# Differentiable Physics-Neural Models enable Learning of Non-Markovian Closures for Accelerated Coarse-Grained Physics Simulations

**Tingkai Xue**
Department of Mechanical Engineering
National University of Singapore
Singapore 119007, Singapore
xue.tingkai@u.nus.edu

**Chinchun Ooi**
Institute of High Performance Computing
Agency for Science, Technology and Research
Singapore 138632, Singapore
ooicc@cfar.a-star.edu.sg

**Zhengwei Ge**
Institute of High Performance Computing
Agency for Science, Technology and Research
Singapore 138632, Singapore
gezw@ihpc.a-star.edu.sg

**Fong Yew Leong**
Institute of High Performance Computing
Agency for Science, Technology and Research
Singapore 138632, Singapore
leongfy@ihpc.a-star.edu.sg

**Hongying Li**
Dept of Mechanical and Aerospace Engineering
Nanyang Technological University
Singapore 639798, Singapore
hongying.li@ntu.edu.sg

**Chang Wei Kang**
Institute of High Performance Computing
Agency for Science, Technology and Research
Singapore 138632, Singapore
kangcw@ihpc.a-star.edu.sg

## Abstract

Numerical simulations provide key insights into many physical, real-world problems. However, while these simulations are solved on a full 3D domain, most analysis only require a reduced set of metrics (e.g. plane-level concentrations). This work presents a hybrid physics-neural model that predicts scalar transport in a complex domain orders of magnitude faster than the 3D simulation (from hours to less than 1 min). This end-to-end differentiable framework jointly learns the physical model parameterization (i.e. orthotropic diffusivity) and a non-Markovian neural closure model to capture unresolved, 'coarse-grained' effects, thereby enabling stable, long time horizon rollouts. This proposed model is data-efficient (learning with 26 training data), and can be flexibly extended to an out-of-distribution scenario

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Machine Learning and the Physical Sciences.

(with a moving source), achieving a Spearman correlation coefficient of 0.96 at the final simulation time. Overall results show that this differentiable physics-neural framework enables fast, accurate, and generalizable coarse-grained surrogates for physical phenomena.

## 1 Introduction

Numerical simulations, e.g. computational fluid dynamics (CFD), are a cornerstone for understanding complex physical phenomena, but their high computational cost can be extremely restrictive. Coarse-graining of simulations is a widely-used alternative to reduce computational cost, especially in multi-scale systems, whereby the dimensionality of the system is reduced through model parameterizations of unresolved fine-scale dynamics [1]. According to the Mori–Zwanzig formalism, this coarse-graining requires memory-dependent closure terms to better account for unresolved interactions across scales. Architectures such as Long Short-Term Memory (LSTM) model were used to predict temporal dynamics in a reduced order basis [2, 3], but accurate, generalizable prediction remains difficult. Recent work in scientific machine learning has also shown the promise of physics-informed surrogates [4, 5], and neural closures have been proposed for modelling of dynamical systems, including subgrid closures for turbulent flows, and neural closures for incomplete multiphysics forecasting [6–10]. However, these efforts rely on purely Markovian closure terms, ignore history-dependent effects, and/or treat physical parameters as fixed rather than trainable quantities.

Inspired by recent innovations whereby Physics guides ML, we propose a hybrid physics–neural surrogate model that provides orders-of-magnitude speed-up over conventional numerical simulations through an analogous coarse-grained-inspired approach for physical consistency. This end-to-end differentiable framework explicitly learns only the non-Markovian closure terms, which potentially explains the model's predictive and extrapolative ability even with a small training dataset ($N_{training} = 26$ 3D full-field simulations). Critically, the proposed model incorporates joint inference of improved physical parameterizations for encoding the physical environment (e.g. diffusivity), a critical step which was found to improve model performance.

The proposed methodology is applied to the transport of a released scalar (e.g. pollutant, airborne virus) in a complex real-world indoor environment. While the full 3D domain greatly impacts the physics, the scalar concentration at human height (1.5 m above ground level) is typically the key metric of interest. Hence, the 'coarse-grained physics' allows acceleration through two key processes without affecting its practical utility: 1) Reduction of problem from the full 3D domain to a 2D plane of interest in the domain; 2) Model-learned implicit parameterization of domain geometry (e.g. obstructions like pillars, and non-linear effects from boundary conditions like the air-conditioners).

For effective long-term simulation, the unresolved physics (e.g. flow orthogonal to the modeled plane) has to be well-learned by the proposed physics-neural model [11]. By combining a finite-volume solver (physics-based) with an LSTM-based neural closure model, we provide a principled approximation of the memory effects as required by the Mori–Zwanzig formalism [12]. Moreover, the differentiable physics-neural framework allows physical quantities (e.g. the orthotropic diffusivity) to be learned simultaneously, permitting more physical model parameterizations [13, 14]. We further demonstrate potential for generalization beyond training distributions, including moving-source scenarios, while maintaining orders-of-magnitude reductions in runtime relative to the full 3D numerical simulation (from hours to $<$ 1 min).

## 2 Problem formulation

Scalar transport is widely studied in urban environments (e.g. pollutant/chemical dispersion, airborne infectious disease). This physics can be encapsulated by the convection-diffusion equation (Eq. 1):

$$\frac{\partial c}{\partial t} = \nabla \cdot \Big( D \nabla c - \vec{v}\, c \Big) + R \tag{1}$$

where $c : \Omega \times [0, T] \to \mathbb{R}_{\geq 0}$ is the scalar quantity of interest in the spatio-temporal domain denoted by $\Omega \subset \mathbb{R}^3$ and $t \in [0, T]$. $D$ is the (potentially orthotropic) diffusivity tensor, $\vec{v}$ is the background velocity field (e.g. derived from CFD), and $R$ denotes sources and sinks in the domain. Initial

conditions $c(\cdot, 0) = c_0$ and boundary conditions are pre-specified (e.g. zero flux condition on walls). While $c(x, y, z, t)$ is a 3D scalar field, the proposed hybrid physics-neural model solves for $\bar{c}(x, y, t) = c(x, y, h, t)$ for a fixed height $h$ as denoted by $\Omega_h$ ($h = 1.5m$ in the rest of this work).

Projecting Eq. 1 onto the 2D plane yields an effective 2D transport equation:

$$\frac{\partial \bar{c}}{\partial t} = \nabla_{xy} \cdot \Big( \bar{D}\, \nabla_{xy} \bar{c} - \bar{\vec{v}}\, \bar{c} \Big) \; + \; \bar{R} \; + \; \underbrace{\mathcal{M}_\theta\big(\bar{c}_{0:t},\, \Phi\big)}_{\text{non-Markovian closure}} \;, \quad (x, y) \in \Omega_h,\; t \in (0, T], \tag{2}$$

where $\bar{D}$, $\bar{\vec{v}}$, and $\bar{R}$ are effective (plane-level) quantities, and $\mathcal{M}_\theta$ is a *history-dependent* closure term that accounts for unresolved effects (e.g., transport normal to the plane), which depends on the historical scalar field $\bar{c}_{0:t}$ and some parameters $\Phi$ (e.g. geometry encodings).

A Markovian module restricts $\mathcal{M}_\theta$ to depend only on $\bar{c}(\cdot, t)$ and limits long time horizon accuracy. In Section 4, we demonstrate that the learning of a non-Markovian closure model and the joint learning of $\bar{D}$ both significantly improve the model's prediction ability, such that qualitatively consistent simulations of out-of-distribution scenarios (e.g. moving source scenarios) can be obtained.

# 3 Method

## 3.1 Dataset

High-fidelity simulation data is obtained from a commercial software (ANSYS Fluent). Multiple convection-diffusion simulations are performed in a 3D indoor environment (with features such as air-conditioning, exhaust fans, and pillars) using the steady-state velocity $\vec{v}$ obtained by solving the governing equations with the imposed boundary conditions (constant outlet velocity at air-cons and zero flux condition on walls) and a constant passive scalar source term $R$ corresponding to a $0.1 \times 0.1 \times 0.1m^3$ volumetric source. The height of the entire 3D domain is 2.46m, which is sufficient for recirculation to occur and appear as non-Markovian effects that need to be captured in a reduced 2D model. The total simulated time is 600s (10min). A total of 53 full-field simulations, each with its own source ($R(x, y)$) were generated, and 26 were used for training. 2D plane-slices were extracted on a $49 \times 26$ structured grid ($\Delta x = \Delta y = 0.5m$), where each grid cell's value is the surface average concentration. The velocity fields ($\vec{v}$) and turbulent diffusivity fields ($D$) from the numerical model are provided in Appendix A.6 Figure 2.

## 3.2 Hybrid physics-neural model

The hybrid physics-neural model has a solver-in-loop setup [15] and is illustrated in Figure 1. Each time step involves a physics-based finite volume solver that computes the diffusion term $D(c^t; \Delta t)$ and advection term $A(c^t; \Delta t)$ from the current concentration field $c^t$, and a non-Markovian neural closure model for $R^t_{NN}$ that accounts for the unresolved physics in the form of an history-dependent change in concentration values. Given a specified release location, a corresponding source term $R$ is also provided. Details of each module are provided in A.1.

The finite volume method is used to solve the convection-diffusion equation. The explicit forward Euler method is used for time advancement while an upwind scheme is used for advection. An orthotropic diffusivity is assumed for computing the diffusion term. Numerical terms are computed using smaller time steps as required to avoid violation of the CFL condition.

Two states are advanced in parallel - a hidden state $h^t$ containing the history of the system, as well as the passive scalar concentration field $c^t$. The hidden state is used to account for unresolved multi-scale effects over multiple time steps [7]. As the concentration ($c^{t+\Delta t}$) is being updated every time step as per Eq. 3, an LSTM module is simultaneously also being updated to yield a history-aware hidden state as per Eq. 4, which is a crucial input to the non-Markovian neural closure model.

$$c^{t+\Delta t} = c^t + A(c^t, \Delta t) + D(c^t, \Delta t) + R\Delta t + R_{NN}(h^t; c^t) \tag{3}$$

$$h^{t+\Delta t} = M(h^t, c^{t+\Delta t}) \tag{4}$$

At each time, the hidden state is first decoded using transpose convolutional layers, which are then concatenated with the current concentration field. A convolutional neural network (CNN) is used to

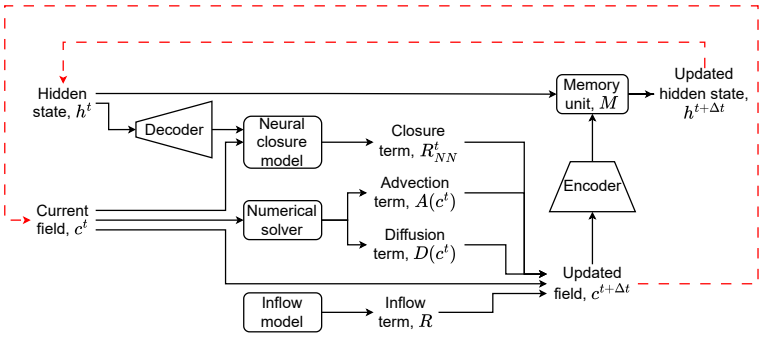


Figure 1: Schematic of time advancement scheme in the proposed hybrid physics-neural model. Dotted lines indicate the flow of hidden state and concentration field from each time step to the next.

compute the closure term due to its suitability for grid-like data and ability to extract local spatial information at various scales [16]. The closure term should also account for local effects, such as advection and diffusion normal to the plane. As such, local properties, i.e. velocity and learnt diffusivity fields, are also provided as inputs. The combination of the closure term, advection term, diffusion term, and source term gives the time-stepped concentration field, $c^{t+\Delta t}$, which is the input to the memory unit to get the time-stepped hidden state $h^{t+\Delta t}$. An LSTM model is used due to its well-established ability to model long-term dependencies [17].

### 3.3 Training

The trainable parameters in this project are the neural network parameters as well as the orthotropic diffusivity values. The diffusivity values are trained as they should reflect unresolved internal features (e.g. pillars) and is in line with the concept of 'coarse-grained' physics.

There are two main difficulties. Firstly, due to the large number of steps in the prediction sequence, unrolling the whole sequence of data for backpropagation through time would require a lot of RAM. Secondly, two parallel but inter-dependent sequences ($c^t$ and $h^t$) are trained simultaneously. Therefore, the 600s of flow time is broken into 60s segments. Except for the first 60s (for which the LSTM states and concentration fields are initialized as zeros), subsequent segments start from the predicted hidden state and concentration field from the previous segment. Within each segment, the prediction at time steps for which a desired concentration field is available in the training dataset is used to compute the loss. Additionally, curriculum learning [18] is employed to train the model with simulations of increasing duration. The model is trained for 500 epochs at each stage, using simulations of 60s, 120s, 300s, and 600s. More details on model training are in Appendix A.2.

Additional ablation studies are conducted to compare (i) the hybrid-neural model vs a data-driven model, and (ii) the non-Markovian vs Markovian model. The standard deviation across triplicate runs with different seeds are also presented as error bars in all subsequent plots. The dataset and the training code are available on https://github.com/IHPC-Physics-AI/Closure-for-Convection-Diffusion.

## 4 Results

**Hybrid physics-neural approach helps with data-efficient training and physical consistency.** As shown in Appendix A.3 Fig. 3, the data loss decreases with training. After training with all 600s of flow time, both the train and test data loss decreased, especially for the longer time rollouts. The initial high loss may be due to numerical errors that led to large predicted concentrations at low concentration regions, thereby increasing the relative error. Fig. 7 further shows that the model is able to start learning and improving, even with just 8 data points (i.e. is data efficient).

Qualitatively, the results are consistent with the physical system, e.g. learnt diffusivity for regions with walls is generally lower, as exemplified by the region circled in red in Fig. 4(d) which resembles the physical layout in Fig. 4(a). A vertical line of low diffusivity is also circled in Fig. 4(c) which is

related to an "air curtain"-like effect caused by the two air-cons nearby. while the closure term in Fig. 4(b) shows negative values at the air-con positions, which concurs with our understanding that air-cons lead to reduction in passive scalar concentration from the plane.

A purely data-driven model is also trained by removing the numerical solver and inflow module, and only retaining the closure term, i.e. $c^{t+\Delta t} = c^t + R^t_{NN}\Delta t$. The resultant loss is significantly higher (Fig. 5) and predictions are highly non-physical (Fig. 6). Results are in Appendix A.4.

**Including memory in the model improves prediction.** A purely Markovian model is tested by removing the hidden state and memory unit, such that the closure term only depends on the current scalar field. As per Appendix A.5, although the loss is comparable initially, the loss at later flow times is higher, as is consistent with the intuition that the memory effect is most obvious at longer time horizons. As observed from Figure 9, features are also significantly different for the predicted closure term, implying that memory is important for learning the non-Markovian effects.

**Joint learning of physical parameters improves performance.** We further compare the effect of joint learning of physical parameterizations such as diffusivity and velocity, and noticed that model performance is enhanced by incorporation of this component, as per Appendix A.6.

**The model is promising for out-of-distribution scenario.** The trained model is used to predict evolution with a moving source, unlike simulations in the training set with static release locations. The relative loss is much lower than a baseline method, where concentration is assumed homogeneous and equal to the average concentration throughout the whole simulation (Appendix A.7 Fig. 11). The prediction is also more meaningful as the model's prediction shows similar features to the numerical simulation (Fig. 12). There is a strong correlation (Spearman correlation coefficient $= 0.96$) between predicted concentration and actual concentration, even at the final time step (Fig. 13). This means the model can be used to identify regions of higher concentration.

## 5 Discussion and Limitations

We demonstrated that a hybrid physics-neural model can deliver fast, long time horizon predictions on a reduced domain (2D plane) while preserving physical consistency through a learned, non-Markovian neural closure model. The physics-neural design proved more data-efficient and physically consistent than its data-driven equivalent, and explicitly modeling memory improved accuracy versus Markovian baselines. Predictions retained meaningful structure and correlated strongly with ground truth, even on a moving source scenario (not in training data). Finally, the learned orthotropic diffusivity qualitatively aligns with aspects of the physical environment (e.g., walls/air-curtain–like regions), reinforcing the value of jointly learning model parameters with the non-Markovian closure.

However, some areas remain for further improvement. Early-time errors on the coarse-grained simulation remain large, which may have resulted from explicit, first-order updates. While it was hypothesized that the closure model may be able to compensate for these errors, the errors remain larger. Also, additional studies to evaluate the generalizability of this methodology (e.g., with different boundary conditions or other physics problems) would provide further insight into areas for improvement. Furthermore, the model's sensitivity to different parameters, such as LSTM size, memory horizon, or grid resolution (i.e. extent of coarse-graining), could all be studied to better understand the model's limitations and capacity for generalization. During the training of our model, a penalty (in the form of L2 regularization) is applied on the neural closure model to constrain the closure model and prevent over-fitting; however, the learned diffusivity and neural closure model remains highly under-constrained. Additional ablation studies may provide greater insight into their respective impact, and better identifiability of the contribution of each of the numerical and closure terms could inspire additional strategies to increase the data-efficiency and predictive performance of the proposed model.

## References


[1] E. J. Parish and K. Duraisamy, "Non-markovian closure models for large eddy simulations using the mori-zwanzig formalism," *Physical Review Fluids*, vol. 2, no. 1, p. 014604, 2017.

[2] T. Nakamura, K. Fukami, K. Hasegawa, Y. Nabae, and K. Fukagata, "Cnn-ae/lstm based turbulent flow forecast on low-dimensional latent space," 2020.

[3] G. Borrelli, L. Guastoni, H. Eivazi, P. Schlatter, and R. Vinuesa, "Predicting the temporal dynamics of turbulent channels through deep learning," *International Journal of Heat and Fluid Flow*, vol. 96, p. 109010, 2022.

[4] G. Karniadakis, Y. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, pp. 1–19, 05 2021.

[5] B. Sanderse, P. Stinis, R. Maulik, and S. E. Ahmed, "Scientific machine learning for closure models in multiscale problems: A review," *arXiv preprint arXiv:2403.02913*, 2024.

[6] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer, "Machine learning–accelerated computational fluid dynamics," *Proceedings of the National Academy of Sciences*, vol. 118, no. 21, p. e2101784118, 2021.

[7] H. A. Melchers, "Machine learning for closure models," Master's thesis, 2022.

[8] A. Gupta and P. F. Lermusiaux, "Generalized neural closure models with interpretability," *Scientific Reports*, vol. 13, Jun 2023.

[9] R. Maulik, O. San, A. Rasheed, and P. Vedula, "Subgrid modelling for two-dimensional turbulence using neural networks," *Journal of Fluid Mechanics*, vol. 858, pp. 122–144, 2019.

[10] S. Pan and K. Duraisamy, "Data-driven discovery of closure models," *SIAM Journal on Applied Dynamical Systems*, vol. 17, no. 4, pp. 2381–2413, 2018.

[11] N. Tathawadekar, N. Doan, C. Silva, and N. Thuerey, "Incomplete to complete multiphysics forecasting: a hybrid approach for learning unknown phenomena," *Data-Centric Engineering*, vol. 4, 11 2023.

[12] R. B. Ruiz, T. Marwah, A. Gu, and A. Risteski, "On the benefits of memory for modeling time-dependent pdes," *arXiv preprint arXiv:2409.02313*, 2024.

[13] N. Thuerey, P. Holl, M. Mueller, P. Schnell, F. Trost, and K. Um, *Physics-based Deep Learning*. WWW, 2021.

[14] H. Wu, F. Wen-Zhen, K. Qinjun, T. Wen-Quan, and Q. Rui, "Predicting effective diffusivity of porous media from images by deep learning," *Scientific Reports*, vol. 9, no. 20387, 2019.

[15] K. Um, R. Brand, Y. R. Fei, P. Holl, and N. Thuerey, "Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 6111–6122, Curran Associates, Inc., 2020.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. Burges, L. Bottou, and K. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[18] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, (New York, NY, USA), p. 41–48, Association for Computing Machinery, 2009.

[19] Y. Yin, I. Ayed, E. de Bézenac, N. Baskiotis, and P. Gallinari, "Leads: Learning dynamical systems that generalize across environments," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 7561–7573, Curran Associates, Inc., 2021.

# A Technical Appendices and Supplementary Material

## A.1 Architecture of neural network modules

This section details the modules composing the model outlined in Fig. 1 and their respective neural architectures. In all modules, LeakyReLu with negative gradient of 0.01 is used as the non-linear activation. KS refers to the kernel size used.

Briefly, the inflow module in Table 1 infers boundary inflows from a user-defined source location. The encoder in Table 2 compresses the current concentration field into a compact latent state while the decoder in Table 3 upsamples the hidden state to the actual physical field resolution so it can be concatenated with the current field variables. Separately, the LSTM in Table 4 ensures the long-time propagation of history across steps in the hidden state, a crucial feature for ensuring the non-Markovian neural closure model in Table 5 has sufficient information for incorporating unresolved physics.

Table 1: Inflow module architecture

| Layer no. | Layer type | Input shape | Specifications | Output shape |
|---|---|---|---|---|
| 1 | Conv2D | (26,49,7) | KS=(3,3) | (26,49,8) |
| 2 | MaxPool2D | (26,49,8) | KS=(2,3), strides=(2,3) | (13,16,8) |
| 3 | Conv2D | (13,16,8) | KS=(3,3) | (13,16,16) |
| 4 | MaxPool2D | (13,16,16) | KS=(2,3), strides=(2,3) | (6,5,16) |
| 5 | Conv2D | (6,5,16) | KS=(3,3) | (6,5,32) |
| 6 | MaxPool2D | (6,5,32) | KS=(2,2), strides=(2,2) | (3,2,32) |
| 7 | Conv2D | (3,2,32) | KS=(3,3) | (3,2,64) |
| 8 | MaxPool2D | (3,2,64) | KS=(3,2), strides=(3,2) | (1,1,64) |
| 9 | Fully Connected | (1,1,64) | features=64 | (1,1,64) |
| 10 | Fully Connected | (1,1,64) | features=64 | (1,1,64) |
| 11 | Fully Connected | (1,1,64) | features=4 | (1,1,4) |

Table 2: Encoder module architecture

| Layer no. | Layer type | Input shape | Specifications | Output shape |
|---|---|---|---|---|
| 1 | Conv2D | (26,49,3) | KS=(3,3) | (26,49,16) |
| 2 | Conv2D | (26,49,16) | KS=(3,3) | (26,49,16) |
| 3 | MaxPool2D | (26,49,16) | KS=(2,3), strides=(2,3) | (13,16,16) |
| 4 | Conv2D | (13,16,16) | KS=(3,3) | (13,16,32) |
| 5 | Conv2D | (13,16,32) | KS=(3,3) | (13,16,32) |
| 6 | MaxPool2D | (13,16,32) | KS=(2,3), strides=(2,3) | (6,5,32) |
| 7 | Conv2D | (6,5,32) | KS=(3,3) | (6,5,64) |
| 8 | Conv2D | (6,5,64) | KS=(3,3) | (6,5,64) |
| 9 | MaxPool2D | (6,5,64) | KS=(2,2), strides=(2,2) | (3,2,64) |
| 10 | Conv2D | (3,2,64) | KS=(3,3) | (3,2,128) |
| 11 | Conv2D | (3,2,128) | KS=(3,3) | (3,2,128) |
| 12 | MaxPool2D | (3,2,128) | KS=(3,2), strides=(3,2) | (1,1,128) |

Table 3: Decoder module architecture

| Layer no. | Layer type | Input shape | Specifications | Output shape |
|---|---|---|---|---|
| 1 | ConvTranspose | (1,1,512) | KS=(3,2), strides=(3,2) | (3,2,128) |
| 2 | Conv2D | (3,2,128) | KS=(3,3) | (3,2,128) |
| 3 | ConvTranspose | (3,2,128) | KS=(2,2), strides=(2,2) | (6,5,64) |
| 4 | Conv2D | (6,5,64) | KS=(3,3) | (6,5,64) |
| 5 | ConvTranspose | (6,5,64) | KS=(2,3), strides=(2,3) | (13,16,32) |
| 6 | Conv2D | (13,16,32) | KS=(3,3) | (13,16,32) |
| 7 | ConvTranspose | (13,16,32) | KS=(2,3), stride=(2,3) | (26,49,16) |
| 8 | Conv2D | (26,49,16) | KS=(3,3) | (26,49,16) |

Table 4: Memory unit architecture

| Layer no. | Layer type | Input shape | Specifications | Output shape |
|---|---|---|---|---|
| 1 | LSTMCell | (1,1,128) | (1,1,128) | (1,1,128) |
| 2 | LSTMCell | (1,1,128) | (1,1,128) | (1,1,128) |
| 3 | LSTMCell | (1,1,128) | (1,1,128) | (1,1,128) |
| 4 | LSTMCell | (1,1,128) | (1,1,128) | (1,1,128) |

Table 5: Neural closure module architecture

| Layer no. | Layer type | Input shape | Specifications | Output shape |
|---|---|---|---|---|
| 1 | Conv2D | (26,49,26) | KS=(3,3) | (26,49,26) |
| 2 | Conv2D | (26,49,26) | KS=(3,3) | (26,49,26) |
| 3 | Conv2D | (26,49,26) | KS=(3,3) | (26,49,1) |

### A.2 Training specifications

More details on the model training settings and hyperparameters are provided in this section. All models are implemented in JAX and trained on a single Nvidia RTX 3090 GPU card. The neural network weights are initialized randomly using Glorot uniform initializers, while the diffusivity values are initialized using the turbulent diffusivity values extracted from the 3D numerical simulation model as per Fig. 2 and trained jointly with the rest of the network. Optimization is performed using the Adam optimizer using a constant learning rate of $10^{-5}$. Each minibatch consists of 5 simulation segments of 60s flow time.

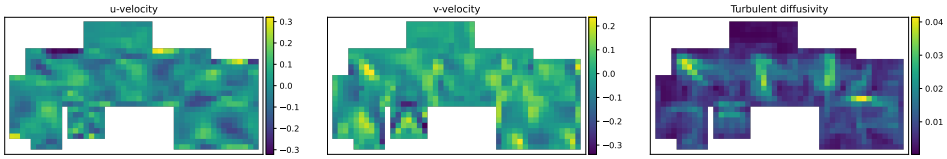


Figure 2: Velocity fields and turbulent diffusivity field (computed as $\nu_t/Sc_t$, where $\nu_t$ is the turbulent viscosity and $Sc_t = 0.9$ is the turbulent Schmidt number) as extracted from the full 3D numerical simulation

The combined loss function is calculated as per Eq. 5:

$$L = L_{data} + \lambda_1 |w|^2 + \lambda_2 \sum_{t=1}^{n_t-1} \sum_{i=0}^{n_x-1} \sum_{j=0}^{n_y-1} (u_{i,j}^t - u_{i,j}^{t-1})^2 \tag{5}$$

where the data loss $L_{data}$ is calculated as the pixel-wise square of the relative error of each cell:

$$L_{data} = \frac{1}{n_x n_y} \sum_{t=0}^{n_t-1} \sum_{i=0}^{n_x-1} \sum_{j=0}^{n_y-1} \left( \frac{u_{i,j}^t - \hat{u}_{i,j}^t}{\hat{u}_{i,j}^t + \varepsilon} \right)^2 I(u_{i,j}^t \text{ is avaliable}) \tag{6}$$

Here, $n_t, n_x, n_y$ are the number of time steps and number of pixels in the horizontal and vertical directions respectively. $u$ is the predicted concentration while $\hat{u}$ is the ground truth concentration. $I(\cdot)$ is the indicator variable giving 1 if the statement is true and 0 otherwise. As the pixel-wise concentration may be zero, a small constant of $\varepsilon = 10^{-9}$ is added in the denominator to prevent the loss term from diverging as the true value approaches zero. Additionally, L2 weight decay is applied to the neural closure model's weights $w$ with a constant factor of $\lambda_1 = 10^{-6}$. This is motivated by [19] where a penalty was imposed on the case-specific component to maximize the learning of the component that is shared across different cases, which in our case is the diffusivity value.

It is also desirable that predictions not oscillate. Therefore, a regularization term is imposed in terms of the pixel-wise sum of squares of the change between consecutive predictions with a constant factor of $\lambda_2 = 10^4$. The magnitude of $\lambda_2$ is relatively higher as the values of the scalar concentrations in this problem are on the order of $10^{-5}$ to $10^{-6}$.

### A.3 Baseline model performance

Overall, the results show that the proposed model can learn well from the relatively small training dataset and produces physically sensible parameters and residuals.

Curriculum training consistently reduces error for the longer rollout times, especially up to 600s, with improving generalization as evidenced by the decrease in test error in Fig. 3. The plots also illustrate relatively higher early-time relative error ($L_{data}$) due to the smaller denominators before the scalar spreads more widely.

Qualitatively, the model learns diffusivity maps that line up with room structure as illustrated in Fig. 4. We observe lower diffusivity near walls/pillars and an "air-curtain–like" region between air-conditioners and a small closure term where plane-normal transport due to the air-conditioning unit lowers the concentration, matching physical intuition.

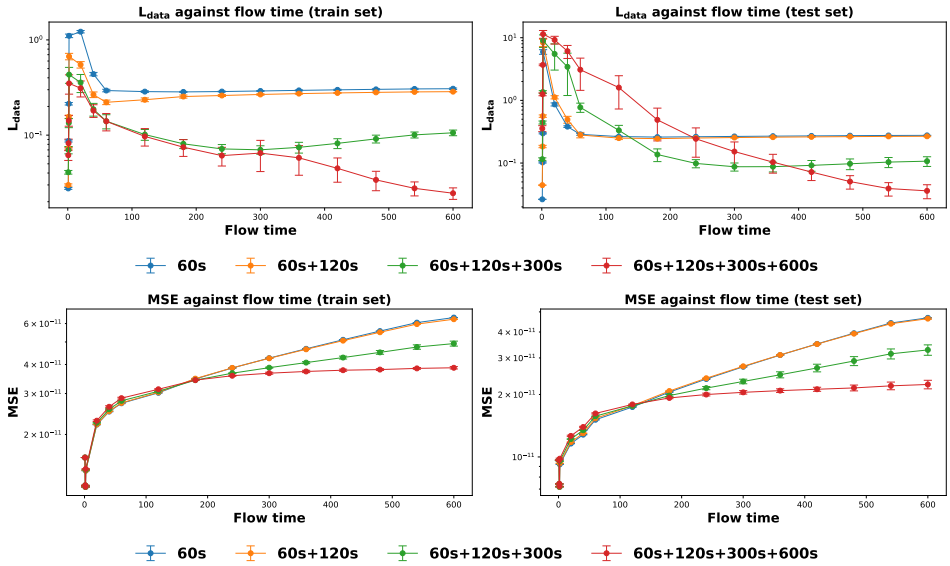


Figure 3: Comparison of $L_{data}$ and MSE against flow time for the (Left) training and (Right) test datasets, with different lines indicating results after each stage of curriculum learning (i.e. after being trained with 60s, 120s, 300s, 600s of flow time respectively). Loss is averaged over all different release locations of the training and test dataset respectively.

Table 6: $L_{data}$ and MSE for the training and test datasets at 4 different simulation times (60s, 120s, 300s, 600s) after training

| | 60s | 120s | 300s | 600s |
|---|---|---|---|---|
| $L_{data}$ (train set) | $0.140$ $\pm 0.029$ | $0.097$ $\pm 0.020$ | $0.065$ $\pm 0.023$ | $0.025$ $\pm 0.003$ |
| $L_{data}$ (test set) | $3.084$ $\pm 1.639$ | $1.601$ $\pm 0.872$ | $0.153$ $\pm 0.064$ | $0.036$ $\pm 0.009$ |
| MSE (train set) | $2.866 \times 10^{-11}$ $\pm 2.568 \times 10^{-13}$ | $3.130 \times 10^{-11}$ $\pm 1.789 \times 10^{-13}$ | $3.669 \times 10^{-11}$ $\pm 3.591 \times 10^{-13}$ | $3.872 \times 10^{-11}$ $\pm 4.486 \times 10^{-13}$ |
| MSE (test set) | $1.621 \times 10^{-11}$ $\pm 1.072 \times 10^{-13}$ | $1.791 \times 10^{-11}$ $\pm 7.060 \times 10^{-14}$ | $2.048 \times 10^{-11}$ $\pm 3.944 \times 10^{-13}$ | $2.235 \times 10^{-11}$ $\pm 1.060 \times 10^{-12}$ |

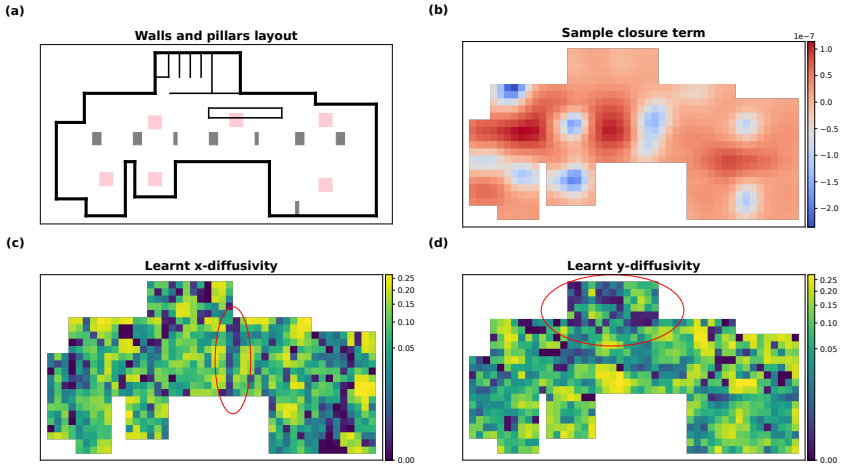


Figure 4: Learned diffusivity with some learned features circled in red and learned closure term. (a) Layout of room, with air-cons and pillars labeled as red and grey boxes respectively. (b) Sample closure term from a time point towards the end of the simulation time. Learned diffusivity along (c) x-direction and (d) y-direction. Yellow, green and blue indicate areas of high, medium and low diffusivity respectively. Low diffusivity regions (e.g. blue-black regions) can result if there are structural obstructions (e.g. internal pillars or walls).

### A.4 Physics-informed vs data-driven approach

We contrast the results from the full physics-neural model with a solely data-driven model in this section. The physics-neural model maintains low, steadily improving errors across long time horizons (Fig. 5 and yields physically consistent spatio-temporal fields as in Fig. 6. In contrast, the data-driven baseline shows large early-time relative errors and non-physical prediction fields. Even under further data scarcity (8 training cases), the physics-neural model exhibits improved prediction relative to the solely data-driven model as illustrated in Fig. 7.

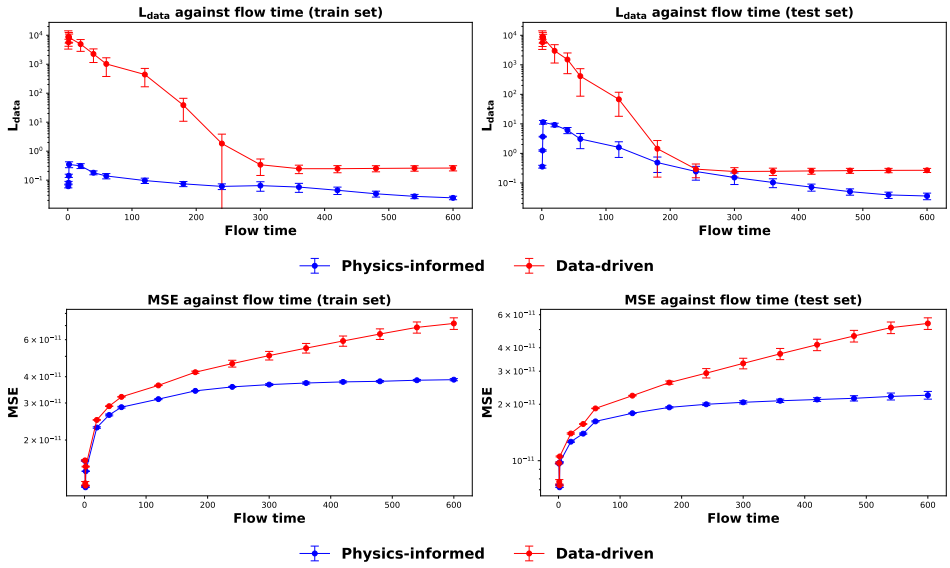


Figure 5: Comparison of $L_{data}$ and MSE against flow time for the (Left) training and (Right) test datasets, based on the physics-neural model and a solely data-driven approach. Errors are averaged over different release locations of the training and test datasets.

After training, the MSE from the physics-neural model at the end of 600s is approximately half that of the data-driven model, as detailed in Table. 7.

Table 7: Test $L_{data}$ and MSE for predictions using the physics-informed approach and data-driven approach at 4 different simulation times (60s, 120s, 300s, 600s) after training

| | 60s | 120s | 300s | 600s |
|---|---|---|---|---|
| Test $L_{data}$ (Physics-informed) | $3.084$ $\pm 1.639$ | $1.601$ $\pm 0.872$ | $0.153$ $\pm 0.064$ | $0.036$ $\pm 0.009$ |
| Test $L_{data}$ (Data-driven) | $413.118$ $\pm 326.486$ | $67.917$ $\pm 49.867$ | $0.243$ $\pm 0.085$ | $0.268$ $\pm 0.042$ |
| Test MSE (Physics-informed) | $1.621 \times 10^{-11}$ $\pm 1.072 \times 10^{-13}$ | $1.791 \times 10^{-11}$ $7.060 \times 10^{-14}$ | $2.048 \times 10^{-11}$ $3.944 \times 10^{-13}$ | $2.235 \times 10^{-11}$ $1.060 \times 10^{-12}$ |
| Test MSE (Data-driven) | $1.900 \times 10^{-11}$ $\pm 4.931 \times 10^{-14}$ | $2.222 \times 10^{-11}$ $\pm 8.402 \times 10^{-14}$ | $3.302 \times 10^{-11}$ $\pm 2.146 \times 10^{-12}$ | $5.392 \times 10^{-11}$ $\pm 3.834 \times 10^{-12}$ |

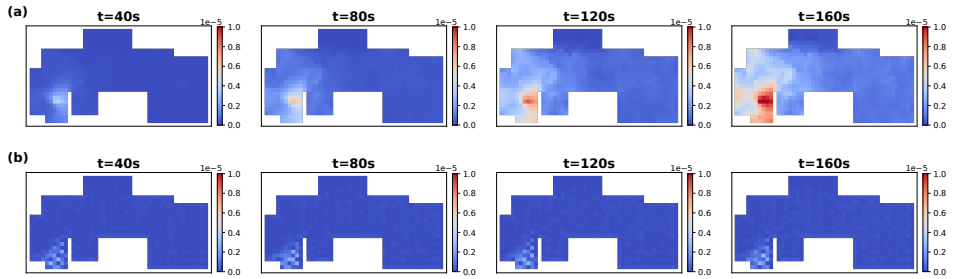


Figure 6: Comparison of simulation predictions on a sample test data using (a) the physics-informed approach and (b) the data-driven approach.

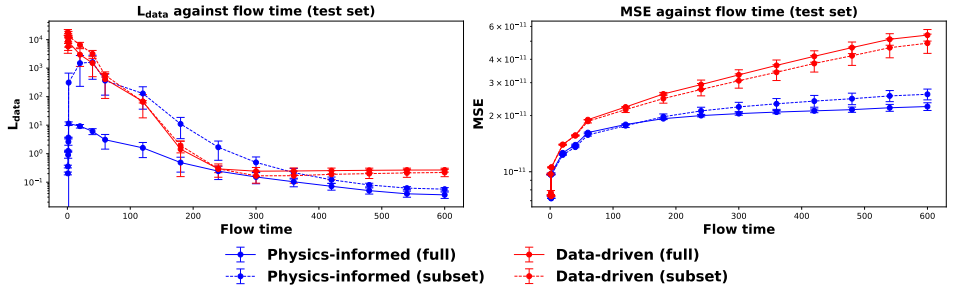


Figure 7: Comparison of $L_{data}$ and MSE against flow time on the test dataset when each model is trained using the full training dataset (26 simulations) or a smaller, randomly selected subset of the training dataset (8 simulations).

### A.5 Non-Markovian vs Markovian approach

Similarly, we compare the relative performance of the model when the history-aware component is removed from the model. In general, removing memory (i.e. a purely Markovian closure) yields competitive early-time losses but degrades at longer time horizons, where dependence on the historical trajectory is increasingly critical. The error curves separate as simulation time increases in Fig. 8, and closure visualizations in Fig. 9 show that the non-Markovian model produces coherent, trajectory-consistent closure patterns, while the Markovian equivalent does not exhibit any similar structure. This confirms the central claim as suggested by the Mori-Zwanzig formalism that memory is necessary to approximate unresolved effects, especially for longer time rollouts.

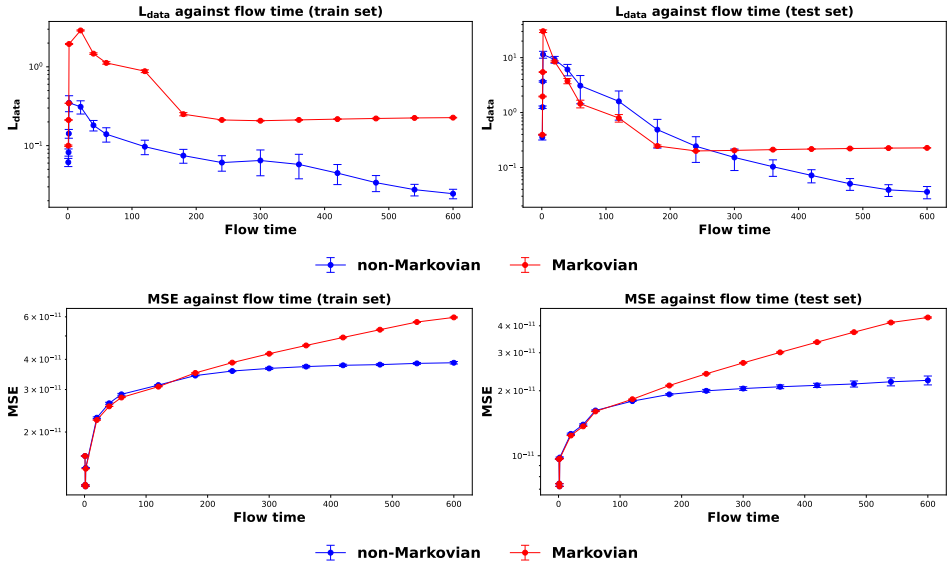


Figure 8: Comparison of $L_{data}$ and MSE against flow time for the (Left) training and (Right) test datasets, using the non-Markovian and Markovian models respectively. Metrics are averaged over different release locations in the training and test datasets.

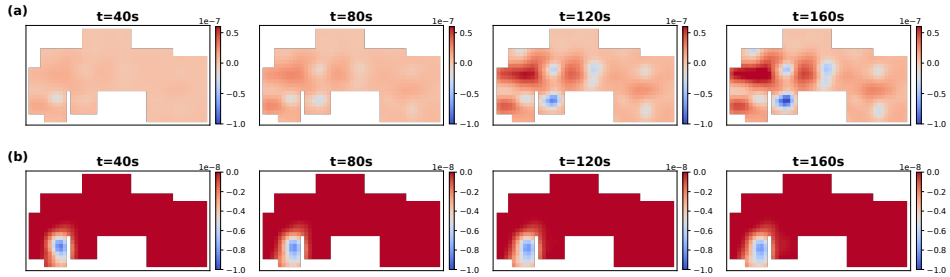


Figure 9: Comparison of predicted closure term on a sample test data using the (a) non-Markovian approach and (b) Markovian approach.

Additional quantitative MSE metrics are provided in Table 8.

Table 8: Test $L_{data}$ and MSE for predictions using the non-Markovian and Markovian approach at 4 different simulation times (60s, 120s, 300s, 600s) after training

| | 60s | 120s | 300s | 600s |
|---|---|---|---|---|
| Test $L_{data}$ (non-Markovian) | $3.084$ $\pm 1.639$ | $1.601$ $\pm 0.872$ | $0.153$ $\pm 0.064$ | $0.036$ $\pm 0.009$ |
| Test $L_{data}$ (Markovian) | $1.449$ $\pm 0.231$ | $0.795$ $\pm 0.125$ | $0.205$ $\pm 0.003$ | $0.228$ $\pm 0.002$ |
| Test MSE (non-Markovian) | $1.621 \times 10^{-11}$ $\pm 1.072 \times 10^{-13}$ | $1.791 \times 10^{-11}$ $\pm 7.060 \times 10^{-14}$ | $2.048 \times 10^{-11}$ $\pm 3.944 \times 10^{-13}$ | $2.235 \times 10^{-11}$ $\pm 1.060 \times 10^{-12}$ |
| Test MSE (Markovian) | $1.604 \times 10^{-11}$ $\pm 3.221 \times 10^{-14}$ | $1.827 \times 10^{-11}$ $\pm 4.726 \times 10^{-14}$ | $2.693 \times 10^{-11}$ $\pm 1.253 \times 10^{-13}$ | $4.371 \times 10^{-11}$ $\pm 2.729 \times 10^{-13}$ |

### A.6 Joint learning of different physical parameterizations

This section further isolates the benefit of jointly learning diffusivity and/or velocity parameters with the neural closure model. From Fig. 2, we observe that learning neither greatly degrades

performance. In contrast, the joint learning of both parameters yields the best improvement in long time horizon prediction. Joint learning of these parameters matter because they allow the model to implicitly encode for obstacles and flow behavior that were previously explicitly captured in the original CFD-derived quantities. Hence, allowing joint physics parameter learning can reduce the learning burden on the neural closure model, thereby improving stability and extrapolation.

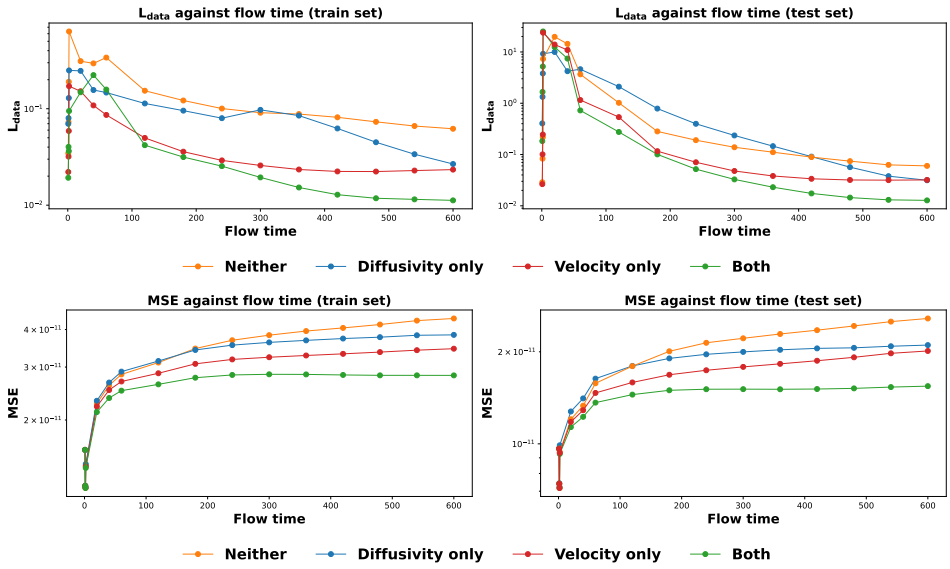


Figure 10: Comparison of $L_{data}$ and MSE against flow time for the (Left) training and (Right) test datasets, for models where different physical terms (diffusivity and/or velocity) are jointly learned with the non-Markovian neural closure model.

### A.7 Moving source scenario

In this section, we investigate the ability of the trained model to generalize to new scenarios. In all previous training data simulations, the source was static and remained at the same location for all simulation time (0 to 600s). In contrast, this scenario involves a dynamically moving source.

Against a simple baseline, the pre-trained model can still achieve fairly low relative errors as shown in Fig. 11. The similarity in MSE is largely due to the difference in magnitudes between the prediction and the ground truth. Critically, the pre-trained model can reproduce salient field features as seen in the CFD snapshots in Fig. 12. The final-time Spearman correlation of $\approx 0.96$ in Fig. 13 further highlights the ability for the model to retain the right physical structure in the evolved field, although the predictive performance can still be further improved. This experiment suggests that this methodology can potentially yield more generalizable models while using a small amount of training data (26 simulations in this instance).

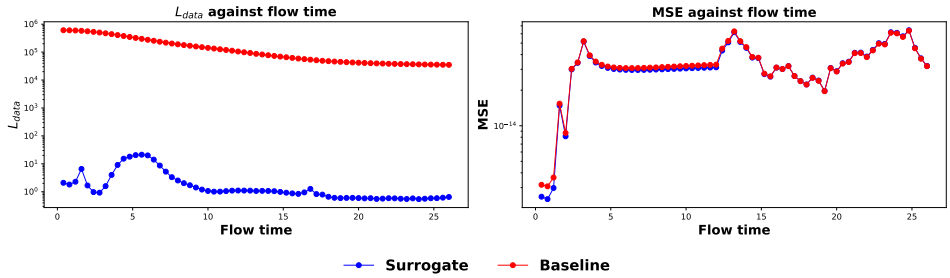


Figure 11: $L_{data}$ and MSE for a moving source scenario comparing predictions from the pre-trained surrogate model and a baseline where the concentration is assumed to be homogeneous and equal to the average concentration throughout the whole simulation.

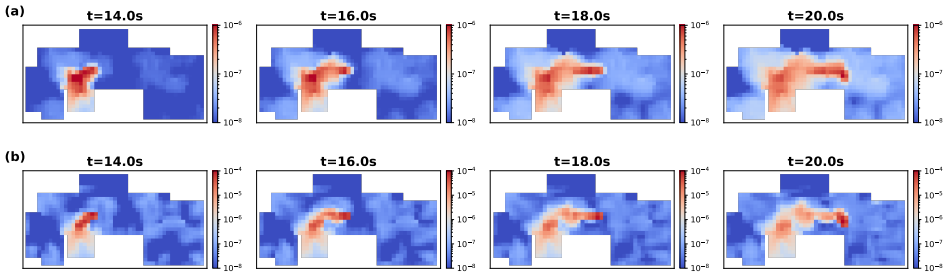


Figure 12: Snapshots of concentration field at different time steps (a) predicted by the physics-neural model (b) simulated by CFD. Note that the colors are on different scales.

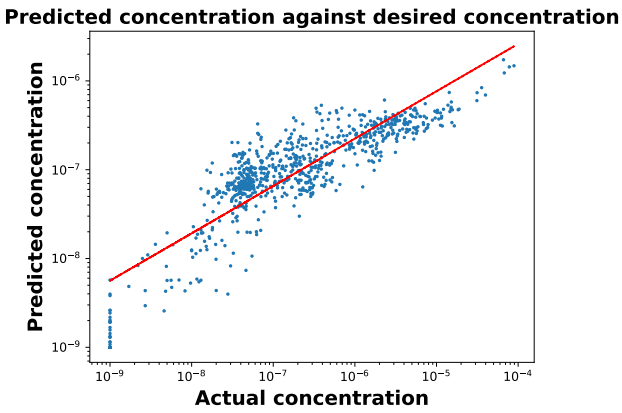


Figure 13: Comparison between concentration predicted by the surrogate model and the actual concentration at the final time step of the scenario of a moving source scenario. The Spearman correlation coefficient was calculated and found to be $0.957$. The red line shows the best linear fit on the log-log scale.