Physically-Informed Flow Matching with Graph Neural Networks for Complex Fluid Dynamics

Xiaozhuang Song[1], **Tianshu Yu**[1]

[1]School of Data Science, The Chinese University of Hong Kong, Shenzhen
xiaozhuangsong1@link.cuhk.edu.cn, yutianshu@cuhk.edu.cn

Abstract

Computational fluid dynamics (CFD) simulations traditionally require extensive computational resources, limiting their utility in many scientific and engineering applications at scale. We introduce Physically-Informed Flow Matching Graph Networks (PIFM-GN), a novel generative framework that directly samples fluid states under specified physical conditions without requiring expensive time-stepping simulations. The key innovation of our approach is the incorporation of incompressibility constraints directly into the flow matching transport process by parameterizing velocity fields through vector potentials, with graph-based curl operators ensuring divergence-free predictions without requiring global pressure-Poisson solves. Experiments on diverse fluid dynamics problems – ranging from two-dimensional surface pressure distributions and complete flow fields, to complex three-dimensional airflow fields – demonstrate that PIFM-GN generates high-fidelity samples with significantly fewer sampling steps than diffusion-based alternatives. Most notably, our model maintains competitive performance even with a single sampling step, a regime where diffusion models completely fail. Our generated samples accurately reproduce the statistical characteristics of target flows, successfully capturing multi-modal pressure distributions across various flow conditions, while achieving significant computational speedups compared to diffusion-based methods. PIFM-GN thus enables efficient generation of fluid states for downstream analysis and design tasks in scientific and engineering applications.

Introduction

Computational fluid dynamics (CFD) simulation plays a critical role in scientific research and engineering design across numerous disciplines, from aerospace engineering to climate modeling (Karniadakis and Sherwin 2013; Verma, Novati, and Petros Koumoutsakos 2018; Kochkov et al. 2021). Despite advances in numerical methods, the high computational cost of traditional simulations, especially for complex geometries and turbulent flows, severely limits their practical large-scale application in design optimization, uncertainty quantification, and parameter space exploration (Pope 2004; Moin and Kim 1982; Maher, Milinski, and Ludwig 2021).

In real-world engineering applications, the primary interest lies in rapidly obtaining specific fluid states under given physical conditions (e.g., Reynolds number, geometry, boundary conditions) (Pope 2000; Caros et al. 2022; Giovanis et al. 2025). From these generated states, crucial statistical properties like mean flows, fluctuation intensities, spatial correlations can be derived (Jané-Ippel et al. 2023; Kochkov et al. 2024). Obtaining such states or their statistical moments traditionally requires intensive numerical simulations (Moin and Kim 1982), involving a lengthy transient phase before reaching equilibrium, small time steps for numerical stability and accuracy (especially for turbulent flows), and iterative solvers for handling constraints like incompressibility (Alfonsi 2009; Zhao et al. 2025; Liu et al. 2024).

Recent advances in generative deep learning have offered promising alternatives for directly sampling from complex distributions (Li et al. 2021; Lino et al. 2023; Gao, Kaltenbach, and Koumoutsakos 2024; Christopher, Baek, and Fioretto 2024; Wang et al. 2024; Jessica et al. 2024). In particular, diffusion models have been applied to fluid flow generation (Zhang et al. 2024; Valencia, Thuerey, and Pfaff 2024; Bastek, Sun, and Kochmann 2025; Xue et al. 2024), enabling direct sampling of states representative of equilibrium distributions without expensive transient simulations (Song, Meng, and Ermon 2021; Valencia, Thuerey, and Pfaff 2024; Lienen et al. 2024). However, these approaches face significant challenges: they typically require many sampling steps to achieve high-quality results and might violate physical conservation laws like incompressibility, necessitating expensive projection steps or post-processing corrections (Toshev et al. 2024; Lippe et al. 2023; Wu et al. 2024). Flow matching (Lipman et al. 2023) offers a more efficient alternative by defining a continuous-time vector field that directly transforms a simple noise distribution to the target distribution, enabling high-quality sampling with substantially fewer integration steps. Existing flow matching approaches have not been extensively adapted to unstructured mesh geometries nor explicitly designed to preserve critical physical constraints inherent in fluid dynamics (Rohbeck et al. 2025; Lipman et al. 2023).

We address these limitations by introducing Physically-Informed Flow Matching Graph Networks (PIFM-GN), a novel generative framework for fluid flows that combines the efficiency of flow matching with strict enforcement of physical conservation laws. Our key innovation is the incorporation of incompressibility constraints directly into the flow matching transport process through a vector potential parameteriza-

tion. By expressing velocity fields as the curl of appropriate potential functions (a vector potential in 3D or a scalar stream function in 2D), PIFM-GN guarantees zero-divergence by construction and eliminates the need for expensive projection steps (Alfonsi 2009; Pope 2004). Furthermore, PIFM-GN operates directly on unstructured meshes using graph neural networks with specialized differential operators that compute curls of potential functions on arbitrary geometries (Pfaff et al. 2021), naturally accommodating complex boundaries and enabling adaptive resolution allocation.

The contributions of this work are primarily threefold: We systematically validate flow matching as a highly efficient generative approach for complex CFD problems, achieving rapid sampling with significantly fewer integration steps than alternatives; We enhance physical fidelity by directly embedding the incompressibility constraint within the flow matching framework, thereby ensuring physically consistent generation without corrective post-processing; and We demonstrate robust, high-quality performance across diverse flow configurations, including challenging single-step sampling regimes where diffusion models typically fail. By combining the computational efficiency of flow matching with rigorous physical constraints, PIFM-GN enables rapid generation of physically-consistent fluid states under varying conditions, offering new insights for accelerated simulation and design optimization in fluid dynamics applications (Guest, Cranmer, and Whiteson 2018; Jumper et al. 2021).

Related Work

Deep Learning for Fluid Dynamics and Physical Systems. Deep learning has revolutionized numerous scientific domains, with fluid flow problems receiving significant attention (Guest, Cranmer, and Whiteson 2018; Jumper et al. 2021; Pathak et al. 2022; Morton et al. 2018; Bar-Sinai et al. 2019; Lino et al. 2023; Xu et al. 2023; Würth et al. 2024). Most early approaches focused on surrogate modeling of mean flows or deterministic time evolution, often employing convolutional neural networks (CNNs) restricted to regular grids (Morton et al. 2018; Bar-Sinai et al. 2019). For complex geometries and adaptive spatial resolution, graph neural networks (GNNs) have emerged as a powerful alternative, operating directly on unstructured meshes (Pfaff et al. 2021; Horie and Mitsume 2022; Lino et al. 2022). These mesh-based approaches have shown particular efficacy in modeling steady and unsteady flows involving complex boundaries and non-uniform spatial gradients (Belbute-Peres, Economon, and Kolter 2020; Lam et al. 2023). Multi-scale GNN architectures further enhance performance by enabling efficient information propagation across the domain (Gao and Ji 2019). Physics-informed approaches like HelmFluid (Xing et al. 2024) further demonstrate the potential of incorporating physical constraints directly into neural architectures.

Generative Models for Physical Fields. Probabilistic models including Gaussian mixture models (GMMs), variational autoencoders (VAEs), and generative adversarial networks (GANs) have been applied to model distributions over physical system states (Goodfellow et al. 2014; Maulik et al. 2020; Drygala et al. 2022; Kim and Lee 2020). Recently, denoising diffusion probabilistic models (DDPMs) have emerged as a powerful class of generative models, demonstrating superior performance in capturing distributional diversity compared to GANs (Ho, Jain, and Abbeel 2020; Nichol and Dhariwal 2021; Dhariwal and Nichol 2021). In the physical sciences, diffusion models have found applications in flow field super-resolution and reconstruction (Shu, Li, and Farimani 2023; Li et al. 2023), uncertainty estimation in under-resolved simulations (Liu and Thuerey 2024), and enhancing the long-term stability of autoregressive rollouts (Lippe et al. 2023; Rühling Cachay et al. 2024; Kohl, Chen, and Thuerey 2023).

Method

Problem Formulation

Given a computational mesh with nodes $\mathcal{V}_M$ and edges $\mathcal{E}_M$, where each node i is located at a fixed spatial position $\mathbf{s}_i \in \mathbb{R}^D$ ($D = 2$ or 3), we represent the fluid state at time t as a collection of field values at the mesh nodes: $\mathbf{x}(t) := \{\mathbf{x}_i(t) \in \mathbb{R}^F | i \in \mathcal{V}_M\}$. Here, $\mathbf{x}_i(t)$ denotes the vector of F physical field variables (e.g., velocity components, pressure) at node i and time t, formally written as $\mathbf{x}_i(t) \equiv \mathbf{x}(\mathbf{s}_i, t, \mathcal{G})$ (Pfaff et al. 2021; Belbute-Peres, Economon, and Kolter 2020). The graph structure $\mathcal{G}$ encodes both the mesh connectivity and conditional features such as boundary conditions and physical parameters. It is important to distinguish between the spatial coordinates $\mathbf{s}_i$ (which remain fixed for each mesh node) and the state variables $\mathbf{x}_i(t)$ (which evolve over time and represent the physical quantities we wish to model). In our generative modeling framework, $\mathbf{x}(t)$ represents points in the feature space rather than physical positions. After an initial transient phase, fluid systems reach statistical equilibrium, where the distribution of states depends only on the boundary conditions and physical parameters, not on the initial conditions (Moin and Kim 1982; Pope 2004). Our goal is to learn this equilibrium distribution of the field values, denoted $p(\mathbf{x}|\mathcal{G})$, and efficiently sample from it, generating physically consistent states without simulating the computationally expensive transient phase (Lino et al. 2023; Gao, Kaltenbach, and Koumoutsakos 2024).

Flow Matching for Generative Modeling

Given a base distribution $p_0(\mathbf{z})$ and a target distribution $p_1(\mathbf{z})$, flow matching constructs a time-dependent vector field $\mathbf{v}_t(\mathbf{x})$ that defines a continuous transformation between these distributions (Lipman et al. 2023; Rohbeck et al. 2025). The vector field satisfies the following ordinary differential equation (ODE):

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{v}_t(\mathbf{x}(t)), \quad \mathbf{x}(0) \sim p_0, \quad t \in [0, 1] \tag{1}$$

The flow matching objective is to learn a vector field $\mathbf{v}_\theta(t, \mathbf{x})$ parameterized by a neural network that approximates the ground truth vector field $\mathbf{v}_t(\mathbf{x})$. Following (Lipman et al. 2023), this can be achieved by minimizing the expected squared error:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t\sim\mathcal{U}(0,1),\mathbf{x}_0\sim p_0,\mathbf{x}_1\sim p_1}\left[\|\mathbf{v}_\theta(t, \mathbf{x}_t) - \mathbf{u}_t(\mathbf{x}_0, \mathbf{x}_1)\|_2^2\right] \tag{2}$$

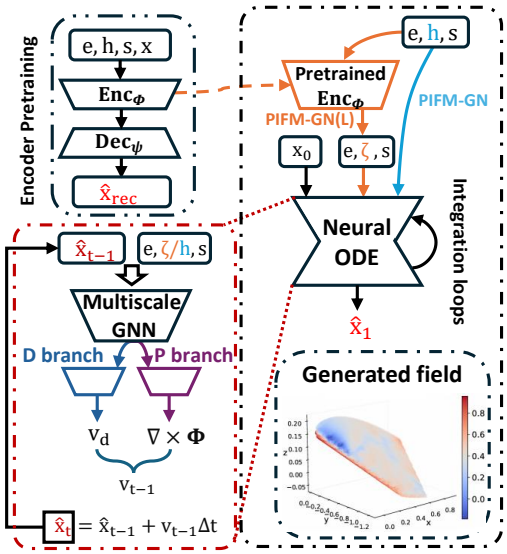

Figure 1: Overview of the PIFM-GN framework. A multi-scale GNN with dual-branch architecture predicts fluid fields, and Neural ODE integration generates new fluid samples. The latent flow matching variant PIFM-GN(L) uses a pre-trained VGAE encodes fluid states into latent space.

where $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ is a simple linear interpolation between source and target points, and the straight-line flow matching field is

$$\mathbf{u}_t(\mathbf{x}_0, \mathbf{x}_1) = \frac{d}{dt}\big((1-t)\mathbf{x}_0 + t\mathbf{x}_1\big) = \mathbf{x}_1 - \mathbf{x}_0.$$

For sampling, we solve the ODE by integrating the learned vector field as $\mathbf{x}(1) = \mathbf{x}(0) + \int_0^1 \mathbf{v}_\theta(t, \mathbf{x}(t))dt$. This integration can be numerically approximated using standard ODE solvers such as Euler or Runge-Kutta methods (Song, Meng, and Ermon 2021; Lienen et al. 2024).

Physically-Informed Flow Matching with Graph Neural Networks

Our PIFM-GN architecture extends the flow matching framework to graph-structured data representing fluid flows on unstructured meshes, while incorporating physical constraints to ensure generated samples satisfy conservation laws (Pfaff et al. 2021; Gao and Ji 2019). Figure 1 illustrates the overall architecture of our PIFM-GN framework.

Multi-Scale Graph Neural Network Architecture Building on the graph representation $\mathcal{G}$ introduced earlier, our architecture processes node features $\mathbf{h} \in \mathbb{R}^{|\mathcal{V}_M| \times F_h}$ and edge features $\mathbf{e} \in \mathbb{R}^{|\mathcal{E}_M| \times F_e}$, where F_h and F_e are the respective feature dimensions. The core of our model is a multi-scale graph neural network that operates at different resolutions, enabling efficient information propagation across the domain—crucial for capturing both local and global flow patterns (Gao and Ji 2019). The architecture follows a hierarchical structure with the following key components:

Embedding Modules: We employ three complementary embeddings to inject temporal, node, and geometric information into our GNN. First, a sinusoidal time embedding $\mathbf{e}_t(t) \in \mathbb{R}^{2d}$ encodes the flow-matching time step $t \in [0, 1]$. Its components are defined for $k = 1, \ldots, d$ as:

$$(\mathbf{e}_t(t))_{2k-1} = \sin(2\pi t\omega_k), (\mathbf{e}_t(t))_{2k} \quad = \cos(2\pi t\omega_k),$$

where $\omega_k = 10000^{-2(k-1)/d}$. This ensures a unique, smooth representation of time (Song, Meng, and Ermon 2021; Lino et al. 2023). Second, the node encoder concatenates each node's physical field values with any conditional features (e.g. boundary conditions, global parameters) and projects them into a learned feature space (Pfaff et al. 2021). Third, the edge encoder augments original edge attributes with radial-basis-function (RBF) distance embeddings to capture multi-scale spatial relations (Pfaff et al. 2021): for each edge (i, j) with Euclidean distance $d_{ij} = \|\mathbf{s}_j - \mathbf{s}_i\|_2$, we compute

$$\phi_k(d_{ij}) = \exp\Big[-\tfrac{(d_{ij}-\mu_k)^2}{2\sigma^2}\Big], \quad k = 1, \ldots, K,$$

where $\{\mu_k\}$ are uniformly spaced RBF centers within a distance cutoff and σ is the RBF bandwidth.

Multi-Scale Graph Processing: The processor consists of a series of downsampling, bottleneck, and upsampling blocks. Each down block applies several message-passing layers at the current resolution before downsampling the graph using a graph pooling operation based on coarsening masks that preserve the topological structure. At the coarsest resolution, message-passing layers in the bottleneck process global features. Each up block receives features from the previous layer, upsamples them to a finer resolution, and combines them with corresponding features from the encoding path through skip connections (Gao and Ji 2019). Message passing at each level follows:

$$m_{ij}^{(l)} = \text{MLP}(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \mathbf{e}_{ij}) \tag{3}$$

$$\mathbf{h}_i^{(l+1)} = \text{MLP}\left(\mathbf{h}_i^{(l)} + \text{AGG}_{j \in \mathcal{N}(i)}\left(m_{ij}^{(l)}\right)\right) \tag{4}$$

where $\mathbf{h}_i^{(l)}$ is the features of node i at layer l, $\mathbf{e}_{ij}$ is the features of edge (i, j), $\mathcal{N}(i)$ is the neighborhood of node i.

Physically-Informed Graph-Based Flow Matching In the flow matching framework, we learn a time-dependent transport vector field $\mathbf{v}_\theta(\mathbf{x}, t)$ that defines a continuous transformation from a simple base distribution to a complex target distribution over fluid states. Importantly, $\mathbf{x}$ here denotes a high-dimensional fluid state (e.g., velocity and pressure fields at mesh nodes), not a physical position. The transformation is governed by the ODE:

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}_\theta(\mathbf{x}, t), \quad t \in [0, 1],$$

which defines a trajectory in state space from $\mathbf{x}(0) \sim p_0$ to $\mathbf{x}(1) \sim p_1$ (Lipman et al. 2023; Song, Meng, and Ermon 2021). To ensure that the generated fluid states are physically plausible, we embed physical constraints—specifically incompressibility—directly into the learned velocity field.

Rather than predicting $\mathbf{v}_\theta$ directly and correcting its divergence post hoc, we enforce the incompressibility condition $\nabla \cdot \mathbf{v}_\theta = 0$ by construction. This is achieved by parameterizing the velocity field as the curl of a learned potential function: $\mathbf{v}_\theta = \nabla \times \Phi_\theta$, where Φ_θ is a scalar stream function in 2D or a vector potential in 3D. This formulation guarantees that the resulting velocity field is divergence-free by definition (Alfonsi 2009; Pope 2004). To support this construction on unstructured meshes, we implement all spatial derivatives as local, parallelizable operations. The gradient of a scalar field f at node i is approximated by:

$$(\nabla f)_i = \frac{1}{\sum_{j \in \mathcal{N}_i} w_{ij}} \sum_{j \in \mathcal{N}_i} w_{ij} \frac{f_j - f_i}{\|\mathbf{s}_j - \mathbf{s}_i\|} \frac{\mathbf{s}_j - \mathbf{s}_i}{\|\mathbf{s}_j - \mathbf{s}_i\|}, \quad (5)$$

where w_{ij} is a geometric weight (e.g., proportional to edge length or dual-cell area), and $\mathbf{s}_i$ denotes the fixed spatial position of node i. Similarly, the divergence of a vector field $\mathbf{v}$ is approximated as:

$$(\nabla \cdot \mathbf{v})_i = \frac{1}{\sum_{j \in \mathcal{N}_i} w_{ij}} \sum_{j \in \mathcal{N}_i} w_{ij} (\mathbf{v}_j - \mathbf{v}_i) \cdot \frac{\mathbf{s}_j - \mathbf{s}_i}{\|\mathbf{s}_j - \mathbf{s}_i\|}. \quad (6)$$

We evaluate $\nabla \Phi_\theta$ using edge-wise finite differences, and then assemble the node-wise curl via fixed linear combinations around minimal cycles (e.g., triangles) in each node's one-ring neighborhood. This graph-based curl operator ensures that the final velocity field is exactly divergence-free, while remaining fully local and efficient to compute.

Latent Flow Matching on Graph To scale our approach to very large meshes and complex 3D domains, we adopt a latent-graph strategy following the graph latent network framework (Pfaff et al. 2021). Following (Valencia, Pfaff, and Thuerey 2025), we used a pretrained Variational Graph Autoencoder (VGAE) with

$$\mathrm{Enc}_\phi : \mathbb{R}^{|\mathcal{V}| \times F} \times \{0,1\}^{|\mathcal{V}| \times |\mathcal{V}|} \to \mathbb{R}^{|\mathcal{V}_L| \times F_L}, \quad (7)$$

$$\mathrm{Dec}_\psi : \mathbb{R}^{|\mathcal{V}_L| \times F_L} \times \{0,1\}^{|\mathcal{V}_L| \times |\mathcal{V}_L|} \to \mathbb{R}^{|\mathcal{V}| \times F}. \quad (8)$$

using a standard reconstruction+ KL objective (Kim and Lee 2020). Once trained, the VGAE weights are frozen, and all subsequent flow-matching is carried out in the reduced latent space. Let $\zeta \in \mathbb{R}^{|\mathcal{V}_L| \times F_L}$ denote the latent node-feature tensor produced by the frozen encoder. The flow matching model receives latent features, edge attributes $\mathbf{e}$ and node positions $\mathbf{s}$ as input. We parametrize the latent velocity as

$$\mathbf{v}_\theta^L(t, \zeta, \mathbf{e}, \mathbf{s}) = \nabla \times \Phi_\theta(t, \zeta, \mathbf{e}, \mathbf{s}), \quad (9)$$

which guarantees divergence-free dynamics in latent space. Decoding via Dec_ψ produces a physical velocity field that remains exactly incompressible (Alfonsi 2009; Pope 2004).

Dual-Branch Output The multiscale GNN processes inputs $\mathbf{x}_{t-1}$ (previous state) and graph structure information ($\mathbf{e}$, $\mathbf{h}$, $\mathbf{s}$), outputting node features that are processed through two parallel branches. The direct branch consists of a simple MLP that outputs vector field predictions $\mathbf{v}_d(t, \mathbf{x}_{t-1}, \mathbf{e}, \mathbf{h}, \mathbf{s})$, while the potential branch (P branch) employs a network that outputs a potential function $\Phi_\theta(t, \mathbf{x}_{t-1}, \mathbf{e}, \mathbf{h}, \mathbf{s})$, which is then processed through a graph-based curl operator. The final velocity is

$$\begin{aligned} \mathbf{v}_\theta(t, \mathbf{x}_{t-1}, \mathbf{e}, \mathbf{h}, \mathbf{s}) = (1 - \lambda)\, \mathbf{v}_d(t, \mathbf{x}_{t-1}, \mathbf{e}, \mathbf{h}, \mathbf{s}) \\ + \lambda\, \nabla \times \Phi_\theta(t, \mathbf{x}_{t-1}, \mathbf{e}, \mathbf{h}, \mathbf{s}) \end{aligned} \quad (10)$$

where λ is a hyperparameter that balances the two contributions. This velocity is used to update the state via $\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{v}_{t-1} \Delta t$. For the latent flow matching variant PIFM-GN (L), the original node features $\mathbf{h}$ are replaced with latent features ζ from the pretrained encoder. We optimize the model by minimizing the mean squared error (MSE) loss between the predicted and target fluid fields.

Experiments

Experimental Setup

Datasets We consider three experimental domains following previous work (Lino et al. 2022; Valencia, Pfaff, and Thuerey 2025): (1) Ellipse: pressure on the wall of a 2D ellipse in quasi-periodic laminar flow, (2) EllipseFlow: velocity and pressure fields around that ellipse, and (3) Wing: pressure on a wing in 3D turbulent flow. The Ellipse and EllipseFlow data were adapted from (Lino et al. 2022), while the Wing dataset was adapted from (Valencia, Pfaff, and Thuerey 2025). For all datasets, we trained on a limited temporal window (10 consecutive states for Ellipse/EllipseFlow and 250 states for Wing). Detailed statistics, dataset split, and additional characteristics of these datasets are provided in the appendix.

Baselines We compare PIFM-GN with its latent flow matching variant PIFM-GN (L) with diffusion-based graph networks including DGN (Valencia, Pfaff, and Thuerey 2025) and LDGN (Valencia, Pfaff, and Thuerey 2025), and other approaches including Gaussian mixture GNN (GM-GNN) and Variational Graph Autoencoder (VGAE). All GNN models in our experiments use 4 scales with a hidden size of 128. Implementation details, detailed configurations, and additional information about the baselines are provided in the appendix.

Evaluation Setup Following (Valencia, Pfaff, and Thuerey 2025), we evaluate our models using several complementary metrics. For sample accuracy, we report the coefficient of determination (R^2) between generated samples and their ground-truth equivalents for the Ellipse and EllipseFlow datasets, with standard deviations across multiple runs. For distributional accuracy, we employ the Wasserstein-2 distance in two forms: (1) W_2^{node}, which treats each node's distribution independently and averages across nodes to assess point-wise statistics accuracy, and (2) W_2^{graph}, which considers the joint distribution across all nodes to capture spatial correlations. We evaluate generalization by testing on out-of-distribution parameters including variations in Reynolds numbers (LowRe, HighRe), ellipse thickness (Thin, Thick), and rotation. A critical aspect of our evaluation is the sampling efficiency, where we assess performance across different numbers of integration/denoising steps (T = 1, T = 10, T = 50), with particular emphasis on single-step (T = 1) performance to highlight computational benefits. We also assess computational efficiency by comparing the time required to generate

Model	Ellipse			EllipseFlow			Wing		
	$T = 50$	$T = 10$	$T = 1$	$T = 50$	$T = 10$	$T = 1$	$T = 50$	$T = 10$	$T = 1$
VGAE	0.986			0.971			0.968		
GM-GNN	0.971			0.973			0.978		
DGN	0.994	0.986	-6.037	0.993	0.964	-7.834	0.984	0.980	-2.522
LDGN	0.994	0.936	0.599	0.989	0.977	0.824	0.977	0.972	0.792
FMGN	0.996	0.988	0.985	0.992	0.987	0.970	0.984	0.984	0.983
FMGN (L)	**0.999**	0.997	0.995	0.992	0.991	0.983	0.983	0.985	0.984
PIFM-GN	**0.999**	0.995	0.993	**0.994**	0.995	0.974	**0.986**	**0.988**	0.986
PIFM-GN (L)	**0.999**	**0.998**	**0.996**	0.993	**0.996**	**0.987**	**0.986**	0.986	**0.987**

Table 1: Sample accuracy (R^2) comparison on the Ellipse, EllipseFlow and Wing datasets under different denoising steps T. Higher is better.

samples under different sampling steps, reporting both absolute times on CPU/GPU and relative speedups. We also conduct ablation studies on our model's physics-informed components and parameter sensitivity. More detailed descriptions and additional evaluations are provided in the appendix.

Sampling Performance Evaluation

We evaluate the performance of our PIFM-GN models in generating fluid states, focusing on sample quality, distributional accuracy, out-of-distribution generalization, and computational efficiency. We compare against several baselines, including diffusion-based models (DGN, LDGN), other flow-matching models (FMGN, FMGN (L)), and traditional graph-based methods (VGAE, GM-GNN).

Overall Sample Quality We first assess the quality of generated samples using visual inspection and the coefficient of determination (R^2) as a quantitative metric, where higher R^2 indicates better agreement with the ground truth. Our PIFM-GN models consistently generate high-quality fluid states. Table 1 provides a comprehensive R^2 comparison across the Ellipse, EllipseFlow, and Wing datasets for different numbers of sampling steps (T = 1, 10, 50). The latent variant, PIFM-GN (L), generally achieves the highest accuracy on the Ellipse dataset across different T, and also shows superior performance on the EllipseFlow dataset at lower sampling steps (T = 10 and T = 1). The direct PIFM-GN performs very competitively, showing slightly better results on the EllipseFlow dataset at T = 50 (0.994 *v.s.* 0.993). Interestingly, for FM-related models, we observe that excessive integration steps (higher T values) may sometimes adversely affect model performance in our experimental settings. For instance, PIFM-GN (L) achieves better results on EllipseFlow with T = 10 (0.996) compared to T = 50 (0.993), and similar patterns can be observed for PIFM-GN on both EllipseFlow and Wing datasets where moderate integration steps yield optimal performance. Both PIFM-GN variants significantly outperform the diffusion-based counterparts and other baselines, especially in the low-T regime. In summary, our PIFM-GN models demonstrate superior accuracy across all datasets, particularly excelling when using fewer sampling steps, which highlights their computational efficiency while maintaining high physical fidelity in complex fluid dynamics applications.

Single-Step Generation Capability

A critical advantage of our PIFM-GN models is their exceptional performance in single-step generation (T = 1), a scenario where traditional diffusion-based methods struggle significantly. This capability represents a substantial computational advantage for fluid dynamics applications, where rapid sample generation is often essential.

Sample Quality in Single-Step Generation As shown in Figure 2 and Figure 3, samples generated by PIFM-GN and PIFM-GN(L) with only T = 1 integration step are visually very close to the ground truth and achieve high R^2 values (e.g., $R^2 = 0.9867$ for PIFM-GN and $R^2 = 0.9871$ for PIFM-GN(L) on Wing; $R^2 = 0.9740$ for PIFM-GN and $R^2 = 0.9878$ for PIFM-GN(L) on EllipseFlow). In contrast, diffusion-based models like DGN and LDGN struggle significantly at T = 1, producing samples with poor visual quality and substantially negative or low R^2 values (e.g., $R^2 = -2.5224$ for DGN and $R^2 = 0.7927$ for LDGN on Wing; $R^2 = -7.8341$ for DGN and $R^2 = 0.8656$ for LDGN on EllipseFlow). GM-GNN performs better than diffusion models at T = 1 but generally lower than PIFM models ($R^2 = 0.9789$ on Wing, $R^2 = 0.9733$ on EllipseFlow).

Distributional Accuracy with Single Step Figure 4 presents the distributional comparison on the Ellipse dataset for T = 1, where both PIFM-GN variants capture the underlying distribution of fluid states remarkably well even with this minimal computational budget. The PIFM-GN(L) model achieves the lowest Wasserstein-2 distances (W_2^{node} and W_2^{graph}) among all models at T = 1, indicating superior distributional accuracy in single-step generation. Diffusion-based models (DGN, LDGN) show pronounced distortions and significantly higher Wasserstein distances at T = 1, highlighting their limitations in low-step sampling regimes.

OOD Generalization We tested generalization to out-of-distribution Reynolds numbers, ellipse thickness, and rotated ellipses in the Ellipse and EllipseFlow datasets. Table 2 presents the mean R^2 on various Ellipse dataset variants at T = 1, including out-of-distribution cases (-LowRe, -HighRe, -Thin, -Thick) and an in-distribution case (-InDist). Our PIFM-GN models, particularly PIFM-GN (L), exhibit excellent generalization, achieving the highest R^2 across all tested variants with only modest degradation compared to

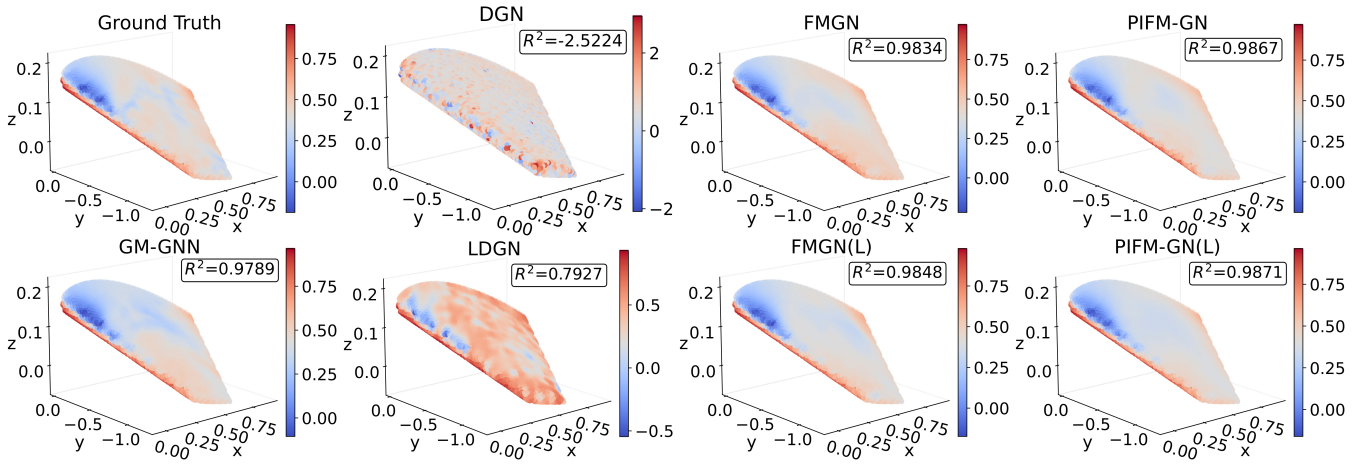

Figure 2: Sample visualizations (R^2) comparison on the Wing dataset with $T = 1$.

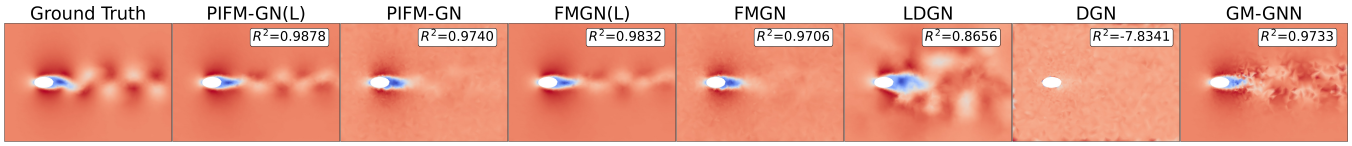

Figure 3: Sample visualizations (R^2) comparison on the EllipseFlow dataset with $T = 1$.

	-LowRe	-HighRe	-Thin	-Thick	-InDist
GM-GNN	0.968	0.895	0.976	0.887	0.971
VGAE	0.989	0.980	0.992	0.714	0.986
DGN	-9.669	-9.236	-8.921	-9.144	-6.037
LDGN	0.771	0.686	0.772	0.819	0.599
FMGN	0.985	0.984	0.983	0.970	0.985
FMGN (L)	0.997	0.994	0.953	0.987	0.995
PIFM-GN	0.989	0.989	0.972	0.983	0.993
PIFM-GN (L)	**0.998**	**0.997**	**0.993**	**0.989**	**0.996**

Table 2: Sample accuracy (R^2) comparison on the Ellipse datasets with ($T = 1$).

$T = 1$			
Model	R^2	CPU (s)	GPU (s)
PIFM-GN	0.986	7.499	0.653
PIFM-GN (L)	**0.987**	1.085	0.227
DGN	-2.241	6.819	0.596
LDGN	0.599	0.986	0.205
$T = 10$			
Model	R^2	CPU (s)	GPU (s)
PIFM-GN	**0.988**	75.312	6.603
PIFM-GN (L)	0.986	11.256	2.307
DGN	0.980	68.579	6.073
LDGN	0.972	10.243	2.302

Table 3: Performance comparison for sample generation and distribution estimation on the Wing dataset. Higher R^2 is better, lower time is better.

the in-distribution performance. For instance, PIFM-GN (L) maintains R^2 above 0.989 on all OOD Ellipse variants at $T = 1$, significantly outperforming baselines like DGN and LDGN which show severe performance drops or complete failure ($R^2 < 0.9$ or even negative). The physically-informed nature of our model—particularly the divergence-free guarantee—provides a strong inductive bias that enhances generalization to unseen conditions. This is especially evident in the EllipseFlow dataset (results not explicitly shown in Table 2 but discussed in text), where the physics-enforced velocity fields remain consistent even under parameter variations, leading to more realistic pressure distributions and vortex formations.

Multi-Step Performance While our models excel at single-step generation, comprehensive evaluations at higher sampling steps ($T = 10$ and $T = 50$) are also conducted to assess their full performance spectrum. As shown in Table 1, PIFM-GN models maintain their superior performance across different sampling budgets, consistently outperforming diffusion-based alternatives. Detailed visualizations and additional results for multi-step generation are provided in the appendix.

Computational Efficiency *v.s.* Performance Table 3 shows the trade-off between sampling steps T and inference cost on the Wing dataset, measuring statistical fidelity by R^2 and reporting CPU/GPU computation times. With single-step integration ($T = 1$), PIFM-GN achieves high accuracy ($R^2 = 0.986$) in 7.499s on CPU and 0.653s on GPU. The latent variant PIFM-GN (L) reduces these times to 1.085s and 0.227s (6.9× and 2.9× speed-ups) while maintaining fidelity ($R^2 = 0.987$). At $T = 10$, both models maintain performance ($R^2 \geq 0.986$) with continued acceleration benefits

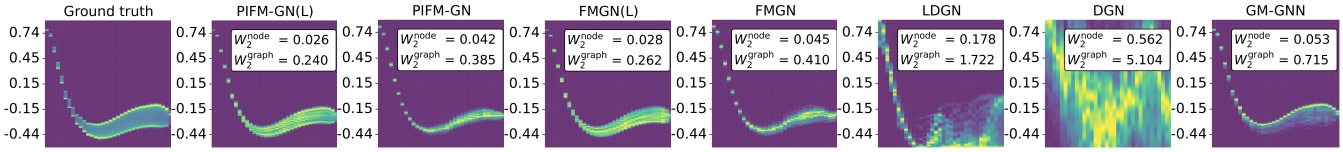

Figure 4: Distributional accuracy on Ellipse dataset (W_2^{graph}, W_2^{node},$T = 1$).

Variant	Ellipse	EllipseFlow	Wing
Full Model	**0.993**	**0.974**	**0.986**
Direct Only	0.988	0.971	0.984
Curl Only	0.975	0.963	0.970
w/o RBF	0.990	0.972	0.985
Full Model	**0.996**	**0.987**	**0.987**
Direct Only	**0.996**	0.981	0.985
Curl Only	0.980	0.970	0.975
w/o RBF	0.995	0.984	0.986

Table 4: Ablation of physics-informed components (R^2 at $T = 1$). "Direct Only" uses only the direct prediction branch. "Curl Only" uses only the potential-based curl branch.

for PIFM-GN (L). The diffusion baseline DGN fails at $T = 1$ ($R^2 = -2.241$) and only recovers at $T = 10$ ($R^2 = 0.980$) with much longer inference times (68.579s CPU, 6.073s GPU). LDGN achieves similar speed-ups but suffers lower fidelity at $T = 1$ ($R^2 = 0.599$). These results demonstrate that our framework achieves superior sample quality with minimal integration steps and significant speed-ups over diffusion methods, making it well-suited for real-time workflows. The superior $T = 1$ performance highlights PIFM-GN's single-step generation capability.

Ablation Studies

Impact of Physics-Informed Components Table 4 shows ablation results for $T = 1$ sampling. The full PIFM-GN model achieves best performance across datasets (R^2 of 0.993, 0.974, 0.986 for Ellipse, EllipseFlow, Wing). Using only the direct branch slightly reduces performance (0.988, 0.971, 0.984), while the curl-only branch causes larger degradation (0.975, 0.963, 0.970). This confirms that curl-based parameterization provides significant benefits while the direct branch contributes complementary information. Removing RBF embeddings also impacts performance (0.990, 0.972, 0.985), highlighting their importance for multi-scale spatial relationships. Similar trends appear in PIFM-GN (L), where the full model achieves superior performance (R^2 of 0.996, 0.987, 0.987). For the Ellipse dataset, the direct-only variant performs equally well (0.996), suggesting that for simpler geometries, direct prediction may suffice. These ablations show each physics-informed component contributes to single-step sampling effectiveness.

Effect of Training Set Size and Dual-Branch Weighting Figure 5(a) shows training set size impact on distributional error (W_2^{graph}) for the Ellipse dataset using $T = 50$ steps. All models improve with more training samples. PIFM-GN

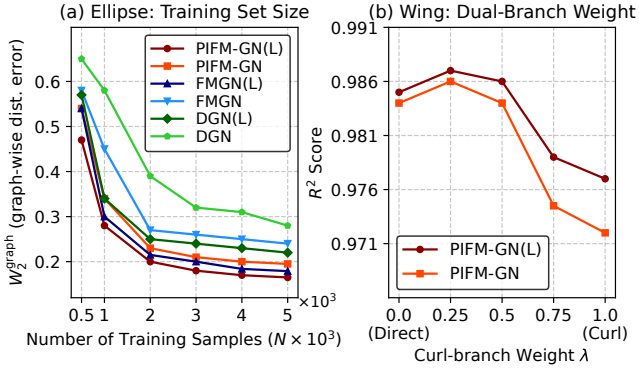

Figure 5: Model sensitivity analysis: (a) Effect of training set size for the Ellipse dataset, measured by graph-wise Wasserstein distance (W_2^{graph}). (b) Impact of dual-branch weighting parameter λ on R^2 performance for the Wing dataset.

(L) consistently achieves lowest error across all training sizes, with FMGN(L) second-best as N increases. PIFM-GN outperforms non-latent FMGN and both diffusion models, highlighting the effectiveness of physics-informed approaches in limited data scenarios. Figure 5(b) investigates dual-branch weighting λ on Wing dataset R^2 scores, where $\lambda = 0$ uses only direct prediction and $\lambda = 1$ uses only curl-based potential. Both PIFM-GN variants achieve optimal performance at $\lambda = 0.25$. Using only direct branch ($\lambda = 0$) yields slightly lower scores, while curl-only ($\lambda = 1$) causes noticeable degradation. This indicates the direct branch effectively learns complex features while moderate potential branch contribution provides beneficial regularization.

Conclusion

In this paper, we have introduced Physically-Informed Flow Matching Graph Networks (PIFM-GN), a novel generative modeling framework that addresses important limitations in computational fluid dynamics by enabling efficient, physically-consistent sampling of fluid states. Our experimental results demonstrate that PIFM-GN achieves superior sample quality and computational efficiency compared to existing methods. By bridging physically-informed modeling with efficient generative approaches, PIFM-GN offers a demonstration of how deep learning methods can complement traditional computational fluid dynamics, enabling new capabilities in scientific computation applications.

Acknowledgments

This work was supported by the National Science and Technology Major Project of China under Grant 2022ZD0116408.

References

Alfonsi, G. 2009. Reynolds-Averaged Navier-Stokes Equations for Turbulence Modeling. *Applied Mechanics Reviews*, 62(4): 040802.

Bar-Sinai, Y.; Hoyer, S.; Hickey, J.; and Brenner, M. P. 2019. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31): 15344–15349.

Bastek, J.-H.; Sun, W.; and Kochmann, D. M. 2025. Physics-informed diffusion models. In *International Conference on Learning Representations*.

Belbute-Peres, F. D. A.; Economon, T.; and Kolter, Z. 2020. Combining differentiable PDE solvers and graph neural networks for fluid flow prediction. In *International Conference on Machine Learning*, 2402–2411. PMLR.

Caros, L.; Buxton, O.; Shigeta, T.; Nagata, T.; Nonomura, T.; Asai, K.; and Vincent, P. 2022. Direct Numerical Simulation of Flow over a Triangular Airfoil Under Martian Conditions. *AIAA Journal*, 60(7): 3961–3972.

Christopher, J. K.; Baek, S.; and Fioretto, N. 2024. Constrained synthesis with projected diffusion models. *Advances in Neural Information Processing Systems*, 37: 89307–89333.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34: 8780–8794.

Drygala, C.; Winhart, B.; di Mare, F.; and Gottschalk, H. 2022. Generative modeling of turbulence. *Physics of Fluids*, 34(3).

Gao, H.; and Ji, S. 2019. Graph u-nets. In *international conference on machine learning*, 2083–2092. PMLR.

Gao, H.; Kaltenbach, S.; and Koumoutsakos, P. 2024. Generative learning for forecasting the dynamics of high-dimensional complex systems. *Nature Communications*, 15(1): 8904.

Giovanis, D. G.; Crabtree, E.; Ghanem, R. G.; and Kevrekidis, I. G. 2025. Generative learning of densities on manifolds. *arXiv preprint arXiv:2503.03963*.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 27.

Guest, D.; Cranmer, K.; and Whiteson, D. 2018. Deep learning and its application to LHC physics. *Annual Review of Nuclear and Particle Science*, 68(1): 161–181.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33: 6840–6851.

Horie, M.; and Mitsume, N. 2022. Physics-embedded neural networks: Graph neural pde solvers with mixed boundary conditions. *Advances in Neural Information Processing Systems*, 35: 23218–23229.

Jané-Ippel, C.; Bempedelis, N.; Palacios, R.; and Laizet, S. 2023. High-fidelity simulations of wake-to-wake interaction in an atmospheric boundary layer over a complex terrain. In *Journal of Physics: Conference Series*, volume 2505, 012033. IOP Publishing.

Jessica, L. S. E.; Arafat, N. A.; Lim, W. X.; Chan, W. L.; and Kong, A. W.-K. 2024. Finite Volume Features, Global Geometry Representations, and Residual Training for Deep Learning-based CFD Simulation. In *International Conference on Machine Learning*, 21555–21576. PMLR.

Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A.; et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873): 583–589.

Karniadakis, G.; and Sherwin, S. 2013. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press.

Kim, J.; and Lee, C. 2020. Deep unsupervised learning of turbulence for inflow generation at various Reynolds numbers. *Journal of Computational Physics*, 406: 109216.

Kochkov, D.; Smith, J. A.; Alieva, A.; Wang, Q.; Brenner, M. P.; and Hoyer, S. 2021. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21): e2101784118.

Kochkov, D.; Yuval, J.; Langmore, I.; Norgaard, P.; Smith, J.; Mooers, G.; Klöwer, M.; Lottes, J.; Rasp, S.; Düben, P.; et al. 2024. Neural general circulation models for weather and climate. *Nature*, 632(8027): 1060–1066.

Kohl, G.; Chen, L.-W.; and Thuerey, N. 2023. Benchmarking autoregressive conditional diffusion models for turbulent flow simulation. *arXiv preprint arXiv:2309.01745*.

Lam, R.; Sanchez-Gonzalez, A.; Willson, M.; Wirnsberger, P.; Fortunato, M.; Alet, F.; Ravuri, S.; Ewalds, T.; Eaton-Rosen, Z.; Hu, W.; et al. 2023. Learning skillful medium-range global weather forecasting. *Science*, 382(6677): 1416–1421.

Li, T.; Lanotte, A. S.; Buzzicotti, M.; Bonaccorso, F.; and Biferale, L. 2023. Multi-scale reconstruction of turbulent rotating flows with generative diffusion models. *Atmosphere*, 15(1): 60.

Li, Z.; Kovachki, N. B.; Azizzadenesheli, K.; Bhattacharya, K.; Stuart, A.; Anandkumar, A.; et al. 2021. Fourier Neural Operator for Parametric Partial Differential Equations. In *International Conference on Learning Representations*.

Lienen, M.; Lüdke, D.; Hansen-Palmus, J.; and Günnemann, S. 2024. From Zero to Turbulence: Generative Modeling for 3D Flow Simulation. In *International Conference on Learning Representations*.

Lino, M.; Fotiadis, S.; Bharath, A. A.; and Cantwell, C. D. 2022. Multi-scale rotation-equivariant graph neural networks for unsteady Eulerian fluid dynamics. *Physics of Fluids*, 34(8).

Lino, M.; Fotiadis, S.; Bharath, A. A.; and Cantwell, C. D. 2023. Current and emerging deep-learning methods for the simulation of fluid dynamics. *Proceedings of the Royal Society A*, 479(2275): 20230058.

Lipman, Y.; Chen, R. T.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2023. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*.

Lippe, P.; Veeling, B.; Perdikaris, P.; Turner, R.; and Brandstetter, J. 2023. Pde-refiner: Achieving accurate long rollouts with neural pde solvers. *Advances in Neural Information Processing Systems*, 36: 67398–67433.

Liu, Q.; and Thuerey, N. 2024. Uncertainty-Aware Surrogate Models for Airfoil Flow Simulations with Denoising Diffusion Probabilistic Models. *AIAA Journal*, 62(8): 2912–2933.

Liu, X.-Y.; Zhu, M.; Lu, L.; Sun, H.; and Wang, J.-X. 2024. Multi-resolution partial differential equations preserved learning framework for spatiotemporal dynamics. *Communications Physics*, 7(1): 31.

Maher, N.; Milinski, S.; and Ludwig, R. 2021. Large ensemble climate model simulations: introduction, overview, and future prospects for utilising multiple types of large ensemble. *Earth System Dynamics*, 12(2): 401–418.

Maulik, R.; Fukami, K.; Ramachandra, N.; Fukagata, K.; and Taira, K. 2020. Probabilistic neural networks for fluid flow surrogate modeling and data recovery. *Physical Review Fluids*, 5(10): 104401.

Moin, P.; and Kim, J. 1982. Numerical investigation of turbulent channel flow. *Journal of fluid mechanics*, 118: 341–377.

Morton, J.; Jameson, A.; Kochenderfer, M. J.; and Witherden, F. 2018. Deep dynamical modeling and control of unsteady fluid flows. In *Advances in Neural Information Processing Systems*.

Nichol, A. Q.; and Dhariwal, P. 2021. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 8162–8171. PMLR.

Pathak, J.; Subramanian, S.; Harrington, P.; Raja, S.; Chattopadhyay, A.; Mardani, M.; Kurth, T.; Hall, D.; Li, Z.; Azizzadenesheli, K.; et al. 2022. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*.

Pfaff, T.; Fortunato, M.; Sanchez-Gonzalez, A.; and Battaglia, P. 2021. Learning Mesh-Based Simulation with Graph Networks. In *International Conference on Learning Representations*.

Pope, S. B. 2000. *Turbulent flows*. Cambridge University Press.

Pope, S. B. 2004. Ten questions concerning the large-eddy simulation of turbulent flows. *New journal of Physics*, 6(1): 35.

Rohbeck, M.; De Brouwer, E.; Bunne, C.; Huetter, J.-C.; Biton, A.; Chen, K. Y.; Regev, A.; and Lopez, R. 2025. Modeling complex system dynamics with flow matching across time and conditions. In *International Conference on Learning Representations*.

Rühling Cachay, S.; Zhao, B.; Joren, H.; and Yu, R. 2024. Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. *Advances in Neural Information Processing Systems*, 36.

Shu, D.; Li, Z.; and Farimani, A. B. 2023. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478: 111972.

Song, J.; Meng, C.; and Ermon, S. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

Toshev, A. P.; Erbesdobler, J. A.; Adams, N. A.; and Brandstetter, J. 2024. Neural SPH: Improved Neural Modeling of Lagrangian Fluid Dynamics. In *International Conference on Machine Learning*, 48428–48452. PMLR.

Valencia, M. L.; Pfaff, T.; and Thuerey, N. 2025. Learning Distributions of Complex Fluid Simulations with Diffusion Graph Networks. In *International Conference on Learning Representations*.

Valencia, M. L.; Thuerey, N.; and Pfaff, T. 2024. Se (3)-equivariant diffusion graph nets: Synthesizing flow fields by denoising invariant latents on graphs. In *ICML 2024 AI for Science Workshop*.

Verma, S.; Novati, G.; and Petros Koumoutsakos. 2018. Efficient Collective Swimming by Harnessing Vortices through Deep Reinforcement Learning. *Proceedings of the National Academy of Sciences*, 115(23): 5849–5854.

Wang, C.; Berner, J.; Li, Z.; Zhou, D.; Wang, J.; Bae, J.; and Anandkumar, A. 2024. Beyond Closure Models: Learning Chaotic-Systems via Physics-Informed Neural Operators. In *NeurIPS Workshop on Data-driven and Differentiable Simulations, Surrogates, and Solvers*.

Wu, H.; Wang, H.; Wang, K.; Wang, W.; Tao, Y.; Chen, C.; Hua, X.-S.; Luo, X.; et al. 2024. Prometheus: Out-of-distribution fluid dynamics modeling with disentangled graph ode. In *International Conference on Machine Learning*.

Würth, T.; Freymuth, N.; Zimmerling, C.; Neumann, G.; and Kärger, L. 2024. Physics-informed MeshGraphNets (PI-MGNs): Neural finite element solvers for non-stationary and nonlinear simulations on arbitrary meshes. *Computer Methods in Applied Mechanics and Engineering*, 429: 117102.

Xing, L.; Wu, H.; Ma, Y.; Wang, J.; and Long, M. 2024. HelmFluid: Learning Helmholtz Dynamics for Interpretable Fluid Prediction. In *International Conference on Machine Learning*, 54673–54697. PMLR.

Xu, Q.; Shi, Y.; Bamber, J.; Tuo, Y.; Ludwig, R.; and Zhu, X. X. 2023. Physics-aware machine learning revolutionizes scientific paradigm for machine learning and process-based hydrology. *arXiv preprint arXiv:2310.05227*.

Xue, X.; Wang, S.; Yao, H.-D.; Davidson, L.; and Coveney, P. V. 2024. Physics informed data-driven near-wall modelling for lattice Boltzmann simulation of high Reynolds number turbulent flows. *Communications Physics*, 7(1): 338.

Zhang, X.; Helwig, J.; Lin, Y.; Xie, Y.; Fu, C.; Wojtowytsch, S.; and Ji, S. 2024. SineNet: Learning Temporal Dynamics in Time-Dependent Partial Differential Equations. In *International Conference on Learning Representations*.

Zhao, S.; Li, Z.; Fan, B.; Wang, Y.; Yang, H.; and Wang, J. 2025. LESnets (Large-Eddy Simulation nets): Physics-informed neural operator for large-eddy simulation of turbulence. *Journal of Computational Physics*, 114125.