Physics-Informed Functional Tucker Method with RKHS Factors for Sparse Spatiotemporal Reconstruction

Zhengyang Liu[1] Jianguo Huang[2] Junxi Hu[1] Yucheng Mi[1] Haojie Cai[1]
Wenxuan Shen[3] Zhikun Zhang[1] Renfeng Peng[5,*] Guangtao Zhang[1,4,*]
Zhiqiang Liu[4,*]
[1]SandGold AI Research, Guangzhou, China
[2]Nanyang Technological University, Singapore
[3]Westlake University, Hangzhou, China
[4]Southern University of Science and Technology, Shenzhen, China
[5]Chinese Academy of Sciences, Beijing, China
pengrenfeng@lsec.cc.ac.cn; tao@sgd-ai.com; liuzq@sustech.edu.cn

Abstract

Reconstructing physical dynamics from fixed sparse sensors and missing frames is a fundamental challenge in scientific research. Sparsity leaves large regions of the field unconstrained, making the inverse problem ill-posed: a model may fit the observed sensors while producing inaccurate off-sensor structures or unreliable predictions between support frames. To this end, we propose PhysFTM—a physics-informed functional Tucker model that parameterizes the Tucker mode factors in reproducing kernel Hilbert spaces (RKHSs) and incorporates PDE constraints through finite-difference residuals evaluated at off-sensor collocation points. These residuals provide additional physical information in regions without direct observational supervision. PhysFTM first reconstructs continuous spatial fields at support frames via an alternating minimization scheme, optimizing the Tucker core tensor and the RKHS-based factors separately. To enable missing-frame reconstruction, PhysFTM constructs a continuous trajectory in Tucker-core space, which is decoded by the learned spatial RKHS-FTM representation to recover the full spatiotemporal field. Experiments on Allen–Cahn, Navier–Stokes, three-dimensional Heat equations, and a three-dimensional gridded Copernicus ocean reanalysis demonstrate accurate 2D/3D reconstruction under two missing-frame protocols. Code is available at https://github.com/liusirzzz/PhysFTM.

1 Introduction

Reconstructing spatiotemporal fields aims to recover full physical fields from a small set of fixed sensor locations, which is a fundamental problem in scientific research [10, 14, 25, 43, 45, 47, 51]. Such sparsity leaves most locations and time instants unconstrained, making the inverse problem severely underdetermined: fine-scale structures such as vortices, fronts, and submesoscale processes may remain unresolved, and a model may fit the observed sensors while producing nonphysical off-sensor fields or unreliable missing-frame predictions [44]. This motivates reconstruction methods that are accurate at sensor locations while improving physical consistency on the missing-frame off-sensor locations.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Data-driven reconstruction methods have made substantial progress in sparse physical-field recovery [22, 48, 49], including sensor-to-field decoders, neural implicit representations, low-rank functional tensors, and generative latent models [6, 17–19]. A common paradigm is to first reconstruct fields or latent states from sparse measurements and then use interpolation, recurrent models, or generative priors to infer the temporal evolution [6]. Functional Tucker models offer compact representations of continuous fields by combining mode-wise factor functions with a low-dimensional core tensor [24, 38].However, these methods remain largely driven by sparse observation fitting or learned priors. Continuous representability alone does not ensure physical consistency where observations are absent.

For systems with an available equation constraint, physical laws provide an additional source of supervision where sparse data provide none [44]. The observation loss is evaluated only at sensor locations, whereas the desired reconstruction is defined over a continuous domain. Finite-difference equation residuals can be imposed at off-sensor collocation points, providing an additional inductive bias in regions where no measurements are available [29]. Thus, PDE residuals are not merely post-hoc physical-consistency metrics or generic smoothing penalties; they reduce the ambiguity of sparse reconstruction by constraining the decoded field in the observation null space. To use such regularization, the representation must be continuously evaluable at sensors, collocation points, and neighboring stencil locations, while remaining compact enough for sparse inverse recovery. This motivates a structured continuous representation that separates spatial coordinate dependence from low-dimensional time-varying coefficients.

Motivated by this view, we propose **PhysFTM**, which represents the continuous dynamics as

$$\hat{u}(\mathbf{x}, t, c) = \mathcal{D}_{\Phi}(G(t))(\mathbf{x}, c),$$

where $\mathcal{D}_{\Phi}$ is a spatially continuous Functional Tucker decoder and $G(t)$ is a continuous trajectory in Tucker-core space. The decoder can be evaluated at sensor locations, dense grids, and collocation points, while temporal evolution is modeled through low-dimensional core coefficients. To make the spatial decoder reliable under sparse supervision, PhysFTM parameterizes its mode-wise factors in reproducing kernel Hilbert spaces (RKHSs) and regularizes their RKHS norms [1, 39]. This replaces unconstrained neural mode functions with explicitly regularized functional factors. Together with finite-difference PDE residuals at collocation points, the resulting RKHS-FTM decoder provides both continuous spatial evaluability and physics-guided off-sensor guidance.

Our contributions are summarized as follows. 1) We propose PhysFTM, a functional Tucker framework that combines RKHS-based spatial factors with a continuous trajectory in Tucker-core space for sparse spatiotemporal reconstruction, paving the way for inquiries of the field at arbitrary times and positions. 2) We introduce a hybrid solve-learn optimization scheme that solves conditionally linear Tucker cores with regularized conjugate gradient method while learning RKHS factors under observation, RKHS, and PDE losses. 3) We provide theoretical guarantees for reconstruction via the proposed PhysFTM framework, and evaluate PhysFTM on Allen–Cahn, Navier–Stokes, Heat equations, and a gridded Copernicus ocean reanalysis, covering 2D/3D reconstruction, showing improved accuracy,off-sensor reconstruction accuracy, physical consistency, and missing frame reconstruction.

2 Related Work

Sparse reconstruction of physical fields. Early deep-learning approaches often transformed sparse measurements into structured grid representations, e.g., through Voronoi-assisted representations, followed by convolutional decoders for full-field recovery [13]. Senseiver [30] encodes unordered sparse sensor sets with attention and decodes field values at query coordinates. SHRED and SINDy-SHRED [16, 42] reconstruct high-dimensional spatiotemporal states from sensor histories using recurrent latent representations or identified latent dynamics. STRIDE [37] combines temporal encoders with modulated implicit decoders for continuous field reconstruction. Additionally, generative approaches were explored for physical field recovery, including conditional neural-field latent diffusion for spatiotemporal turbulence [9] and diffusion in Functional Tucker space for full-field dynamics from irregular sparse observations [6]. Existing approaches often rely on neural decoders that remain weakly constrained in off-sensor regions under sparse observations.

Functional tensor and low-rank continuous representations. Low-rank tensor decompositions provide compact multilinear representations for high-dimensional data, with Tucker decomposition

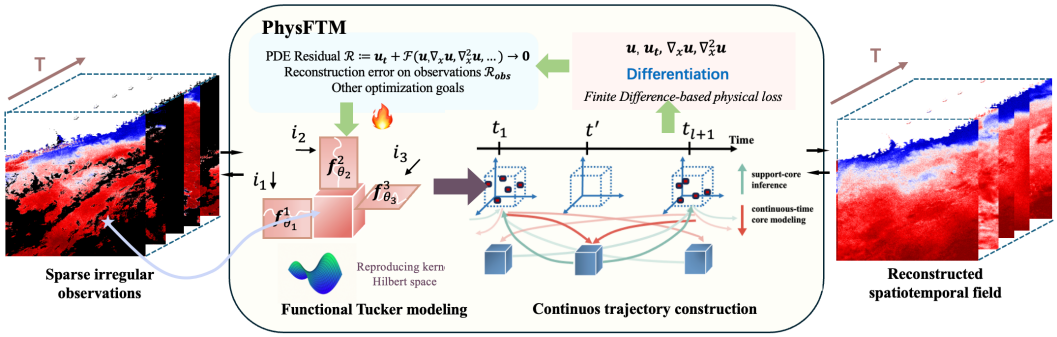

Figure 1: Overall framework of PhysFTM. PhysFTM reconstructs sparse grid-based sensor observations by learning an RKHS-regularized Functional Tucker spatial decoder constrained by observation loss and finite-difference PDE residuals. The inferred Tucker cores are then modeled as a continuous trajectory in core space and decoded back to the physical domain to obtain the reconstructed spatiotemporal field.

being one of the classical formulations based on mode factors and a core tensor [15, 21, 38]. To extend such representations beyond fixed grids, one may consider continuous or functional variants that replace discrete factor matrices with coordinate-dependent factor functions, enabling field reconstruction and evaluation at arbitrary locations; see, e.g., [12, 24]. Functional Tucker models, where observations are reconstructed by shared latent functions and time-dependent Tucker cores [6], were adapted as latent spaces for sparse physical dynamics. Related functional tensor approaches include [8, 33, 40, 41], which offer alternative representations and priors for continuous tensor modeling.

Physics-informed reconstruction and PDE supervision. Physics-informed learning incorporates governing equations into neural approximation by penalizing PDE residuals at collocation points. PINNs are the canonical example, using residual losses together with data, boundary, or initial-condition terms to solve forward and inverse PDE problems [26]; broader physics-informed machine learning has been reviewed as a way to combine data with scientific constraints [20]. Physics-informed neural operators extend this idea to operator learning by combining data losses with PDE residuals or physical consistency terms [23, 46]. Discrete and spatiotemporal architectures such as PhyCRNet impose residual constraints in convolutional-recurrent models for PDE dynamics [29]. In reconstruction settings, Hidden Fluid Mechanics uses Navier–Stokes constraints to infer velocity and pressure fields from indirect observations [27], and physics-informed diffusion models incorporate physical conditions to improve high-fidelity flow-field reconstruction [31]. These works show that PDE residuals can provide supervision beyond labeled samples and improve physical consistency.

3 Methodology of PhysFTM

3.1 Problem setup and PhysFTM decomposition

Sparse observations. For clarity, we describe the formulation for a single trajectory in two spatial dimensions and omit the trajectory index. Let $u(\mathbf{x}, t, c)$ denote the c-th physical channel at time t and spatial coordinate $\mathbf{x} = (x, y)$. The data are represented by $Y \in \mathbb{R}^{H\times W\times T\times C}$, where $H \times W$ is the spatial grid resolution, T is the number of frames, and C is the number of physical channels. Only a sparse subset of entries is available as observations. We denote the observation mask by $M \in \{0, 1\}^{H\times W\times T\times C}$ and the observed index set by

$$\Omega = \{(h, w, t, c) : M_{h,w,t,c} = 1\}.$$

All trajectories share a common nominal time grid. Sensor locations are fixed over time within each trajectory, and other spatial locations are referred to as *off-sensor locations*. Frames with sensor observations are referred to as *support frames*, while frames without sensor observations are

referred to as *missing frames*. We aim to reconstruct the full field $\hat{u}(\mathbf{x}, t, c)$ on both support and missing frames from these sparse observations.

Spatial–temporal factorization. PhysFTM combines a spatially continuous Functional Tucker decoder with a time-dependent Tucker core:

$$\hat{u}(\mathbf{x}, t, c) = \mathcal{D}_\Phi\big(G(t)\big)(\mathbf{x}, c), \tag{1}$$

where $\mathcal{D}_\Phi$ is a spatial decoder defined by RKHS-parameterized Functional Tucker factors, and $G(t)$ is the time-dependent Tucker core. Sparse sensor observations in support frames are used to infer the corresponding cores $G(t_j)$. A continuous-time model then estimates the cores of missing frames from these inferred support-frame cores, and $\mathcal{D}_\Phi$ decodes them into reconstructed fields. The spatial decoder provides a continuous spatial representation and can be regularized by PDE residuals, while temporal modeling operates in the compact Tucker-core space.

3.2 Physics-informed RKHS Functional Tucker decoder

RKHS mode factors and Tucker decoding. For each active spatial mode $d \in \{x, y\}$, PhysFTM defines R_d factor functions $\{\phi_{d,r}\}_{r=1}^{R_d}$. Each factor belongs to a reproducing kernel Hilbert space induced by a one-dimensional positive definite kernel k_d and is represented by a finite kernel expansion

$$\phi_{d,r}(\xi) = \sum_{m=1}^{M_d} a_{d,m,r}\, k_d(\xi, z_{d,m}), \tag{2}$$

where $\{z_{d,m}\}_{m=1}^{M_d}$ are inducing points and $A_d = [a_{d,m,r}] \in \mathbb{R}^{M_d \times R_d}$ are learnable RKHS coefficients. By using Eq. (2), we provide the following spatial decoders via Tucker decomposition. For a scalar field at a support frame t, the spatial decoder is

$$\hat{u}(t, x, y) = \sum_{p=1}^{R_x} \sum_{q=1}^{R_y} G_{p,q,t}\, \phi_{x,p}(x)\phi_{y,q}(y), \tag{3}$$

where G is a core tensor. For multichannel fields, the spatial decoder is modeled by

$$\hat{u}(t, x, y, c) = \sum_{p=1}^{R_x} \sum_{q=1}^{R_y} G_{p,q,t,c}\, \phi_{x,p}(x)\phi_{y,q}(y). \tag{4}$$

Note that factors $\phi_{x,p}(x)$ and $\phi_{y,q}(y)$ are shared between channels, and the channel index is stored in the core tensor. The decoder can be evaluated at observed sensors, dense grid points, and off-grid PDE collocation points.

RKHS and physics-informed loss functions and alternating-minimization strategy. PhysFTM exploits the block structure of Eq. (1).In general, we propose an alternating-minimization strategy to optimize the factors and core tensor. Specifically, with fixed RKHS factors, support-time Tucker cores can be inferred by solving a least-squares problem. For RKHS factors, since the factors are into 143 the PDE residuals, they are learned by gradient descent under observation, RKHS, and PDE losses.

For a mini-batch and fixed factors, the support-time core sequence is obtained by

$$\{G_t^\star\}_{t=1}^{T_s} = \arg\min_{\{G_t\}} \sum_{t=1}^{T_s} \big\| M_t \odot (\Phi\, \mathrm{vec}(G_t) - Y_t) \big\|_2^2 + \lambda_{\mathrm{ridge}} \sum_{t=1}^{T_s} \|G_t\|_F^2 + \beta_{\mathrm{TV}} \sum_{t=2}^{T_s} \|G_t - G_{t-1}\|_F^2. \tag{5}$$

Here T_s denotes the number of support frames, $\Phi(\mathbf{x}) = \bigotimes_{d=1}^{d_s} \phi_d(x_d)$ is the tensor-product spatial feature, and M_t is the sparse observation mask at time t. The subproblem is solved by conjugate gradients using implicit matrix-vector products. Note that the matrix $\Phi(\mathbf{x})$ is never explicitly computed.

After solving the cores, the RKHS basis coefficients are updated by AdamW by minimizing

$$\mathcal{L}_{\mathrm{basis}} = \mathcal{L}_{\mathrm{obs}} + \lambda_{\mathrm{RKHS}}\mathcal{L}_{\mathrm{RKHS}} + \lambda_{\mathrm{PDE}}\mathcal{L}_{\mathrm{PDE}} + \lambda_t \mathcal{L}_{\mathrm{time}}, \tag{6}$$

with the loss terms defined below. The observation loss is

$$\mathcal{L}_{\text{obs}} = \frac{1}{|\Omega|} \sum_{(t,h,w,c)\in\Omega} \left(\hat{Y}_{t,h,w,c} - Y_{t,h,w,c}\right)^2 \tag{7}$$

This is the RMSE over observed scalar entries of the channelwise normalized fields. For fixed factors, the core update uses the squared-error subproblem in Eq. (5). Note that in practice, for a mini-batch of N_b trajectories, let Ω_b denote the observed index set of trajectory b, we evaluate the loss on each Ω_b separately and compute the weighted average by Eq. (5).

The RKHS regularization term controls the complexity of the spatial factor functions:

$$\mathcal{L}_{\text{RKHS}} = \sum_{d\in\{x,y\}} \text{tr}(A_d^\top K_d A_d), \tag{8}$$

where K_d is the kernel Gram matrix over inducing points of mode $d \in \{x, y\}$. Given a mini-batch of trajectories and spatial collocation points $\mathcal{Q}$, the decoder evaluates $\hat{u}(\mathbf{x}, t, c)$ for each $\mathbf{x} \in \mathcal{Q}$. For a dataset-specific finite-difference residual operator $\mathcal{R}_h$, the PDE loss is

$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_b|\mathcal{Q}|T'} \sum_{b=1}^{N_b} \sum_{\mathbf{x}\in\mathcal{Q}} \sum_{t\in\mathcal{T}'} \left\|\mathcal{R}_h[\hat{u}_b](\mathbf{x}, t)\right\|_2^2, \tag{9}$$

where $\mathcal{T}'$ contains the timestamps of support frames with valid finite-difference stencils, and $T' = |\mathcal{T}'|$. The derivative stencils and valid spatial and temporal points depend on the dataset; see Appendix B.

The time-kernel term regularizes the inferred support-frame core sequence:

$$\mathcal{L}_{\text{time}} = \frac{1}{D_G} \sum_{b=1}^{N_b} \text{tr}\left((G_b^\flat)^\top K_t^{-1} G_b^\flat\right), \tag{10}$$

where $G_b^\flat \in \mathbb{R}^{T_s\times D_G}$ is the flattened core sequence of trajectory b, T_s is the number of support frames, $D_G = CR_xR_y$ is the flattened-core dimension, and K_t is an RBF kernel matrix over the timestamps of support frames. For three-dimensional fields, the RKHS regularization also includes the z-mode factors, and $D_G = CR_xR_yR_z$.

RKHS-Tucker reconstruction analysis. We provide the reconstruction error analysis of the RKHS-Tucker formulation in Appendix C.

Theorem 1 (Fixed-rank sparse reconstruction)**.** *Assume that 1) the observation points are i.i.d. samples from ρ; 2) the noise is conditionally sub-Gaussian; 3) the tensor RKHS integral operator admits a product spectral decomposition; 4) the target field satisfies the source condition $u = T^v g$ with $v > 1/2$; 5) and the mode-wise eigenvalues satisfy $\lambda_{j,m} \le K_0 m^{-\beta}$ with $\beta(2v-1) > 1$. Then, for a rank-r RKHS-Tucker estimator $\hat{u}_r$, with probability at least $1-\delta$,*

$$\|\hat{u}_r - u\|_{L^2(\rho)} \lesssim C_A r^{-s} + C_S\sqrt{\frac{r^d + \log(1/\delta)}{n}}, \qquad s = \frac{\beta(2v-1)-1}{2}. \tag{11}$$

Note that the first term in Eq. (11) is the truncation error induced by the finite Tucker rank, and the second term comes from the statistical error induced by sparse sampling.

Corollary 1 (Sample complexity)**.** *Let the truncation rank be $r_\star \asymp n^{1/(2s+d)}$, which balances approximation and statistical errors. Then, under the conditions of Theorem 1, if the sample size satisfies*

$$n \gtrsim \left(\frac{C}{\epsilon}\right)^{\frac{2(\beta(2v-1)-1+d)}{\beta(2v-1)-1}} + \frac{C^2\log(1/\delta)}{\epsilon^2}, \tag{12}$$

the L^2-norm $\|\hat{u}_{r_\star} - u\|_{L^2(\rho)}$ is smaller than ϵ with probability at least $1-\delta$.

3.3 Continuous-time core trajectory and missing frames reconstruction

After learning the spatial RKHS-FTM decoder, PhysFTM freezes the RKHS factor functions and represents temporal variation only through the time-dependent Tucker core. The Tucker core provides a compact coefficient representation for modeling temporal dynamics under the spatial decoder. We derive the temporal model from support-core estimates inferred with the RKHS decoder learned during the physics-constrained spatial stage.

Support-core inference with a mean-core reference. For a new trajectory, the learned RKHS factors are frozen and reused as a deterministic spatial decoder. The remaining task is to infersupport-frame cores from sparse sensor measurements. Under severe sparsity, this inverse problem is highly underdetermined: the sensing operator observes only a small subset of entries, leaving a large null space in core space. If the core is solved directly with a ridge penalty, the solution can be biased toward a minimum-norm core; in practice, especially with finite-iteration CG and extremely sparse masks, this often leads to near-zero cores and low-energy reconstructions.

Therefore, we introduce a residual correction:

$$G_t = G_t^{\mathrm{ref}} + \Delta G_t \tag{13}$$

on the data-driven mean-core reference G_t^{ref} computed from the training set,

$$G_t^{\mathrm{ref}} = \frac{1}{N_{\mathrm{train}}} \sum_{i=1}^{N_{\mathrm{train}}} G_{i,t}^{\mathrm{train}}. \tag{14}$$

Instead of regularizing the absolute core, we penalize only the correction. The core-correction subproblem is defined on the complete native time grid:

$$\begin{aligned} \Delta G^\star = \arg\min_{\{\Delta G_t\}_{t=1}^T} \quad & \sum_{t=1}^{T} \left\| M_t \odot \left(\Phi \,\mathrm{vec}(G_t^{\mathrm{ref}} + \Delta G_t) - Y_t \right) \right\|_2^2 \\ & + \lambda_{\mathrm{supp}} \sum_{t=1}^{T} \|\Delta G_t\|_F^2 \\ & + \beta_{\mathrm{supp}} \sum_{t=2}^{T} \|\Delta G_t - \Delta G_{t-1}\|_F^2. \end{aligned} \tag{15}$$

Here T is the number of native timestamps. The observation mask is zero at missing frames, while the last summand couples corrections at adjacent native timestamps. The inferred support-time cores are

$$\bar{G}_t = G_t^{\mathrm{ref}} + \Delta G_t^\star. \tag{16}$$

This residual formulation keeps the solution close to a plausible training-set core trajectory when observations are insufficient, while still allowing sparse measurements to correct the trajectory through ΔG.

Temporal core modeling. Let $\{(t_j, \bar{G}(t_j))\}_{j=1}^{T_s}$ denote the inferred support-frame cores and their timestamps. A continuous-time model H_ψ estimates the cores of missing frames:

$$\hat{G}(\tau) = H_\psi\big(\{(t_j, \bar{G}(t_j))\}_{j=1}^{T_s}, \tau\big), \qquad \tau \in \mathcal{T}_{\mathrm{miss}}, \tag{17}$$

where $\mathcal{T}_{\mathrm{miss}}$ contains the timestamps of missing frames within $[t_1, t_{T_s}]$.

We implement H_ψ as a learned residual correction to cubic Hermite interpolation in core space. Let $B(\tau)$ denote the Hermite interpolant through the support-frame cores. Between two adjacent support-frame timestamps t_a and t_b, the estimated core is

$$\hat{G}(\tau) = B(\tau) + \gamma(\tau)\Delta_\psi(\tau), \qquad \gamma(\tau) = \sin^2\left(\pi \frac{\tau - t_a}{t_b - t_a}\right), \tag{18}$$

where $\Delta_\psi(\tau)$ is generated by a lightweight ODE-RNN in flattened core space. The gate is zero at both endpoints, preserving the inferred support-frame cores.

Missing-frame decoding. The reconstructed field at spatial location $\mathbf{x}$ and missing-frame timestamp τ is obtained through the learned spatial decoder:

$$\hat{u}(\mathbf{x}, \tau, c) = \mathcal{D}_\Phi\big(\hat{G}(\tau)\big)(\mathbf{x}, c). \tag{19}$$

4 Experiments

4.1 Experimental setup

Datasets. We use **Allen–Cahn (AC2D)**, a scalar reactiondiffusion benchmark with 200 trajectories of 201 frames on a 256×256 grid, split into 160/20/20 train/validation/test trajectories; **Navier–Stokes (NS2D)**, an incompressible flow benchmark with (u, v, ω) channels and 800 trajectories of 201 frames on a 128×128 grid, split into 640/80/80; **Heat3D**, a scalar three-dimensional heat-diffusion benchmark with 200 trajectories of 50 frames on a 64^3 grid, split into 160/20/20 train/validation/test trajectories; and **Ocean3D**, a three-dimensional ocean-velocity reanalysis dataset derived from the Copernicus Marine GLORYS12V1 product [11], with (u, v) channels and 100 sequences of 90 frames on a $32 \times 64 \times 64$ grid, split chronologically into 80/10/10 train/validation/test sequences. Dataset details are in Appendix D.

Observation and missing-frame protocols. During training, 15% of spatial locations are sampled to simulate sparse grid-based sensors. The sensor mask is fixed over time within each trajectory and shared by physical channels, while different trajectories use different masks. We refer to the sensor positions of each trajectory as sensor locations and all remaining spatial locations as off-sensor locations. We report spatial sensor ratios $s \in \{0.01, 0.03, 0.15\}$. We use two missing-frame protocols: random-drop-half independently retains internal frames as support frames with probability 0.5 and always retains both endpoints, whereas temporal-interval-missing withholds sensor observations from a contiguous internal block of 20% of the frames. Frame masks are trajectory specific; full protocol and implementation details are in Appendix D.2.

Compared Methods. We compare PhysFTM with SDIFT [6], LRTFR [24], SIREN-MLP [32], PINNs [26], and RecFNO [50]. Their conditioning, training, and test-time protocols are summarized in Appendix 5.

Metrics. We report root mean squared error (RMSE) at sensor locations in support frames (S-sup), off-sensor locations in support frames (Off-sup), sensor locations in missing frames (S-miss), and off-sensor locations in missing frames (Off-miss). The four disjoint subsets and aggregation rules are defined in Appendix D.1. All metrics use inverse-normalized physical variables.

4.2 Reconstruction results

Overall reconstruction under missing-frame protocols. We first compare reconstruction at $s = 0.03$ across the four datasets and both missing-frame protocols. Table 1 reports metrics separately for support frames and missing frames, each evaluated at sensor locations and off-sensor locations.

PhysFTM obtains the lowest RMSE in most reported dataset–protocol–subset comparisons. The gains are especially clear at off-sensor locations in support frames on AC2D, NS2D, and Heat3D. These results support the effectiveness of the proposed representation for fixed-sensor reconstruction with missing frames across both 2D and 3D grids.

Complementary physical diagnostics. Table 2 reports NS2D missing-frame accuracy together with divergence, curl-derived vorticity, and kinetic-energy-spectrum errors. The diagnostics are computed on inverse-normalized full grids of missing frames; their formulas and cross-frame reducers are given in Appendix B.1.

PhysFTM has the lowest Off-miss, vorticity, and spectrum errors under both protocols, while PINNs has the lowest divergence residual. The result indicates a favorable accuracy–consistency trade-off. Complete three-ratio diagnostics are reported in Appendix B.1.

4.3 Model analysis

The effects of individual components and block-update strategies are evaluated through ablations. Table 3 compares RKHS regularization, PDE regularization, and core/factor block-update strategies with their corresponding effective objectives, without temporal missingness.

Table 1: Reconstruction RMSE on four datasets at spatial sensor ratio $s = 0.03$. The two column groups report the four disjoint evaluation subsets, defined by sensor/off-sensor locations and support/missing frames, under random-drop-half and temporal-interval-missing. Lower is better. For Heat3D and Ocean3D, the PhysFTM interval-missing results use the random-drop-half-trained temporal checkpoint. (**Bold**: best, Underline: second best)

Method	Random-drop-half				Temporal-interval-missing			
	S-sup	Off-sup	S-miss	Off-miss	S-sup	Off-sup	S-miss	Off-miss
				AC2D				
SDIFT	0.35804	0.38080	0.38030	0.38151	0.35699	0.36199	0.39071	0.39259
LRTFR	0.31170	0.32220	0.31340	0.32038	0.31006	0.31676	0.32059	0.32781
PhysFTM	**0.01735**	**0.08710**	0.05035	**0.09988**	**0.01633**	**0.09397**	0.03953	**0.07924**
RecFNO	0.01848	0.45283	**0.02178**	0.45523	0.01811	0.45875	**0.02116**	0.43680
SIREN-MLP	0.34868	0.34932	0.35013	0.35076	0.34605	0.34664	0.35819	0.35898
PINNs	0.35166	0.35243	0.35268	0.35331	0.34874	0.34943	0.36508	0.36622
				NS2D				
SDIFT	0.52867	0.50970	0.51003	0.51027	0.53019	0.53052	0.50434	0.50275
LRTFR	0.13245	0.14720	0.13447	0.14576	0.13515	0.14682	0.19025	0.19661
PhysFTM	**0.00669**	**0.06210**	**0.03240**	**0.06498**	**0.00435**	**0.06282**	**0.14949**	**0.15056**
RecFNO	0.06810	0.09757	0.07820	0.10441	0.06906	0.09910	0.35803	0.35943
SIREN-MLP	0.24555	0.24526	0.24465	0.24441	0.24797	0.24777	0.21369	0.21354
PINNs	0.34254	0.34177	0.34101	0.34055	0.34722	0.34657	0.31892	0.31848
				Heat3D				
SDIFT	0.05811	0.05808	0.06303	0.06272	0.06284	0.06268	0.05016	0.04989
LRTFR	0.01212	0.01222	0.01309	0.01315	0.01210	0.01225	0.00577	0.00577
PhysFTM	**0.00108**	**0.00125**	**0.00575**	**0.00574**	**0.00122**	**0.00137**	**0.00310**	**0.00305**
RecFNO	0.28736	0.77847	0.26372	0.72034	0.28629	0.77526	0.22597	0.64187
SIREN-MLP	0.04080	0.04087	0.04211	0.04205	0.04342	0.04344	0.02749	0.02742
PINNs	0.04000	0.04011	0.04271	0.04264	0.04314	0.04319	0.03353	0.03334
				Ocean3D				
SDIFT	0.27863	0.28065	0.27585	0.27838	0.27494	0.27725	0.28712	0.28941
LRTFR	0.11038	0.11450	0.11767	0.12155	0.10506	0.10829	**0.18062**	**0.18377**
PhysFTM	**0.00632**	**0.06574**	**0.07133**	**0.08858**	**0.00595**	**0.06382**	0.18985	0.18565
RecFNO	0.73827	0.97074	0.77688	0.96302	0.71841	0.94379	0.99191	1.05383
SIREN-MLP	0.20158	0.20168	0.20024	0.20094	0.19773	0.19833	0.21199	0.21151
PINNs	0.30163	0.30100	0.30232	0.30194	0.29765	0.29739	0.31870	0.31725

Table 2: NS2D missing-frame reconstruction and complementary physical diagnostics at $s = 0.03$. Divergence, vorticity, and spectrum errors are evaluated on full grids of missing frames; Off-miss is evaluated at off-sensor locations in missing frames. Lower is better.

Protocol	Method	Off-miss	Divergence	Vort. RMSE	Spectrum
Random	SDIFT	0.510274	0.352407	0.916343	0.877309
	LRTFR	0.145757	0.459044	0.237923	0.049954
	PhysFTM	**0.064976**	0.134170	**0.109304**	**0.033599**
	RecFNO	0.104413	0.459014	0.118608	0.067430
	SIREN-MLP	0.244405	1.887604	0.623477	0.408029
	PINNs	0.340554	**0.120243**	0.586433	0.398297
Interval	SDIFT	0.502748	0.344546	0.931635	0.892291
	LRTFR	0.196613	0.616958	0.272195	0.055034
	PhysFTM	**0.150558**	0.129873	**0.150436**	**0.036346**
	RecFNO	0.359434	0.342736	0.211528	0.055423
	SIREN-MLP	0.213537	1.863564	0.546894	0.368662
	PINNs	0.318479	**0.120243**	0.586433	0.349142

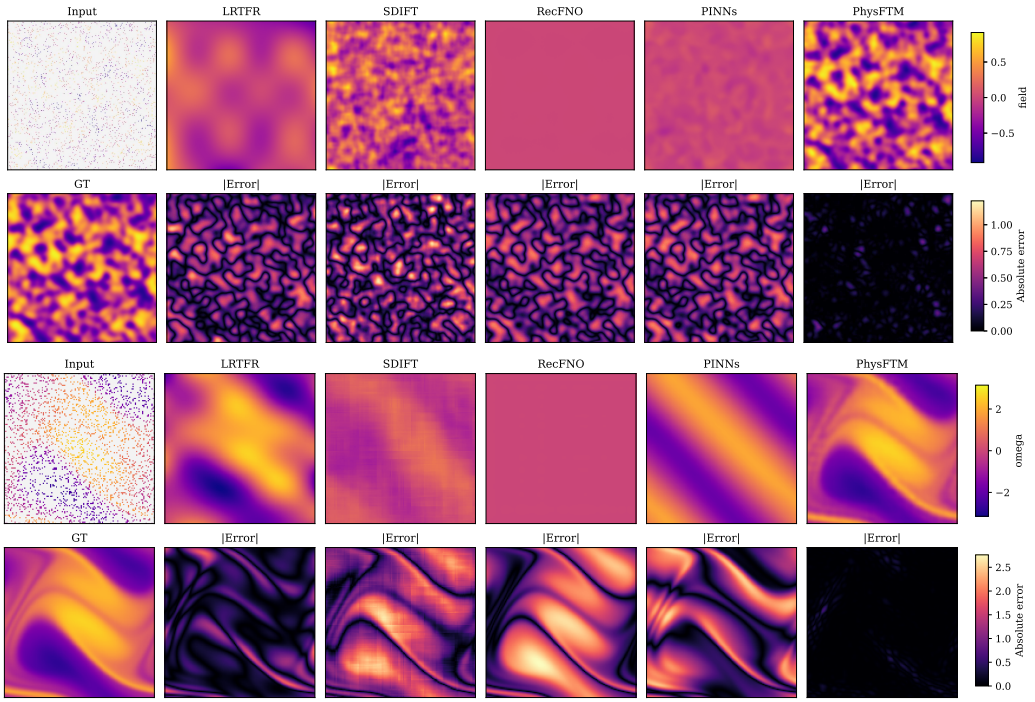

Figure 2: Qualitative comparisons on AC2D (top) at $s = 0.03$ and the NS2D vorticity channel (bottom) at $s = 0.15$. The upper row of each block shows the sparse input and predictions from LRTFR, SDIFT, RecFNO, PINNs, and PhysFTM; the lower row shows the ground truth and the corresponding absolute-error maps.

Table 3: Component and block-update-strategy ablation on AC2D and NS2D at $s = 0.03$, without temporal missingness. RMSEs use all evaluated frames; PDE is evaluated at disjoint held-out off-sensor points and centered-stencil-valid internal times.

Data	Variant	PDE	Full	Sensor	Off-sensor
AC2D	PhysFTM	0.053250	**0.088078**	0.015185	**0.089395**
	w/o physics	0.064986	0.091135	0.010852	0.092520
	w/o RKHS reg.	0.048491	0.096622	0.027551	0.097985
	solve-only	1.155543	0.171035	**0.008657**	0.173653
	gradient-only	1.76×10^{-5}	0.352111	0.351441	0.352132
NS2D	PhysFTM	0.662575	**0.060270**	0.004189	**0.061196**
	w/o physics	1.045838	0.085478	0.009674	0.086781
	w/o RKHS reg.	1.703260	0.113713	0.016850	0.115421
	solve-only	1.625978	0.105585	**0.000653**	0.107207
	gradient-only	**0.450487**	0.090639	0.073760	0.091112

Table 4: RMSE at non-grid-aligned times within the continuous support-frame time range, using ground truth directly recorded by the PDE solver. "Learned" is the gated ODE-RNN correction to cubic Hermite interpolation in Tucker-core space.

Dataset	s	Learned	Linear	Cubic
AC2D	0.01	**0.180029**	0.189510	0.189556
AC2D	0.03	**0.085552**	0.090391	0.090388
NS2D	0.01	**0.130469**	0.137667	0.137453
NS2D	0.03	**0.055546**	0.064605	0.063790

The complete model gives the lowest Full and Off-sensor RMSE on both datasets. Although solve-only achieves the lowest Sensor RMSE, its substantially higher Off-sensor RMSE indicates that accurate fitting at sensor locations does not necessarily translate into accurate reconstruction elsewhere. The gradient-only variant obtains a low PDE residual on AC2D together with a high reconstruction RMSE. These comparisons assess the combined choices of block updates and their effective objectives. Overall, PhysFTM offers the most favorable trade-off among all evaluated variants between sensor fitting, off-sensor reconstruction, and physical consistency. The block updates and active objective terms are specified in Appendix F. Additional spatial and collocation ablations are in Appendix A.

Reconstruction at non-grid-aligned times. We evaluate the learned core trajectory at 100 non-grid-aligned timestamps within the interval spanned by the timestamps of support frames. These evaluation timestamps do not coincide with the original frame timestamps and are therefore distinct from missing frames. For both AC2D and NS2D, the ground truth is recorded directly by rerunning the PDE solver at the perturbed physical timestamps rather than interpolating stored frames. The timestamp and split rules are given in Appendix D.3.

The learned trajectory reduces reconstruction RMSE at non-grid-aligned times relative to both interpolation baselines in all four settings. This experiment supports continuous temporal reconstruction within the interval spanned by the timestamps of support frames on AC2D and NS2D.

5 Conclusion

We introduce PhysFTM, a functional Tucker method for reconstructing continuous spatiotemporal dynamics from sparse sensors and missing frames. By parameterizing Tucker factors by RKHS and enforcing governing equations via finite-difference residuals, the proposed framework is able to improve the physical consistency and reconstruction accuracy. Experiments on Allen–Cahn, Navier–Stokes, Heat equations, and a gridded Copernicus ocean reanalysis demonstrate accurate and physically consistent reconstruction across 2D and 3D grids.

The main limitations are as follows. First, the current framework considers sparse observations collected from fixed spatial sensors, with missing temporal frames. Extending it to more general observation settings, where measurements are available at arbitrary space-time locations, remains an interesting direction. Second, PhysFTM relies on known governing equations to regularize the reconstruction, which may limit its applicability when the underlying physical models are inaccurate or incomplete. Third, sparse observations generally do not uniquely determine the underlying physical field. Current PhysFTM returns a single regularized field without quantifying the associated reconstruction uncertainty. Therefore, we may extend the method to observations at arbitrary space-time coordinates and incorporate uncertainty quantification in the future to characterize reconstruction ambiguity and provide reliable estimates for downstream analysis.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 42450181) and by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Grant No. AoE/P-601/23-N).

References

[1] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

[2] Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

[3] Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005. doi: 10.1214/009053605000000282.

[4] Alain Berlinet and Christine Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers, Boston, MA, 2004. doi: 10.1007/978-1-4419-9096-9.

[5] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007. doi: 10.1007/s10208-006-0196-8.

[6] Panqi Chen, Yifan Sun, Lei Cheng, Yang Yang, Weichang Li, Yang Liu, Weiqing Liu, Jiang Bian, and Shikai Fang. Generating full-field evolution of physical dynamics from irregular sparse observations. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2505.09284.

[7] Felipe Cucker and Ding-Xuan Zhou. *Learning Theory: An Approximation Theory Viewpoint*, volume 24 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2007. ISBN 978-0-521-86559-3.

[8] Sergey Dolgov, Daniel Kressner, and Christoph Strössner. Functional tucker approximation using chebyshev interpolation. *SIAM Journal on Scientific Computing*, 43(3):A2190–A2210, 2021. doi: 10.1137/20M1356944.

[9] Pan Du, Meet Hemant Parikh, Xiantao Fan, Xin-Yang Liu, and Jian-Xun Wang. Conditional neural field latent diffusion model for generating spatiotemporal turbulence. *Nature Communications*, 15(10416), 2024. doi: 10.1038/s41467-024-54712-1.

[10] N. Benjamin Erichson, Lionel Mathelin, Zhewei Yao, Steven L. Brunton, Michael W. Mahoney, and J. Nathan Kutz. Shallow neural networks for fluid flow reconstruction with limited sensors. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 476 (2238):20200097, 2020. doi: 10.1098/rspa.2020.0097.

[11] E.U. Copernicus Marine Service Information (CMEMS). Global Ocean Physics Reanalysis. Marine Data Store. URL `https://doi.org/10.48670/moi-00021`. Product ID: `GLOBAL_MULTIYEAR_PHY_001_030`.

[12] Shikai Fang, Xin Yu, Zheng Wang, Shibo Li, Robert M. Kirby, and Shandian Zhe. Functional Bayesian Tucker decomposition for continuous-indexed tensor data. In *International Conference on Learning Representations*, 2024.

[13] Kai Fukami, Romit Maulik, Nesar Ramachandra, Koji Fukagata, and Kunihiko Taira. Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning. *Nature Machine Intelligence*, 3(11):945–951, 2021. doi: 10.1038/s42256-021-00402-2.

[14] Bin Gao, Renfeng Peng, and Ya-xiang Yuan. Riemannian preconditioned algorithms for tensor completion via tensor ring decomposition. *Computational Optimization and Applications*, 88 (2):443–468, 2024.

[15] Bin Gao, Renfeng Peng, and Ya-xiang Yuan. Low-rank optimization on Tucker tensor varieties. *Mathematical Programming*, 214:357–407, 2025. doi: 10.1007/s10107-024-02186-w.

[16] Mars Liyao Gao, Jan P. Williams, and J. Nathan Kutz. Sparse identification of nonlinear dynamics and koopman operators with shallow recurrent decoder networks. *Proceedings of the National Academy of Sciences*, 2026. doi: 10.1073/pnas.2508144123. Also available as arXiv:2501.13329.

[17] Tao Han, Zhenghao Chen, Song Guo, Wanghan Xu, Wanli Ouyang, and Lei Bai. Climate science data can be compressed efficiently by dual-stage extreme compression with a variational auto-encoder transformer. *Communications Earth & Environment*, 6(1):955, 2025.

[18] Tao Han, Zhibin Wen, Zhenghao Chen, Dazhao Du, Song Guo, and Lei Bai. Benchmarking physics-informed time-series models for operational global station weather forecasting, 2026. URL `https://arxiv.org/abs/2406.14399`.

[19] Tao Han, Zhibin Wen, Zhenghao Chen, Fenghua Lin, Junyu Gao, Song Guo, and Lei Bai. Generative 3d gaussian splatting for arbitrary-resolution atmospheric downscaling and forecasting. *arXiv preprint arXiv:2604.07928*, 2026.

[20] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[21] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009. doi: 10.1137/07070111X.

[22] Juan Li, Yajie Bai, Xuerong Cui, Lei Li, Bin Jiang, Shibao Li, and Jungang Yang. Oceanic 3-d thermohaline field reconstruction with multidimensional features using sabnn. *IEEE Journal of Oceanic Engineering*, 50(2):1273–1289, 2025. doi: 10.1109/JOE.2025.3535591.

[23] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024. doi: 10.1145/3648506.

[24] Yisi Luo, Xile Zhao, Zhemin Li, Michael K. Ng, and Deyu Meng. Low-rank tensor function representation for multi-dimensional data recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3351–3369, 2024. doi: 10.1109/TPAMI.2023.3341688.

[25] Krithika Manohar, Bingni W. Brunton, J. Nathan Kutz, and Steven L. Brunton. Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns. *IEEE Control Systems Magazine*, 38(3):63–86, 2018. doi: 10.1109/MCS.2018.2810460.

[26] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

[27] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[28] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2006. ISBN 978-0-262-18253-9. doi: 10.7551/mitpress/3206.001.0001.

[29] Pu Ren, Chengping Rao, Yang Liu, Jian-Xun Wang, and Hao Sun. Phycrnet: Physics-informed convolutional-recurrent network for solving spatiotemporal pdes. *Computer Methods in Applied Mechanics and Engineering*, 389:114399, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2021.114399. URL `https://www.sciencedirect.com/science/article/pii/S0045782521006514`.

[30] Javier E. Santos, Zachary R. Fox, Arvind T. Mohan, Daniel O'Malley, Hari Viswanathan, and Nicholas Lubbers. Development of the Senseiver for efficient field reconstruction from sparse observations. *Nature Machine Intelligence*, 5(11):1317–1325, 2023. doi: 10.1038/s42256-023-00746-x.

[31] Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, 2023.

[32] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, 2020.

[33] Noah Steidle, Joppe De Jonghe, and Mariya Ishteva. Adaptive subspace modeling with functional Tucker decomposition. *arXiv preprint arXiv:2603.25530*, 2026.

[34] Michael L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Series in Statistics. Springer, New York, NY, 1999. ISBN 978-1-4612-7166-6. doi: 10.1007/978-1-4612-1494-6.

[35] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Information Science and Statistics. Springer, New York, NY, 2008. ISBN 978-0-387-77241-7. doi: 10.1007/978-0-387-77242-4.

[36] Ingo Steinwart, Don Hush, and Clint Scovel. Optimal rates for regularized least squares regression. In *Proceedings of the 22nd Annual Conference on Learning Theory*, COLT 2009, pages 79–93, 2009.

[37] Yanjie Tong and Peng Chen. From sparse sensors to continuous fields: STRIDE for spatiotemporal reconstruction. *arXiv preprint arXiv:2602.04201*, 2026.

[38] Ledyard R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966. doi: 10.1007/BF02289464.

[39] Grace Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, PA, 1990. doi: 10.1137/1.9781611970128.

[40] Zheng Wang and Shandian Zhe. Nonparametric factor trajectory learning for dynamic tensor decomposition. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23459–23469, 2022.

[41] Zheng Wang, Shikai Fang, Shibo Li, and Shandian Zhe. Dynamic tensor decomposition via neural diffusion-reaction processes. In *Advances in Neural Information Processing Systems*, 2023.

[42] Jan P. Williams, Olivia Zahn, and J. Nathan Kutz. Sensing with shallow recurrent decoder networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 480(2298):20240054, 2024. doi: 10.1098/rspa.2024.0054.

[43] Shengfeng Xu, Zhenxu Sun, Renfang Huang, Dilong Guo, Guowei Yang, and Shengjun Ju. A practical approach to flow field reconstruction with sparse or incomplete data through physics informed neural network. *Acta Mechanica Sinica*, 39(3):322302, 2023.

[44] Shengfeng Xu, Chang Yan, Guangtao Zhang, Zhenxu Sun, Renfang Huang, Shengjun Ju, Dilong Guo, and Guowei Yang. Spatiotemporal parallel physics-informed neural networks: A framework to solve inverse problems in fluid mechanics. *Physics of Fluids*, 35(6):065141, 06 2023. ISSN 1070-6631. doi: 10.1063/5.0155087. URL `https://doi.org/10.1063/5.0155087`.

[45] Shengfeng Xu, Chang Yan, Guangtao Zhang, Zhenxu Sun, Renfang Huang, Shengjun Ju, Dilong Guo, and Guowei Yang. Spatiotemporal parallel physics-informed neural networks: A framework to solve inverse problems in fluid mechanics. *Physics of Fluids*, 35(6), 2023.

[46] Huiyu Yang, Jianghang Gu, Yuntian Chen, Yuanwei Bin, Jianchun Wang, and Shiyi Chen. Spatially-aware transformer operator for real-time aerodynamic evaluations of arbitrary three-dimensional vehicles. *Journal of Computational Physics*, page 114456, 2025.

[47] Guangtao Zhang, Yiting Duan, Guanyu Pan, Qijing Chen, Huiyu Yang, and Zhikun Zhang. Data-driven discovery of state-changes in underlying system from hidden change-points in partial differential equations with spatiotemporal varying coefficients. *Journal of Computational and Applied Mathematics*, 474:116962, 2026.

[48] Yi Zhang, Yiting Duan, Xiangjun Wang, and Zhikun Zhang. Solving fokker–planck-kolmogorov equation by distribution self-adaptation normalized physics-informed neural networks. *Physica A: Statistical Mechanics and its Applications*, page 131251, 2026.

[49] Zhikun Zhanga, Guanyu Panc, Xiangjun Wangd, Yong Xua, and Guangtao Zhange. Parameter identification for partial differential equation with jump discontinuities in coefficients by markov switching model and physics-informed machine learning. *arXiv preprint arXiv:2510.14656*, 2025.

[50] Xiaoyu Zhao, Xiaoqian Chen, Zhiqiang Gong, Weien Zhou, Wen Yao, and Yunyang Zhang. RecFNO: A resolution-invariant flow and heat field reconstruction method from sparse observations via fourier neural operator. *International Journal of Thermal Sciences*, 195:108619, 2024. doi: 10.1016/j.ijthermalsci.2023.108619.

[51] Fang Zhu, Keyue Sun, Guangtao Zhang, and Junxiang Yang. Diffuse-interface modeling and energy-stable numerical framework for the heat transfer-coupled two-phase fluids in contact with solids. *Journal of Computational Physics*, 524:113699, 2025.

Table 5: Task-level comparison protocols. "Native" denotes prescribed dataset timestamps; "coordinate" denotes direct coordinate evaluation. Baseline budgets and checkpoint selection follow their executed run configurations.

Method	Test input	Training / fitting	Supervision / equation use	Test optimization	Spatial scope	Time interface / selection
SDIFT	Sparse spatiotemporal observations	Cross-trajectory diffusion training	Dense training trajectories; no test missing-frame targets	Posterior sampling	2D/3D	Native-grid protocols; executed checkpoint
LRTFR	Sparse coordinate–value tuples	Per-trajectory representation fitting	Sensor measurements; no equation loss	Per-trajectory fit	2D/3D	Native-grid evaluation; fitted objective
SIREN-MLP	Coordinates; no test-mask conditioning	Train once on training split	Training fields; no equation loss	None	2D/3D	Coordinate output; executed checkpoint
PINNs	Coordinates; no test-mask conditioning	Train once on training split	Training fields and implemented equation residual	None	2D/3D	Coordinate output; executed checkpoint
RecFNO	Gridded sparse field and mask	Cross-trajectory operator training	Dense training fields; no equation loss	None	2D/3D	Prescribed-grid support/missing frames; executed checkpoint
PhysFTM	Sparse support-frame sensor values	Cross-trajectory decoder/head training	Observations, missing-frame targets, and equation loss where enabled	Support-core CG solve	2D/3D	Native-grid and AC/NS in-range non-grid-aligned reconstruction; spatial: training dense RMSE; temporal: validation missing-frame relative ℓ_2

A Comparison methods and spatial analysis

Compared methods. SDIFT [6] is a cross-trajectory diffusion model conditioned on sparse spatiotemporal observations. LRTFR [24] fits a functional tensor representation to the sensor coordinate–value measurements of each target trajectory. RecFNO [50] is trained across trajectories and maps gridded sparse-field/mask inputs to prescribed-grid outputs. SIREN-MLP [32] and PINNs [26] are coordinate-to-field controls trained once on the training split and evaluated directly at test coordinates without test-time adaptation; they share the same dense prediction across spatial ratios and temporal protocols, while only the evaluated subsets change. The earlier response used "SIREN-INR" for the same SIREN-MLP implementation.

PhysFTM trains the spatial stage for 500 epochs, where one epoch traverses the complete training loader, and selects the spatial checkpoint by the lowest dense-field training RMSE after inverse normalization, averaging evaluation mini-batch RMSEs. Its temporal head is trained for 20 epochs with AdamW and selects the checkpoint with the lowest validation relative ℓ_2 error over missing frames, using decoded fields at all spatial locations after inverse normalization. RecFNO is evaluated on prescribed-grid support frames and missing frames.

The Panel A comparisons isolate basis parameterization and core optimization, whereas Panel B isolates collocation placement. These diagnostics are separate from the component/optimization ablation in Table 3, whose PDE column uses a disjoint held-out off-sensor set and internal centered-stencil times.

B Finite-difference residuals and physical diagnostics

All derivatives are evaluated after inverse normalization. AC2D uses the Allen–Cahn residual

$$\mathcal{R}_{\mathrm{AC}} = D_t\hat{u} - \lambda_{\mathrm{AC}}\big(\epsilon^2(D_{xx} + D_{yy})\hat{u} - \hat{u}^3 + \hat{u}\big).$$

NS2D stores (u, v, ω) rather than pressure. Its held-out residual reuses the centered finite-difference vorticity-equation component of the training operator,

$$\mathcal{R}_{\mathrm{NS}} = D_t\hat{\omega} + \hat{u}D_x\hat{\omega} + \hat{v}D_y\hat{\omega} - \nu(D_{xx} + D_{yy})\hat{\omega} - f,$$

and excludes the first and last temporal points. Training collocation and held-out off-sensor evaluation points are disjoint. Heat3D uses $\mathcal{R}_{\mathrm{heat}} = D_t\hat{u} - 0.01(D_{xx} + D_{yy} + D_{zz})\hat{u}$ in the spatial stage

Table 6: Additional spatial analysis at $s = 0.03$. Panel A evaluates reconstruction at off-sensor locations in support frames. Panel B varies the equation-collocation placement; the complete model uses off-sensor collocation. (**Bold**: best, Underline: second best)

Panel A: off-sensor reconstruction in support frames

Method	AC2D			NS2D		
	RMSE	SSIM	PDE	RMSE	SSIM	PDE
Linear interp.	0.1275	0.6379	0.5877	0.0897	0.9322	1.1605
Cubic interp.	0.1051	0.7632	0.7174	0.0653	**0.9656**	0.8339
SIREN-MLP	0.2907	0.1747	1.1703	0.2155	0.5836	2.1245
MLP-FTM	0.1049	0.7515	0.2282	0.0859	0.9116	4.0426
MLP-FTM+CG	0.1026	0.7719	0.1995	0.0832	0.9276	1.2277
FTM-RKHS	0.0925	0.7893	0.1532	0.0868	0.9468	0.7371
PhysFTM	**0.0894**	**0.7938**	**0.1325**	**0.0612**	0.9636	**0.3500**

Panel B: equation-collocation placement

Placement	AC2D		NS2D	
	RMSE	PDE	RMSE	PDE
w/o equation	0.0925	0.1532	0.0868	0.7371
sensor locations	0.0898	0.1358	0.0622	0.3584
off-sensor locations	**0.0884**	**0.1316**	**0.0609**	**0.3338**
full domain	0.0897	0.1344	0.0618	0.3556

with weight 10^{-4}, while its temporal equation weight is zero. Ocean3D has no governing-equation loss in the reported configuration.

For NS training, the equation-based loss is the sum of three mean squared residuals,

$$\begin{aligned}\mathcal{L}_{\mathrm{PDE}}^{\mathrm{NS}} = \mathrm{mean}(\mathcal{R}_{\mathrm{NS}}^2) + \mathrm{mean}\left[(D_x\hat{u} + D_y\hat{v})^2\right] \\ + \mathrm{mean}\left[(\hat{\omega} - (D_x\hat{v} - D_y\hat{u}))^2\right],\end{aligned} \tag{20}$$

multiplied by the overall PDE-loss weight. The reported NS equation-residual metric is the mean squared vorticity-equation residual, $\mathrm{mean}(\mathcal{R}_{\mathrm{NS}}^2)$. The main prediction and ground-truth PDE exports use this same single-equation convention, while the unweighted sum of the three constraints is retained as a separate auxiliary export. The ground-truth export evaluates the numerical equation residual of the reference field.

B.1 Missing-frame physical diagnostics

The NS2D diagnostics are evaluated on inverse-normalized full grids of missing frames. Spatial derivatives use centered differences, and Fourier wavenumbers use the physical domain lengths. Divergence measures the predicted velocity field itself, $|\nabla \cdot \hat{\mathbf{u}}|$. Vorticity is derived independently from predicted and reference velocities and then compared:

$$D_{\mathrm{div}} = \left[\frac{1}{M}\sum_{n,i,\mathbf{x}} |\partial_x\hat{u} + \partial_y\hat{v}|^2\right]^{1/2},$$

$$D_{\mathrm{vort}} = \left[\frac{1}{M}\sum_{n,i,\mathbf{x}} |(\partial_x\hat{v} - \partial_y\hat{u}) - (\partial_x v - \partial_y u)|^2\right]^{1/2}.$$

Here the sums pool all test trajectories, their missing frames, and all full-grid points before taking the square root. For the spectrum diagnostic, the spatial means of (u, v) are removed, a radial shell-sum kinetic-energy spectrum $E_{n,i}(k)$ is formed, and the frame error is

$$e_{n,i}^{\mathrm{spec}} = \frac{\|\widehat{E}_{n,i} - E_{n,i}\|_2}{\|E_{n,i}\|_2}.$$

Frame errors are averaged within each trajectory, after which the evaluator computes the mean and standard deviation of the trajectory means. The tables report the mean component.

Table 7: Aggregation rules for missing-frame physical diagnostics.

Diagnostic	Missing-frame/spatial reduction	Trajectory reduction
Divergence error	Global RMS over all missing frames and grid points	Included in the same global pool
Vorticity RMSE	Global RMSE over all missing frames and grid points	Included in the same global pool
Spectrum error	Per-frame relative L_2, then missing-frame mean	Mean/std of trajectory means

Table 8: Complete NS2D missing-frame diagnostics across spatial sensor ratios. Lower is better.(**Bold**: best, Underline: second best)

Protocol	s	Method	Off-miss	Divergence	Vort. RMSE	Spectrum
Random	0.01	SDIFT	0.509261	0.307569	0.876079	0.865337
		LRTFR	0.170679	0.475408	0.237857	**0.048102**
		PhysFTM	**0.130659**	0.274083	**0.232358**	0.104305
		RecFNO	0.413223	0.514580	0.396548	0.161395
		SIREN-MLP	0.244360	1.886754	0.623300	0.409266
		PINNs	0.341072	**0.120243**	0.586433	0.402015
	0.03	SDIFT	0.510274	0.352407	0.916343	0.877309
		LRTFR	0.145757	0.459044	0.237923	0.049954
		PhysFTM	**0.064976**	0.134170	**0.109304**	**0.033599**
		RecFNO	0.104413	0.459014	0.118608	0.067430
		SIREN-MLP	0.244405	1.887604	0.623477	0.408029
		PINNs	0.340554	**0.120243**	0.586433	0.398297
	0.15	SDIFT	0.510589	0.782450	1.288744	0.882962
		LRTFR	0.138756	0.467318	0.236704	0.050423
		PhysFTM	**0.033357**	**0.029145**	**0.042455**	**0.029433**
		RecFNO	0.040051	0.460688	0.729675	0.041657
		SIREN-MLP	0.244476	1.887873	0.623739	0.407336
		PINNs	0.340193	0.120243	0.586433	0.399324
Interval	0.01	SDIFT	0.492242	0.304216	0.881166	0.870480
		LRTFR	0.206649	0.889423	0.311479	**0.069965**
		PhysFTM	**0.164821**	0.270320	**0.251761**	0.128132
		RecFNO	0.478397	0.374108	0.347071	0.178211
		SIREN-MLP	0.206503	1.858348	0.528179	0.356109
		PINNs	0.306602	**0.120243**	0.586433	0.346441
	0.03	SDIFT	0.502748	0.344546	0.931635	0.892291
		LRTFR	0.196613	0.616958	0.272195	0.055034
		PhysFTM	**0.150558**	0.129873	**0.150436**	**0.036346**
		RecFNO	0.359434	0.342736	0.211528	0.055423
		SIREN-MLP	0.213537	1.863564	0.546894	0.368662
		PINNs	0.318479	**0.120243**	0.586433	0.349142
	0.15	SDIFT	0.485171	0.816494	1.383403	0.925256
		LRTFR	0.172046	0.550371	0.254336	0.061462
		PhysFTM	**0.134567**	**0.026805**	**0.105232**	**0.010536**
		RecFNO	0.318191	0.321772	0.172333	0.061395
		SIREN-MLP	0.203190	1.853913	0.520432	0.374622
		PINNs	0.300163	0.120243	0.586433	0.373544

C Theoretical analysis on spatial reconstruction

In this section, we conduct an idealized spatial reconstruction error analysis for the RKHS-parameterized Functional Tucker Model (FTM-RKHS) under sparse observations. We aim to characterize how the modewise truncation level, kernel spectrum, and number of measurement samples jointly determine the reconstruction error. The i.i.d. spatial-sampling assumption is an analytical idealization and is not identical to the fixed-within-trajectory sensor protocol used in the experiments. Moreover, the spectral truncation level r in this analysis is not identified with the implemented Tucker rank.

Let $u : \mathcal{Z} \to \mathbb{R}$ denote the target field, where $\mathcal{Z}$ contains trajectory, time, spatial coordinates, and channel indices. The field is measured only on a sparse set $\Omega = \{z_i\}_{i=1}^n$, with $n = |\Omega|$, through measurements $y_i = u(z_i) + \varepsilon_i$, where $\mathbb{E}[\varepsilon_i \mid z_i] = 0$. The task is to recover the field u via the measurements y_i.

For the sake of theoretical analysis, we consider the following assumptions. For high-probability bounds, we assume that the noise is conditionally sub-Gaussian, i.e., $\mathbb{E}\left[\exp(t\varepsilon_i) \mid z_i\right] \leq$

$\exp\left(\frac{\sigma^2 t^2}{2}\right),\ t \in \mathbb{R}$. We assume that the observation distribution echoes the target distribution ρ over the full space $\mathcal{Z}$, i.e., the observations z_i are sampled i.i.d. from ρ.

From one-dimensional RKHS to tensor-product RKHS. We first recall the one-dimensional RKHS construction for each mode, following the standard spectral characterization of RKHSs induced by compact positive kernel integral operators [1, 4, 35]. Let $(\mathcal{Z}_j, \mathcal{B}_j, \rho_j)$, $j = 1, \dots, d$, be the probability space associated with the j-th mode, where $\mathcal{Z}_j$ is the input domain, $\mathcal{B}_j$ is the corresponding sigma-algebra, and ρ_j is the probability measure on $\mathcal{Z}_j$. Let $\mathcal{H}_j$ be an RKHS on $\mathcal{Z}_j$ with kernel k_j, the kernel integral operator is defined by $(T_j f)(z_j) = \int_{\mathcal{Z}_j} k_j(z_j, z_j') f(z_j')\, d\rho_j(z_j')$. Assume that each $T_j : L^2(\rho_j) \to L^2(\rho_j)$ is compact, self-adjoint, and positive semidefinite, so that the Mercer-type spectral decomposition applies [7, 35]. Then, there exist eigenpairs $T_j \phi_{j,m} = \lambda_{j,m} \phi_{j,m}$, with $\lambda_{j,1} \geq \lambda_{j,2} \geq \cdots > 0$, where $\{\phi_{j,m}\}_{m \geq 1}$ is orthonormal in $L^2(\rho_j)$.

For $f_j = \sum_{m \geq 1} a_{j,m} \phi_{j,m}$ and $g_j = \sum_{m \geq 1} b_{j,m} \phi_{j,m}$, the one-dimensional L^2 and RKHS inner products are defined by

$$\langle f_j, g_j \rangle_{L^2(\rho_j)} := \sum_{m \geq 1} a_{j,m} b_{j,m}, \qquad \langle f_j, g_j \rangle_{\mathcal{H}_j} := \sum_{m \geq 1} \frac{a_{j,m} b_{j,m}}{\lambda_{j,m}},$$

which induces the norms $\|f_j\|^2_{L^2(\rho_j)} = \sum_{m \geq 1} a_{j,m}^2$ and $\|f_j\|^2_{\mathcal{H}_j} = \sum_{m \geq 1} a_{j,m}^2 / \lambda_{j,m}$. Equivalently, $\{\sqrt{\lambda_{j,m}} \phi_{j,m}\}_{m \geq 1}$ forms an orthonormal system in $\mathcal{H}_j$.

We now extend the construction to the tensor setting using the standard tensor-product RKHS construction [1, 4]. Let $\mathcal{Z} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_d$, and assume that the full coordinate distribution is $\rho = \rho_1 \otimes \cdots \otimes \rho_d$. For the sake of brevity, we assume that the observation distribution echoes the target distribution.

Define the tensor-product RKHS $\mathcal{H} = \bigotimes_{j=1}^d \mathcal{H}_j$ with tensor-product kernel $K(z, z') = \prod_{j=1}^d k_j(z_j, z_j')$, which is positive definite by the closure properties of reproducing kernels under tensor products [1, 4]. For a multi-index $\boldsymbol{m} = (m_1, \dots, m_d) \in \mathbb{N}^d$, define

$$\Phi_{\boldsymbol{m}}(z) = \prod_{j=1}^d \phi_{j,m_j}(z_j), \qquad \Lambda_{\boldsymbol{m}} = \prod_{j=1}^d \lambda_{j,m_j}.$$

Because ρ is a product measure, it holds that $\langle \Phi_{\boldsymbol{m}}, \Phi_{\boldsymbol{n}} \rangle_{L^2(\rho)} = \prod_{j=1}^d \delta_{m_j n_j} = \delta_{\boldsymbol{m}\boldsymbol{n}}$, i.e., $\{\Phi_{\boldsymbol{m}}\}_{\boldsymbol{m} \in \mathbb{N}^d}$ is orthonormal in $L^2(\rho)$.

The tensor-product integral operator is $T = \bigotimes_{j=1}^d T_j$, and the preceding definitions imply $T\Phi_{\boldsymbol{m}} = \Lambda_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}$. Thus, the tensor spectrum follows from the one-dimensional spectra and the product-measure structure.

For two functions $f = \sum_{\boldsymbol{m}} a_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}$ and $g = \sum_{\boldsymbol{m}} b_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}$, the tensor-product RKHS inner product is given by

$$\langle f, g \rangle_{\mathcal{H}} = \sum_{\boldsymbol{m} \in \mathbb{N}^d} \frac{a_{\boldsymbol{m}} b_{\boldsymbol{m}}}{\Lambda_{\boldsymbol{m}}} = \sum_{\boldsymbol{m} \in \mathbb{N}^d} \frac{a_{\boldsymbol{m}} b_{\boldsymbol{m}}}{\prod_{j=1}^d \lambda_{j,m_j}},$$

which introduces the norm

$$\|f\|^2_{L^2(\rho)} = \sum_{\boldsymbol{m} \in \mathbb{N}^d} a_{\boldsymbol{m}}^2, \qquad \|f\|^2_{\mathcal{H}} = \sum_{\boldsymbol{m} \in \mathbb{N}^d} \frac{a_{\boldsymbol{m}}^2}{\Lambda_{\boldsymbol{m}}}.$$

Truncated RKHS-Tucker function class. For a fixed truncation level r, define the rectangular modewise truncation region

$$\mathcal{R}_r = \left\{ \boldsymbol{m} \in \mathbb{N}^d : 1 \leq m_j \leq r,\ \forall j = 1, \dots, d \right\}.$$

Its complement is $\mathbb{N}^d \setminus \mathcal{R}_r = \{\boldsymbol{m} \in \mathbb{N}^d : \exists j,\ m_j > r\}$, which contains all index tuples where at least one mode enters the high-frequency region $\{r+1, r+2, \dots\}$.

Using the tensor-product bases defined above, the truncated RKHS-Tucker function class is

$$\mathcal{F}_r = \Bigg\{ f : \mathcal{Z} \to \mathbb{R} \;\Bigg|\; f(z) = \sum_{\boldsymbol{m} \in \mathcal{R}_r} \theta_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}(z), \\ \Theta = (\theta_{\boldsymbol{m}})_{\boldsymbol{m} \in \mathcal{R}_r} \text{ admits a Tucker low-rank structure} \Bigg\}.$$

Let $u_r^\star$ be the best approximation of u within $\mathcal{F}_r$, namely

$$u_r^\star = \arg\min_{f \in \mathcal{F}_r} \mathcal{R}(f), \qquad \mathcal{R}(f) = \mathbb{E}_{z \sim \rho}\left[(f(z) - u(z))^2\right].$$

The empirical estimator is denoted by $\hat{u}_r$. The total reconstruction error satisfies

$$\|\hat{u}_r - u\|_{L^2(\rho)} \le \underbrace{\|u_r^\star - u\|_{L^2(\rho)}}_{\text{approximation error}} + \underbrace{\|\hat{u}_r - u_r^\star\|_{L^2(\rho)}}_{\text{statistical error}}.$$

Approximation error. We evaluate the approximation error by using a specific RKHS-Tucker function in $\mathcal{F}_r$. Assume that the target function satisfies the following tensor source condition, which is a standard smoothness assumption in spectral regularization and kernel learning theory [5, 7, 36]:

$$u = T^v g, \qquad \|g\|_{L^2(\rho)} \le C_g, \qquad v > \frac{1}{2}.$$

Using the spectral notation defined above, it holds that $g = \sum_{\boldsymbol{m}} g_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}$ and $u = \sum_{\boldsymbol{m}} \Lambda_{\boldsymbol{m}}^v g_{\boldsymbol{m}} \Phi_{\boldsymbol{m}}$. Let $P_r^{\otimes d}$ denote the tensor-product projection onto the span of bases indexed by $\mathcal{R}_r$. By the RKHS norm, the truncation error can be given by

$$\mathcal{E}_{\text{app},\mathcal{H}}^2(r) := \|(I - P_r^{\otimes d})u\|_{\mathcal{H}}^2 = \sum_{\boldsymbol{m} \notin \mathcal{R}_r} \Lambda_{\boldsymbol{m}}^{2v-1} g_{\boldsymbol{m}}^2.$$

The exponent $(2v-1)$ appears because the RKHS norm divides the squared coefficient by the corresponding kernel eigenvalue.

Assume further that each mode has polynomial eigenvalue decay $\lambda_{j,m} \le K_0 m^{-\beta}$, a standard capacity condition for kernel integral operators [5, 36]. Let $\gamma = \beta(2v-1)$. If $\gamma > 1$, then the one-dimensional spectral tail satisfies $\sum_{m > r} \lambda_{j,m}^{2v-1} \lesssim r^{1-\gamma}$, while the corresponding low-frequency sum is uniformly bounded in r. Therefore, the truncation error satisfies

$$\mathcal{E}_{\text{app},\mathcal{H}}^2(r) \lesssim C r^{1-\beta(2v-1)},$$

or equivalently,

$$\mathcal{E}_{\text{app},\mathcal{H}}(r) \lesssim C_A r^{-s}, \qquad s = \frac{\beta(2v-1)-1}{2}.$$

Here C_A absorbs the source radius C_g, the spectral decay constant K_0, the tensor order d, and the bounded low-frequency spectral sums; in particular, C_A is independent of r. Under the continuous embedding $\|h\|_{L^2(\rho)} \le C_{\text{emb}} \|h\|_{\mathcal{H}}$, the same rate gives

$$\|u_r^\star - u\|_{L^2(\rho)} \le C_{\text{emb}} \mathcal{E}_{\text{app},\mathcal{H}}(r) \lesssim C_A r^{-s}, \qquad s = \frac{\beta(2v-1)-1}{2}.$$

where C_A additionally absorbs C_{emb}.

Statistical error. For the statistical component, define the truncated tensor effective dimension as a finite-dimensional analogue of the effective dimension used in kernel ridge regression and regularized least squares [5, 36]:

$$d_{\text{eff},r}^{\text{tensor}}(\lambda) = \sum_{m_1 \le r, \ldots, m_d \le r} \frac{\prod_{j=1}^d \lambda_{j,m_j}}{\prod_{j=1}^d \lambda_{j,m_j} + \lambda}.$$

Here $\lambda > 0$ is the regularization parameter. It determines the learnability of each tensor spectral direction: directions with $\prod_{j=1}^d \lambda_{j,m_j} \gg \lambda$ are effectively retained, while directions with

$\prod_{j=1}^{d} \lambda_{j,m_j} \ll \lambda$ are suppressed by regularization. Thus, $d_{\mathrm{eff},r}^{\mathrm{tensor}}(\lambda)$ measures the effective number of learnable tensor spectral directions within the truncated RKHS class. Equivalently, the same statistical term can be obtained through localized Rademacher complexity for squared-loss regression under boundedness and local curvature conditions [2, 3].

Under bounded-kernel, sub-Gaussian noise, and local curvature assumptions, standard kernel regression and localized Rademacher arguments yield that, with probability at least $1-\delta$ [3, 5],

$$\mathcal{E}_{\mathrm{stat}}(r,n,\delta) = \|\hat{u}_r - u_r^\star\|_{L^2(\rho)} \lesssim C_S \sqrt{\frac{d_{\mathrm{eff},r}^{\mathrm{tensor}}(\lambda) + \log(1/\delta)}{n}}.$$

Since $d_{\mathrm{eff},r}^{\mathrm{tensor}}(\lambda) \le r^d$, a conservative rank-based bound is

$$\mathcal{E}_{\mathrm{stat}}(r,n,\delta) \lesssim C_S \sqrt{\frac{r^d + \log(1/\delta)}{n}}.$$

Balancing reconstruction rate. Combining the approximation and statistical bounds, we yield

$$\|\hat{u}_r - u\|_{L^2(\rho)} \lesssim C_A r^{-s} + C_S \sqrt{\frac{r^d + \log(1/\delta)}{n}}, \qquad s = \frac{\beta(2v-1)-1}{2}.$$

Ignoring the lower-order concentration term, the optimal truncation level is obtained by balancing $C_A r^{-s} \asymp C_S r^{d/2} n^{-1/2}$, which gives

$$r_\star \asymp n^{\frac{1}{2s+d}}.$$

Substituting $r_\star$ into the total error, the right-hand side boils down to $Cn^{-\frac{s}{2s+d}}$.

Since $2s = \beta(2v-1)-1$, we obtain

$$\|\hat{u}_{r_\star} - u\|_{L^2(\rho)} \lesssim Cn^{-\frac{\beta(2v-1)-1}{2(\beta(2v-1)-1+d)}} + C\sqrt{\frac{\log(1/\delta)}{n}}.$$

This rate reflects two effects: the kernel spectrum controls how fast the rectangular truncation error decays, while the retained tensor-product spectral dimension controls the statistical term.

Sample complexity. Finally, the sample complexity can be inferred from the balanced rate. To ensure that $\|\hat{u}_r - u\|_{L^2(\rho)} \le \epsilon$ with probability at least $1-\delta$, it suffices to choose

$$n \gtrsim \left(\frac{2C}{\epsilon}\right)^{\frac{2(\beta(2v-1)-1+d)}{\beta(2v-1)-1}} + \frac{4C^2\log(1/\delta)}{\epsilon^2}.$$

Specialization to the Matérn-$3/2$ kernel. We now adopt the preceding bound to the Matérn-$3/2$ kernel. For a one-dimensional mode, the Matérn kernel with smoothness parameter $\nu_{\mathrm{M}} = 3/2$ has polynomial spectral decay $\lambda_{j,m} \lesssim m^{-\beta}$, with $\beta = 2\nu_{\mathrm{M}} + 1 = 4$, under the standard one-dimensional Sobolev/Matérn spectral correspondence [28, 34]. The value of β is determined by the kernel spectrum, whereas the source exponent v describes the regularity of the target field relative to the kernel integral operator.

If the target field satisfies the second-order source condition $u = T^2 g$ with $\|g\|_{L^2(\rho)} \le C_g$, then $v = 2$, so $\gamma = \beta(2v-1) = 12$ and $s = (\gamma-1)/2 = 11/2$. The approximation and total reconstruction bounds become

$$\mathcal{E}_{\mathrm{app}}(r) \lesssim C_A r^{-11/2}, \qquad \|\hat{u}_r - u\|_{L^2(\rho)} \lesssim C_A r^{-11/2} + C_S \sqrt{\frac{r^d + \log(1/\delta)}{n}}.$$

Balancing $r^{-11/2} \asymp r^{d/2} n^{-1/2}$ gives $r_\star \asymp n^{1/(11+d)}$. Substituting this choice yields

$$\|\hat{u}_{r_\star} - u\|_{L^2(\rho)} \lesssim Cn^{-\frac{11}{2(11+d)}} + C\sqrt{\frac{\log(1/\delta)}{n}}.$$

This rate shows that, under a Matérn-$3/2$ kernel and a second-order source condition, the approximation term decays rapidly with the truncation level, while the statistical term still depends on the effective tensor dimension induced by the retained modes.

Table 9: Datasets used in the experiments. Shapes are trajectories × time × spatial axes × channels.

Dataset	Split	Stored tensor	Channels	Domain and time span
AC2D	160/20/20	$200 \times 201 \times 256 \times 256$	scalar u	$[0,1)^2, t \in [0,2]$
NS2D	640/80/80	$800 \times 201 \times 128 \times 128 \times 3$	(u, v, ω)	$[0,1)^2, t \in [0,20]$
Heat3D	160/20/20	$200 \times 50 \times 64 \times 64 \times 64 \times 1$	temperature u	$[0,1]^3, t \in [0,4]$
Ocean3D	80/10/10	$100 \times 90 \times 32 \times 64 \times 64 \times 2$	(u, v)	depth–latitude–longitude, 90 days

D Dataset Information

The experiments involve three synthetic benchmarks and a gridded ocean reanalysis. Table 9 summarizes the tensors from the datasets.

AC2D and NS2D. AC2D solves the periodic two-dimensional Allen–Cahn equation with $\lambda = 1$ and $\epsilon = 0.01$ and saves 201 frames at interval 0.01. NS2D solves the forced periodic vorticity equation with viscosity 10^{-4}, saves 201 frames at interval 0.1, and stores (u, v, ω) after downsampling to 128^2. The AC2D and NS2D training and held-out equation-residual evaluators use centered differences in both space and time and discard the first and last time points.

Heat3D. Heat3D contains 200 independent scalar-temperature trajectories. The trajectories are permuted with seed 123 and split 160/20/20. All trajectories share 50 physical save times in $[0, 4]$; these times are slightly nonuniform because the explicit-Euler step and the save interval are rounded, while the logical model coordinate is a uniform 50-point grid in $[0, 1]$. All 64^3 voxels are valid, the spatial grid includes both endpoints, and the generator imposes homogeneous Dirichlet conditions on all six faces.

Ocean3D. Ocean3D is constructed from daily-mean horizontal currents (u, v) in the CMEMS product `GLOBAL_MULTIYEAR_PHY_001_030`. The local source consists of annual NetCDF files, starting with `uv_fulldepth_1993-01-01_to_1993-12-31.nc`. Files are ordered chronologically, and each file is divided into nonoverlapping 90-day sequences beginning at its first record. Trailing records that do not form a complete sequence are discarded; sequences never cross file boundaries. We retain the first 100 sequences, comprising 9000 daily fields from 1993 to 2017 and split them sequentially into 80 training, 10 validation, and 10 test sequences. The first and last retained dates of these splits are 1993-01-01 to 2012-12-25, 2013-01-01 to 2015-06-29, and 2015-06-30 to 2017-12-26, respectively; these endpoint ranges do not imply uninterrupted daily coverage because 5 or 6 trailing days are omitted in each year. We directly retain the source-grid crop of 64×64 horizontal points spanning latitudes [-2.6666667461, 2.5833332539] and longitudes [167.3333282471, 172.5833282471] degrees, without horizontal interpolation. We select zero-based depth indices 0 to 31; these 32 nonuniform levels range from 0.494024992 to 541.088928 m.Each sequence has shape $90 \times 32 \times 64 \times 64 \times 2$.

D.1 Evaluation subsets

For trajectory b, let $\mathcal{X}_b$ be its evaluation grid, $\mathcal{X}_{\text{sen}}^{(b)} \subset \mathcal{X}_b$ its fixed set of sensor locations, and $\mathcal{I}_{\text{sup}}^{(b,p)}$ and $\mathcal{I}_{\text{miss}}^{(b,p)}$ the indices of support frames and missing frames under protocol p, respectively. The four disjoint subsets are

$$\mathcal{A}_{\text{S-sup}}^{(b,p)} = \mathcal{I}_{\text{sup}}^{(b,p)} \times \mathcal{X}_{\text{sen}}^{(b)}, \qquad \mathcal{A}_{\text{Off-sup}}^{(b,p)} = \mathcal{I}_{\text{sup}}^{(b,p)} \times (\mathcal{X}_b \setminus \mathcal{X}_{\text{sen}}^{(b)}),$$
$$\mathcal{A}_{\text{S-miss}}^{(b,p)} = \mathcal{I}_{\text{miss}}^{(b,p)} \times \mathcal{X}_{\text{sen}}^{(b)}, \qquad \mathcal{A}_{\text{Off-miss}}^{(b,p)} = \mathcal{I}_{\text{miss}}^{(b,p)} \times (\mathcal{X}_b \setminus \mathcal{X}_{\text{sen}}^{(b)}).$$

Support frames provide sparse measurements only at sensor locations, not complete spatial fields. Missing frames provide no sensor measurements to the model; their ground-truth fields, including values at sensor locations, are retained for evaluation. Let C_D denote the number of channels. For each evaluation subset, we pool squared errors across trajectories, selected spacetime points, and channels, divide by the total number of selected scalar values, and take the square root. An example

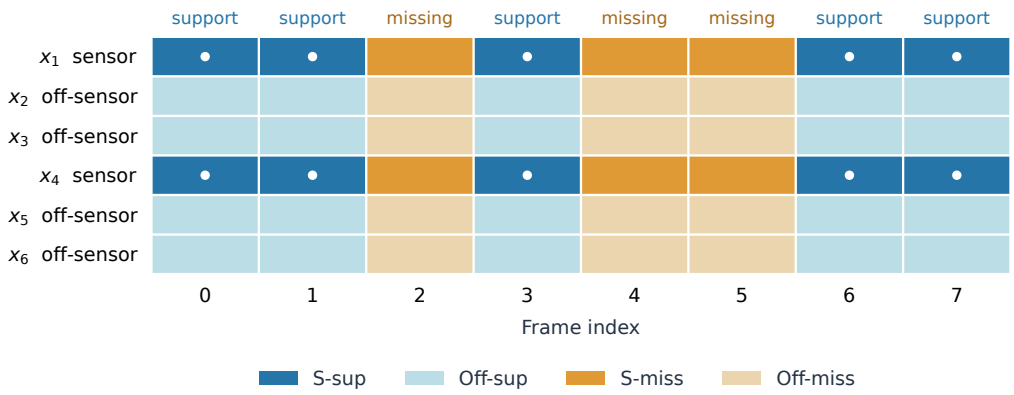

Figure 3: Four evaluation subsets defined by sensor/off-sensor locations and support/missing frames. White dots indicate sensor measurements provided to the model.

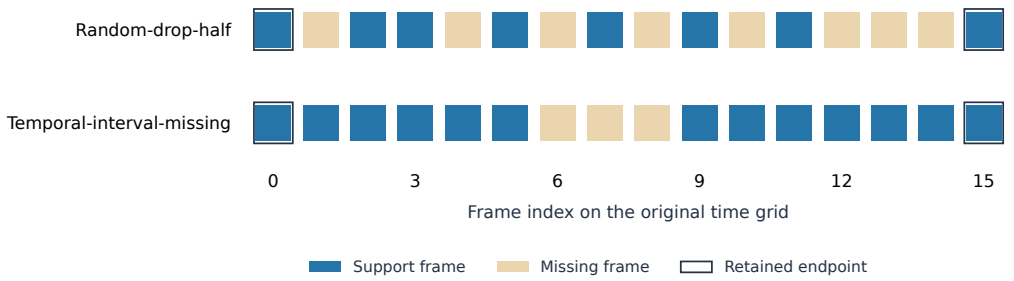

Figure 4: Two missing-frame protocols on the original time grid: randomly withholding frames or withholding a contiguous block. Both endpoints remain support frames.

is shown in Figure 3.

$$\mathrm{RMSE}_D(\mathcal{A}) = \left[\frac{\sum_b \sum_{(t,\mathbf{x})\in\mathcal{A}^{(b)}} \sum_{c=1}^{C_D} |\hat{u}_b(t,\mathbf{x},c) - u_b(t,\mathbf{x},c)|^2}{C_D \sum_b |\mathcal{A}^{(b)}|} \right]^{1/2}.$$

D.2 Missing-frame protocols

Let T be the max timestamp of the trajectory. Let $M_t^{(b)}$ be the spatial mask fixed within the trajectory, and let $K_t^{(b,p)} \in \{0,1\}$ be a frame gate specific to the trajectory. The observation mask is $\widetilde{M}_t^{(b,p)} = K_t^{(b,p)} M_t^{(b)}$; when $K_t = 1$, the frame is a support frame with sparse sensor measurements, and when $K_t = 0$, it is a missing frame with all sensor measurements withheld across depths and channels. This mask affects the input of the model, not the availability of ground truth for evaluation.

Random-drop-half. Both endpoints are retained, while internal frames are sampled independently:

$$K_0^{(b)} = K_{T-1}^{(b)} = 1, \qquad K_t^{(b)} \overset{\text{i.i.d.}}{\sim} \mathrm{Bernoulli}(0.5), \quad t = 1, \ldots, T-2.$$

The implementation uses `numpy.random.RandomState` and split seed $10^5 \cdot 123 + 10g(s) + i_{\text{split}}$, where $g(0.01) = 1$, $g(0.03) = 3$, $g(0.15) = 15$, and $i_{\text{split}} \in \{0,1,2\}$.

Temporal-interval-missing. A contiguous internal block is sampled independently for each trajectory:

$$L_{\text{miss}} = \max(1, \mathrm{round}(0.2T)), \qquad a_b \sim \mathrm{Uniform}\{1, \ldots, T - L_{\text{miss}} - 1\},$$

and $K_t^{(b)} = 0$ for $t \in [a_b, a_b + L_{\text{miss}})$. Both endpoints remain support frames.An example is shown in Figure 4.

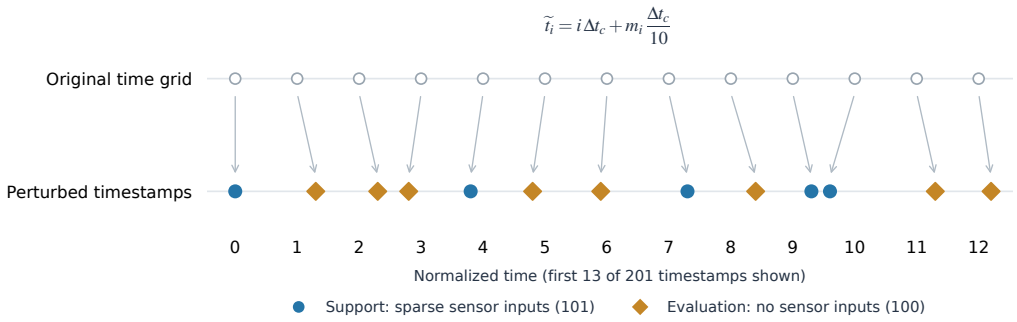

Figure 5: Internal timestamps are perturbed before being split into support and evaluation sets, while both endpoints remain fixed. Evaluation ground truth is recorded directly by the PDE solver.

D.3 Reconstruction at non-grid-aligned times

This AC2D/NS2D ablation is separate from the native-grid missing-frame protocols. Its evaluation timestamps are non-grid-aligned relative to the original coarse time grid and lie within the interval spanned by the timestamps of support frames. They are not defined as missing frames from the original stored sequence. Starting from 201 coarse times $t_i^{(0)} = i\Delta t_c$, both endpoints are fixed and every internal time is perturbed by

$$\widetilde{t}_i = i\Delta t_c + m_i \frac{\Delta t_c}{10}, \qquad m_i \overset{\text{i.i.d.}}{\sim} \text{Uniform}\{-4,-3,-2,-1,1,2,3,4\}, \quad i = 1,\dots,199.$$

AC2D uses $\Delta t_c = 0.01$ on $[0,2]$, and NS2D uses $\Delta t_c = 0.1$ on $[0,20]$.100 evaluation indices are sampled without replacement from $\{1,\dots,199\}$:

$$I_e \subset \{1,\dots,199\}, \quad |I_e| = \lfloor 201 \times 0.5 \rfloor = 100, \qquad I_s = \{0,\dots,200\} \setminus I_e, \quad |I_s| = 101.$$

Sensor measurements are provided only at the timestamps indexed by I_s; the timestamps indexed by I_e are used for reconstruction evaluation without providing their sensor measurements to the model. The endpoints therefore always belong to the support set. AC2D and NS2D use the same normalized offset vector and index-level partition between support frames and evaluation snapshots at non-grid-aligned times, shared across all test trajectories and sensors. Ground-truth states are saved directly by rerunning each PDE solver at the perturbed timestamps, not by interpolating uniformly saved snapshots. An example is shown in Figure 5.

E Complete reconstruction results across sensor ratios

Tables 10–13 report the complete three-ratio four-subset comparison. Rankings, when discussed, use only available values within the same dataset, temporal protocol, spatial ratio, and subset.

F Implementation Details

The spatial decoder uses a Matérn-$3/2$ kernel. Each active spatial mode has its own factor matrix; two-dimensional data use two modes, while Heat3D and Ocean3D use three. The number of inducing points along each mode equals that mode's grid side length in the reported default configuration; the reported AC2D sensitivity baseline therefore uses 256 inducing points. Spatial factors are shared across channels, with the channel index stored in the Tucker core.

The spatial stage uses batch size 8 and is trained for 500 epochs, where one epoch traverses the complete training loader. For each mini-batch, the conditionally linear cores are updated by conjugate gradients and the RKHS factors are updated with AdamW. The learning rate is 2×10^{-4}; the RKHS, core-ridge, core-TV, and temporal-GP weights are 10^{-4}, 10^{-2}, 10^{-2}, and 10^{-5}, respectively. Core updates use five CG iterations and tolerance 10^{-3}.

Spatial checkpoints are selected by the lowest dense-field RMSE on the training trajectories after inverse normalization, averaging the RMSE values of evaluation mini-batches. Dense training fields

are used for spatial checkpoint selection; spatial reconstruction gradients use only available sensor measurements, together with the stated regularizers and equation constraints.

The complete model samples equation-collocation centers from off-sensor locations on the spatial grid. For each trajectory, the set of sensor locations is the union of spatial locations with sensor measurements at any saved time and in any channel; the eligible collocation set is its spatial complement. AC2D and NS2D use two-dimensional grid centers, while Heat3D uses three-dimensional voxel centers. For Heat3D, centers whose finite-difference stencils would require moving the center inward from a boundary are excluded before sampling. The off-sensor restriction applies to collocation centers; neighboring stencil evaluations may lie at sensor locations. Ocean3D uses no governing-equation loss in the reported setting. AC2D and NS2D use centered spatial and temporal differences and exclude the temporal endpoints from residual training and evaluation.

Before each AdamW factor update, the gradient of each RKHS coefficient matrix is left-multiplied by the inverse Gram matrix of its spatial-mode kernel, $K_d^{-1}\nabla_{A_d}\mathcal{L}$. The spatial AdamW optimizer uses weight decay 10^{-2}; the gradient-only core updater uses the same weight decay.

We compare block-update strategies with their corresponding effective objectives. The complete model alternates CG core solves with gradient-based RKHS-factor updates. The solve-only variant retains the same ridge-TV core subproblem, but updates each spatial factor by solving for target factor values with a value-ridge penalty and projecting them to RKHS coefficients with kernel ridge regression. These factor updates use observational least squares and kernel projection. The gradient-only variant updates both cores and factors with AdamW using the RMSE over available sensor measurements, RKHS penalty, equation-based loss, and time-kernel penalty. The RKHS penalty depends only on the factors, whereas the time-kernel penalty depends only on the cores. Core-ridge and temporal-TV penalties belong to the CG subproblem. The time-kernel term is defined in Eq. (10).

Fields are normalized with channel-wise train-set statistics. Validation and test splits reuse these statistics, and all reported RMSEs, residuals, and physical diagnostics are computed after inverse normalization. The Ocean3D RecFNO RMSE export originally aggregates both channels jointly; the values reported in Table 1 use the corresponding per-channel convention by dividing those four exported RMSE columns by $\sqrt{2}$.

At inference, the spatial decoder is frozen and the core-correction sequence is solved around the train-set mean-core reference on the complete native time grid with ridge 10^{-2}, TV weight 10^{-2}, 30 CG iterations, tolerance 10^{-4}, and batch size 32. Missing frames have zero observation masks; after the joint solve, the cores at the timestamps of support frames are extracted as support anchors.

The residual ODE-RNN temporal head operates in Tucker-core space, uses hidden dimension 256, and is trained for 20 epochs with AdamW (batch size 4, learning rate 10^{-3}, and weight decay 10^{-6}, except that the learned residual-scale parameter has zero weight decay). Let $B_i(t)$ be the cubic Hermite baseline constructed from the inferred support cores, and let $C_\psi(t)$ denote the actual correction added to that baseline, including the interval gate and the learned bounded residual scale. At the timestamps of missing frames, we minimize the mean squared error between $C_\psi(t)$ and $G_i^{\mathrm{train}}(t) - B_i(t)$, where G_i^{train} is the frozen training-core sequence saved by the spatial stage. Equivalently, the objective is the missing-frame MSE between the predicted core $B_i(t) + C_\psi(t)$ and $G_i^{\mathrm{train}}(t)$. The core-loss weight is 1; decoded-field, spatial-gradient, and temporal-equation losses are disabled. We select the checkpoint with the lowest validation relative ℓ_2 error over missing frames, computed from decoded fields after inverse normalization.

$$\mathcal{L}_{\mathrm{temporal}} = \mathrm{MSE}_{i,t\in\mathcal{T}_{\mathrm{miss}}^{(i)}}\left(C_\psi(t), G_i^{\mathrm{train}}(t) - B_i(t)\right). \tag{21}$$

The MSE averages all scalar coefficients of the selected cores. Missing-frame validation uses all spatial locations at the selected times.

Table 10: Complete four-subset reconstruction RMSE on AC2D across spatial sensor ratios. Lower is better. (**Bold**: best, Underline: second best)

Protocol	s	Method	S-sup	Off-sup	S-miss	Off-miss
Random-drop-half	0.01	SDIFT	0.373381	0.381700	0.378977	0.381200
		LRTFR	0.311380	0.329100	0.308521	0.328814
		PhysFTM	**0.009898**	**0.187900**	0.042055	**0.191673**
		RecFNO	0.020782	0.697782	**0.021170**	0.696267
		SIREN-MLP	0.351270	0.350988	0.349308	0.349007
		PINNs	0.354653	0.354871	0.350601	0.350840
	0.03	SDIFT	0.358035	0.380800	0.380301	0.381514
		LRTFR	0.311700	0.322200	0.313404	0.320383
		PhysFTM	**0.017354**	**0.087100**	0.050349	**0.099881**
		RecFNO	0.018483	0.452826	**0.021782**	0.455233
		SIREN-MLP	0.348683	0.349317	0.350125	0.350761
		PINNs	0.351663	0.352425	0.352679	0.353309
	0.15	SDIFT	0.294393	0.374100	0.371273	0.371736
		LRTFR	0.320074	0.317200	0.319075	0.320370
		PhysFTM	0.029960	**0.035900**	0.057787	**0.062454**
		RecFNO	**0.011095**	0.230709	**0.011986**	0.231022
		SIREN-MLP	0.350102	0.350055	0.350040	0.349975
		PINNs	0.353934	0.353904	0.351765	0.351766
Temporal-interval-missing	0.01	SDIFT	0.372720	0.379670	0.372029	0.373811
		LRTFR	0.309972	0.330427	0.306973	0.326716
		PhysFTM	**0.006558**	**0.189538**	0.032476	**0.183397**
		RecFNO	0.020743	0.701355	**0.028592**	0.681053
		SIREN-MLP	0.351543	0.351296	0.342732	0.342199
		PINNs	0.353767	0.354026	0.347975	0.348079
	0.03	SDIFT	0.356991	0.361986	0.390707	0.392586
		LRTFR	0.310055	0.316757	0.320590	0.327806
		PhysFTM	**0.016328**	**0.093969**	0.039527	**0.079239**
		RecFNO	0.018107	0.458752	**0.021163**	0.436798
		SIREN-MLP	0.346053	0.346642	0.358185	0.358978
		PINNs	0.348742	0.349428	0.365077	0.366217
	0.15	SDIFT	0.294740	0.299284	0.349127	0.350140
		LRTFR	0.320387	0.321854	0.305895	0.307641
		PhysFTM	0.031414	**0.040888**	0.045196	**0.048007**
		RecFNO	**0.011196**	0.234129	**0.012961**	0.218094
		SIREN-MLP	0.353258	0.353191	0.332458	0.332446
		PINNs	0.356292	0.356274	0.338695	0.338699

Table 11: Complete four-subset reconstruction RMSE on NS2D across spatial sensor ratios. Lower is better. (**Bold**: best, Underline: second best)

Protocol	s	Method	S-sup	Off-sup	S-miss	Off-miss
Random-drop-half	0.01	SDIFT	0.518934	0.512800	0.510791	0.509261
		LRTFR	0.123841	0.172000	0.126188	0.170679
		PhysFTM	**0.007849**	**0.137700**	**0.030927**	**0.130659**
		RecFNO	0.261520	0.418079	0.262461	0.413223
		SIREN-MLP	0.244818	0.245620	0.243905	0.244360
		PINNs	0.338968	0.341335	0.339094	0.341072
	0.03	SDIFT	0.528668	0.509700	0.510029	0.510274
		LRTFR	0.132450	0.147200	0.134470	0.145757
		PhysFTM	**0.006687**	**0.062100**	**0.032396**	**0.064976**
		RecFNO	0.068098	0.097567	0.078196	0.104413
		SIREN-MLP	0.245551	0.245255	0.244651	0.244405
		PINNs	0.342539	0.341771	0.341006	0.340554
	0.15	SDIFT	0.609226	0.510500	0.510615	0.510589
		LRTFR	0.135290	0.139400	0.136582	0.138756
		PhysFTM	**0.005888**	0.020300	**0.032096**	**0.033357**
		RecFNO	0.007173	**0.007929**	0.039964	0.040051
		SIREN-MLP	0.245407	0.245486	0.244408	0.244476
		PINNs	0.342011	0.342206	0.340038	0.340193
Temporal-interval-missing	0.01	SDIFT	0.522547	0.520767	0.494061	0.492242
		LRTFR	0.127082	0.173888	0.179695	0.206649
		PhysFTM	**0.004291**	**0.141533**	**0.136380**	**0.164821**
		RecFNO	0.267930	0.428534	0.405006	0.478397
		SIREN-MLP	0.249049	0.249814	0.206410	0.206503
		PINNs	0.346718	0.349269	0.306146	0.306602
	0.03	SDIFT	0.530194	0.530518	0.504341	0.502748
		LRTFR	0.135148	0.146821	0.190250	0.196613
		PhysFTM	**0.004346**	**0.062824**	**0.149493**	**0.150558**
		RecFNO	0.069056	0.099100	0.358033	0.359434
		SIREN-MLP	0.247968	0.247773	0.213687	0.213537
		PINNs	0.347219	0.346569	0.318917	0.318479
	0.15	SDIFT	0.610478	0.609820	0.485120	0.485171
		LRTFR	0.138750	0.140954	0.170839	0.172046
		PhysFTM	**0.005690**	0.012067	**0.134833**	**0.134567**
		RecFNO	0.007474	**0.008253**	0.318252	0.318191
		SIREN-MLP	0.250208	0.250287	0.203004	0.203190
		PINNs	0.350376	0.350661	0.300498	0.300163

Table 12: Complete four-subset reconstruction RMSE on Heat3D across spatial sensor ratios. Lower is better. (**Bold**: best, Underline: second best)

Protocol	s	Method	S-sup	Off-sup	S-miss	Off-miss
Random-drop-half	0.01	SDIFT	0.058457	0.058594	0.061825	0.062284
		LRTFR	0.010659	0.010815	0.012014	0.012549
		PhysFTM	**0.001344**	**0.001907**	**0.006379**	**0.006824**
		RecFNO	0.339625	0.888925	0.307609	0.813064
		SIREN-MLP	0.040199	0.040240	0.042215	0.042397
		PINNs	0.040309	0.040409	0.041993	0.042295
	0.03	SDIFT	0.058111	0.058081	0.063030	0.062716
		LRTFR	0.012121	0.012216	0.013093	0.013146
		PhysFTM	**0.001083**	**0.001250**	**0.005745**	**0.005744**
		RecFNO	0.287363	0.778472	0.263722	0.720340
		SIREN-MLP	0.040803	0.040865	0.042108	0.042053
		PINNs	0.040002	0.040113	0.042714	0.042640
	0.15	SDIFT	0.059938	0.059993	0.060820	0.060836
		LRTFR	0.012518	0.012578	0.011607	0.011652
		PhysFTM	**0.001015**	**0.001040**	**0.005282**	**0.005295**
		RecFNO	0.171750	0.322953	0.158704	0.258328
		SIREN-MLP	0.041383	0.041470	0.041042	0.041105
		PINNs	0.041778	0.041929	0.040826	0.040917
Temporal-interval-missing	0.01	SDIFT	0.059872	0.060202	0.060550	0.060789
		LRTFR	0.011883	0.012311	0.012546	0.012876
		PhysFTM	**0.001293**	**0.002132**	**0.007193**	**0.007070**
		RecFNO	0.320509	0.837414	0.321806	0.904206
		SIREN-MLP	0.040678	0.040735	0.038204	0.038412
		PINNs	0.040636	0.040735	0.043391	0.044007
	0.03	SDIFT	0.062840	0.062680	0.050158	0.049889
		LRTFR	0.012100	0.012250	0.005766	0.005767
		PhysFTM	**0.001218**	**0.001372**	**0.003103**	**0.003046**
		RecFNO	0.286290	0.775257	0.225969	0.641868
		SIREN-MLP	0.043423	0.043440	0.027492	0.027421
		PINNs	0.043136	0.043190	0.033532	0.033341
	0.15	SDIFT	0.063097	0.063136	0.047558	0.047574
		LRTFR	0.011794	0.011881	0.009763	0.009595
		PhysFTM	**0.001105**	**0.001136**	**0.003756**	**0.003682**
		RecFNO	0.175643	0.317359	0.122943	0.169169
		SIREN-MLP	0.043686	0.043784	0.025485	0.025478
		PINNs	0.044045	0.044195	0.027758	0.027711

Table 13: Complete four-subset reconstruction RMSE on Ocean across spatial sensor ratios. Lower is better. (**Bold**: best, Underline: second best)

Protocol	s	Method	S-sup	Off-sup	S-miss	Off-miss
Random-drop-half	0.01	SDIFT	0.281171	0.278458	0.280994	0.280682
		LRTFR	0.113447	0.126022	0.122721	0.133850
		PhysFTM	**0.006599**	**0.115414**	**0.076277**	**0.125325**
		RecFNO	0.758963	1.005671	0.793739	1.002127
		SIREN-MLP	0.200624	0.203220	0.196868	0.200081
		PINNs	0.297127	0.301674	0.295973	0.301402
	0.03	SDIFT	0.278627	0.280655	0.275848	0.278376
		LRTFR	0.110377	0.114503	0.117666	0.121548
		PhysFTM	**0.006319**	**0.065738**	**0.071328**	**0.088581**
		RecFNO	0.738273	0.970741	0.776879	0.963019
		SIREN-MLP	0.201582	0.201681	0.200241	0.200943
		PINNs	0.301625	0.300997	0.302323	0.301937
	0.15	SDIFT	0.275481	0.276918	0.281502	0.282907
		LRTFR	0.109859	0.110456	0.127294	0.127726
		PhysFTM	**0.007103**	**0.017565**	**0.080431**	**0.080677**
		RecFNO	0.711010	0.854068	0.755037	0.861364
		SIREN-MLP	0.199900	0.200691	0.201670	0.202316
		PINNs	0.298829	0.300486	0.301210	0.302968
Temporal-interval-missing	0.01	SDIFT	0.283314	0.282061	0.272620	0.270666
		LRTFR	0.104156	0.119385	**0.171227**	**0.182007**
		PhysFTM	**0.005952**	**0.115920**	0.191076	0.194097
		RecFNO	0.763253	1.007166	0.944159	0.992394
		SIREN-MLP	0.201337	0.204394	0.182836	0.184601
		PINNs	0.299977	0.306116	0.282315	0.282459
	0.03	SDIFT	0.274940	0.277254	0.287116	0.289410
		LRTFR	0.105062	0.108289	**0.180623**	**0.183770**
		PhysFTM	**0.005946**	**0.063821**	0.189849	0.185648
		RecFNO	0.718410	0.943793	0.991909	1.053829
		SIREN-MLP	0.197733	0.198327	0.211994	0.211505
		PINNs	0.297650	0.297391	0.318698	0.317248
	0.15	SDIFT	0.277062	0.278427	0.284215	0.285884
		LRTFR	0.110311	0.110758	0.190074	0.190728
		PhysFTM	**0.007105**	**0.017647**	**0.159473**	**0.158267**
		RecFNO	0.712010	0.855126	0.930787	0.906187
		SIREN-MLP	0.202733	0.203547	0.189757	0.190244
		PINNs	0.300200	0.301877	0.299327	0.301165

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction summarize the proposed method, the target problem setting, and the empirical scope of the work.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Conclusion discusses reconstruction ambiguity and the scope of fixed sensors, missing frames, and in-range queries. Appendix C states the idealized assumptions used in the theoretical analysis.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The details about theory and assumptions are in Section C

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: The method, losses, datasets, metrics, splits, and implementation settings are described in Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes].

Justification: The current manuscript describes the implementation and data layout but does not yet provide an anonymized public code and data release with complete access instructions. Code including data download/generation code are available at https://github.com/liusirzzz/PhysFTM

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes].

Justification: The observation protocols, data splits, training and test settings, and optimization hyperparameters are described in Section 4.1 and Appendices A, D, and F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No].

Justification: The experiments use fixed splits and a fixed random seed, and the current tables report point estimates rather than error bars or confidence intervals.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No].

   Justification: It does not yet provide complete GPU model, memory, wall-clock training time, and total compute accounting for every experiment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification:The work uses scientific simulation data, does not involve human subjects or sensitive personal data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No].

    Justification: Potential positive and negative societal impacts are not discussed in this manuscript.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The paper does not release high-risk assets such as language models, human-image generators, scraped personal datasets, or tools intended for dual-use misuse. The proposed model targets scientific fluid-field reconstruction from paired numerical or reanalysis data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No].

    Justification: The paper credits the baseline methods and identifies the data sources, but does not provide a complete account of the licenses and terms of use for all existing assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: The current submission does not release a new dataset, public model checkpoint, or other standalone asset. The manuscript documents the proposed method and implementation details.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: The research does not involve crowdsourcing, participant studies, user annotation, or any other human-subject data collection.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

Justification: The research does not involve crowdsourcing or human subjects, so IRB or equivalent human-subjects approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A].

Justification: LLMs are not used as an important, original, or non-standard component of the core method, experiments, or evaluation.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.