GEOMETRY-AWARE LATENT AUTOREGRESSIVE GENERATIVE MODEL FOR PDES IN COMPLEX DOMAINS

Anonymous authors
Paper under double-blind review

ABSTRACT

Solving multiphysics partial differential equations (PDEs) remains a major challenge in scientific computing, especially for highly complex μm-scale tortuous geometries critical to energy and chemical engineering. We address this challenge by proposing a Geometry-aware Latent Autoregressive generative Model for PDEs (GeoLAMP), which solves physics within highly irregular and tortuous structures by decoupling flow and transport physics. GeoLAMP introduces a dual-encoder architecture on graph representations to jointly capture global topology and fine-scale geometric features, enabling an effective transition from real-space fields to compact latent representations. In the latent space, we propose a causal self-attention transformer with flow matching to model temporal dynamics, allowing stable and scalable block-wise autoregressive prediction. In addition, we propose a grid–graph data fusion scheme that projects low-resolution grid-based approximate priors onto graph representations, improving prediction of flow in tortuous structures. We establish three multiphysics benchmark datasets in complex geometries, covering reactive flow, heat convection, and elasticity. GeoLAMP consistently achieves the most stable autoregression performance on these datasets. Our results provide a systematic study of geometry-aware learning for PDEs in μm-scale complex geometries and offer new insights into block-wise time marching of latent autoregressive PDE modeling via a flow matching framework.

1 INTRODUCTION

Scientific machine learning for partial differential equations (PDEs) has gained substantial attention and has multiple methodological directions, ranging from physics-informed fitting (Raissi et al., 2019; Karniadakis et al., 2021; Pang et al., 2019) to operator learning (Li et al., 2021; Lu et al., 2021; Li et al., 2023a; Azizzadenesheli et al., 2024) and latent modeling (Wu et al., 2022; Iakovlev et al., 2023; Kontolati et al., 2024). It shows strong potential to reshape conventional computational workflows in many scientific and engineering scenarios, including weather forecasting (Chen et al., 2024), turbulence reconstruction (Wang et al., 2020), and geological engineering (Wen et al., 2022). However, compared with the widely studied physical problems, multiphysics processes occurring inside complex meso-scale structures (μm) have received far less attention (Chen et al., 2022). More importantly, meso-scale physics often governs macroscopic scientific and engineering behavior, such as natural hydrogen (Mathur & Mukerji, 2025), chip cooling (Van Erp et al., 2020) and battery design (Lu et al., 2020). This cross-scale dependence poses significant computational challenges. Therefore, it is necessary to develop and apply deep learning methods specifically tailored to physical systems defined on meso-scale complex geometries (Li et al., 2013; Agrawal et al., 2001).

To handle complex geometries, several machine-learning-based studies have proposed efficient solutions by incorporating geometric information into model architectures (Bronstein et al., 2021; Brandstetter et al., 2022; Takamoto et al., 2022). Point-cloud methods represent complex geometries as unordered sets of spatial points, thereby avoiding the need for structured grids (Qi et al., 2017; Li et al., 2023a; Wu et al., 2024; Kashefi & Mukerji, 2021; Liang & Zhao, 2013; Wang et al., 2019). Graph-based methods further augment these representations with explicit neighborhood connectivity, enabling message passing or graph-kernel operators to model local physical interactions and long-range geometry-dependent transport (Li et al., 2020; 2023c; Pfaff et al., 2021; Sanchez-Gonzalez et al., 2020; Chung et al., 2024). More recently, GeoPT (Wu et al., 2026) introduces lifted geometric pre-training, which augments large-scale geometry data with synthetic dynamics to bridge

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

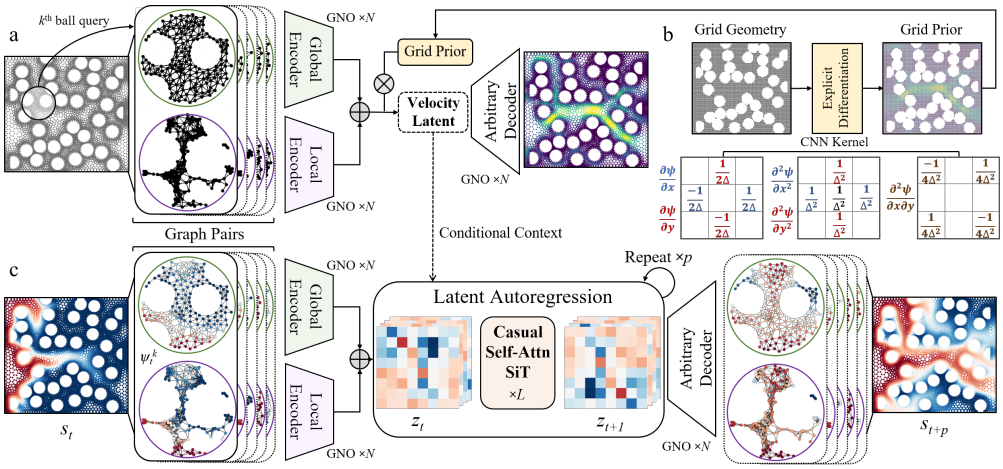


Figure 1: Schematic of GeoLAMP architecture. (a) Prediction of tortuous fluid flow on graph pairs. (b) Grid-based flow prior generation. (c) Autoregression of time-evolved physics.

the gap between static shape representation and physics simulation. However, real-space geometric complexity increases dramatically with domain size and problem scale, especially for meso-scale structures represented by millions of voxels.

To address this difficulty, latent-space methods (Gao et al., 2026; Li et al., 2025; Wu et al., 2022; Kontolati et al., 2024) encode real-space data into compact latent representations and model temporal evolution in latent space, leading to substantially improved efficiency and flexibility through compression. For complex domains, latent modeling is more appealing if the geometry-aware backbones can compress irregular point clouds or meshes into regular and compact latent representations while preserving real-space information (Wang et al., 2019; Li et al., 2020). Meso-scale geometries characterized by highly tortuous and heterogeneous structures calls for more reffned geometry-aware encoder–decoder designs that can compress complex domains while preserving geometry-sensitive physical information.

Once the dynamics are represented in a latent space, diffusion-based generative modeling (Ho et al., 2020; Lipman et al., 2023; Peebles & Xie, 2023; Liu et al., 2023b; Nichol & Dhariwal, 2021; Song et al., 2021b;a; Karras et al., 2022) provides a principled paradigm for accurate and stable temporal prediction. At scale, latent tokenization and transformer backbones have further shown that high-dimensional generation becomes more efficient when the model operates in a compact latent space (van den Oord et al., 2017; Esser et al., 2021; Rombach et al., 2022; Peebles & Xie, 2023; Ma et al., 2024). These ideas are increasingly relevant to scientific surrogate modeling (Lippe et al., 2023; Hao et al., 2024; Li et al., 2025).

In this study, we propose a **Geo**metry-aware **L**atent **A**utoregressive generative **M**odel for **P**DEs (GeoLAMP) to learn time-dependent multiphysics behavior in complex meso-scale geometries through latent representations, as shown in Fig. 1. Specifically, GeoLAMP consists of three main components: (i) Global and Local Encoders (GE & LE) uses multiple graph neural operator (GNO) layers to encode global/local information from graph pairs of query points sampled by the farthest-point sampling (FPS) and covariance-based sampling (CBS), separately; (ii) Causal Self-attention Transformer performs block-wise temporal autoregression by applying masked self-attention over timestep tokens; and (iii) A grid–graph fusion scheme embeds a convolution-derived velocity prior into the graph structure, enhancing prediction by embedding physical priors. GeoLAMP first encodes real-space information into a latent space using GE and LE. The velocity field is then predicted from pure geometry and prior information, and the temporal dynamics are autoregressively generated from the initial latent state conditioned on it. Finally, the generated latent representations are decoded back to real space at arbitrary spatial locations and time points. We constructed three meso-scale physics datasets. GeoLAMP achieves the most stable autoregression behavior on them, maintaining low errors throughout the entire rollout horizon.

Contributions (i) A dual-GNO encoder combines farthest-point coverage with covariance-selected interior detail on irregular meshes, while a coarse-grid flow prior supports geometry-to-velocity

prediction. (ii) A target-isolated causal mask lets flow matching generate an entire latent block in parallel, with autoregression across committed blocks. Shared frame-wise parameters allow a frozen model to vary both history length C and one-call output length P by rebuilding the mask; Section 5.3 quantifies the accuracy and serial-call tradeoff. (iii) We evaluate the design on reactive flow, heat convection, and elasticity in complex geometries, including geometry baselines and a matched direct-regression control.

2 Related Work

Existing datasets for learning PDE dynamics fall short of the regime we target on at least one axis as shown in Table 1. Our three datasets occupy exactly this intersection of tortuous volumetric domains, time-dependent rollout and multiphysics coupling, and this regime motivates two design choices that distinguish GeoLAMP from prior work.

Table 1: Comparison of widely used AI-for-PDE datasets with ours.

Dataset	Complex geometry	Time-dependent	Multi-physics
FNO suite - NS (Li et al., 2021)	×	✓	×
PDEBench (Takamoto et al., 2022)	×	✓	×
The Well (Ohana et al., 2024)	×	✓	✓
Geo-FNO Elasticity (Li et al., 2023b)	✓ (Single object)	×	×
Geo-FNO Airfoil/Pipe (Li et al., 2023b)	✓ (Single object)	×	×
ShapeNet-Car (Umetani & Bickel, 2018)	✓ (Single object)	×	×
AirfRANS (Bonnet et al., 2022)	✓ (Single object)	×	×
Ahmed-Body (Li et al., 2023c)	✓ (Single object)	×	×
Ours (3 datasets)	✓ (Tortuous domain)	✓	✓

The first concerns geometry representation. Existing geometry-aware solvers convert an irregular shape into a grid-compatible form: Geo-FNO (Li et al., 2023b) learns a diffeomorphic deformation to a latent regular grid, NUNO (Liu et al., 2023a) partitions the domain into individually gridded subdomains, GINO (Li et al., 2023c) projects the mesh onto a regular latent grid through GNO layers, and attention-based solvers (Li et al., 2023a; Wu et al., 2024) operate directly on mesh points or learned slices. These designs target domains dominated by a single smooth boundary such as an airfoil, a car body and a pipe, whereas tortuous meso-scale domains are multiply connected with tens of internal boundaries. GeoLAMP therefore represents the whole volumetric domain rather than a surface: global encoder provides globally uniform coverage, and local encoder densifies pore and channel interiors. Together, the dual encoder effectively recover the physics in tortuous domains.

The second concerns temporal prediction. Building on latent diffusion and flow matching (Rombach et al., 2022; Peebles & Xie, 2023; Ma et al., 2024; Lipman et al., 2023), denoising-based rollout improves long-horizon stability over direct regression for PDEs (Lippe et al., 2023; Hao et al., 2024; Rozet et al., 2025); the latent flow-matching solver of Li et al. (2025), the closest generative relative, performs next-step generation on a regular latent grid. GeoLAMP-B instead generates a block of P frames in one flow-matching solve, each noisy target group attending to all context groups but not to other targets, and shared frame-wise parameters let a frozen checkpoint change the context and block lengths at inference by rebuilding the causal mask (Section 5.3). Together, the domain-level encoding and block-wise latent generation address the regime of Table 1 that prior benchmarks and solvers do not cover.

3 Problem Setup in Complex Geometry

We focus on solving multiphysics PDEs with K solution variables, denoted by $s^K(x,t)$, in a complex domain $\Omega \subseteq \mathbb{R}^d$ over time $t \in [0,T]$, subject to boundary conditions $BC(x,t)$ and initial conditions $IC(x)$, as defined in Eq. 1. Herein, $\mathcal{L}a$ denotes a physics-dependent differential operator parameterized by coefficients or source terms a. We construct three datasets covering different physical processes and meso-scale structures: flow-reactive transport in circular-packed porous structures, convective heat transfer in stochastic-field-generated structures, and elasticity in

foam structures. All datasets are generated using the finite element method (FEM) in COMSOL (Multiphysics, 1998), with detailed descriptions provided in Appendix A.

$$\begin{aligned} &\mathcal{L}_a \circ s^K(x,t) = 0, && x \in \Omega \\ &s^K(x,t) = BC(x,t), && x \in \partial\Omega,\ t \in [0,T], \\ &s^K(x,0) = IC(x), && x \in \Omega \end{aligned} \tag{1}$$

4 Geometry-aware latent generative model

4.1 Encoder and decoder

The physical information in real space is organized from unstructured mesh outputs as point sets, denoted by $x_i, s_i^K i = 1^N$ at time t. Since the full point set is typically dense and computationally expensive, directly mapping all real-space points to latent representations is inefficient. To address this, we design global and local encoders (GE & LE) that compress the full field through tailored point selection while preserving as much geometry-sensitive physical information as possible. The arbitrary decoder maps the generated latent representations back to real-space points at arbitrary resolutions. All of them are built upon graph neural operators (GNOs) (Li et al., 2020), which aggregate information from points within a query ball B of radius r_b and map it onto a regular latent grid $y_j, g_j^{D\,M}{}_{j=1}$ at time t through Eq. 2, following a Green's function approximation.

$$g_j(y^0) \approx \sum_{x \in B} \kappa(x, y_j^0)\, s(x)\, \mu(x) \tag{2}$$

Then, followed with multiple GNO blocks, the real space information is compressed to latent representations, where in each layer, the approximation follows

$$g_j(y^l) \approx \sum_{y_j^{l-1} \in B^l} \kappa(y^l, y_j^{l-1})\, s(y_j^{l-1})\, \mu(y_j^{l-1}) \tag{3}$$

In order to comprehensively capture the global and local information, GE and LE takes different embedded points from real space, and they are trained together as a variational auto-encoders (VAE). The GE&LE selections are shown in Appendix B.

Global encoder Given the full input point cloud $\{x_i\}_{i=1}^N$, where $x_i \in \mathbb{R}^d$, GE automatically selects n_g representative points using the farthest point sampling (FPS) (Moenning & Dodgson, 2003) as the input. Starting from an initial point x_{i_1}, the sampled set at step t is denoted as $S_t = \{x_{i_1}, x_{i_2}, \ldots, x_{i_t}\}$, at each iteration, the next point is selected by maximizing its minimum Euclidean distance to the existing sampled set:

$$x_{i_{t+1}} = \arg\max_{x \in P \setminus S_t} \left(\min_{x_j \in S_t} \|x - x_j\|_2 \right). \tag{4}$$

This greedy procedure approximately solves the max–min sampling objective:

$$\max_{S \subset P,\ |S| = n_g} \ \min_{x \in P} \min_{x_j \in S} \|x - x_j\|_2. \tag{5}$$

Therefore, GE intends to cover the full spatial domain and avoids local clustering, providing a robust geometric representation for global information, including inlet, outlet and boundary nodes. It captures domain-scale trends from inlet to outlet and along the boundary.

Local encoder From the points not selected by the global encoder, the local encoder (LE) selects n_l additional points to resolve pore and channel interiors. We use covariance-based sampling (CBS): a neighborhood score derived from the point covariance favors locally isotropic point distributions that complement the global FPS set. Given a point set $\{x_i\}_{i=1}^N$, where $x_i \in \mathbb{R}^d$, for each point x_i, its k-nearest neighbors are first constructed as

$$\mathcal{N}_k(x_i) = \{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}. \tag{6}$$

The local geometric variation around x_i is characterized by the covariance matrix of neighbor points:

$$\mathbf{C}_i = \frac{1}{k} \sum_{x_j \in \mathcal{N}_k(x_i)} (x_j - \bar{x}_i)(x_j - \bar{x}_i)^\top, \tag{7}$$

where the local centroid is defined as

$$\bar{x}_i = \frac{1}{k} \sum_{x_j \in \mathcal{N}_k(x_i)} x_j. \tag{8}$$

By performing eigenvalue decomposition on $\mathbf{C}_i$, we obtain $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_d$, and define a local neighborhood-isotropy score as

$$\kappa_i = \frac{\lambda_1}{\sum_{m=1}^{d} \lambda_m}. \tag{9}$$

A larger κ_i indicates a more isotropic local neighborhood; it does not directly measure boundary curvature. LE selects the n_l highest-scoring points among those not chosen by FPS. This densifies interior pores while the FPS branch retains domain-wide coverage, as illustrated in Appendix B.

Arbitrary decoder The arbitrary decoder (AD) is designed to decode the generated latent representations back to real-space points at arbitrary locations and time steps. To handle meshes with variable numbers of nodes, we introduce a simple masking strategy into the GNO block, where information is propagated only through valid graph nodes while empty nodes are masked out. Moreover, the full field can be reconstructed at any desired resolution through iterative decoding with AD.

4.2 Flow matching and autoregressive generation

After mapping the original physical-space states into a compact latent token space, we apply flow matching to learn a transport from a Gaussian distribution to the latent data distribution, enabling autoregressive generation in the latent space. Specifically, we develop two prediction methods by modifying the scalable interpolant transformer (Ma et al., 2024) (SiT) backbone, as illustrated in Fig. 2. In the first method, the previous two latent states are patchified into tokens and used as conditional context through the cross-attention mechanism of the transformer block for next-step prediction. In addition, auxiliary information, in particular the velocity field predicted by the preceding flow stage, is incorporated into the conditional context, providing the model with richer physics-relevant guidance. We refer to this method as GeoLAMP-S in Fig. 2(a), where "S" denotes single-step prediction. In the second method, GeoLAMP-B, each conditioning frame and each noisy target frame is patchified into a spatial token group. We train with $M = 8$ conditioning frames and $M = 8$ target frames. The context-to-target mask lets conditioning groups attend to all conditioning groups; each target group attends to every conditioning group and its own spatial tokens, but not to other target groups (Fig. 2(b)). Thus each flow-matching vector-field evaluation covers all M future latent frames in parallel. Autoregression occurs between completed blocks: committed predictions become context for the next block. The frame-wise embedders, transformer layers, and output head are reused across groups, so inference with different conditioning and output lengths changes the token stream and parameter-free mask without changing learned weights. Section 5.3 measures the resulting accuracy tradeoff. Further flow-matching details are in Appendix C.

4.3 Grid-based flow prior

We compute a cheap, low-resolution flow field on a regular grid and fuse it into the encoder, as illustrated in Fig. 1(b). Given a binary structure at resolution G, we solve the incompressible Navier–Stokes equations explicitly using *fixed* convolution kernels rather than learned ones. Spatial derivatives of any field ψ on the grid are realized by three second-order stencils applied as convolutions,

$$\partial_x \psi \approx K_x * \psi, \qquad \partial^2_{xx} \psi \approx K_{xx} * \psi, \qquad \partial^2_{xy} \psi \approx K_{xy} * \psi, \tag{10}$$

with the central difference $K_x = [-1, 0, 1]/2\Delta$, the Laplacian stencil $K_{xx} = [1, -2, 1]/\Delta^2$, and the four-corner mixed stencil $K_{xy} = [\pm 1]/4\Delta^2$, and their transposes in y. Applied to u, v and p, these kernels yield all advective, viscous and pressure terms as batched convolutions, and the fields are marched explicitly to steady state on the coarse grid at negligible cost relative to the full simulation (details in Appendix D.1). The converged field is compressed into a prior tensor Φ carrying the two velocity components, the velocity magnitude and the pressure. Because averaging across a wall would invent flow through solid, Φ is downsampled to the 32×32 latent grid by a *mask-aware* pooling that

averages over fluid cells only,

$$\mathrm{pool}(\Phi)_j = \frac{\sum_{c\in\mathcal{C}_j} \Phi_c\, m_c}{\sum_{c\in\mathcal{C}_j} m_c}, \qquad \phi_j = \frac{1}{|\mathcal{C}_j|}\sum_{c\in\mathcal{C}_j} m_c, \tag{11}$$

where $\mathcal{C}_j$ are the fine cells pooled into latent cell j and $m_c \in \{0,1\}$ is the fluid indicator. The pooled prior is concatenated onto the grid features produced by the GNO encoders and mixed by convolution.

$$\tilde{g}_j = W\big[\, g_j \,\|\, \mathrm{pool}(\Phi)_j \,\|\, \phi_j \,\big], \tag{12}$$

This grid–graph fusion supplies the global pressure and flow structure that the local geometric features alone cannot express. The fused encoder then predicts the steady velocity field from geometry alone, and that predicted field is what conditions the autoregressive transport rollout; this is the sense in which flow and transport are decoupled, and the two stages are evaluated separately in Section 5.1.

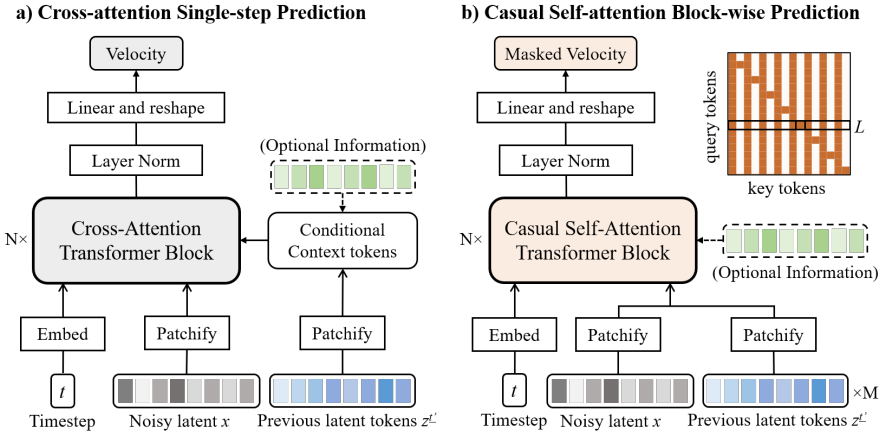


Figure 2: Illustration of latent-space autoregressive generative paradigms

5 Experiment

5.1 Model accuracy

Geometry-to-velocity prediction The steady velocity field is predicted first, directly from geometry, and only then conditions the autoregressive stage. We therefore evaluate this first stage on its own. We compare against the same five geometry-aware backbones, each trained *prior-matched*, i.e. receiving exactly the coarse-prior and fluid-fraction channels that GeoLAMP's encoder consumes. Results for the two velocity components and the full velocity vector are reported in Table 2. GeoLAMP attains the lowest error on every component of both datasets. The coarse prior lowers the error of every model; the geometry-only ablation and example predictions are given in Appendix D.2. Under both scenario, GeoLAMP exhibits the best performance on tortuous flow predictions.

Table 2: Relative L_2 error for geometry-to-velocity prediction with the coarse flow prior. u and v are the streamwise and transverse components; $|\mathbf{u}|$ is the error of the full velocity vector, $\|\mathbf{u}_{\mathrm{pred}}-\mathbf{u}\|/\|\mathbf{u}\|$. Best/second-best are **bold**/underlined.

	Reactive Flow			Heat Convection		
Model	u	v	$\lvert\mathbf{u}\rvert$	u	v	$\lvert\mathbf{u}\rvert$
OFormer (Li et al., 2023a)	<u>0.125</u>	<u>0.183</u>	<u>0.143</u>	0.503	0.665	0.546
GINO (Li et al., 2023c)	0.206	0.278	0.229	0.450	0.569	0.482
Transolver (Wu et al., 2024)	0.131	0.199	0.153	<u>0.364</u>	<u>0.485</u>	<u>0.396</u>
Geo-FNO (Li et al., 2023b)	0.142	0.213	0.165	0.382	0.495	0.412
NUNO (Liu et al., 2023a)	0.451	0.573	0.488	0.568	0.690	0.601
GeoLAMP	**0.097**	**0.135**	**0.109**	**0.177**	**0.246**	**0.196**

Autoregressive prediction Given the predicted velocity field, the transport dynamics are then rolled out in latent space. We conduct autoregressive experiments on three datasets using 4096 global–local (G&L) points sampled from the method in Section. 4.1. We compare GeoLAMP with recent methods and evaluate performance using the relative L_2 error over the full autoregressive rollout as well as at the final prediction step, as reported in Table 3. For both single-step and block-wise autoregression, GeoLAMP achieves the best last-step prediction accuracy across all three datasets, demonstrating its effectiveness in long-horizon forecasting. GeoLAMP-B's last-step errors exceed its rollout means but remain the lowest among the blockwise methods on all three datasets. Comparing GeoLAMP-S with the matched Latent-DiT control probes flow matching against DDPM noise prediction. Comparing GeoLAMP-B with Latent-Det. probes generative flow matching against direct latent-block regression; the largest last-step gap occurs on Elasticity.

Table 3: Relative L_2 errors for autoregressive mean and last-step predictions on three datasets. Best/second-best results are **bold**/underlined. The -S/-B labels denote the single-step/blockwise protocols evaluated here.

	Heat Convection		Reactive Flow		Elasticity	
Model	Mean	Last	Mean	Last	Mean	Last
(-S) OFormer (Li et al., 2023a)	0.0060	0.0134	<u>0.1101</u>	<u>0.1384</u>	9.29	144.4
(-S) GINO (Li et al., 2023c)	0.0050	0.0074	0.1535	0.1675	6.31	143.5
(-S) Transolver (Wu et al., 2024)	0.0052	0.0266	0.4766	0.4717	**0.2520**	1.3703
(-S) Latent-DiT	<u>0.0037</u>	<u>0.0073</u>	0.1464	0.1529	0.4340	<u>0.5892</u>
(-S) GeoLAMP	**0.0033**	**0.0063**	**0.1071**	**0.1183**	<u>0.4081</u>	**0.5154**
(-B) Geo-FNO (Li et al., 2023b)	0.0041	0.0071	0.1369	0.1580	<u>0.3676</u>	<u>0.5420</u>
(-B) NUNO (Liu et al., 2023a)	0.0043	0.0073	0.1528	0.1514	0.7160	3.8620
(-B) Latent-Det.	<u>0.0035</u>	<u>0.0056</u>	<u>0.0924</u>	<u>0.1030</u>	0.4395	1.9324
(-B) GeoLAMP	**0.0025**	**0.0046**	**0.0789**	**0.0813**	**0.3460**	**0.3823**

5.2 Effects of geometry representation

We further examine the contribution of the proposed GE & LE architecture to autoregressive prediction and evaluate the model's ability to reconstruct physical fields on full-mesh point sets.

Designed global & local encoder To demonstrate the effectiveness of the proposed global & local dual-encoder desgin, we compare GeoLAMP with a variant that encodes purely randomly sampled 4096 points (RS). The autoregressive accuracy on the three datasets is reported in Table 4. The higher accuracy on G&L points demonstrates that carefully designed point selection is essential for bridging real and latent space, and is crucial for accurate latent-space prediction.

Full-mesh generation by arbitrary decoder Originating from the flexibility of the GNO-based decoder, the AD enables the generated latent states to be decoded back to real-space fields on arbitrary points. We further evaluate the performance using both the RS and the G&L point sets. The autoregression results from G&L points for three datasets are shown in Fig. 3. The accuracy is given in Table 4, the G&L point set not only achieves higher accuracy at its own locations, but also leads to better full-mesh recovery. The whole comparison of results with RS and G&L points highlight a key distinction in scientific PDEs learning: spatial representation directly influence solution accuracy. This effect is further amplified in complex geometries, where physical behavior can vary sharply across subregions due to tortuous pathways and localized geometric features.

Table 4: Relative L_2 error comparison across different sampling method and corresponding full-mesh recovery. We report the autoregression mean and the last-step error. Best results are in **bold**.

	Heat Convection		Reactive Flow		Elasticity	
Locations	Mean	Last	Mean	Last	Mean	Last
RS	0.0045	0.0088	0.1178	0.1262	**0.3459**	**0.3811**
G&L	**0.0025**	**0.0046**	**0.0913**	**0.0995**	0.3460	0.3823
Full mesh from RS	0.0045	0.0080	0.1152	0.1349	0.5351	0.5573
Full mesh from G&L	**0.0040**	**0.0077**	**0.1065**	**0.1247**	**0.3561**	**0.3684**

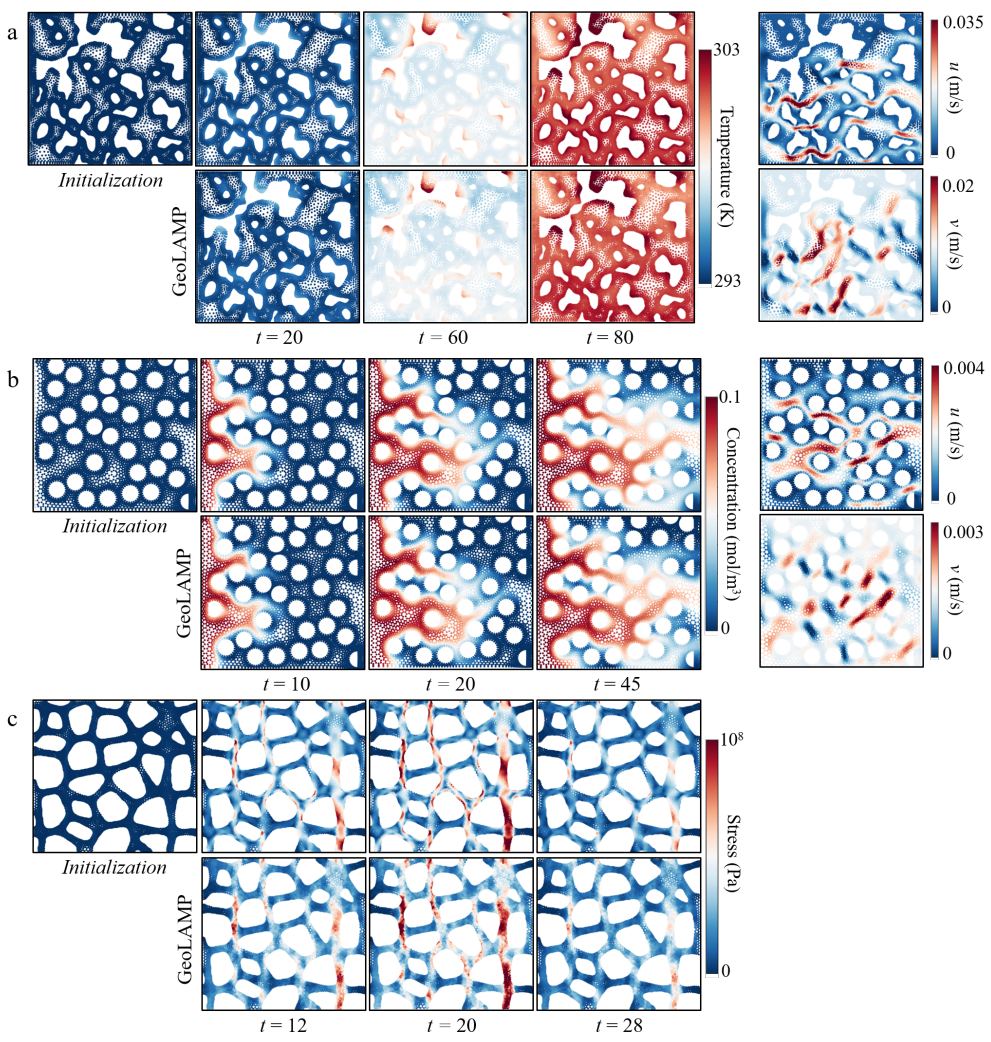


Figure 3: GeoLAMP results for velocity prediction (top for streamwise velocity prediction and bottom for transverse velocity prediction) and autoregression (bottom for prediction vs. top for ground truth). (a) Heat convection in stochastic-field structure. (b) Reactive flow in sphere-packed structure. (c) Elasticity in foam structure.

5.3 Inference-time rollout flexibility

GeoLAMP's causal block-wise attention allows the conditioning length C and output length P to be varied at inference time without retraining. Because frame groups share model parameters, changing their number requires only rebuilding the parameter-free attention mask. We evaluate this flexibility using the frozen pretrained GeoLAMP-B models on 200 held-out trajectories per dataset, reporting errors normalized by the native $C = P = 8$ configuration (Table 5).

With $C = 8$, increasing P to 10 or 12 increases error by at most 1% across datasets. Doubling P to 16 preserves near-native error for heat convection ($1.01\times$) and elasticity ($0.977\times$), but increases error to $1.42\times$ native for reactive flow. Thus, larger output blocks can reduce the number of autoregressive rollout iterations while maintaining near-native accuracy on some datasets. With $P = 8$, shortening the conditioning history to $C = 6$ or 7 increases error by at most 6% or 2%, respectively. However, using a single conditioning frame increases error to $2.74\times$ native for heat convection and $2.04\times$ for reactive flow, compared with only $1.04\times$ for elasticity. These results reveal distinct sensitivities to output block length and conditioning history. Inference-time flexibility therefore enables task-dependent configuration choices, rather than uniform robustness to arbitrary block lengths.

Table 5: Inference-time block-length flexibility of pretrained GeoLAMP-B models. Values are horizon-averaged physical relative L_2 errors normalized by the corresponding native $C = P = 8$ rollout (lower is better; 1.00 is native). Left: output length at $C = 8$, with commit stride P. Right: conditioning length at $P = 8$.

	Output length ($C = 8$)					History length ($P = 8$)			
Dataset	P=8	P=10	P=12	P=14	P=16	C=1	C=4	C=6	C=7
Heat convection	1.00	0.991	0.991	0.994	1.01	2.74	1.07	1.01	1.00
Reactive flow	1.00	1.00	1.01	1.14	1.42	2.04	1.19	1.06	1.02
Elasticity	1.00	0.964	0.993	0.986	0.977	1.04	1.07	1.03	1.01

5.4 Attention variations along the temporal axis

We analyze temporal changes by isolating the attention from noisy target queries N_j to conditioning frames C_i. For each query, the attention weights renormalized over C_1–C_8 are shown as line plots in Fig. 4 (a)–(c), measuring the relative preference among conditioning frames. For heat convection and elasticity, temperature and stress increase continuously across the entire domain throughout the process, as shown in Fig. 3. As a result, nearby states remain spatiotemporally consistent and continuous, gaining stronger attention weights. In contrast, reactive flow displays a different pattern. For near-future prediction, the initial conditioning frames remain nearly as important as the most recent frames. For later target frames, the attention gradually shifts toward recent inputs, while the contribution from the earliest frames decreases substantially. This behavior likely originates from the physics. As shown in Fig. 3, the reaction fronts preferentially migrate along different paths of lower resistance, exhibiting time-lagged spatial heterogeneity. Together, these observations suggest that capturing long-term dependencies could be more beneficial for PDEs autoregressive learning.

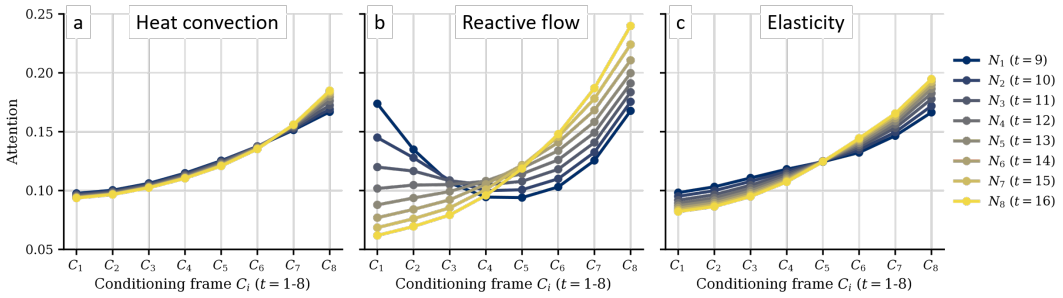


Figure 4: Attention strength along the temporal axis for the three dynamics.

6 Conclusions

In this work, we propose GeoLAMP for multiphysics PDEs learning in meso-scale complex 2D geometries. The specially designed dual encoder enables the model to capture both global physical behavior and local features. By decoupling flow from transport, the velocity field is predicted from geometry before the transport dynamics are rolled out, and fusing a cheap grid-based flow prior into the graph encoder enhances tortuous flow predictions. We propose a casual self-attention transformer in a flow matching framework, and achieve block-wise autoregressive prediction. We construct three complex-geometry datasets, including heat convection, reactive flow, and elasticity. GeoLAMP achieves best accuracy in long-time autoregression and flexible rollout policy, while demonstrating strong flexibility in geometry representation for tortuous structures. Several directions remain for future work. First, although the dual-encoder architecture improves geometry-aware latent representation, the model still faces challenges in reconstructing high-quality physical fields in highly complex geometries. Second, while block-wise inference improves inference efficiency, the training cost increases with the block size. We plan to address this by introducing more efficient attention mechanisms. Moreover, meso-scale multiphysics encompasses a broad range of scientific and engineering problems in 3D, which remain to be further explored. Overall, GeoLAMP demonstrates promising potential as a fast surrogate model in solving multiphysics in tortuous complex domains.

AI USE STATEMENT

In this work, we used generative AI tools for [formulate mathematical claims, design or provide feedback on research methodology or experiments, implement methods, clean and reformat dataset, support qualitative and thematic data analysis]. We have not used generative AI tools for [generate synthetic data sets, help develop theoretical models or conceptual frameworks, provide critical ingredients for proving mathematical claims, interpret results], and [assist in the writing of proofs, propose or refine hypotheses, assist with translation] are not applicable to this work. Additionally, we used generative AI tools for [create or modify scientific figures or images, draft parts of a research paper, brainstorming, edit a research paper to improve readability]. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies surrogate modeling of multiphysics PDEs on synthetic meso-scale geometries. All three datasets are generated by the authors using finite element method following settings in Appendix A, which will be released upon acceptance. The authors declare no conflicts of interest.

REPRODUCIBILITY STATEMENT

We have made the following efforts to support reproducibility. The full generation procedure for all three datasets, including geometries, governing equations, material properties, boundary conditions is described in Appendix A. Complete architectural hyperparameters for GeoLAMP and every baseline are given in Appendix F, and all training configurations (learning rate, weight decay, scheduler, and epochs) are listed in Table 8. All experiments were run on a single NVIDIA A100 GPU.

REFERENCES

Kapil Agrawal, Peter N Loezos, Madhava Syamlal, and Sankaran Sundaresan. The role of meso-scale structures in rapid gas–solid flows. *Journal of Fluid Mechanics*, 445:151–185, 2001.

Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6:320–328, 2024. doi: 10.1038/s42254-024-00712-5. URL https://doi.org/10.1038/s42254-024-00712-5.

Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. *Advances in Neural Information Processing Systems*, 35:23463–23478, 2022.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural pde solvers. *arXiv preprint arXiv:2202.03376*, 2022. doi: 10.48550/arXiv.2202.03376. URL https://arxiv.org/abs/2202.03376.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. doi: 10.48550/arXiv.2104.13478. URL https://arxiv.org/abs/2104.13478.

Lei Chen, Xiaohui Zhong, Hao Li, Jie Wu, Bo Lu, Deliang Chen, Shang-Ping Xie, Libo Wu, Qingchen Chao, Chensen Lin, et al. A machine learning model that outperforms conventional global subseasonal forecast models. *Nature Communications*, 15(1):6425, 2024.

Li Chen, An He, Jianlin Zhao, Qinjun Kang, Zeng-Yao Li, Jan Carmeliet, Naoki Shikazono, and Wen-Quan Tao. Pore-scale modeling of complex transport phenomena in porous media. *Progress in Energy and Combustion Science*, 88:100968, 2022.

Jaehong Chung, Rasool Ahmad, WaiChing Sun, Wei Cai, and Tapan Mukerji. Prediction of effective elastic moduli of rocks using graph neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421:116780, 2024.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. *arXiv preprint arXiv:2012.09841*, 2021. doi: 10.48550/arXiv.2012.09841. URL https://arxiv.org/abs/2012.09841.

Tianrun Gao, Haoren Zheng, Wenhao Deng, Haodong Feng, Tao Zhang, Ruiqi Feng, Qianyi Chen, and Tailin Wu. Gencp: Towards generative modeling paradigm of coupled physics. *arXiv preprint arXiv:2601.19541*, 2026.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17616–17635, 2024. doi: 10.48550/arXiv.2403.03542. URL https://proceedings.mlr.press/v235/hao24d.html.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020. doi: 10.48550/arXiv.2006.11239. URL https://arxiv.org/abs/2006.11239.

Valerii Iakovlev, Markus Heinonen, and Harri Lähdesmäki. Learning space-time continuous latent neural pdes from partially observed states. *Advances in Neural Information Processing Systems*, 36:26372–26395, 2023.

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.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022. doi: 10.48550/arXiv.2206.00364. URL https://arxiv.org/abs/2206.00364.

Ali Kashefi and Tapan Mukerji. Point-cloud deep learning of porous media for permeability prediction. *Physics of Fluids*, 33(9), 2021.

Katiana Kontolati, Somdatta Goswami, George Em Karniadakis, and Michael D. Shields. Learning nonlinear operators in latent spaces for real-time predictions of complex dynamics in physical systems. *Nature Communications*, 15:5101, 2024. doi: 10.1038/s41467-024-49411-w. URL https://doi.org/10.1038/s41467-024-49411-w.

Jinghai Li, Wei Ge, Wei Wang, Ning Yang, Xinhua Liu, Limin Wang, Xianfeng He, Xiaowei Wang, Junwu Wang, and Mooson Kwauk. *From multiscale modeling to meso-science*. Springer, 2013.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2023a. doi: 10.48550/arXiv.2205.13671. URL https://arxiv.org/abs/2205.13671.

Zijie Li, Saurabh Patil, Francis Ogoke, Dule Shu, Wilson Zhen, Michael Schneier, John R Buchanan Jr, and Amir Barati Farimani. Latent neural pde solver: A reduced-order modeling framework for partial differential equations. *Journal of Computational Physics*, 524:113705, 2025.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020. doi: 10.48550/arXiv.2003.03485. URL https://arxiv.org/abs/2003.03485.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2021. doi: 10.48550/arXiv.2010.08895. URL https://arxiv.org/abs/2010.08895.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023b. doi: 10.48550/arXiv.2207.05209. URL https://jmlr.org/papers/v24/23-0064.html.

Zongyi Li, Nikola Borislavov Kovachki, Christopher Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d pdes. *arXiv preprint arXiv:2309.00583*, 2023c. doi: 10.48550/arXiv.2309.00583. URL https://arxiv.org/abs/2309.00583.

Jian Liang and Hongkai Zhao. Solving partial differential equations on point clouds. *SIAM Journal on Scientific Computing*, 35(3):A1461–A1486, 2013.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. doi: 10.48550/arXiv.2210.02747. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. Pderefiner: Achieving accurate long rollouts with neural pde solvers. *arXiv preprint arXiv:2308.05732*, 2023. doi: 10.48550/arXiv.2308.05732. URL https://arxiv.org/abs/2308.05732.

Songming Liu, Zhongkai Hao, Chengyang Ying, Hang Su, Ze Cheng, and Jun Zhu. Nuno: A general framework for learning parametric pdes with non-uniform data. In *International Conference on Machine Learning*, pp. 21658–21671. PMLR, 2023a.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2023b. doi: 10.48550/arXiv.2209.03003. URL https://arxiv.org/abs/2209.03003.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021. doi: 10.1038/s42256-021-00302-5. URL https://doi.org/10.1038/s42256-021-00302-5.

Xuekun Lu, Antonio Bertei, Donal P Finegan, Chun Tan, Sohrab R Daemi, Julia S Weaving, Kieran B O'Regan, Thomas MM Heenan, Gareth Hinds, Emma Kendrick, et al. 3d microstructure design of lithium-ion battery electrodes assisted by x-ray nano-computed tomography and modelling. *Nature communications*, 11(1):2079, 2020.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Yashee Mathur and Tapan Mukerji. Changes in physical properties of rocks during serpentinization and implications for natural hydrogen exploration. *The Leading Edge*, 44(6):479–487, 2025.

Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory, 2003.

COMSOL Multiphysics. Introduction to comsol multiphysics®. *COMSOL Multiphysics, Burlington, MA, accessed Feb*, 9(2018):32, 1998.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021. doi: 10.48550/arXiv.2102.09672. URL https://arxiv.org/abs/2102.09672.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B Dalziel, Drummond B Fielding, et al. The well: a large-scale collection of diverse physics simulations for machine learning. *Advances in Neural Information Processing Systems*, 37:44989–45037, 2024.

Joachim Ohser and Katja Schladitz. *3D images of materials structures: processing and analysis*. John Wiley & Sons, 2009.

Guofei Pang, Lu Lu, and George Em Karniadakis. fpinns: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4):A2603–A2626, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2023. doi: 10.48550/arXiv.2212.09748. URL https://arxiv.org/abs/2212.09748.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2021. doi: 10.48550/arXiv.2010.03409. URL https://arxiv.org/abs/2010.03409.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature-learning-on-point-sets-in-a-metric-space.

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. URL https://doi.org/10.1016/j.jcp.2018.10.045.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2022. doi: 10.48550/arXiv.2112.10752. URL https://arxiv.org/abs/2112.10752.

François Rozet, Ruben Ohana, Michael McCabe, Gilles Louppe, François Lanusse, and Shirley Ho. Lost in latent space: An empirical study of latent diffusion models for physics emulation. *arXiv preprint arXiv:2507.02608*, 2025.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *arXiv preprint arXiv:2002.09405*, 2020. doi: 10.48550/arXiv.2002.09405. URL https://arxiv.org/abs/2002.09405.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2021a. doi: 10.48550/arXiv.2010.02502. URL https://arxiv.org/abs/2010.02502.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2021b. doi: 10.48550/arXiv.2011.13456. URL https://arxiv.org/abs/2011.13456.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. *arXiv preprint arXiv:2210.07182*, 2022. doi: 10.48550/arXiv.2210.07182. URL https://arxiv.org/abs/2210.07182.

Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics (TOG)*, 37(4):1–10, 2018.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/7210-neural-discrete-representation-learning.

Remco Van Erp, Reza Soleimanzadeh, Luca Nela, Georgios Kampitsis, and Elison Matioli. Co-designing electronics with microfluidics for more sustainable cooling. *Nature*, 585(7824):211–216, 2020.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1457–1466, 2020.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics*, 38(5):146:1–146:12, 2019. doi: 10.1145/3326362. URL https://doi.org/10.1145/3326362.

Zi Wang, Li Chen, Hangkai Wei, Zhenxue Dai, Qinjun Kang, and Wen-Quan Tao. Pore-scale study of mineral dissolution in heterogeneous structures and deep learning prediction of permeability. *Physics of Fluids*, 34(11), 2022.

Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M Benson. U-fno—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705, 2024. doi: 10.48550/arXiv.2402.02366. URL https://proceedings.mlr.press/v235/wu24r.html.

Haixu Wu, Minghao Guo, Zongyi Li, Zhiyang Dou, Mingsheng Long, Kaiming He, and Wojciech Matusik. Geopt: Scaling physics simulation via lifted geometric pre-training. *arXiv preprint arXiv:2602.20399*, 2026.

Tailin Wu, Takashi Maruyama, and Jure Leskovec. Learning to accelerate partial differential equations via latent global evolution. *Advances in Neural Information Processing Systems*, 35:2240–2253, 2022.

A Dataset Generation

Overall, we reconstruct three distinct meso-scale geometries for three different physical problems, as shown in Fig. 5. For each dataset, 2,000 structures are generated, and the corresponding physical PDEs are solved to obtain 2,000 spatio-temporal solution sequences. For these different structures, the porosity changes from 0.55 to 0.70. Each dataset is then randomly split into training, validation, and test sets with 1,500, 300, and 200 sequences, respectively. All simulation is implemented in COMSOL (Multiphysics, 1998). We show one sequences of each dataset in Fig. 6.

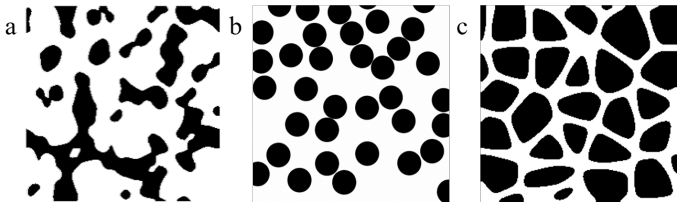


Figure 5: Meso-scale structure reconstruction

Heat convection in stochastic-field structure. The geometry is generated using a stochastic field method implemented in GeoDict (Ohser & Schladitz, 2009), as shown in Fig. 5(a), yielding a highly heterogeneous porous structure within a $24{,}000 \times 24{,}000, \mu\mathrm{m}^2$ domain. The porosity of 2000 structures varies from 0.68 to 0.70. The flow is governed by the incompressible Navier–Stokes equations coupled with an energy transport equation, where heat exchange occurs at the solid–fluid interface. Water is used as the working fluid. Periodic boundary conditions are imposed along the flow direction with a pressure drop of $3, \mathrm{Pa}$, allowing the fluid to exit through the outlet and re-enter from the inlet while continuously exchanging heat with the solid phase under a prescribed heat flux of $1000, \mathrm{W/m}^2$. In total, temperature field is saved at 100 time steps for the autoregression task. This setup represents pore-scale heat-transfer processes commonly encountered in battery systems and low-heat-flux thermal management applications.

$$\begin{aligned}
&\rho\left(\tfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u}, && x \in \Omega_f \\
&\nabla\cdot\mathbf{u} = 0, && x \in \Omega_f \\
&\tfrac{\partial T}{\partial t} + \mathbf{u}\cdot\nabla T = \alpha\nabla^2 T, && x \in \Omega_f \\
&-k_T\nabla T\cdot\mathbf{n} = q, && x \in \partial\Omega_s
\end{aligned} \tag{13}$$

Reactive flow in sphere-packed structure. The complex geometry is constructed using a stochastic Monte Carlo sphere-packing process (Wang et al., 2022), as shown in Fig. 5(b). The porosity of 2000 structures varies from 0.61 to 0.68. Specifically, multiple spheres with radius $14, \mu\mathrm{m}$ are randomly initialized and iteratively adjusted within a $240 \times 240, \mu\mathrm{m}^2$ computational domain. In this domain, the incompressible Navier–Stokes equations are coupled with a solute transport equation, where reactants are advected from the left inlet to the right outlet and consumed at the solid surfaces through a surface reaction term. The inlet velocity, inlet concentration, and surface reaction rate are set to $7.2 \times 10^{-4}, \mathrm{m/s}$, $0.1, \mathrm{mol/m^3}$, and $10^4, \mathrm{m/s}$, respectively. In total, concentration field is saved at 60 time steps for the autoregression task. This setup captures key mechanisms in subsurface reactive transport and is representative of applications such as CO_2 sequestration and natural hydrogen generation.

$$
\begin{aligned}
&\rho\left(\tfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u}, && x \in \Omega_f \\
&\nabla\cdot\mathbf{u} = 0, && x \in \Omega_f \\
&\tfrac{\partial c}{\partial t} + \mathbf{u}\cdot\nabla c = D\nabla^2 c, && x \in \Omega_f \\
&-D\nabla c\cdot\mathbf{n} = kc, && x \in \partial\Omega_s
\end{aligned}
\tag{14}
$$

Elasticity in foam structure. The porous geometry is generated using a foam reconstruction method implemented in GeoDict (Ohser & Schladitz, 2009), as shown in Fig. 5(c), within a $2{,}400 \times 2{,}400, \mu\mathrm{m}^2$ domain. The porosity of 2000 structures varies from 0.55 to 0.66. The mechanical response is modeled by infinitesimal linear elasticity under dynamic loading conditions. Copper is used as the solid material, reflecting its common use in metallic foams. Although the governing equations are quasi-static, time-dependent boundary displacements are applied to mimic realistic loading scenarios. In total, stress field is saved at 40 time steps. It should be mentioned that only first 38 timesteps are used, since the stress field at the last two steps approach zero, leading to analysis error. This dataset evaluates the model's ability to capture deformation responses under temporally varying boundary conditions in complex porous media.

$$
\begin{aligned}
&\nabla\cdot\sigma = 0, && x \in \Omega \\
&\sigma = \lambda(\nabla\cdot u)I + 2\mu\varepsilon, && x \in \Omega \\
&\varepsilon = \tfrac{1}{2}(\nabla u + (\nabla u)^\top), && x \in \Omega \\
&u(x,t) = -0.001H\cdot\tfrac{1-\cos(2\pi t)}{2}, && x \in \Gamma_{top}
\end{aligned}
\tag{15}
$$

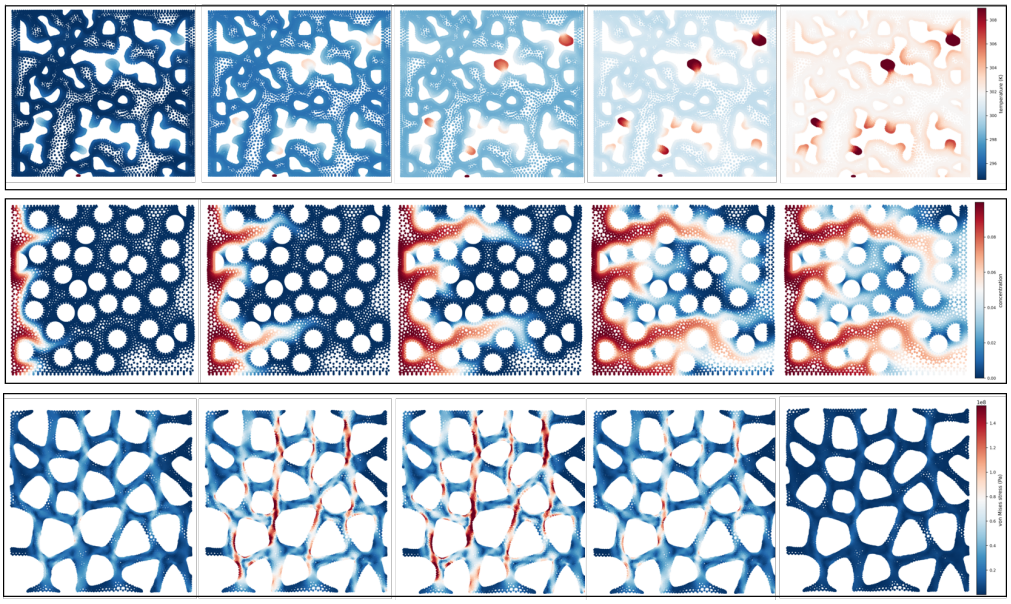


Figure 6: Meso-scale physics datasets

B POINT SELECTION EXAMPLES

Fig. 7 shows the global–local point sets of Section 4.1 on three geometries from each dataset. FPS first draws $n_g = 2048$ points from the full mesh, and CBS then selects $n_l = 2048$ points with the largest scores from the remaining nodes, using $k = 20$ nearest neighbors. The two sets are therefore disjoint. The point sets shown are the ones used in training. FPS spreads its points evenly over the whole domain, including the solid boundaries and the domain edges, and so provides the global backbone of the geometry. CBS points concentrate inside the local pores and tortuous channels, adds resolution where the global set is sparse rather than repeating it. The CBS points play an important role in learning the local physical details inside pores and channels, which carries critical information of tortuous flow-transport physics.

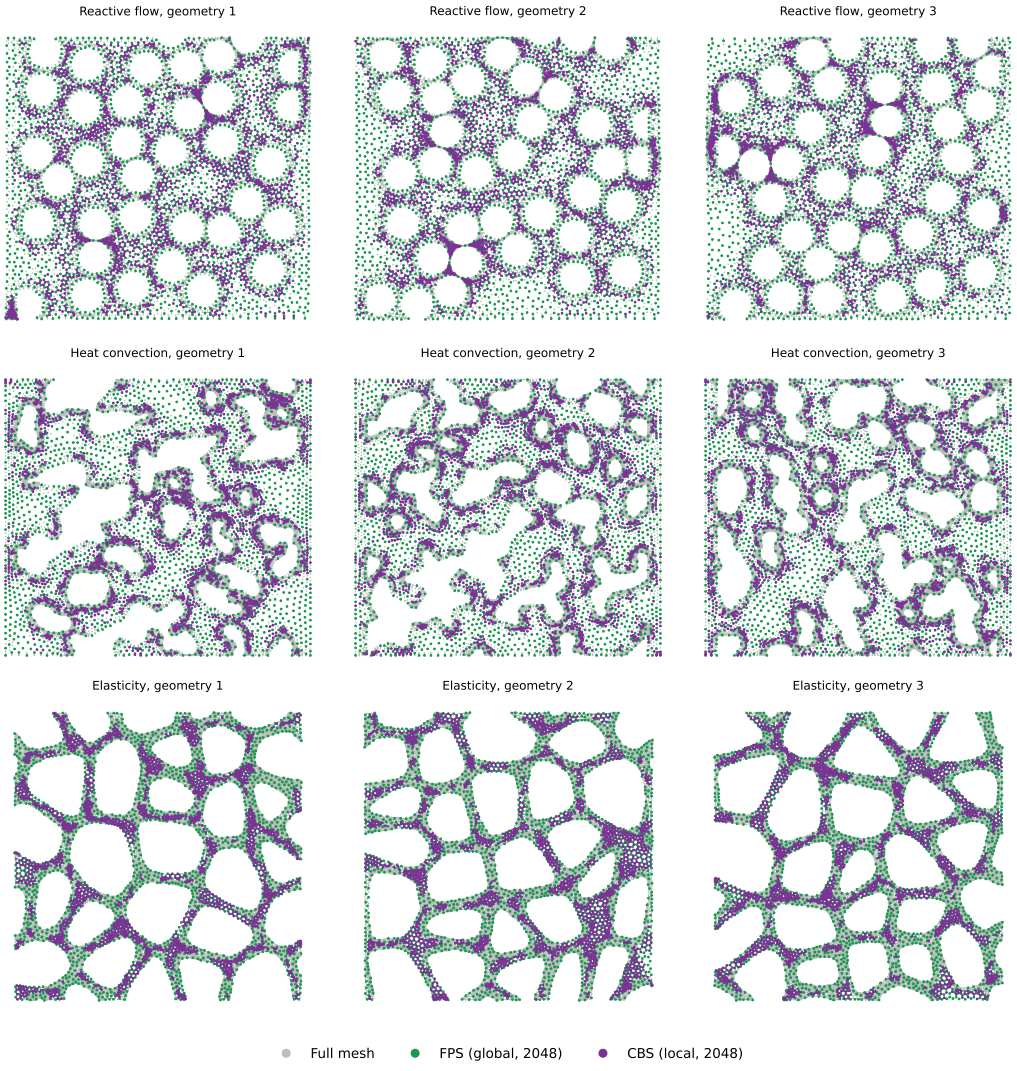


Figure 7: Global (FPS, green) and local (CBS, purple) point selections on the full mesh (gray) for three geometries of reactive flow (top), heat convection (middle) and elasticity (bottom). Each set contains 2048 points.

C FLOW MATCHING IN GEOLAMP

For both prediction paradigms, we train the latent generative model using a linear flow-matching objective. Let $\mathbf{z}$ denote the clean target latent tokens and let $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ denote standard Gaussian

noise with the same shape as $\mathbf{z}$. Given $t \sim \mathcal{U}(0,1)$, we construct the interpolated latent state

$$\mathbf{x}_t = (1-t)\boldsymbol{\epsilon} + t\mathbf{z}, \tag{16}$$

which transports samples from the Gaussian distribution at $t = 0$ to the latent data distribution at $t = 1$. The corresponding target velocity is

$$\mathbf{u}_t = \frac{d\mathbf{x}_t}{dt} = \mathbf{z} - \boldsymbol{\epsilon}. \tag{17}$$

The neural network is trained to predict this velocity field from the noisy latent state, the flow-matching time, and the conditional context tokens $\mathbf{c}$. The training objective is

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{\mathbf{z},\boldsymbol{\epsilon},t}\left[\left\|\mathbf{v}_\theta(\mathbf{x}_t, t, \mathbf{c}) - \mathbf{u}_t\right\|_2^2\right]. \tag{18}$$

For block-wise prediction, the clean latent variable $\mathbf{z}$ corresponds only to the target block. Given a conditioning window $[\mathbf{z}_s, \ldots, \mathbf{z}_{s+M-1}]$, the model predicts the target block $[\mathbf{z}_{s+M}, \ldots, \mathbf{z}_{s+2M-1}]$. Flow matching is applied only to this target block, while the conditioning block is provided through $\mathbf{c}$. Equivalently, the block-wise objective is an average over target latent positions:

$$\mathcal{L}_{\mathrm{FM}}^{\mathrm{block}} = \mathbb{E}_{\mathbf{z},\boldsymbol{\epsilon},t}\left[\frac{1}{|\Omega_{\mathrm{tar}}|}\sum_{i\in\Omega_{\mathrm{tar}}}\left\|\mathbf{v}_\theta(\mathbf{x}_t, t, \mathbf{c})_i - \mathbf{u}_{t,i}\right\|_2^2\right], \tag{19}$$

where Ω_{tar} denotes the set of target-block latent positions. In implementation, this target selection is performed implicitly by constructing $\mathbf{x}_t$ only over the target block and by extracting velocity predictions only from the noisy target-token positions.

During inference, samples are generated by solving the learned probability-flow ODE

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{v}_\theta(\mathbf{x}_t, t, \mathbf{c}), \qquad \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad t \in [0,1]. \tag{20}$$

The terminal latent state $\mathbf{x}_1$ gives the predicted target latent block. For autoregressive rollout, the predicted latent states are appended to the conditioning window and the procedure is repeated. The terminal latents are then passed through the decoder to reconstruct the corresponding physical-space states. For the reported test rollouts, we use Euler integration with 50 discretization steps.

D Coarse flow prior and velocity prediction

D.1 Coarse flow prior: method and settings

The prior solver applies the fixed stencils of Eq. 10 to u, v and p, so all advective, viscous and pressure terms are evaluated as batched convolutions on GPU. The velocity is marched explicitly to steady state; the pressure is updated by a five-point Jacobi iteration of $\nabla^2 p = -\rho(u_x^2 + 2u_y v_x + v_y^2)$ with an added $\nabla\cdot\mathbf{u}/\Delta t$ damping term. No-slip is imposed by masking solid cells, and a fixed pressure drop drives the flow between inlet and outlet buffers. The result is a coarse, explicit approximation of the physics solved in the dataset.

The grid-based flow prior of Section 4.3 is precomputed once per geometry and cached, so it adds no cost during training or inference. The solver settings used for all reported results are listed in Table 6. The occupancy mask is rasterized at $G = 256$ and the converged field is pooled to the 32×32 latent grid with an exact kernel of size $k = G/32 = 8$. Because the occupancy grid uses a square bounding box while the physical domain is not square, the padding rows above the domain are forced solid; leaving them fluid would give the solver a frictionless bypass channel along the top boundary and corrupt the pressure field.

Convergence is monitored by two criteria evaluated together: the relative change of the velocity field between checks, and a flux-balance measure computed as the interquartile spread of the column-wise flux over interior columns, normalized by its median. The columns adjacent to the inlet and outlet buffers are excluded from this measure, since the buffers are forced fully fluid and therefore carry artificially more flux. A sequence is accepted only if both criteria are met; sequences that fail, or

that produce any non-finite value, are marked invalid and assigned a zero prior together with a zero fluid-fraction channel, so the encoder can distinguish a missing prior from a genuinely stagnant region. For the reported datasets, 99.6% of sequences converged with a median flux imbalance of 0.011, against an acceptance threshold of 95% and 0.05 respectively.

Two properties of the scheme are worth stating plainly. First, no-slip is imposed by masking solid cells rather than by a boundary discretization, so accuracy degrades near walls; the field is used as a carrier of global connectivity and pressure structure, and should not be read as a quantitative near-wall velocity. Second, the collocated grid admits a checkerboard pressure mode, which is monitored explicitly and was found to remain small at the reported settings.

Table 6: Solver and pooling settings for the coarse flow prior.

Setting	Value	Note
Solve resolution G	256	pooled to 32×32, kernel $k = 8$
Time step Δt	0.2	explicit marching to steady state
Max iterations	100,000	median convergence $\approx$ 5,000
Velocity residual tolerance	10^{-4}	relative change between checks
Flux-balance tolerance	0.05	interquartile spread / median
Inlet–outlet buffer	8 cells	fixed pressure drop, fully fluid
Transverse boundary	free-slip	preserves the y-flip symmetry
Validity gate	$\geq 95\%$	achieved 99.6%

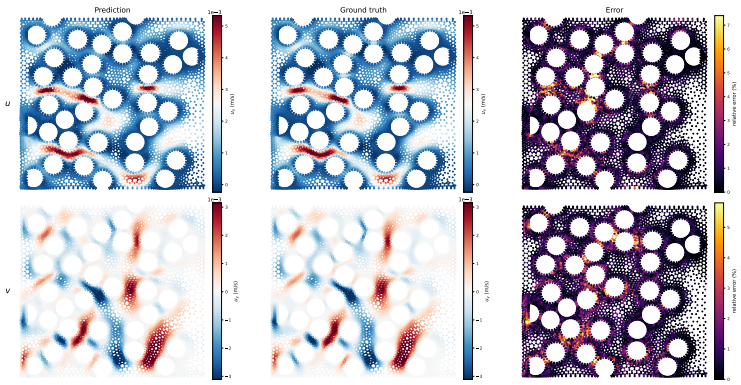
D.2 Velocity prediction results

Table 7 reports the geometry-only counterpart of Table 2: every model is trained from geometry alone, without the coarse-prior channels. GeoLAMP is again the most accurate on every component, and its margin is largest in this setting (reactive flow 0.153 vs. 0.263 for Geo-FNO; heat convection 0.482 vs. 0.608). Without the prior, however, several baselines approach the 0.891 error of a zero prediction and recover little beyond the mean field; on heat convection OFormer and Transolver reach $v \approx 1.000$, i.e. they predict an almost vanishing transverse velocity. Adding the prior lowers the vector error of every baseline, by 1.4–$4.2\times$ on reactive flow and 1.3–$2.2\times$ on heat convection, and of GeoLAMP by $1.4\times$ and $2.5\times$ respectively.

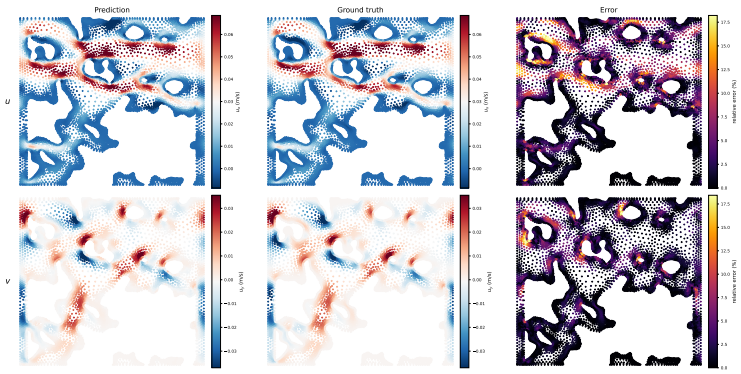
Fig. 8 shows GeoLAMP predictions with the prior on a held-out structure of each dataset. The preferential flow channels are reproduced in both components, and the largest errors concentrate at pore throats, where the masked no-slip treatment of the coarse prior is least accurate.

Table 7: Relative L_2 error for geometry-to-velocity prediction *without* the coarse flow prior (geometry only). Columns as in Table 2. Best/second-best among geometry-only models are **bold**/underlined.

	Reactive Flow			Heat Convection		
Model	u	v	$\lvert\mathbf{u}\rvert$	u	v	$\lvert\mathbf{u}\rvert$
OFormer (Li et al., 2023a)	0.464	0.716	0.546	0.700	1.000	0.788
GINO (Li et al., 2023c)	0.415	0.583	0.468	0.722	0.879	0.764
Transolver (Wu et al., 2024)	0.564	0.831	0.649	0.792	1.000	0.853
Geo-FNO (Li et al., 2023b)	<u>0.229</u>	<u>0.334</u>	<u>0.263</u>	<u>0.571</u>	<u>0.710</u>	<u>0.608</u>
NUNO (Liu et al., 2023a)	0.640	0.846	0.704	0.754	0.921	0.801
GeoLAMP	**0.137**	**0.188**	**0.153**	**0.456**	**0.564**	**0.482**



(a) Reactive flow, test structure 400



(b) Heat convection, test structure 751

Figure 8: Geometry-to-velocity prediction of GeoLAMP with the coarse flow prior on held-out structures. In each panel, columns show the prediction, the ground truth and the pointwise relative error; rows show the streamwise (u) and transverse (v) velocity components.

E COMMIT STEP INFLUENCE

The blockwise prediction paradigm provides the flexibility to commit different numbers of predicted steps during inference. Specifically, after predicting a block of future latent states, the model can commit only the first few steps and then re-predict the next block, or commit more steps at once before the next autoregressive update. We investigate this effect on three datasets by varying the number of committed steps k, as shown in Fig. 9. GeoLAMP-B permits the commit stride k to vary while keeping the predicted block length fixed. Across the three datasets, larger k generally yields lower rollout error, particularly at later timesteps and for Elasticity. This trend is consistent with the reduced number of autoregressive updates when more predicted steps are committed at once.

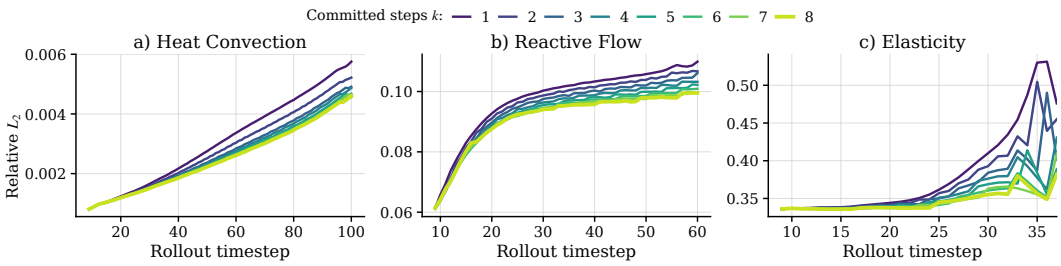


Figure 9: Relative L_2 rollout error for different committed steps during blockwise prediction.

F MODEL HYPERPARAMETER AND IMPLEMENTATION DETAILS

F.1 MODEL HYPERPARAMETER

Global & Local VAE. The autoencoder compresses each per-timestep scalar field on the unstructured mesh into a $64 \times 8 \times 8$ latent. Two independent GNO encoders use radius-graph integral transforms with radius 0.15 and MLP layers $[80, 80, 80]$, projecting the input points onto a shared 32×32 grid with 128 channels. The two encoded grids are summed element-wise. The CNN encoder uses a base hidden width of 128, channel multipliers $(1, 2, 4)$, two ResNet blocks per resolution level, and self-attention at resolutions 16×16 and 8×8, producing an $8 \times 8 \times 64$ bottleneck with mean and log-variance. The KL weight is set to 10^{-6}. The decoder mirrors the CNN encoder and reconstructs a $32 \times 32 \times 128$ grid, followed by a GNO decoder with MLP layers $[512, 256]$ and radius 0.15.

Arbitrary Decoder. The arbitrary decoder shares the CNN-decoder and GNO-decoder backend with the VAE. It takes a predicted $8 \times 8 \times 64$ latent and upsamples it through three resolution levels with base hidden width 128, channel multipliers $(1, 2, 4)$, two ResNet blocks per level, and self-attention at resolutions 8×8 and 16×16. The CNN decoder outputs a $32 \times 32 \times 128$ feature grid. A GNO decoder with MLP layers $[512, 256]$ and radius 0.15 maps the grid features to the full mesh with approximately 12–21 $\times 10^3$ valid vertices and produces a one-channel field prediction.

GeoLAMP-S. GeoLAMP-S denoises a $64 \times 8 \times 8$ target latent under conditioning. With `patch_size` $= 1$, each latent is patch-embedded into 64 tokens with hidden width 768 and sinusoidal 2-D positional embeddings. The flow-matching time and autoregressive step index are embedded using sinusoidal-MLP timestep embedders and combined as the AdaLN-Zero conditioning vector. The transformer uses 12 blocks, hidden width 768, 12 attention heads, and $\mathrm{mlp_ratio} = 4$. Each block contains AdaLN-modulated self-attention, cross-attention to context tokens, and an AdaLN-modulated GeGLU feed-forward layer. The final AdaLN-modulated linear layer maps the tokens back to 64-channel patches.

GeoLAMP-B. GeoLAMP-B uses the same backbone and hyperparameter settings as GeoLAMP-S, but performs non-overlapping block-wise prediction. It predicts $M = 8$ future latent states in one flow-matching evaluation. Conditional and noisy target latents are patch-embedded into a causal token sequence, and the noisy target token groups are unpatchified and concatenated to produce the eight predicted latent states. The velocity here is the flow-matching vector field, not the physical velocity of Section 5.1.

GINO. GINO uses the same encoder-decoder structure as the Global & Local VAE (changed to AE), with the variational bottleneck replaced by a deterministic autoencoder, and are trained together. The FNO operates on the 8×8 latent grid. A 1×1 lifting convolution maps the input to a 752-channel hidden representation, using 192 input channels for reaction flow and heat convection and 128 input channels for elasticity. The trunk contains 10 spectral blocks with $(4, 4)$ Fourier modes, complex-valued mode-wise weights, 1×1 pointwise convolutional residual bypasses, GELU activations, and outer residual connections. A final 1×1 projection convolution maps the hidden tensor to 64 output channels.

OFormer. OFormer operates directly on 4096 mesh nodes sampled by FPS and CBS. The encoder uses a $\mathrm{Conv2d}(5 \to 128)$ stem with kernel size $(2, 1)$, learned node-type embeddings, and relative positional encodings. The encoder contains six Galerkin linear-attention blocks with hidden dimension 128, head dimension 128, one attention head, and feed-forward hidden dimension 512. The decoder uses Gaussian Fourier coordinate features, a two-layer MLP with hidden dimension 128, one Galerkin cross-attention block, one linear-attention block, and a three-layer MLP prediction head.

Transolver. Transolver operates directly on 4096 real-space query points from the concatenated FPS and CBS point clouds. The input feature dimension is 6 for reaction flow and heat convection and 4 for elasticity. A preprocessing MLP maps the input to width 256, followed by a learned placeholder vector and a sinusoidal physical-time embedding processed by a two-layer SiLU MLP. The trunk contains 6 Transolver blocks with hidden size 256, 8 heads, 32 slice tokens per head, dropout 0, and $\mathrm{mlp_ratio} = 2$. A final LayerNorm and linear head predict one scalar field value per point.

Geo-FNO. Geo-FNO learns a deformation from the irregular input domain to a regular computational grid and applies an FNO trunk on that grid, mapping back to the physical points through the inverse deformation. It is trained on all three datasets under the same blockwise M-to-M ($M = 8$) autoregression protocol, with matched model size, identical training strategy, and identical inference protocol as GeoLAMP-B, and is evaluated on the same 4096 G&L points of the test set.

NUNO. NUNO partitions the non-uniform point set into subdomains, interpolates each subdomain onto a local regular grid, and applies an FNO per subdomain before scattering the predictions back to the original points. It follows the same blockwise M-to-M ($M = 8$) protocol, matched model size, training strategy, inference protocol, and 4096 G&L evaluation points as Geo-FNO and GeoLAMP-B.

Latent-Det. Latent-Det. is a matched deterministic control for the generative formulation. It uses the same frozen Global & Local VAE, the same latent tokens, the same causal block mask, the same $M = 8$ blockwise protocol and commit stride, matched transformer depth and width, and identical conditioning as GeoLAMP-B. The flow-matching objective and its iterative ODE sampler are replaced by direct MSE regression of the same target latent block, so inference uses a single forward pass per block rather than 50 Euler steps.

Latent-DiT. Latent-DiT denoises a $64 \times 8 \times 8$ target latent using DDPM noise prediction. With $\texttt{patch_size} = 1$, the noisy target latent is embedded into 64 tokens with hidden width 768 and fixed sinusoidal 2-D positional embeddings. The diffusion timestep is embedded using a sinusoidal-MLP timestep embedder for AdaLN-Zero modulation. The transformer uses 12 blocks, hidden width 768, 12 attention heads, and $\mathrm{mlp_ratio} = 4$. Each block contains AdaLN-modulated self-attention, cross-attention to context tokens, and an AdaLN-modulated GeGLU feed-forward layer. The final AdaLN-modulated linear layer maps the tokens back to 64-channel patches.

F.2 Training settings

The training configurations are summarized in Table 8. For the heat convection and reactive flow datasets, the input includes the velocity field as an optional context. For the elasticity dataset, the model takes only the von Mises stress field as input. The VAE and latent dynamics model are trained separately. The arbitrary decoder is also trained independently, using precomputed latent vectors from partial points as input and full-mesh data as the target. All training tasks are conducted on a single NVIDIA A100 GPU with 80 GB of memory.

Table 8: Training hyperparameters for baseline models, GeoLAMP variants, the VAE, and the arbitrary decoder.

Model	Learning rate	Weight decay	Scheduler	Epochs
GINO	10^{-4}	10^{-4}	cosine, 500-step warmup	200
OFormer	10^{-4}	10^{-4}	cosine, 500-step warmup	200
Transolver	3×10^{-4}	10^{-4}	cosine, 500-step warmup	200
GeoFNO	10^{-4}	10^{-4}	cosine, 500-step warmup	200
NUNO	10^{-4}	10^{-4}	cosine, 500-step warmup	200
Latent-DiT	10^{-4}	10^{-4}	cosine, 500-step warmup	200
Latent-Det.	10^{-4}	10^{-4}	cosine, 500-step warmup	200
GeoLAMP-S	10^{-4}	10^{-4}	cosine, 500-step warmup	200
GeoLAMP-B	2.5×10^{-5}	10^{-4}	cosine, 1000-step warmup	200
VAE	3×10^{-4}	10^{-4}	ReduceLROnPlateau (factor 0.5, pat. 5)	50
AD	10^{-4}	10^{-4}	ReduceLROnPlateau (factor 0.5, pat. 10)	100