in the opposite order of the forward computation. For the t-th Sinkhorn iteration, the normalized weights associated with the row and column log-sum-exp reductions are:

$$\begin{aligned} h_{ij}^t &= \frac{\exp\left(\overline{S}_{\text{init},ij}^{l-1,l} + v_j^t\right)}{\sum_{k=1}^{N^l} \exp\left(\overline{S}_{\text{init},ik}^{l-1,l} + v_k^t\right)}, \\ q_{ij}^{t+1} &= \frac{\exp\left(\overline{S}_{\text{init},ij}^{l-1,l} + u_i^{t+1}\right)}{\sum_{k=1}^{N^{l-1}} \exp\left(\overline{S}_{\text{init},kj}^{l-1,l} + u_k^{t+1}\right)}, \end{aligned} \tag{39}$$

where $t \in \{0, \ldots, T-1\}$. Furthermore, the local derivatives of the row and column dual updates at iteration t are:

$$\begin{aligned} \frac{\partial u_i^{t+1}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} &= \frac{\partial u_i^{t+1}}{\partial v_j^t} = -h_{ij}^t, \\ \frac{\partial v_j^{t+1}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} &= \frac{\partial v_j^{t+1}}{\partial u_i^{t+1}} = -q_{ij}^{t+1}. \end{aligned} \tag{40}$$

The complete reverse-mode gradient of the scaled initial assignment is:

$$\frac{\partial \mathcal{L}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} = \underbrace{\frac{\partial \mathcal{L}}{\partial Z_{ij}^{l-1,l}}}_{\text{final log-assignment}} - \underbrace{\sum_{t=0}^{T-1} q_{ij}^{t+1} \frac{\partial \mathcal{L}}{\partial v_j^{t+1}}}_{\text{column dual updates}} - \underbrace{\sum_{t=0}^{T-1} h_{ij}^t \frac{\partial \mathcal{L}}{\partial u_i^{t+1}}}_{\text{row dual updates}}. \tag{41}$$

Propagating through the temperature scaling gives the gradient with respect to the initial assignment matrix:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial S_{\text{init},ij}^{l-1,l}} &= \frac{1}{\tau} \frac{\partial \mathcal{L}}{\partial \overline{S}_{\text{init},ij}^{l-1,l}} \\ &= \frac{1}{\tau}\left[\frac{\partial \mathcal{L}}{\partial Z_{ij}^{l-1,l}} - \sum_{t=0}^{T-1}\left(q_{ij}^{t+1} \frac{\partial \mathcal{L}}{\partial v_j^{t+1}} + h_{ij}^t \frac{\partial \mathcal{L}}{\partial u_i^{t+1}}\right)\right]. \end{aligned} \tag{42}$$

Similar to Equation (38), all gradients with respect to the dual variables in Equation (41), including $\partial \mathcal{L}/\partial v_j^{t+1}$ and $\partial \mathcal{L}/\partial u_i^{t+1}$, are explicitly accumulated by the fused backward kernels in reverse-mode order, while $\mathbf{h}^t$ and $\mathbf{q}^t$ are reconstructed on demand from the scaled initial assignment and cached dual variables. When $T = 0$, the summations are empty and Equation (42) reduces to Equation (34). When $T = 1$, Equation (42) reduces to Equation (38).

Computational Efficiency of CoTAP Iteration

To evaluate the computational efficiency of CoTAP Iteration, we compare it with the standard log-domain Sinkhorn iteration implementation based on PyTorch. All experiments use

N^{l-1}	N^l	Standard Log-domain Sinkhorn Iteration			CoTAP Iteration		
		Forward (ms) ↓	Forward+Backward (ms) ↓	Peak Memory (GB) ↓	Forward (ms) ↓	Forward+Backward (ms) ↓	Peak Memory (GB) ↓
4,096	512	1.34	3.87	0.50	0.74	3.57	0.22
32,768	1,024	44.16	110.76	8.00	12.05	71.19	3.50
65,536	2,048	176.24	OOM	OOM	49.47	466.42	14.00

Table 8: Performance comparison between standard Sinkhorn iterations and CoTAP Iteration. Peak Memory denotes the peak incremental GPU memory during forward-and-backward computation. OOM indicates that the computation exceeds the available memory of a 24-GB GPU.

FP32, a batch size of 4, and four Sinkhorn iterations on an NVIDIA GeForce RTX 4090. We report the runtime over three runs after one warm-up run and the peak GPU memory during forward-and-backward computation.

As shown in Table 8, CoTAP Iteration substantially reduces the forward runtime. When $(N^{l-1}, N^l) = (32{,}768, 1{,}024)$, it reduces the forward time from 44.16 ms to 12.05 ms and the forward-and-backward time from 110.76 ms to 71.19 ms, corresponding to reductions of 72.7% and 35.7%, respectively. CoTAP Iteration also substantially reduces training memory. For $(N^{l-1}, N^l) = (4{,}096, 5{,}12)$ and $(N^{l-1}, N^l) = (32{,}768, 1{,}024)$, CoTAP Iteration decreases the peak incremental memory from 0.50 GB and 8.00 GB to 0.22 GB and 3.50 GB, respectively, yielding reductions of approximately 56%. At the largest scale of $(65{,}536, 2{,}048)$, the standard implementation runs out of memory during the backward pass, whereas CoTAP Iteration completes forward-and-backward computation with 14.00 GB of peak incremental memory.

These results show that the fused forward and explicit fused backward effectively reduce the materialization and computational-graph storage of assignment-sized intermediate tensors, enabling CoTAP to scale more efficiently to large cross-space assignments.

Appendix F: Implementation Details

Benchmarks

We mainly adopt benchmark configurations and data splits as in previous works (Wu et al. 2024; Hu et al. 2026).

Airfoil. The Airfoil benchmark (Li et al. 2023a) evaluates Mach number prediction around different airfoil geometries. Geometries are generated by deforming the baseline NACA-0012 airfoil and discretized into structured meshes of size 221×51. The input and output are the two-dimensional grid coordinates ($221 \times 51 \times 2$) and Mach numbers ($221 \times 51 \times 1$), respectively. The dataset contains 1,000 training samples and 200 testing samples.

Darcy. The Darcy benchmark (Li et al. 2020a) evaluates steady-state pressure-field prediction under different porous-medium distributions, with the original physical process discretized on a 421×421 standard grid. The input consists of the two-dimensional coordinates ($421 \times 421 \times 2$) and corresponding porous-medium coefficient ($421 \times 421 \times 1$) at each grid point. The output is the pressure at each grid point, with a shape of $421 \times 421 \times 1$. For the multiple-resolution evaluation, we construct Darcy benchmark variants at different downsampled resolutions, including 85×85, 141×141, and 211×211. The dataset contains 1,000 training samples and 200 testing samples.

Elasticity. The Elasticity benchmark (Li et al. 2023a) evaluates the prediction of internal stress distributions for different material structures. Each structure is represented by 972 point-cloud observations. The input consists of the two-dimensional coordinates of these points, with a shape of 972×2 The output is the stress value at each point, with a shape of 972×1. The dataset contains 1,000 training samples and 200 testing samples.

Navier-Stokes (NS2D). The Navier-Stokes benchmark (Li et al. 2020a), denoted as NS2D, evaluates autoregressive prediction of two-dimensional time-dependent flow fields. The flow field is discretized on a 64×64 Cartesian grid. The input consists of the velocity fields from the previous 10 time steps, with a shape of $64 \times 64 \times 10 \times 1$. The output contains the velocity fields for the subsequent 10 time steps with a shape of $64 \times 64 \times 10 \times 1$. The dataset contains 1,000 training samples and 200 testing samples.

Pipe. The Pipe benchmark (Li et al. 2023a) evaluates velocity-field prediction. The geometries are generated by varying the pipe centerline and are discretized into structured meshes of size 129×129. The input consists of the two-dimensional coordinates of the mesh points, with a shape of $129 \times 129 \times 2$ The output is the velocity at each point, with a shape of $129 \times 129 \times 1$. The dataset contains 1,000 training samples and 200 testing samples.

Plasticity. The Plasticity benchmark (Li et al. 2023a) evaluates the future deformation of a plastic material. For each sample, the die geometry is discretized into a structured mesh of size 101×31. The input includes the two-dimensional coordinates ($101 \times 31 \times 2$) and corresponding external force at each mesh point ($101 \times 31 \times 1$). The output describes the deformation of each point along four directions over the subsequent 20 time steps, with a shape of $101 \times 31 \times 20 \times 4$. Unlike the autoregressive prediction used for NS2D, Plasticity encodes each target time step as a time embedding and concatenates it into the model input, allowing the deformation at each time step to be predicted independently. The dataset contains 900 training samples with different die shapes and 80 testing samples.

AirfRANS. AirfRANS (Bonnet et al. 2022) describes two-dimensional incompressible steady-state Reynolds-Averaged Navier-Stokes flows over different airfoil geometries, Reynolds numbers, and angles of attack. Each case is represented by a point cloud containing a varying number of observation points. We denote the number of observation points in the i-th case by N^i_{AirfRANS}.

The input consists of two-dimensional spatial coordinates ($N^i_{\text{AirfRANS}} \times 2$), two-dimensional inlet velocity ($N^i_{\text{AirfRANS}} \times 2$), Euclidean distance to the airfoil ($N^i_{\text{AirfRANS}} \times 1$), and two-dimensional outward-pointing unit surface normal ($N^i_{\text{AirfRANS}} \times 2$). The output consists of the two-dimensional velocity ($N^i_{\text{AirfRANS}} \times 2$), pressure ($N^i_{\text{AirfRANS}} \times 1$), and turbulent kinematic viscosity ($N^i_{\text{AirfRANS}} \times 1$). AirfRANS contains 1,000 samples in total. Following previous works (Wu et al. 2024; Hu et al. 2026), we use different data splits according to the evaluation setting. For the main experiment, 720, 80, and 200 samples are used for training, validation, and testing, respectively. The detailed configurations of the OOD experiments are provided in Appendix H.

Evaluation Metrics

We use relative L2 error to evaluate prediction accuracy across all benchmarks. For AirfRANS, we additionally evaluate the corresponding aerodynamic force coefficients and their Spearman rank correlations as previous works (Wu et al. 2024; Hu et al. 2026).

Relative L2 Error. For each test sample, the relative L2 error, termed $\mathcal{E}_{\text{rL2}}$, is defined as the L2 error between the predicted and ground-truth physical fields normalized by the L2 norm of the ground truth:

$$\mathcal{E}_{\text{rL2}} = \frac{\left\| Y - \widehat{Y} \right\|_2}{\|Y\|_2}, \tag{43}$$

where Y and $\widehat{Y}$ denote the ground-truth and predicted physical fields, respectively, and $\|\cdot\|_2$ denotes the L2 norm. The reported result is averaged over all testing samples.

Lift Coefficient. The lift coefficient C_L is used to evaluate global aerodynamic performance on AirfRANS. It is computed by integrating the pressure and viscous forces over the airfoil surface to obtain the lift force, which is subsequently normalized by the freestream dynamic pressure and reference area. The detailed computation follows Transolver (Wu et al. 2024) and LinearNO (Hu et al. 2026).

Spearman Rank Correlation. The Spearman rank correlation ρ_L measures the ranking consistency between the predicted and ground-truth lift coefficients. Given n testing samples, it is defined as the Pearson correlation between the ranks of the ground-truth and predicted lift coefficient sequences:

$$\rho_L = \frac{\text{Cov}\left(\text{rank}(\mathbf{c}_L), \text{rank}(\widehat{\mathbf{c}}_L)\right)}{\sigma_{\text{rank}(\mathbf{c}_L)} \sigma_{\text{rank}(\widehat{\mathbf{c}}_L)}}, \tag{44}$$

where $\mathbf{c}_L$ and $\widehat{\mathbf{c}}_L$ denote the ground-truth and predicted lift coefficient sequences over all testing samples, respectively. $\text{Cov}(\cdot,\cdot)$ denotes the covariance, $\text{rank}(\cdot)$ denotes the ranking operation, and $\sigma_{\text{rank}(\cdot)}$ denotes the standard deviation. A value of ρ_L closer to 1 indicates stronger ranking consistency.

Baselines

For baselines, we primarily use the results reported by LinearNO (Hu et al. 2026). For the multiple-resolution evaluation, we implement and evaluate the baselines using their publicly available code.

MoNo

CoTAP. For all benchmarks, the temperature coefficient τ and the number of iterations T in CoTAP are set to 1.0 and 8, respectively. Each assignment projector $\mathcal{P}^l(\cdot)$ is implemented as an MLP with three hidden linear layers. Except for Darcy, we further apply row-wise weight normalization to the final linear layer of $\mathcal{P}^1(\cdot)$. Specifically, each row of its weight matrix is first normalized by its L2 norm and then multiplied by a learnable scale initialized to 3.0. This design prevents the model from increasing the assignment scores of specific latent tokens merely by enlarging the norms of the corresponding output weights during training, thereby improving the stability of initial assignment learning.

Training. All experiments are conducted with a random seed of 0. MoNo and MoNo-light use the same training hyper-parameters on the standard benchmarks, as summarized in Table 9. Compared with LinearNO and Transolver, MoNo and MoNo-light use the same or fewer total optimization steps for each benchmark, ensuring a fair comparison.

For the multiple-resolution evaluation on Darcy, MoNo and MoNo-light are trained for 500 epochs using the AdamW optimizer, relative L2 loss, a OneCycle scheduler with 100 warmup epochs, and a batch size of 4. For MoNo, the learning rate is set to 8×10^{-4} at all resolutions. We observe that MoNo-light does not fully converge with a learning rate of 8×10^{-4}. Therefore, we use slightly larger learning rates while keeping all other training hyper-parameters unchanged. Specifically, the learning rate is set to 1×10^{-3} at resolutions of 85×85 and 141×141, and to 1.4×10^{-3} at a resolution of 211×211.

Following previous works (Wu et al. 2024; Hu et al. 2026), we sample 32,000 points from each AirfRANS case during training and repeatedly perform sampling and prediction during inference until all points are covered. Since MoNo relies on stable mapping, we find that the fully random sampling strategy with a batch size of 1 adopted in previous works causes severe training instability. To address this issue, each sampled case contains all surface points together with randomly sampled volume points, yielding 32,000 points in total. We further set the batch size to 4 and increase the number of training epochs to preserve the same total number of optimization steps. Both MoNo and MoNo-light use the relative L2 loss, AdamW optimizer, and OneCycle scheduler, with the learning rate and number of warmup epochs set to 2×10^{-4} and 100, respectively. Following previous works, we jointly supervise the prediction of physical fields in the surrounding region and on the surface, assigning both loss terms a weight of 1.0. Both OOD Reynolds and OOD Angles experiments use the same training hyper-parameters as the main AirfRANS experiment.

Hardware and Software Environment. All experiments are conducted using a single NVIDIA GeForce RTX 4090 GPU with up to 600 GB of system memory. The core software environment consists of PyTorch 2.8.0, CUDA 12.8, cuDNN 9.10.2, and Triton 3.4.0.

Benchmark	Loss	Epochs	Warmup Epochs	LR	Optimizer	Scheduler	Batch Size
Airfoil	rL2	500	100	8×10^{-4}	AdamW	OneCycle	4
Elasticity	rL2	500	100	8×10^{-4}	AdamW	OneCycle	4
NS2D	rL2	500	100	8×10^{-4}	AdamW	OneCycle	2
Pipe	rL2	500	100	8×10^{-4}	AdamW	OneCycle	4
Plasticity	rL2	500	100	8×10^{-4}	AdamW	OneCycle	8

Table 9: Training hyper-parameters on the standard benchmarks. MoNo and MoNo-light use the same settings.

Appendix G: Efficiency Analysis

Table 10 reports the parameter counts and computational costs. All results use FP32 inference with a batch size of one. All models use two-channel inputs and single-channel outputs for evaluation. MoNo-light consistently achieves the lowest computational cost across all input sizes, while MoNo also exhibits increasingly clear computational advantages and outperforms all baselines at larger input scales.

Appendix H: Detailed OOD Results

The training and test ranges for the OOD Reynolds and OOD Angles settings are summarized in Table 11. Table 12 further presents the complete OOD results on AirfRANS, including the lift coefficient error C_L and its Spearman rank correlation ρ_L. MoNo achieves the best performance across all metrics under both OOD settings. On OOD Reynolds, MoNo achieves a C_L error of 0.1050, outperforming the strongest baseline, Transolver, by 35.3%, while improving the ρ_L from 0.9951 (LinearNO) to 0.9983. On OOD Angles, MoNo further achieves a C_L error of 0.0782, outperforming the strongest baseline, LinearNO, by 20.8%, and improves ρ_L from 0.9963 to 0.9970. These results demonstrate that MoNo can extrapolate not only local physical fields but also global aerodynamic performance under unseen flow conditions.

Appendix I: More Ablation Studies

Latent Token Number and Token Dimensions. Table 13 analyzes the effects of the number of latent tokens N^1 in the first latent space Ω^1 and the token dimensions of anchor tokens and physical state tokens. Increasing N^1 and the token dimensions initially improves performance across the three tasks, after which performance gradually saturates, and further scaling provides no consistent gains.

Compared with the initial configuration using $N^1 = 256$ and token dimensions of 48, MoNo-light uses $N^1 = 512$ and token dimensions of 96, reducing the errors on Airfoil, Elasticity, and Pipe by 20.0%, 66.9%, and 27.0%, respectively. MoNo further uses $N^1 = 1024$ and token dimensions of 192, reducing the errors on the three tasks by 20.0%, 74.0%, and 43.2%, respectively, compared with the initial configuration. These results show that MoNo-light achieves a favorable balance between model scale and overall solution accuracy, while MoNo further improves overall accuracy through increased representational capacity.

CoTAP Hyper-parameters. Table 14 analyzes the effects of the temperature coefficient τ and the number of iterations T with MoNo-light. The temperature coefficient τ controls the sharpness of the assignment distribution. A smaller τ weakens entropy regularization and produces sharper assignments, whereas a larger τ yields smoother assignments. With $T = 8$, both $\tau = 0.5$ and $\tau = 1.0$ achieve stable

Model	Parameters (M)	Input Points N^0	GFLOPs
GNOT	6.73	1,024	6.94
		2,048	13.88
		4,096	27.76
		8,192	55.52
		16,384	111.04
		32,768	222.09
		65,536	444.18
Transolver	2.81	1,024	3.08
		2,048	6.14
		4,096	12.27
		8,192	24.53
		16,384	49.06
		32,768	98.10
		65,536	196.19
Transolver++	1.74	1,024	1.90
		2,048	3.79
		4,096	7.58
		8,192	15.15
		16,384	30.30
		32,768	60.59
		65,536	121.17
LinearNO	1.77	1,024	2.08
		2,048	4.15
		4,096	8.30
		8,192	16.60
		16,384	33.21
		32,768	66.41
		65,536	132.83
MoNo-light	1.73	1,024	1.23
		2,048	1.58
		4,096	2.29
		8,192	3.70
		16,384	6.52
		32,768	12.17
		65,536	23.46
MoNo	6.90	1,024	8.41
		2,048	9.82
		4,096	12.64
		8,192	18.28
		16,384	29.56
		32,768	52.13
		65,536	97.26

Table 10: Efficiency comparison with different numbers of input points. MoNo achieves higher computational efficiency.

Split	OOD Reynolds: Reynolds Number Range	OOD Angles: AoA Range
Training Set	$[3 \times 10^6, 5 \times 10^6]$	$[-2.5^\circ, 12.5^\circ]$
Test Set	$[2 \times 10^6, 3 \times 10^6] \cup [5 \times 10^6, 6 \times 10^6]$	$[-5^\circ, -2.5^\circ] \cup [12.5^\circ, 15^\circ]$

Table 11: Training and test ranges for the AirfRANS OOD Reynolds and OOD Angles evaluations.

Method	AirfRANS (OOD Reynolds)				AirfRANS (OOD Angles)			
	Vol. (↓)	Surf. (↓)	C_L (↓)	ρ_L (↑)	Vol. (↓)	Surf. (↓)	C_L (↓)	ρ_L (↑)
MLP	0.0669	0.1153	0.6205	0.9578	0.1309	0.3311	0.4128	0.9572
PointNet (Qi et al. 2017)	0.0838	0.1403	0.3836	0.9806	0.2021	0.4649	0.4425	0.9784
Graph U-Net (Bronstein et al. 2016)	0.0538	0.1168	0.4664	0.9645	0.0979	0.2391	0.3756	0.9816
MeshGraphNet (Pfaff et al. 2020)	0.2789	0.2382	1.7718	0.7631	0.4902	1.1071	0.6525	0.8927
GNO (Li et al. 2020b)	0.0833	0.1562	0.4408	0.9878	0.1626	0.2359	0.3038	0.9884
GALERKIN (Cao 2021)	0.0330	0.0972	0.4615	0.9826	0.0577	0.2773	0.3814	0.9821
GNOT (Hao et al. 2023)	0.0305	0.0959	0.3268	0.9865	0.0471	0.3466	0.3497	0.9868
GINO (Li et al. 2023b)	0.0839	0.1825	0.4180	0.9645	0.1589	0.2469	0.2583	0.9923
LNO (Wang and Wang 2024)	0.0825	0.1762	0.7440	0.9399	0.0346	0.0790	0.3732	0.9905
Transolver (Wu et al. 2024)	0.0122	0.0550	<u>0.1622</u>	0.9904	0.0480	0.2335	0.2438	0.9948
LinearNO (Hu et al. 2026)	0.0112	0.0372	0.2400	<u>0.9951</u>	0.0464	0.2500	0.0987	<u>0.9963</u>
MoNo-Light (ours)	<u>0.0090</u>	<u>0.0146</u>	0.1816	0.9936	<u>0.0222</u>	<u>0.0553</u>	<u>0.0951</u>	0.9953
MoNo (ours)	**0.0066**	**0.0121**	**0.1050**	**0.9983**	**0.0097**	**0.0248**	**0.0782**	**0.9970**

Table 12: Comparison on the AirfRANS OOD Reynolds and OOD Angles benchmarks. We evaluate both the surrounding (Vol.) and surface (Surf.) physical fields, as well as the lift coefficient (C_L). ρ_L denotes Spearman's rank correlation for C_L.

N^1	D_g & D_f	Params. (M)	Airfoil	Elasticity	Pipe
256	48	0.44	0.0060	0.0127	0.0037
512	48	0.46	0.0053	0.0123	0.0036
512	96	1.73	0.0048	0.0042	0.0027
512	192	6.71	0.0048	0.0033	0.0022
1024	192	6.90	0.0048	0.0033	0.0021
1024	384	26.77	0.0046	0.0035	0.0021
2048	384	27.50	0.0049	0.0031	0.0023

Table 13: Ablation studies of the number of tokens in the first latent space and token dimensions in relative L2 error (↓).

τ	T	Airfoil	Elasticity	Pipe
0.5	8	0.0048	0.0041	0.0023
4.0	8	0.0050	0.0046	0.0027
1.0	4	0.0045	0.0043	0.0026
1.0	12	0.0048	0.0054	0.0024
1.0	8	0.0048	0.0042	0.0027

Table 14: Ablation studies of τ and T in CoTAP by relative L2 error (↓).

Input for $\mathbf{S}_{\text{init}}^{l-1,l}$	Reduction Factor	Airfoil	Elasticity	Pipe
F_{enc}^{l-1}	$\times 2$	0.0050	0.0058	0.0031
G^{l-1}	$\times 1$	0.0046	0.0050	0.0026
G^{l-1}	$\times 4$	0.0048	0.0064	0.0030
G^{l-1}	$\times 2$	0.0048	0.0042	0.0027

Table 15: Ablation studies of the input for initial assignment construction and the token reduction factor between adjacent latent spaces in relative L2 error (↓).

performance, whereas increasing τ to 4.0 reduces prediction performance. These results indicate that latent tokens require sufficiently sharp assignments to better represent local physical information. Increasing the number of iterations allows the assignment matrix to satisfy the marginal constraints more closely, effectively imposing stronger normalization, but also makes convergence more difficult. The results show that increasing T from 8 to 12 provides no consistent performance improvement and noticeably reduces prediction accuracy on Elasticity.

In all experiments, we use $\tau = 1.0$ and $T = 8$ as the default configuration to balance assignment sharpness, marginal constraint enforcement, and training convergence.

Architectural Designs. Table 15 analyzes the input for initial assignment construction and the token reduction factor between adjacent latent spaces. Constructing assignments from F_{enc}^{l-1} enables case-specific projections conditioned on the current physical states but substantially reduces the stability of CoTAP across latent spaces. With limited training data, this additional flexibility encourages the model to fit complex case-specific cross-space projections instead of sufficiently learning physical interactions within each latent space.

For the token reduction factor, $\times 1$ preserves the number of tokens and only redistributes them between adjacent spaces,

thereby providing no effective hierarchical information aggregation. In contrast, $\times 4$ compresses physical states too rapidly and causes substantial information loss. Therefore, we construct CoTAP from G^{l-1} and use a token reduction factor of $\times 2$ by default to balance cross-space projection stability, progressive information aggregation, and representational capacity at each scale.

Appendix J: The Use of Large Language Models

All major technical contributions and theoretical derivations in this work were conceived and developed by the authors. Large language models (LLMs) were used for language polishing, auxiliary analysis, and checking theoretical derivations. All LLM-assisted descriptions and analyses were carefully reviewed and verified by the authors.

References

Alkin, B.; Fürst, A.; Schmid, S. L.; Gruber, L.; Holzleitner, M.; and Brandstetter, J. 2024. Universal physics transformers: A framework for efficiently scaling neural operators. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Bonnet, F.; Mazari, J. A.; Cinnella, P.; et al. 2022. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Bronstein, M. M.; Bruna, J.; LeCun, Y.; Szlam, A.; and Vandergheynst, P. 2016. Geometric deep learning: going beyond euclidean data. *arXiv preprint arXiv:1611.08097*.

Cao, S. 2021. Choose a transformer: Fourier or galerkin. *Advances in neural information processing systems*, 34: 24924–24940.

Hao, Z.; Wang, Z.; Su, H.; Ying, C.; Dong, Y.; Liu, S.; Cheng, Z.; Song, J.; and Zhu, J. 2023. Gnot: A general neural operator transformer for operator learning. In *International conference on machine learning*, 12556–12569. PMLR.

Herde, M.; Raonić, B.; Rohner, T.; Käppeli, R.; Molinaro, R.; De Bezenac, E.; and Mishra, S. 2024. Poseidon: Efficient foundation models for pdes. *Advances in Neural Information Processing Systems*, 37: 72525–72624.

Hu, W.; Liu, S.; Qiao, P.; Sun, Z.; and Dou, Y. 2026. Transolver is a linear transformer: Revisiting physics-attention through the lens of linear attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 408–416.

Karniadakis, G. E.; Kevrekidis, I. G.; Lu, L.; Perdikaris, P.; Wang, S.; and Yang, L. 2021. Physics-informed machine learning. *Nature Reviews Physics*, 3(6): 422–440.

Kovachki, N.; Li, Z.; Liu, B.; Azizzadenesheli, K.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2023. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89): 1–97.

Li, X.; Li, Z.; Kovachki, N.; and Anandkumar, A. 2025. Geometric operator learning with optimal transport. *arXiv preprint arXiv:2507.20065*.

Li, Z.; Huang, D. Z.; Liu, B.; and Anandkumar, A. 2023a. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388): 1–26.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020a. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020b. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*.

Li, Z.; Kovachki, N.; Choy, C.; Li, B.; Kossaifi, J.; Otta, S.; Nabian, M. A.; Stadler, M.; Hundt, C.; Azizzadenesheli, K.; et al. 2023b. Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems*, 36: 35836–35854.

Li, Z.; Meidani, K.; and Farimani, A. B. 2022. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*.

Li, Z.; Shu, D.; and Barati Farimani, A. 2023. Scalable transformer for pde surrogate modeling. *Advances in Neural Information Processing Systems*, 36: 28010–28039.

Liu, P.; Wang, P.; Ren, X.; Yuan, H.; Hao, Z.; Xu, C.; Cai, S.; and Ni, D. 2025. Aerogto: An efficient graph-transformer operator for learning large-scale aerodynamics of 3d vehicle geometries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 18924–18932.

Lu, L.; Jin, P.; and Karniadakis, G. E. 2019. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*.

Luo, H.; Wu, H.; Zhou, H.; Xing, L.; Di, Y.; Wang, J.; and Long, M. 2025a. Transolver++: An accurate neural solver for pdes on million-scale geometries. *arXiv preprint arXiv:2502.02414*.

Luo, K.; Zhao, J.; Wang, Y.; Li, J.; Wen, J.; Liang, J.; Soekmadji, H.; and Liao, S. 2025b. Physics-informed neural networks for PDE problems: A comprehensive review. *Artificial Intelligence Review*, 58(10): 323.

Ma, Y.; Wu, H.; Zhou, H.; Weng, H.; Wang, J.; and Long, M. 2026. Physense: Sensor placement optimization for accurate physics sensing. *Advances in Neural Information Processing Systems*, 38: 35697–35725.

McCabe, M.; Régaldo-Saint Blancard, B.; Parker, L.; Ohana, R.; Cranmer, M.; Bietti, A.; Eickenberg, M.; Golkar, S.; Krawezik, G.; Lanusse, F.; et al. 2025. Multiple physics pretraining for spatiotemporal surrogate models. *Advances in Neural Information Processing Systems*, 37: 119301–119335.

Niaki, S. A.; Haghighat, E.; Campbell, T.; Poursartip, A.; and Vaziri, R. 2021. Physics-informed neural network for modelling the thermochemical curing process of composite-tool

systems during manufacture. *Computer Methods in Applied Mechanics and Engineering*, 384: 113959.

Pfaff, T.; Fortunato, M.; Sanchez-Gonzalez, A.; and Battaglia, P. 2020. Learning mesh-based simulation with graph networks. In *International conference on learning representations*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, K.; Wang, F.; Dong, Z.; and Sun, J. 2026. SpiderSolver: A Geometry-Aware Transformer for Solving PDEs on Complex Geometries. *Advances in Neural Information Processing Systems*, 38: 152983–153013.

Raissi, M.; Yazdani, A.; and Karniadakis, G. E. 2020. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481): 1026–1030.

Serrano, L.; Wang, T. X.; Naour, E. L.; Vittaut, J.-N.; and Gallinari, P. 2024. Aroma: Preserving spatial structure for latent pde modeling with local neural fields. *arXiv preprint arXiv:2406.02176*.

Sinkhorn, R. 1967. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 74(4): 402–405.

Tran, A.; Mathews, A.; Xie, L.; and Ong, C. S. 2021. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Villani, C.; et al. 2009. *Optimal transport: old and new*, volume 338. Springer.

Wandel, N.; Weinmann, M.; Neidlin, M.; and Klein, R. 2022. Spline-pinn: Approaching pdes without data using fast, physics-informed hermite-spline cnns. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 8529–8538.

Wang, T.; and Wang, C. 2024. Latent neural operator for solving forward and inverse pde problems. *Advances in Neural Information Processing Systems*, 37: 33085–33107.

Wang, Z.; Yang, Z.; Fu, D.; and Liu, Z. 2025. Learning Material-Geometry-Aware Fourier Neural Operator for Stress-Strain Prediction and Parameter Inversion. *International Journal of Applied Mechanics*, 17: 01108.

Wen, G.; Li, Z.; Azizzadenesheli, K.; Anandkumar, A.; and Benson, S. M. 2022. U-FNO—An enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163: 104180.

Wen, S.; Kumbhat, A.; Lingsch, L.; Mousavi, S.; Zhao, Y.; Chandrashekar, P.; and Mishra, S. 2026. Geometry aware operator transformer as an efficient and accurate neural surrogate for pdes on arbitrary domains. *Advances in Neural Information Processing Systems*, 38: 155423–155501.

Wu, G.; and Wu, Z. 2026. A multi-objective optimization framework for adaptive weighting in physics-informed machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, 26885–26893.

Wu, H.; Hu, T.; Luo, H.; Wang, J.; and Long, M. 2023. Solving high-dimensional pdes with latent spectral models. *arXiv preprint arXiv:2301.12664*.

Wu, H.; Luo, H.; Wang, H.; Wang, J.; and Long, M. 2024. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*.

Xiao, Z.; Hao, Z.; Lin, B.; Deng, Z.; and Su, H. 2023. Improved operator learning by orthogonal attention. *arXiv preprint arXiv:2310.12487*.

Yang, H.-R.; and Ren, C.-X. 2025. A Physics-preserved Transfer Learning Method for Differential Equations. *arXiv preprint arXiv:2505.01281*.

Yang, Z.; Qiu, Z.; and Fu, D. 2023. DMIS: Dynamic mesh-based importance sampling for training physics-informed neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 5375–5383.

Zhou, H.; Ma, Y.; Wu, H.; Wang, H.; and Long, M. 2024. Unisolver: Pde-conditional transformers towards universal neural pde solvers. *arXiv preprint arXiv:2405.17527*.