TURTLES: A Large-Scale Benchmark for Turbulent 3D Neural PDE Surrogates

Anonymous Author(s)
Affiliation
Address
email

Abstract

Machine-learning surrogates are increasingly used to accelerate computational fluid dynamics (CFD), yet progress is limited by the lack of benchmarks capturing realistic, time-dependent turbulent flows. We introduce TURTLES, a 13 TB dataset of high-fidelity implicit large-eddy simulations of three-dimensional turbulent cylinder wakes in the shear-layer transition regime. The dataset contains 380 trajectories of 400 time steps each, with 3–9 million points per frame sampled on irregular point clouds, spanning variations in geometry, Reynolds number, and angle of attack. Unlike existing datasets, TURTLES captures fully three-dimensional turbulence with an active energy cascade driven by vortex stretching, combining (i) dense irregular meshes, (ii) long-horizon temporal dynamics, and (iii) physically realistic turbulent behavior. We benchmark state-of-the-art neural operators on long autoregressive rollouts and identify key failure modes, including temporal instability, loss of small-scale energy, and challenges in learning on large irregular domains. TURTLES provides a new standard benchmark for evaluating surrogate models in high-resolution, industrial-scale CFD regimes. *Project page*: https://turtlesprojectml-alt.github.io/

1 Introduction

The simulation of physical phenomena is central to industrial workflows in design (Allen et al., 2022), forecasting (Richardson and Lynch, 1922), control (Brunton and Kutz, 2022), and decision-making (Lam et al., 2023). These phenomena are typically governed by partial differential equations describing the spatio-temporal evolution of physical fields. Because such systems are rarely analytically tractable, they are solved numerically, often at substantial computational cost (Greenspan and Ames, 1970; Quarteroni and Valli, 2008). Yet downstream decisions rarely require the full resolution produced by high-fidelity solvers (Bonnet et al., 2023; Zhiyin, 2015). This mismatch has motivated surrogate models: low-cost approximations that reproduce task-relevant physical behavior while avoiding repeated full-scale simulation. Deep learning has emerged as a powerful framework for surrogate modeling (Long et al., 2018; de Bezenac et al., 2018), with rapid progress in operator learning (Li et al., 2020; Lu et al., 2021; Serrano et al., 2024; Wu et al., 2024a; Alkin et al., 2024a), neural simulation (Brandstetter et al., 2022), and generative emulation (Serrano et al., 2025; Li et al., 2025; Koupaï et al., 2025). Its adoption in realistic simulation settings, however, remains limited by a circular dependency: accurate surrogates require large, high-quality datasets, but such datasets must typically be generated by the same numerical solvers or physical experiments that surrogates are meant to replace.

Recent datasets have begun to address this bottleneck, but each leaves a critical ‘axis uncovered. The Well (Ohana et al., 2025) provides approximately 15 TB of dynamical trajectories across sixteen physical settings on regular grids, with most three-dimensional cases capped at 128^3 resolution. DrivAerNet (Elrefaie et al., 2025) reaches industrial-scale three-dimensional geometries with tens

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

of millions of degrees of freedom on irregular meshes, but only at steady state. EAGLE (Janny et al., 2023) captures dynamic flows on irregular geometries, yet its $\mathcal{O}(10^3)$ points per frame are too coarse to resolve realistic three-dimensional flow structures. RealPDEBench (Hu et al., 2026) pairs real-world PIV measurements with matched simulations, but its fluid measurements are two-dimensional laser-sheet observations with only in-plane velocity components. It is therefore valuable for simulation–experiment alignment, but does not directly expose the three-dimensional velocity fields and multiscale structures needed to assess turbulent flow surrogates.

More broadly, many machine learning benchmarks conflate temporal complexity with turbulence (Oommen et al., 2025). A flow may be unsteady, chaotic, or visually complex without containing the mechanisms that define three-dimensional turbulence. Several datasets described as "dynamic fluid" benchmarks fall into this category: they consist of unsteady laminar flows, chaotic two-dimensional systems, or three-dimensional dynamics dominated by buoyancy, radiative forcing, magnetic effects, or other mechanisms rather than by an inertial cascade (Pfaff et al., 2021; Takamoto et al., 2024). In three dimensions, turbulence is sustained by vortex stretching, represented by the $(\boldsymbol{\omega} \cdot \nabla)\mathbf{u}$ term in the vorticity equation, which transfers kinetic energy from large eddies to progressively smaller scales until viscous dissipation closes the energy budget (Parnaudeau et al., 2008a). In two dimensions, this mechanism is absent; in other forced systems, it may be secondary. A surrogate trained where vortex stretching is absent or marginal is therefore not learning three-dimensional turbulence, but rather interpolation of unsteady or chaotic fields without the cascade physics that define turbulent flow. Standard error metrics often hide this distinction, which becomes visible only through energy spectra, fluctuation statistics, and regime-transition behavior.

At the same time, the machine learning community has made substantial progress on the ingredients required for realistic surrogate modeling: long-horizon stability on regular grids (McCabe et al., 2024; Lippe et al., 2023), generalization (Huang et al., 2022; Koupaï et al., 2024; Subramanian et al., 2023; Takamoto et al., 2023), neural operators for dense unstructured static fields (Li et al., 2023b; Wen et al., 2025; Luo et al., 2025b), and generative emulators for chaotic dynamics (Zhou et al., 2025; Pedersen et al., 2025). What remains missing is a benchmark that stresses these capabilities jointly: dynamic three-dimensional trajectories, dense irregular discretizations, long temporal horizons, and physically turbulent flow driven by vortex stretching.

To fill this gap, we introduce TURTLES, a 13 TB dataset of high-fidelity simulations of three-dimensional cylinder wakes in the shear-layer transition regime. The dataset contains 380 trajectories of 400 time steps each, with 3–9 million degrees of freedom per frame sampled on irregular point clouds. Across trajectories, we vary the cylinder cross-section, Reynolds number, and angle of attack, producing geometry-dependent wakes that span subcritical to transitional regimes. In this range, shear-layer transition generates a turbulent near wake, so the dataset contains an active three-dimensional cascade in the simulated fields themselves.

TURTLES makes three contributions to surrogate modeling for PDEs. First, it provides dynamic three-dimensional flow data on dense irregular discretizations, reaching millions of points per frame rather than relying on regular grids or coarse point clouds. Second, it enables long-horizon evaluation over hundreds of autoregressive time steps in chaotic, geometry-dependent wakes. Third, it supports physically grounded assessment of turbulent surrogates through high-frequency quantities such as fluctuation statistics, near-wake spectra, and intermittent shedding-mode behavior. By combining industrial-scale spatial resolution, long temporal trajectories, irregular geometry, and physically turbulent flow, TURTLES targets a regime that existing benchmarks do not jointly cover. It exposes whether learned surrogates can remain stable, resolve multiscale turbulent content, and generalize across geometry-dependent flow regimes, rather than only minimizing standard field-wise errors.

2 Related Work

Operator Learning Operator learning provides a way to train neural networks to approximate PDE solution maps directly between function spaces. Early neural operators such as DeepONet and FNO showed that these models can act as mesh-independent surrogates for PDEs (Lu et al., 2021; Li et al., 2020). Since then, the field has moved toward models that are more accurate on complex geometries, better at resolving multiscale and high-frequency features, and scalable to very large meshes.

Recent work has explored several directions. Multiresolution and wavelet-based operators improve the representation of fine-scale structures (Gupta et al., 2021), while latent and geometry-aware

formulations extend neural operators to irregular domains (Tran et al., 2023; Li et al., 2023c). Attention-based models such as OFormer, GNOT and Transolver further improve operator learning on point clouds and arbitrary domains by using graph or transformer encoders to handle irregular inputs and complex PDE dynamics (Li et al., 2023a; Hao et al., 2023; Wu et al., 2024b). However, full attention remains expensive for high-resolution fields.

This has motivated more efficient neural operator surrogates. Perceiver-style and latent-token models such as Aroma and UPT compress large inputs into compact latent representations, reducing the cost of learning on dense meshes (Wang and Wang, 2024; Alkin et al., 2024b). Hagnberger et al. (2025); Koupaï et al. (2025) proposed continuous convolution for learning hierarchical neural operators that can operate on very dense grids, allowing million point trajectory generation. Wen et al. (2025) uses geometry-aware multiscale graph/attention encoding to scale neural PDE surrogates to arbitrary large meshes, including industrial 3D CFD domains.

Datasets for Machine Learning Applications A key bottleneck in training neural operators for fluid dynamics is the limited availability of high-quality datasets. Generating such data requires solving the Navier-Stokes equations at high spatial and temporal resolution, which comes with substantial computational and storage costs. As a result, most existing datasets are restricted in either fidelity, dimensionality, or physical complexity.

A substantial body of work (Bonnet et al., 2023; Sung et al., 2025; Elrefaie et al., 2025; Kanchi et al., 2025; Paischer et al., 2025) has focused on performing steady-state flow simulations through the Reynolds-averaged Navier-Stokes (RANS) equations, where all turbulence fluctuations are modelled rather than computed directly. While these approaches significantly reduce computational complexity, RANS-based simulations suffer from well-documented limitations in capturing complex phenomena such as sharp pressure gradients, laminar-to-turbulent transition and flow separation (Wilcox et al., 1998). Furthermore, by definition, steady-state solutions neglect temporal dynamics, preventing operators from learning transient flow phenomena.

Therefore, several benchmarks based on unsteady flows have been introduced. A common cost-reducing strategy is to restrict simulations to two-dimensional domains (Huang et al., 2019; Viquerat and Hachem, 2020; Jakob et al., 2021; Pfaff et al., 2021; Han et al., 2022; Xu et al., 2023; Janny et al., 2023; Tali et al., 2024; Luo et al., 2024; Dapelo and Bridgeman, 2025), at the expense of losing relevant three-dimensional flow phenomena. Alternatively, higher-fidelity transient datasets generated using large-eddy simulation (LES) or direct numerical simulation (DNS) have been explored in well-known (multi-)physics benchmarks (Hassan et al., 2023; Chung et al., 2023; Koehler et al., 2024; Takamoto et al., 2024; Ohana et al., 2025; Mao et al., 2026). There, they are typically limited to regular grids and idealized configurations (e.g., homogeneous isotropic turbulence, no objects within the domain, periodic boundary conditions). These flow problems do not introduce geometric complexities or capture the anisotropic nature of real-world turbulent flows. It should be noted that scale-resolving simulations for machine learning applications have been conducted by Ashton et al. (2024, 2025) on the Ahmed and Windsor car bodies, respectively. However, due to storage constraints, only time-averaged volume data was made publicly available.

To the best of our knowledge, there is no dataset that combines three-dimensional, time-resolved high-fidelity CFD simulations with systematic variations in complex geometries while fully resolving a broad range of turbulent scales. The present study aims to address this gap by performing implicit large eddy simulations on irregular grids for the canonical cylinder flow case across a range of inflow and geometric configurations.

3 Dataset Presentation

The dataset captures a wide range of fluid–object interactions and wake dynamics through systematic variations in geometric parameters and inflow conditions. To accurately resolve the resulting flow features and their temporal evolution, costly high-fidelity numerical simulations are performed. Generating a large number of such trajectories requires a robust and fully automated data generation pipeline. The remainder of this section describes the components of this pipeline, including the governing fluid flow model, the definition of the parameter sampling space, the automated simulation setup, and the subsequent data processing.

Fluid flow simulation The fluid dynamics are governed by the Navier–Stokes equations, a system of equations that expresses the conservation of mass, momentum, and energy for a compressible fluid. Given a set of inflow and boundary conditions, characterized by the Reynolds number Re, the Mach number Ma and Prandtl number Pr, these equations describe how a fluid evolves over space and time. In this study, we focus our attention on the evolution of the pressure p, density ρ and velocity components u, v, w.

At sufficiently high Reynolds numbers, the Navier-Stokes equations give rise to turbulence, whose multi-scale, chaotic dynamics make direct numerical solution computationally prohibitive. We solve these equations using an implicit Large Eddy Simulation (iLES) approach, in which the most energetic turbulent scales are computed directly, while the truncation error of the numerical scheme acts as a model for the behaviour of the smaller, unresolved scales. In this context, the Flux Reconstruction (FR) scheme (Huynh, 2007) has proven to be effective and is implemented using PyFR (Witherden et al., 2014), an open-source, high-order accurate GPU-based solver. The reader is referred to Appendix D for quantity definitions, and to Appendix E for a detailed description of the compressible Navier-Stokes equations and the solution method used in this study.

Sampling space The dataset is constructed within a four-dimensional parameter space. Variations in geometry are defined using the super-ellipse (or Lamé curve), which consists of all points (x, y) satisfying:

$$\left|\frac{x}{a}\right|^n + \left|\frac{y}{b}\right|^n = 1, \tag{1}$$

where a and b represent half the horizontal and vertical dimensions of the object, respectively, as illustrated in Figure 1a. The aspect ratio is defined as $AR = a/b$. To reduce the number of variable parameters, $b = 1/2$ is fixed for all cases, while a is sampled log-uniformly over the range $[1/6, 3/2]$. This results in aspect ratios spanning from 1/3 to 3, with an equal proportion of cases having $AR < 1$ and $AR > 1$. The exponent n is selected from the discrete set $\{1, 2, 5, \infty\}$, corresponding to diamond, circular, rounded rectangle and rectangular shapes, respectively.

The inflow conditions are characterized by Reynolds numbers $Re \in [1000, 3900]$, where the length scale used in the definition of Re is based on the object height, $L = 2b$. Within this range, the wake is expected to be turbulent while the boundary layer remains laminar, consistent with observations from circular cylinder flow studies (Lienhard, 1966). The inflow angle of attack α is the counter-clockwise angle between the x-axis and the object's centreline. It is varied between 0 and 45 degrees, introducing unique flow separation behaviour for each configuration. Limiting the range to $45°$ prevents generating geometrically redundant cases. Both Re and α are varied uniformly and restricted to integer values for simplicity.

To obtain representative samples of the parameter space, we employ a Sobol' sequence (Sobol', 1967), which is a quasi-random low-discrepancy sequence designed to minimize clustering and gaps in high-dimensional sampling. Unlike related methods such as Latin Hypercube Sampling, it supports sequential sampling, making it well suited for dataset generation where the total number of cases need not be specified in advance. A four-dimensional unit hypercube is sampled $N_s = 380$ times and subsequently transformed to the previously defined (log-)uniform or discrete parameter ranges. The resulting space is visualized in Figure 1b together with histograms of the different parameters in *(c-f)*.

Simulation setup For each case, the super-elliptical object is placed inside the 3D computational domain in Figure 2a at an angle α. The flow is prescribed through boundary conditions placed on the walls (**A**-**G**). **A** is the inlet, **B**-**D** are the outlet and **E** is the object surface. Walls **F** and **G** represent the side-walls of the domain. They are assigned periodic boundary conditions, which means flow leaving **F** enters back through **G**. An automated routine is implemented to generate the computational mesh using Gmsh (Geuzaine and Remacle, 2009). In figure Figure 2b, we show the resulting mesh in the xy-plane for the domain presented in Figure 2a. This two-dimensional mesh is then extended into three dimensions by a 30-layer structured extrusion in the span-wise direction. The flux reconstruction method places solution nodes inside these elements. A single element contains 75-125 points, yielding total grid sizes ranging from 3.1M to 9.2M unique solution points across all cases. Additional information on the domain definition and meshing procedure are given in Appendix F.

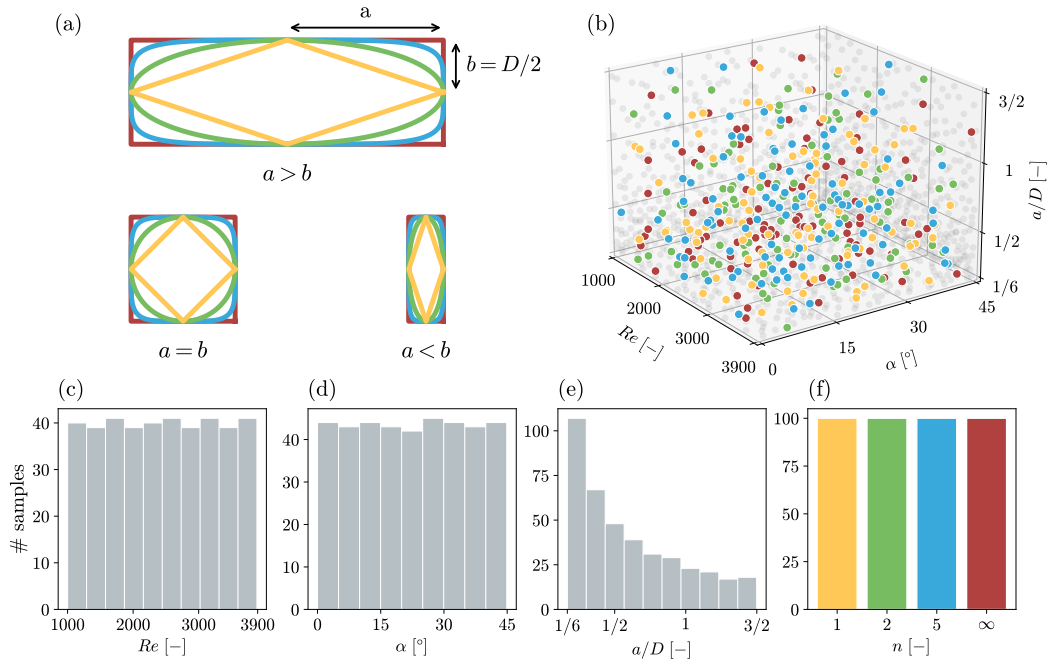


Figure 1: Overview of the parameter sampling space. (a) Definition of the hyper-elliptical geometry. (b) The $N_s = 380$ samples visualized in three dimensions, including projections on the x, y, z planes in grey for reference. (c-f) Histograms of the sampled parameters. Note that the colours used for different values of n in (f) correspond with those in (a) and (b).

The numerical framework is validated against four benchmark cases: a circular cylinder ($a = 0.5$, $n = 2$) at $Re = 1000$ and $Re = 3900$, a square cylinder ($a = 0.5$, $n = \infty$) at $Re = 2000$ and an elongated rectangle ($a = 2.5$, $n = \infty$) at $Re = 3000$. Overall, the results indicate good agreement with both DNS and experimental data reported in the literature. A detailed comparison is provided in Appendix G.

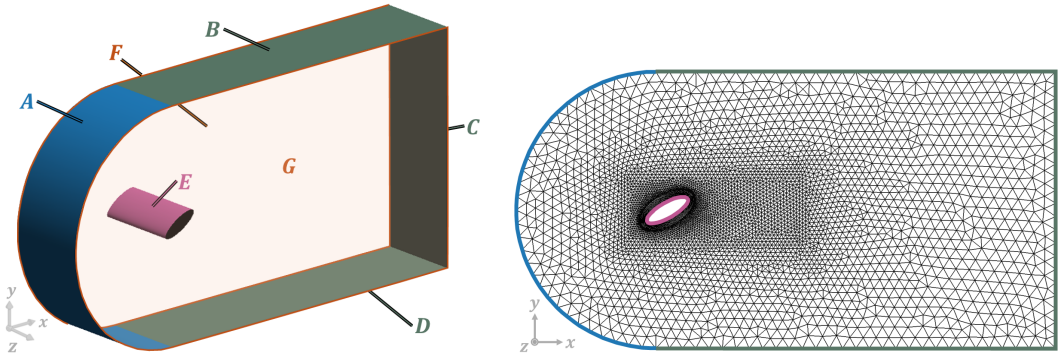


Figure 2: Schematic overview of the domain. (a) Three-dimensional view illustrating the physical boundaries, including the inlet (**A**), outlets (**B**–**D**), object surface (**E**), and periodic side-walls (**F**, **G**). (b) xy-plane of the computational mesh, which is extended into three dimensions via a structured extrusion in the span-wise (z) direction.

Data processing Processing steps are applied to limit the total data size while ensuring the solutions remain physically representative. Data collection starts only after the initial transient has been washed away according to the numerical convergence statistic proposed by Boudet et al. (2015). Subsequently,

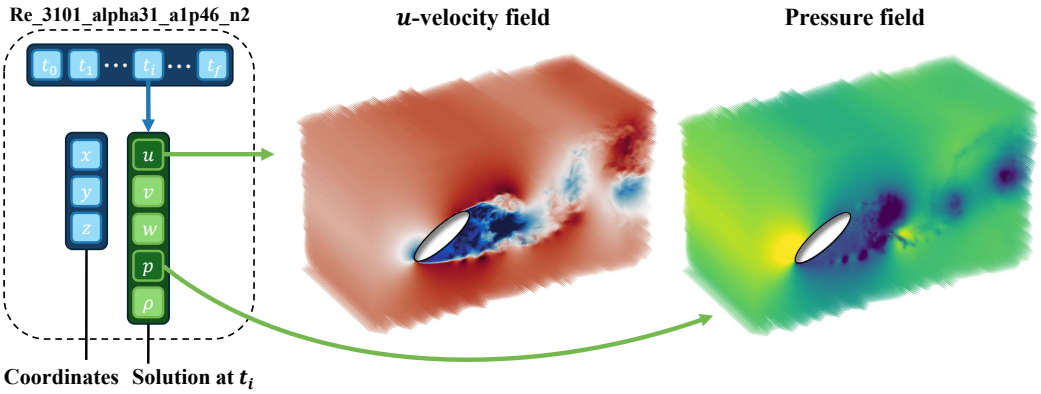


Figure 3: Overview of the flow case data structure. For each case, a single set of coordinates is stored with the solutions at 400 time-steps (e.g. $t_0, t_1, ..., t_f$, where $t_f = 399$). A solution comprises the three-dimensional pressure (p), density (ρ), and velocity (u, v, w) fields.

400 three-dimensional snapshots are saved of the three velocity components, pressure and density: $\{u, v, w, p, \rho\}$ at fixed time intervals Δt_s. A snapshot is restricted to all solution nodes inside the rectangular box given by $x/D \in [-3, 10]$, $y/D \in [-3.5, 3.5]$ and $z/D \in [0, 2\pi]$. The sampling interval Δt_s exceeds the solver time-step to manage storage constraints but is chosen such that the inertial range of the turbulence spectrum is retained for all cases. Ultimately, one trajectory covers a time-period in which most characteristic frequencies are repeated multiple times. For instance, four vortex shedding cycles are captured during $400\Delta t_s$ in the rounded cylinder case ($a = 0.5, n = 2$). Figure 3 visualizes the data structure of a single flow case. A case directory contains one set of node coordinates, as the grid remains constant throughout the entire trajectory, accompanied with the corresponding solutions for all 400 time-steps. Appendix H goes into more detail on the statistic used to measure the initial transient and how the interval Δt_s is defined.

4 Benchmarking Setup

Motivation and benchmark. To better demonstrate the value of this dataset, we evaluate the performance of current neural operators on these data. The machine learning task considered here is temporal prediction of several physical fields: pressure, density, and the three velocity components in the x, y, and z directions.

This dataset is designed to highlight some of the limitations of existing neural operators, especially for dynamic forecasting. In particular, handling dense meshes remains computationally expensive, stability over long rollout horizons is often poor, and high-frequency features are frequently underrepresented or smoothed out. More broadly, since the dataset is parametric, with variations in obstacle geometry, angle of attack, and Reynolds number, it provides a challenging benchmark for assessing the ability of neural networks to generalize across a rich distribution of physical configurations and to serve as accurate neural surrogates.

We evaluate several recent state-of-the-art neural operator architectures from the literature: UPT (Alkin et al., 2024b), GINO (Li et al., 2023c), Transolver++ (Luo et al., 2025a), and the continuous layer proposed in CALM-PDE (Hagnberger et al., 2025), combined with the scaling improvements introduced in ECHO for larger meshes (Koupaï et al., 2025).

Training setup. The models are trained to predict the temporal residual between time t and time $t + 1$ from a single input frame:

$$\mathrm{NO}(u_t, Re) + u_t \mapsto u_{t+1},$$

where the Reynolds number Re is also provided as an input. Learning the residual is generally easier for neural networks than directly predicting the full next state. The training loss is a standard mean squared error, computed on the predicted residual. Since the meshes are very large, a subset of points is sampled during training, and each model predicts the corresponding points at the next time step.

All models are trained for 100,000 steps with a batch size of 2, for approximately 20 hours on a single H100 GPU. We use the Adam optimizer with a weight decay of 10^{-3} and a learning rate of 10^{-4}. All models are trained with a linear warmup over the first 5,000 steps, followed by a cosine learning-rate scheduler.

Inference and evaluation setup. At inference time, each model rolls out the full dynamics using only the initial condition and the Reynolds number. Each trajectory contains 400 time steps, while the number of mesh nodes varies across trajectories; we therefore use a batch size of 1 and predict all nodes in each frame by iterating over subsets of points, since the full mesh cannot be processed in a single forward pass. The rollout produces predictions $\hat{u}_{t,n,c}$ and reference values $u_{t,n,c}$, where $t \in \{1, \dots, T_{\text{out}}\}$ indexes rollout steps, $n \in \{1, \dots, N\}$ indexes mesh nodes, and $c \in \{1, \dots, C\}$ indexes physical channels, with $C = 5$ corresponding to p, ρ, u, v, w.

We report four quantities computed from the autoregressive rollout. First, the relative L^2 error is

$$\text{Rel-}L^2 = \frac{1}{T_{\text{out}}} \sum_{t=1}^{T_{\text{out}}} \frac{\|\hat{u}_t - u_t\|_2}{\|u_t\|_2 + \varepsilon}, \qquad \varepsilon = 10^{-8}.$$

Second, because the physical channels span several orders of magnitude and unnormalised RMSE is dominated by p and ρ, we report the variance-scaled RMSE

$$\text{VRMSE} = \frac{1}{T_{\text{out}}} \sum_{t=1}^{T_{\text{out}}} \frac{\|\hat{u}_t - u_t\|_2}{\|u_t - \bar{u}_t\|_2 + \varepsilon}, \qquad \varepsilon = 10^{-8},$$

where $\bar{u}_t$ denotes the per-channel spatial mean of u_t; a predictor that outputs this per-frame spatial mean attains $\text{VRMSE} = 1$.

Third, to assess multi-scale accuracy, we compare isotropic energy spectra. For each frame, we extract a thin z-slice near the midplane and resample it onto a regular $H_g \times W_g$ grid with $H_g = W_g = 256$ using inverse-distance-weighted k-nearest neighbours with $k = 8$. Let $U_t(\mathbf{k})$ denote the two-dimensional Fourier transform of the resampled field. The isotropic power spectrum is

$$P_t(\kappa) = \frac{1}{|S_\kappa|} \sum_{\mathbf{k} \in S_\kappa} |U_t(\mathbf{k})|^2, \qquad S_\kappa = \left\{ \mathbf{k} : \kappa - \frac{\Delta\kappa}{2} \le \|\mathbf{k}\| < \kappa + \frac{\Delta\kappa}{2} \right\},$$

averaged over channels. We split the wavenumber support uniformly in $\log \kappa$ into low-, mid-, and high-frequency bands, $B_{\text{low}}, B_{\text{mid}}, B_{\text{high}}$, and report the RMS log-power discrepancy in each band:

$$\text{specRMSE}_B = \frac{1}{T_{\text{out}}} \sum_{t=1}^{T_{\text{out}}} \sqrt{\frac{1}{|B|} \sum_{\kappa \in B} \left(\log P_t^{\hat{u}}(\kappa) - \log P_t^{u}(\kappa) \right)^2}.$$

These three bands probe large, intermediate, and small flow scales, respectively, and penalise spectral mismatches that pointwise RMSE can average away.

Finally, we report inference latency as the wall-clock time required to predict one trajectory on the full mesh. Encoder–decoder baselines such as UPT and CALM-PDE encode the input mesh once and decode query points in chunks, whereas the monolithic Transolver baseline processes the mesh through a tile partition; all models use the same per-frame timing protocol, differing only in kernel-launch granularity.

5 Results

We evaluate the models along three complementary axes: rollout accuracy, spectral fidelity, and latency. These criteria are designed to separate errors that are often conflated in field-wise metrics. A model may achieve low pointwise error while dissipating small-scale turbulent content, or may perform well over short horizons while drifting during long autoregressive rollouts.

Dynamics forecasting. In table 1, we report models performance on the first frame prediction. It shows that CALM-PDE obtains the best field-level accuracy among the reported models, with the lowest relative L^2 error (6.80×10^{-2}) and the lowest VRMSE (1.33×10^{-2}). UPT is competitive in

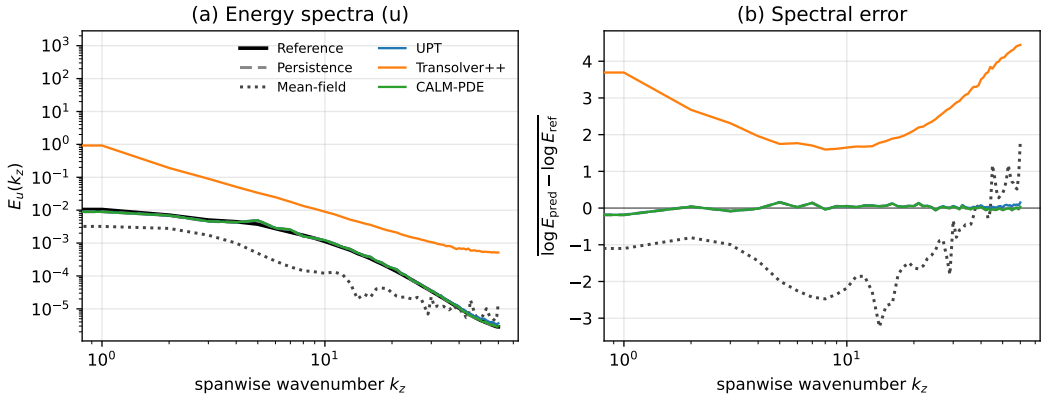


Figure 4: Spectral fidelity of autoregressive rollouts. Panel (a) shows reference and predicted isotropic energy spectra averaged over test trajectories. Panel (b) shows the log-power discrepancy across wavenumbers, with low-, mid-, and high-frequency bands indicated.

relative L^2 error, but its VRMSE is more than twice as large, indicating larger errors after accounting for the spatial variance of each frame. Transolver++ is substantially less accurate in this setting: its relative L^2 error is higher, and its VRMSE is close to the mean-field level. The spectral metrics further show that UPT and CALM-PDE preserve the low-, mid-, and high-frequency content with similar accuracy, while Transolver++ exhibits a much larger high-frequency error. This suggests that Transolver++ over-smooths or misrepresents small-scale structures in the turbulent wake.

Table 1: Next frame results on the test set. Lower is better for all metrics. Spectral errors are RMS log-power discrepancies computed over low-, mid-, and high-frequency wavenumber bands.

	Field error		Spectral error		
Model	Rel-L^2 ↓	VRMSE ↓	Low ↓	Mid ↓	High ↓
UPT	7.01e-2	2.90e-2	4.69e-2	2.33e-2	4.42e-2
GINO	1.32e-1	1.21	1.32e-1	7.31e-2	4.48e-2
Transolver++	1.08e-1	1.06	8.02e-2	2.89e-2	4.41e-1
CALM-PDE	6.80e-2	1.33e-2	4.66e-2	2.38e-2	4.44e-2

Long-range stability. Table 2 highlights a clear difference between short-horizon accuracy and long-horizon rollout stability. CALM-PDE remains stable throughout the autoregressive rollout, with VRMSE increasing only from 4.61×10^{-1} in the first 50 steps to 7.89×10^{-1} in the last rollout window. In contrast, UPT begins with a moderate error but drifts rapidly over time, reaching a VRMSE of 5.12 in the final window. Transolver++ is unstable from the beginning of the rollout and its error grows sharply, reaching 31.24 over steps 301–399. These results indicate that CALM-PDE better preserves the dynamics of the wake attractor over long autoregressive horizons, whereas the other reported baselines accumulate errors that eventually dominate the trajectory.

Table 2: Temporal stability of autoregressive rollouts. Entries report VRMSE averaged over different rollout windows. Lower is better.

Model	1–50 ↓	51–150 ↓	151–300 ↓	301–399 ↓
UPT	6.51e-1	1.56	3.31	5.12
GINO	3.18	10.32	24.31	33.12
Transolver++	2.33	8.32	19.33	31.24
CALM-PDE	4.61e-1	5.46e-1	6.52e-1	7.89e-1

Spectral fidelity. Pointwise errors do not fully reveal whether a surrogate preserves the multi-scale structure of the turbulent wake. We therefore compare isotropic energy spectra between predicted and

Table 3: Inference latency of surrogate models. Lower is better.

Model	Latency ↓
UPT	835 (s)
GINO	4097 (s)
Transolver++	1148 (s)
CALM-PDE	1073 (s)

reference trajectories. In Figure 4, spectra are averaged over test trajectories and rollout times, and the spectral error is shown as a function of wavenumber. A model that tracks large coherent structures but dissipates small eddies is expected to match low wavenumbers while underestimating high-wavenumber energy. We also report a persistence baseline, defined by the one-step rule $\hat{u}_{t+1} = u_t$; under autoregressive rollout, this corresponds to repeating the initial state. Because this baseline preserves a real turbulent snapshot, it can match the time-averaged energy spectrum reasonably well despite not predicting the temporal evolution of the wake. This highlights that spectral agreement alone is insufficient and should be interpreted together with field-level and long-horizon rollout errors. Among the learned models, CALM-PDE most closely matches the reference spectrum across the resolved range, with near-zero log-power discrepancy over low and intermediate wavenumbers. UPT follows the correct spectral decay but shows larger high-wavenumber errors, indicating reduced fidelity at small scales. In contrast, Transolver++ strongly overpredicts spectral energy, especially at intermediate and high wavenumbers, suggesting spurious small-scale content during rollout.

Latency. Finally, we compare the full-mesh inference latency of the neural surrogates in Table 3. All evaluated methods remain costly at this spatial resolution, requiring hundreds to thousands of seconds to predict a full trajectory. UPT is the fastest baseline, followed by CALM-PDE and Transolver++, while GINO is substantially slower than the other methods. These results show that, even when surrogate models improve rollout accuracy, scaling inference to dense irregular meshes with millions of points remains a major computational bottleneck.

Together, these evaluations expose complementary failure modes of neural PDE surrogates on TURTLES: short-horizon prediction error, long-horizon instability, loss of high-frequency turbulent energy, and insufficient computational acceleration on dense irregular meshes.

6 Conclusion

We introduced TURTLES, a large-scale benchmark for learning neural PDE surrogates in realistic three-dimensional turbulent flow regimes. The dataset contains 380 high-fidelity iLES trajectories of cylinder wakes across variations in geometry, Reynolds number, and angle of attack, with millions of irregularly distributed solution points per frame and hundreds of time steps per trajectory. By combining dense unstructured spatial discretizations, long temporal horizons, and physically turbulent wake dynamics, TURTLES targets a regime that is not jointly covered by existing machine-learning CFD benchmarks.

Our benchmark experiments show that current neural operators still face major challenges in this setting. While several models achieve reasonable short-horizon prediction accuracy, long autoregressive rollouts reveal substantial differences in stability. Spectral diagnostics further show that pointwise errors alone are insufficient: models may appear accurate while losing or misrepresenting turbulent energy at small scales.

TURTLES is intended to support progress toward neural surrogates that are not only accurate on average, but also stable over long horizons, faithful to multiscale turbulent structure, and efficient on industrial-scale meshes. Future work may extend the benchmark to additional geometries, higher Reynolds-number regimes, richer boundary conditions, and uncertainty-aware or generative surrogate models. We hope that TURTLES will provide a useful testbed for developing and evaluating next-generation neural operators for high-resolution computational fluid dynamics.

References

James Ahrens, Berk Geveci, and Charles Law. ParaView: An end-user tool for large data visualization. In *Visualization Handbook*. Elesvier, 2005. ISBN 978-0123875822.

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal Physics Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a. URL http://arxiv.org/abs/2402.12365.

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers. *arXiv preprint arXiv:2402.12365*, 2024b.

Kelsey R. Allen, Tatiana Lopez-Guevara, Kimberly Stachenfeld, Alvaro Sanchez-Gonzalez, Peter Battaglia, Jessica Hamrick, and Tobias Pfaff. Physical design using differentiable learned simulators, 2 2022. URL http://arxiv.org/abs/2202.00728.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, April 2024. doi: 10.1145/3620665.3640366. URL https://docs.pytorch.org/assets/pytorch2-2.pdf.

Neil Ashton, Danielle C. Maddix, Samuel Gundry, and Parisa M. Shabestari. Ahmedml: High-fidelity computational fluid dynamics dataset for incompressible, low-speed bluff body aerodynamics, 2024. URL https://arxiv.org/abs/2407.20801.

Neil Ashton, Jordan B. Angel, Aditya S. Ghate, Gaetan K. W. Kenway, Man Long Wong, Cetin Kiris, Astrid Walle, Danielle C. Maddix, and Gary Page. Windsorml: High-fidelity computational fluid dynamics dataset for automotive aerodynamics, 2025. URL https://arxiv.org/abs/2407.19320.

H. Bai and M. M. Alam. Dependence of square cylinder wake on reynolds number. *Physics of Fluids*, 30(1):015102, 2018. ISSN 1070-6631. doi: 10.1063/1.4996945. URL https://doi.org/10.1063/1.4996945.

Florent Bonnet, Ahmed Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier-stokes solutions, 2023. URL https://arxiv.org/abs/2212.07564.

J. Boudet, J.-F. Monier, and F. Gao. Implementation of a roughness element to trip transition in large-eddy simulation. *Journal of Thermal Science*, 24(1):30–36, 02 2015. ISSN 1993-033X. doi: 10.1007/s11630-015-0752-8. URL https://doi.org/10.1007/s11630-015-0752-8.

Johannes Brandstetter, Daniel E Worrall, and Max Welling. Message passing neural pde solvers. *International Conference on Learning Representations*, 2022.

C. Brun, S. Aubrun, T. Goossens, and Ph. Ravier. Coherent structures and their frequency signature in the separated shear layer on the sides of a square cylinder. *Flow, Turbulence and Combustion*, 81(1):97–114, Jul 2008. ISSN 1573-1987. doi: 10.1007/s10494-008-9152-4. URL https://doi.org/10.1007/s10494-008-9152-4.

Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2 edition, 2022.

Wai Tong Chung, Bassem Akoush, Pushan Sharma, Alex Tamkin, Ki Sung Jung, Jacqueline H. Chen, Jack Guo, Davy Brouzet, Mohsen Talei, Bruno Savard, Alexei Y. Poludnenko, and Matthias Ihme. Turbulence in focus: Benchmarking scaling behavior of 3d volumetric super-resolution with blastnet 2.0 data, 2023. URL https://arxiv.org/abs/2309.13457.

Andrea Cimarelli, Adriano Leonforte, and Diego Angeli. Direct numerical simulation of the flow around a rectangular cylinder at a moderately high Reynolds number. *Journal of Wind Engineering and Industrial Aerodynamics*, 174:39–49, March 2018. doi: 10.1016/j.jweia.2017.12.020.

Davide Dapelo and John Bridgeman. Fluidgpt: a proof-of-concept model for attention-based flow pattern generation in cfd, 01 2025.

Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. In *ICLR*, 2018.

Mohamed Elrefaie, Florin Morar, Angela Dai, and Faez Ahmed. Drivaernet++: A large-scale multi-modal car dataset with computational fluid dynamics simulations and deep learning benchmarks, 2025. URL https://arxiv.org/abs/2406.09624.

Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79:1309 – 1331, 09 2009. doi: 10.1002/nme.2579.

Sandip Ghosal. An analysis of numerical errors in large-eddy simulations of turbulence. *Journal of Computational Physics*, 125(1):187–206, 1996. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1996.0088. URL https://www.sciencedirect.com/science/article/pii/S0021999196900881.

Donald Greenspan and William F. Ames. Numerical methods for partial differential equations. *The American Mathematical Monthly*, 77, 1970. ISSN 00029890. doi: 10.2307/2318149.

Gaurav Gupta, Xiongye Xiao, and Paul Bogdan. Multiwavelet-based operator learning for differential equations. *Advances in neural information processing systems*, 34:24048–24062, 2021.

Jan Hagnberger, Daniel Musekamp, and Mathias Niepert. CALM-PDE: Continuous and adaptive convolutions for latent space modeling of time-dependent PDEs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=0r4yzkvt9j.

Xu Han, Han Gao, Tobias Pfaff, Jian-Xun Wang, and Li-Ping Liu. Predicting physics in mesh-reduced space with temporal attention, 2022. URL https://arxiv.org/abs/2201.09113.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. Gnot: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, pages 12556–12569. PMLR, 2023.

Sheikh Md Shakeel Hassan, Arthur Feeney, Akash Dhruv, Jihoon Kim, Youngjoon Suh, Jaiyoung Ryu, Yoonjin Won, and Aparna Chandramowlishwaran. Bubbleml: A multi-physics dataset and benchmarks for machine learning, 2023. URL https://arxiv.org/abs/2307.14623.

Peiyan Hu, Haodong Feng, Hongyuan Liu, Tongtong Yan, Wenhao Deng, Tianrun Gao, Rong Zheng, Haoren Zheng, Chenglei Yu, Chuanrui Wang, Kaiwen Li, Zhi-Ming Ma, Dezhi Zhou, Xingcai Lu, Dixia Fan, and Tailin Wu. Realpdebench: A benchmark for complex physical systems with real-world data. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=y3oHMcoItR. Oral Presentation.

Cheng Huang, Karthik Duraisamy, and Charles L. Merkle. Investigations and improvement of robustness of reduced-order models of reacting flow, 2019. URL https://arxiv.org/abs/1907.07846.

Xiang Huang, Zhanhong Ye, Hongsheng Liu, Shi Bei Ji, Zidong Wang, Kang Yang, Yang Li, Min Wang, Haotian CHU, Fan Yu, Bei Hua, Lei Chen, and Bin Dong. Meta-auto-decoder for solving parametric partial differential equations. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=PwlW5Jri1Xt.

H.T. Huynh. A flux reconstruction approach to high-order schemes including discontinuous galerkin methods. *AIAA Paper AIAA 20074079*, pages 1–42, 01 2007.

Jakob Jakob, Markus Gross, and Tobias Günther. A fluid flow data set for machine learning and its application to neural flow map interpolation. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1279–1289, 2021. doi: 10.1109/TVCG.2020.3028947.

Steeven Janny, Aurélien Béneteau, Madiha Nadri, Julie Digne, Nicolas Thome, and Christian Wolf. Eagle: Large-scale learning of turbulent fluid dynamics with mesh transformers, 2023. URL https://arxiv.org/abs/2302.10803.

Hongyi Jiang and Liang Cheng. Strouhal–reynolds number relationship for flow past a circular cylinder. *Journal of Fluid Mechanics*, 832:170–188, 2017. doi: 10.1017/jfm.2017.685.

Rohit Sunil Kanchi, Benjamin Melanson, Nithin Somasekharan, Shaowu Pan, and Sicheng He. Unifoil: A universal dataset of airfoils in transitional and turbulent regimes for subsonic and transonic flows, 2025. URL https://arxiv.org/abs/2505.21124.

Felix Koehler, Simon Niedermayr, Rüdiger Westermann, and Nils Thuerey. Apebench: A benchmark for autoregressive neural emulators of pdes, 2024. URL https://arxiv.org/abs/2411.00180.

Jean Kossaifi, Nikola Kovachki, Zongyi Li, David Pitt, Miguel Liu-Schiaffini, Robert Joseph George, Boris Bonev, Kamyar Azizzadenesheli, Julius Berner, Valentin Duruisseaux, and Anima Anandkumar. A library for learning neural operators. *arXiv preprint arXiv:2412.10354*, 2025.

Armand Kassaï Koupaï, Jorge Mifsut Benet, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Geps: Boosting generalization in parametric pde neural solvers through adaptive conditioning. *Advances in Neural Information Processing Systems*, 38, 2024.

Armand Kassaï Koupaï, Lise Le Boudec, Louis Serrano, and Patrick Gallinari. ENMA: Tokenwise autoregression for continuous neural PDE operators. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=3CYXSMFv55.

Armand Kassaï Koupaï, Lise Le Boudec, and Patrick Gallinari. Efficient generative transformer operators for million-point pdes, 2025. URL https://arxiv.org/abs/2512.04974.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Learning skillful medium-range global weather forecasting. *Science*, 382(6677): 1416–1421, 2023. doi: 10.1126/science.adi2336. URL https://www.science.org/doi/abs/10.1126/science.adi2336.

O. Lehmkuhl, I. Rodríguez, R. Borrell, and A. Oliva. Low-frequency unsteadiness in the vortex formation region of a circular cylinder. *Physics of Fluids*, 25(8):085109, August 2013. ISSN 1070-6631. doi: 10.1063/1.4818641. URL https://doi.org/10.1063/1.4818641. _eprint: https://pubs.aip.org/aip/pof/article-pdf/doi/10.1063/1.4818641/13982198/085109_1_online.pdf.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

Zijie Li, Anthony Zhou, and Amir Barati Farimani. Generative latent neural pde solver using flow matching. *arXiv preprint arXiv:2503.22600*, 2025. URL http://arxiv.org/abs/2503.22600.

Zong-Yi Li, Nikola B. Kovachki, Chris 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*, abs/2309.00583, 2023b. URL https://api.semanticscholar.org/CorpusID:261494027.

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*, 2020.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-informed neural operator for large-scale 3d pdes. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 35836–35854. Curran Associates, Inc., 2023c. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70518ea42831f02afc3a2828993935ad-Paper-Conference.pdf.

John H. Lienhard. Synopsis of lift, drag, and vortex frequency data for rigid circular cylinders. Technical Extension Service Bulletin 300, Washington State University, College of Engineering, Research Division, Pullman, WA, 1966.

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.

Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-net: Learning PDEs from data. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3208–3216. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/long18a.html.

Luiz M. Lourenço. Characteristics of the plane turbulent near wake of a circular cylinder. 1993. URL https://api.semanticscholar.org/CorpusID:210618974.

Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *Nat Mach Intell*, 3:218–229, 10 2021. doi: 10.1038/s42256-021-00302-5. URL http://arxiv.org/abs/1910.03193http://dx.doi.org/10.1038/s42256-021-00302-5.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=AM7iAh0krx.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=AM7iAh0krx.

Yining Luo, Yingfa Chen, and Zhen Zhang. Cfdbench: A large-scale benchmark for machine learning methods in fluid dynamics, 2024. URL https://arxiv.org/abs/2310.05963.

Runze Mao, Rui Zhang, Xuan Bai, Tianhao Wu, Teng Zhang, Zhenyi Chen, Minqi Lin, Bocheng Zeng, Yangchen Xu, Yingxuan Xiang, Haoze Zhang, Shubham Goswami, Pierre A. Dawe, Yifan Xu, Zhenhua An, Mengtao Yan, Xiaoyi Lu, Yi Wang, Rongbo Bai, Haobu Gao, Xiaohang Fang, Han Li, Hao Sun, and Zhi X. Chen. Benchmarking neural surrogates on realistic spatiotemporal multiphysics flows, 2026. URL https://arxiv.org/abs/2512.18595.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=DKSI3bULiZ.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Keaton Burns, Stuart B. Dalziel, Drummond B. Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R. Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H. Parker, Miles Cranmer, and Shirley Ho. The well: a large-scale collection of diverse physics simulations for machine learning, 2025. URL https://arxiv.org/abs/2412.00568.

Vivek Oommen, Aniruddha Bora, Zhen Zhang, and George Em Karniadakis. Integrating neural operators with diffusion models improves spectral representation in turbulence modeling, 2025. URL https://arxiv.org/abs/2409.08477.

Fabian Paischer, Leo Cotteleer, Yann Dreze, Richard Kurle, Dylan Rubini, Maurits Bleeker, Tobias Kronlachner, and Johannes Brandstetter. Going with the speed of sound: Pushing neural surrogates into highly-turbulent transonic regimes. *arXiv preprint arXiv:2511.21474*, 2025.

Philippe Parnaudeau, Johan Carlier, Dominique Heitz, and Eric Lamballais. Experimental and numerical studies of the flow over a circular cylinder at reynolds number 3900. *Physics of Fluids*, 20(8):085101, 2008a. doi: 10.1063/1.2957018.

Philippe Parnaudeau, Johan Carlier, Dominique Heitz, and Eric Lamballais. Experimental and numerical studies of the flow over a circular cylinder at Reynolds number 3900. *Physics of Fluids*, 20(8):085101, August 2008b. ISSN 1070-6631. doi: 10.1063/1.2957018. URL https://doi.org/10.1063/1.2957018. _eprint: https://pubs.aip.org/aip/pof/article-pdf/doi/10.1063/1.2957018/15878439/085101_1_online.pdf.

Christian Pedersen, Laure Zanna, and Joan Bruna. Thermalizer: Stable autoregressive neural emulation of spatiotemporal chaos. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=73zrUyI5kB.

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

Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

Alfio Maria Quarteroni and Alberto Valli. *Numerical Approximation of Partial Differential Equations*. Springer Series in Computational Mathematics, 2008. URL https://api.semanticscholar.org/CorpusID:120584359.

Lewis Fry Richardson and Peter Lynch. Weather prediction by numerical process, 1922. URL https://api.semanticscholar.org/CorpusID:62588316.

Louis Serrano, Lise Le Boudec, Armand Kassaï Koupaï, Thomas X Wang, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Operator learning with neural fields: Tackling pdes on general geometries. *Advances in Neural Information Processing Systems*, 36, 2024.

Louis Serrano, Armand Kassaï Koupaï, Thomas X. Wang, Pierre Erbacher, and Patrick Gallinari. Zebra: In-context generative pretraining for solving parametric PDEs. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=22kNOkkokU.

K. R. Sharma and S. Dutta. Flow Over a Square Cylinder at Intermediate Reynolds Numbers. *Journal of Fluids Engineering*, 144(12):121303, August 2022. ISSN 0098-2202. doi: 10.1115/1.4055126. URL https://doi.org/10.1115/1.4055126. _eprint: https://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/144/12/121303/6911777/fe_144_12_121303.pdf.

I.M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(67)90144-9. URL https://www.sciencedirect.com/science/article/pii/0041555367901449.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. *Advances in Neural Information Processing Systems 37 (NeurIPS 2023)*, 2023.

Bane Sullivan and Alexander Kaszynski. PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). *Journal of Open Source Software*, 4(37):1450, May 2019. doi: 10.21105/joss.01450. URL https://doi.org/10.21105/joss.01450.

Nicholas Sung, Steven Spreizer, Mohamed Elrefaie, Kaira Samuel, Matthew C. Jones, and Faez Ahmed. Blendednet: A blended wing body aircraft dataset and surrogate model for aerodynamic predictions. In *Volume 3B: 51st Design Automation Conference (DAC)*, IDETC-CIE2025. American Society of Mechanical Engineers, August 2025. doi: 10.1115/detc2025-168977. URL http://dx.doi.org/10.1115/DETC2025-168977.

Makoto Takamoto, Francesco Alesiani, and Mathias Niepert. Learning neural pde solvers with parameter-guided channel attention. *International Conference on Machine Learning (ICML)*, 2023.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning, 2024. URL https://arxiv.org/abs/2210.07182.

Ronak Tali, Ali Rabeh, Cheng-Hau Yang, Mehdi Shadkhah, Samundra Karki, Abhisek Upadhyaya, Suriya Dhakshinamoorthy, Marjan Saadati, Soumik Sarkar, Adarsh Krishnamurthy, Chinmay Hegde, Aditya Balu, and Baskar Ganapathysubramanian. Flowbench: A large scale benchmark for flow simulation over complex geometries, 2024. URL https://arxiv.org/abs/2409.18032.

The HDF Group. Hierarchical Data Format, version 5. URL https://www.hdfgroup.org/HDF5/. Version X.Y.Z.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tmIiMPl4IPa.

B. C. Vermeire, F. D. Witherden, and P. E. Vincent. On the utility of GPU accelerated high-order methods for unsteady flow simulations: A comparison with industry-standard tools. *Journal of Computational Physics*, 334:497–521, 2017. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2016.12.049. URL https://www.sciencedirect.com/science/article/pii/S0021999116307136.

P.E. Vincent, P. Castonguay, and A. Jameson. Insights from von neumann analysis of high-order flux reconstruction schemes. *Journal of Computational Physics*, 230(22):8134–8154, 2011. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2011.07.013. URL https://www.sciencedirect.com/science/article/pii/S0021999111004323.

Jonathan Viquerat and Elie Hachem. A supervised neural network for drag prediction of arbitrary 2d shapes in laminar flows at low reynolds number. *Computers Fluids*, 210:104645, 2020. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2020.104645. URL https://www.sciencedirect.com/science/article/pii/S0045793020302164.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse PDE problems. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=VLw8ZyKfcm.

Shizheng Wen, Arsh Kumbhat, Levi Lingsch, Sepehr Mousavi, Yizhou Zhao, Praveen Chandrashekar, and Siddhartha Mishra. Geometry aware operator transformer as an efficient and accurate neural surrogate for PDEs on arbitrary domains. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=HXFvNkNt0n.

David C Wilcox et al. *Turbulence modeling for CFD*, volume 2. DCW industries La Canada, CA, 1998.

F.D. Witherden, A.M. Farrington, and P.E. Vincent. Pyfr: An open source framework for solving advection–diffusion type problems on streaming architectures using the flux reconstruction approach. *Computer Physics Communications*, 185(11):3028–3040, 2014. ISSN 0010-4655. doi: https://doi.org/10.1016/j.cpc.2014.07.011. URL https://www.sciencedirect.com/science/article/pii/S0010465514002549.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A Fast Transformer Solver for PDEs on General Geometries. In *International Conference on Machine Learning*, 2024a. URL http://arxiv.org/abs/2402.02366.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *International Conference on Machine Learning*, 2024b.

Wenzhuo Xu, Noelia Grande Gutierrez, and Christopher McComb. Megaflow2d: A parametric dataset for machine learning super-resolution in computational fluid dynamics simulations. In *Proceedings of Cyber-Physical Systems and Internet of Things Week 2023*, CPS-IoT Week '23, page 100–104, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400700491. doi: 10.1145/3576914.3587552. URL https://doi.org/10.1145/3576914.3587552.

Yang Zhiyin. Large-eddy simulation: Past, present and the future. *Chinese Journal of Aeronautics*, 28(1):11–24, 2015. doi: 10.1016/j.cja.2014.12.007.

Anthony Zhou, Alexander Wikner, Amaury Lancelin, Pedram Hassanzadeh, and Amir Barati Farimani. Reframing generative models for physical systems using stochastic interpolants, 2025. URL https://arxiv.org/abs/2509.26282.

A Broader impact

TURTLES provides high-fidelity three-dimensional turbulent flow trajectories on dense irregular meshes, and is intended to serve as a benchmark for the development of neural surrogate models in computational fluid dynamics. We anticipate the following uses:

1. evaluating geometry-aware operator-learning models on dynamic, physically turbulent three-dimensional flow rather than on regular-grid or two-dimensional benchmarks;
2. studying the long-horizon stability and multiscale spectral fidelity of learned PDE surrogates through the evaluation protocol introduced in Section 5;
3. informing the design of new architectures for irregular meshes with millions of degrees of freedom, where memory and inference cost become first-order constraints;
4. extending point-cloud and graph-based benchmarks toward industrially relevant fluid-dynamics regimes;
5. supporting the development of surrogate solvers that accelerate early-stage design exploration in aerospace and automotive engineering;
6. studying the generalisation behaviour of neural operators across geometry, Reynolds number, and angle of attack in a controlled parametric setting.

We do not anticipate direct negative societal impact: the dataset is synthetic and contains no personally identifiable, geographic, or sensitive information. As with any data-driven surrogate, downstream practitioners should validate predictions before relying on them for safety-critical engineering decisions; surrogate models trained on TURTLES are not substitutes for verified high-fidelity simulation in such contexts.

B Reproducibility statement

We provide an anonymous repository containing (i) the code [1] used to generate every table and figure in the main paper; and (ii) a download link for the preprocessed dataset along with instructions for re-running the data generation pipeline (mesh generation with Gmsh, simulation with PyFR, post-processing into HDF5).

All baseline experiments were conducted on a single NVIDIA H100 80 GB GPU per run, with approximately 20 hours of wall-clock training time per model. Evaluation runs use the same hardware. Hyperparameters, data subsampling, optimisation schedule, and inference protocol are documented in Appendix I.

C Description of software

This work uses computational fluid dynamics, mesh-generation, and machine learning tools, all of them open source.

PyFR (Witherden et al., 2014). PyFR is an open-source GPU-accelerated solver for advection–diffusion-type problems, based on the Flux Reconstruction high-order accurate scheme. We use PyFR for all iLES simulations. We used version 2.0.3 of PyFR with CUDA 12 on NVIDIA H100 GPUs. PyFR is released under the BSD 3-Clause license.

Gmsh (Geuzaine and Remacle, 2009). Gmsh is an open-source three-dimensional finite-element mesh generator with a built-in CAD engine and post-processor. We use Gmsh, scripted from Python, to build all 380 meshes in the dataset (block-structured near-wall regions, unstructured wake regions, structured spanwise extrusion). We used version 4.11. Gmsh is released under the GNU GPL.

HDF5 / h5py (The HDF Group). The dataset is stored in HDF5 chunks (one coordinate file plus four temporal chunks of 100 frames per trajectory) and read at training and evaluation time through h5py. HDF5 is released under a permissive BSD-style licence; h5py is released under the BSD 3-Clause license.

[1] https://anonymous.4open.science/r/turtlescode-DC26/README.md

PyVista (Sullivan and Kaszynski, 2019). PyVista is an open-source tool for 3D visualization and mesh analysis built on top of the Visualization Toolkit (VTK). In this work, we use it to post-process the physical fields produced by our simulations. PyVista is released as free and open-source software under the MIT License.

ParaView (Ahrens et al., 2005). We use ParaView for qualitative visualisation of the simulated and predicted flow fields. We used version 5.13. ParaView is released under the BSD 3-Clause license.

PyTorch (Ansel et al., 2024). All neural-network training and inference is implemented in PyTorch. We used PyTorch 2.8 with CUDA 12.8 and Python 3.12. PyTorch is released under the BSD 3-Clause license.

neuraloperator (Kossaifi et al., 2025). The GINO baseline is built on the official `neuraloperator` library, which provides the GNO and FNO building blocks used in the encoder–FNO–decoder pipeline described in Appendix I. We used version 1.0.2. `neuraloperator` is released under the MIT license.

D Constant and Dimensionless Quantities

In compressible flow simulations, the dynamics are primarily governed by the Reynolds number (Re), Mach number (Ma) and Prandtl number (Pr). In addition, cylinder flows exhibit periodic unsteady behaviour which warrants the introduction of a non-dimensional frequency, namely the Strouhal number (St). We mathematically define these quantities as follows:

$$Re = \frac{\rho U L}{\mu}, \quad Ma = \frac{U}{c}, \quad Pr = \frac{\mu c_p}{k}, \quad St = \frac{fL}{U}. \tag{2}$$

U and L denote the characteristic velocity and length scale of the problem, respectively. In the present study, U is equal to the inflow velocity U_∞, and $L = 2b$ where $b = 0.5$ represents half the length of the object's front side facing the flow. Furthermore, ρ is the fluid density, μ the dynamic viscosity, c the speed of sound, c_p the specific heat, k the thermal conductivity, and f is a characteristic frequency associated with the flow. For all flow cases considered in this work, the Mach number $Ma = 0.2$ and Prandtl number $Pr = 0.71$ are kept constant while the Reynolds number is varied. Finally, time t is always presented in dimensionless form, normalized by the characteristic time scale L/U.

E Implicit Large Eddy Simulation

The Navier-Stokes equations represent the fundamental laws governing fluid dynamics. For a Cartesian coordinate system, Equation 3, Equation 4, and Equation 5 respectively denote a fluid's conservation of mass, momentum and energy:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0, \tag{3}$$

$$\frac{\partial}{\partial t}\rho u_i + \frac{\partial}{\partial x_j}(\rho u_i u_j) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\tau_{ij}, \tag{4}$$

$$\frac{\partial}{\partial t}E + \frac{\partial}{\partial x_j}(E u_j) = \frac{\partial}{\partial x_j}\left[u_i \tau_{ij} - q_j - p u_j\right], \tag{5}$$

where ρ is the fluid density, E is the total energy and u_i denotes the velocity in the x_i direction (with $i = 1, 2, 3$ corresponding to the x, y, z axes). Note that these equations utilize Einstein notation where a repeated index (i, j, k) implies a summation over all spatial dimensions. Furthermore, the stress tensor, strain tensor and heat flux are defined as:

$$\tau_{ij} = 2\mu\left[S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\right], \qquad S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \qquad q_j = \frac{\mu}{Pr}\frac{\partial h}{\partial x_j}. \tag{6}$$

Here, μ is the fluid's dynamic viscosity, Pr is the Prandtl number, and h is the specific enthalpy. Finally, the ideal gas equation of state provides a relation coupling the pressure to the total the energy:

$$p = (\gamma - 1)(E - \frac{1}{2}\rho u_i u_i), \tag{7}$$

where γ is the specific heat ratio. A solution to this system of equations only exists under simplifying assumptions, and thus numerical methods are used to approximate solutions for complex flow problems. The domain is then discretised into a grid on which flow variables are iteratively solved. To capture the entire turbulence spectrum of a fluid, Direct Numerical Simulation (DNS) employs a grid fine enough to resolve the smallest physical structures. The computational overhead is often intractable for industrially relevant applications, as the cost of fully resolving all turbulent scales in 3D Navier-Stokes simulations grows with Re^3, where Re denotes the Reynolds number. Instead, Reynolds-averaged Navier-Stokes (RANS) based simulations decompose an instantaneous fluid quantity into its time-averaged and fluctuating component, whereby the effects of the latter are represented by a turbulence model rather than computed directly. While this allows a significant reduction in computational cost, these methods exhibit limited accuracy for complex flow phenomena (e.g. laminar-to-turbulent transition, separation, three-dimensional effects) (Wilcox et al., 1998).

Large Eddy Simulation (LES) provides a high-fidelity alternative for simulating turbulent flows at reasonable cost. It is based on the separation between the energy-containing large scales of turbulence and the smaller high-frequency scales through a filter operation (Pope, 2000):

$$u_i(x,t) = \hat{u}_i(x,t) + u_i''(x,t). \tag{8}$$

The filtered flow variables ($\hat{u}_i$) are also referred to as the resolved variables, as they are directly solved for in the LES framework. The residual of the filter operation, u_i'', represents the unresolved turbulent motions. Numerous filtering strategies exist, but often the computational mesh itself acts as a filter where the local grid size, Δ_{grid}, dictates the cut-off between represented turbulent scales and *sub-grid scales* (SGS). The scale separation of Equation 8 is physically motivated by each component's distinct behaviour. The resolved scales are highly energetic, anisotropic, and strongly influenced by the flow geometry, which requires their direct computation. In contrast, the unresolved scales are more isotropic, carry less energy and exhibit a more universal behaviour, which allows for unified approaches to modelling their effects. Because these smaller scales are not resolved, LES utilizes significantly coarser meshes than DNS, leading to reduced computational complexity.

In explicit LES, an explicit model is used to replicate the effects of SGS turbulence, which entails providing the correct energy drain from the resolved flow to the smaller unresolved scales. This approach assumes that numerical errors associated with the discretisation scheme are small. In practice, this is often not the case, as even high-order schemes exhibit truncation errors of similar magnitude to SGS model contributions (Ghosal, 1996). Both truncation errors and SGS model terms are typically most significant at the smallest resolved scales. Due to the strong non-linear nature of Navier-Stokes, their combined effect can be difficult to anticipate. One way to address this is challenge is to apply an explicit filter at a scale larger than the grid spacing ($\Delta_{filter} > \Delta_{grid}$), effectively shifting the SGS model's activity away from the high-frequency truncation errors. However, this significantly increases computational cost as much finer grids are required to maintain the same level of resolved turbulence.

Alternatively, by recognising that some numerical schemes exhibit errors of similar dissipative nature as the effect of the sub-grid scales, implicit LES (iLES) abandons the use of an explicit SGS model and lets the truncation error serve as the energy drain from the resolved flow. The present study adopts this approach by employing the Flux Reconstruction (FR) or Correction Procedure via Reconstruction (CPR) scheme (Huynh, 2007). CPR functions well in the context of iLES as the scheme is inherently dissipative at high wave numbers (Vincent et al., 2011).

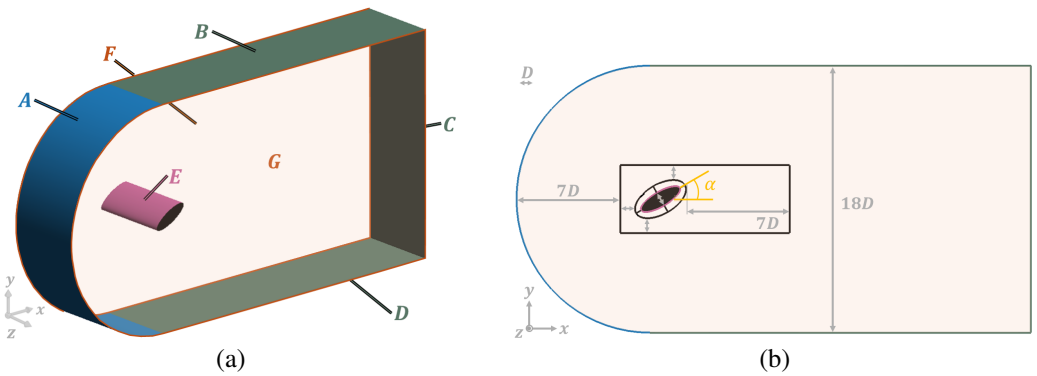
F Simulation setup

Domain and Boundary Conditions The compressible Navier-Stokes equations are solved within the three-dimensional domain in Figure 5a. This domain is generated by extruding the 2D plane depicted in Figure 5b by a spanwise length of 2π. The centre of the super-elliptical object **E** (coloured

Table 4: Boundary conditions of the simulation. Surface symbols refer to boundaries indicated in Figure 5a

Category	Symbol (in Figure 5a)	Boundary condition
Inlet	**A**	Riemann invariant
Outlet	**B-D**	Riemann invariant
Wall	**E**	Adiabatic no-slip wall
Left side	**F**	Periodic
Right side	**G**	Periodic

in pink) is placed at the origin $(x, y) = (0, 0)$ and rotated according to its prescribed inflow angle α. Rotating the object, rather than modifying the inflow angle of the flow field, is more convenient as it ensures the wake always propagates in the horizontal direction, which simplifies meshing and deciding what solution region to extract during data processing. The five blocks defined by the black lines in Figure 5b have no physical meaning but are used to control the automated meshing. The four blocks directly surrounding the physical object define the structured meshing region by placing a super-ellipse with a', b' and n' (see section 3) and connecting it to object **E** with parameters a, b and n. In all cases, $a' = a + D/2$ and $b' = b + D/2$ in order to create a thick enough structured mesh region that can capture the initial development of separated shear layers. n' is discretely defined as $1.5, 2, 5$, and 10 for $n = 1, 2, 5$, and ∞, respectively. These values mitigate mesh skewness, which can otherwise lead to numerical issues. The rectangular box is used to control the mesh nearby and behind the object, where a fine resolution is required to capture complex wake structures. Its front, top and bottom boundaries are located $1D$ from the structured mesh region, while the back edge extends to $7D$ behind the super-ellipses. Finally, the inlet **A** is positioned at $7D$ ahead of the rectangular wake box while the outlet curves **B-D** always extend until $x/D = 25$. Table 4 provides an overview of the boundary conditions placed on the domain.



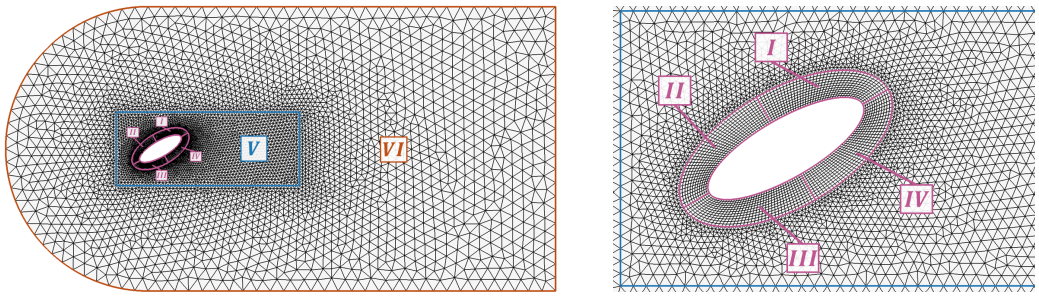
(a) (b)

Figure 5: Schematic overview of the computational domain. (a) Three-dimensional view illustrating the physical boundaries, including the inlet (**A**), outlets (**B**–**D**), object surface (**E**), and periodic sidewalls (**F**, **G**). (b) 2D side view showing characteristic dimensions relative to the cylinder diameter D. Note that black lines have no physical implications but indicate regions used for mesh refinement.

Meshing Generating a high-quality mesh is essential for achieving accurate and stable CFD simulations. Key considerations include minimizing element distortions, maintaining low skewness, ensuring gradual transitions in cell size, and providing adequate resolution. The latter is of particular importance in implicit large eddy simulations, where it is assumed that a significant portion of turbulence scales is directly resolved. In practice, a high-quality mesh is attained through case-specific iterative refinement, which is not feasible when constructing a large dataset. It was therefore chosen to adopt a simple meshing strategy that can be automated, accommodates a wide range of flow cases and limits the number of variable parameters. The routine is implemented using Python and Gmsh (Geuzaine and Remacle, 2009), an open source three-dimensional finite element mesh generator. Gmsh supports the generation of high-order meshes, which improve accuracy for domains with curved boundaries.

The xy-plane is partitioned into six blocks as illustrated in Figure 6. The dimensions of each block are defined in Figure 5b and remain consistent across all flow cases. Blocks **V** and **VI** are discretized using unstructured triangular elements, as they provide flexibility in dealing with geometric variations. For each block, a target cell size is set: l_{wb} (for **V**) and l_{ff} (for **VI**). Gmsh targets these sizes when possible while ensuring smooth transitions between regions of differing resolution. Blocks **I**-**IV** employ structured layers to give precise control over the near-wall discretization. Within these regions, the first element thickness l_t is chosen such that $y^+ < 1$. All other wall-normal cell sizes are determined through a geometric progression with a growth rate of 1.021. Furthermore, the cell size in the wall-parallel direction is set by l_p.

Mesh resolution is highly dependent on Reynolds number; for increasing Re, a finer mesh is required to capture the smaller turbulent structures. To determine the appropriate values of the parameters l_{wb}, l_{ff}, l_t, and l_p, mesh convergence studies are conducted at the edge cases of $Re = 1000$ and $Re = 3900$ in Appendix G. Between these Reynolds numbers, cell sizes are scaled using a power law interpolation. This resulting two-dimensional mesh is extended into three dimensions by a 30-layer structured extrusion in the span-wise direction.



(a) (b)

Figure 6: Two-dimensional (xy-)plane of the computational mesh. The full three-dimensional domain is a 30-layer extrusion in the z direction. (a) Full domain. (b) Zoomed-in view of the geometric object.

G Validation

The simulation is validated using DNS and experimental data of four benchmark cases: a circular cylinder flow ($a = 0.5$, $n = 2$) at $Re = 1000$ and $Re = 3900$, an elongated rectangle ($a = 2.5$, $n = \infty$) at $Re = 3000$ and a square rectangle ($a = 0.5$, $n = \infty$) at $Re = 2000$. These cases were selected to span a broad range of Reynolds numbers and to assess performance on sharp-edged and rounded geometries with varying aspect ratios. Figure 7 presents the overall statistics for each case together with reference data from literature. In general, the simulation shows good agreement with DNS and experimental results. Discrepancies are observed in drag prediction for very bluff bodies, but this appears to be a common trend for the studies included in the figure. These cases are discussed in detail below, with the cylinder flow cases also being used for a mesh convergence study.

Circular cylinder ($a = 0.5$, $n = 2$) **at** $Re = 1000$ **and** $Re = 3900$

The circular cylinder flow case is is used for a mesh convergence study, ensuring the numerical results are independent of spatial discretization. Simulations are conducted for 1500 non-dimensional time units on increasingly refined grids and compared against DNS data from Lehmkuhl et al. (2013). Refinements are achieved by adjusting the parameters l_{wb}, l_{ff}, l_t and l_p described in Appendix F, yielding computational meshes with 12.9M, 20.0M, 32.6M solution points, hereafter referred to as the coarse, medium and fine grids, respectively.

The flow around a circular cylinder at $Re = 3900$ has been studied extensively, yet varying results are reported in the literature. Lehmkuhl et al. (2013) addressed these inconsistencies by demonstrating that the flow exhibits bistable behaviour, oscillating between two modes, characterized by the length of the separated wake region, also referred to as the *recirculation length*. Previous studies relied on

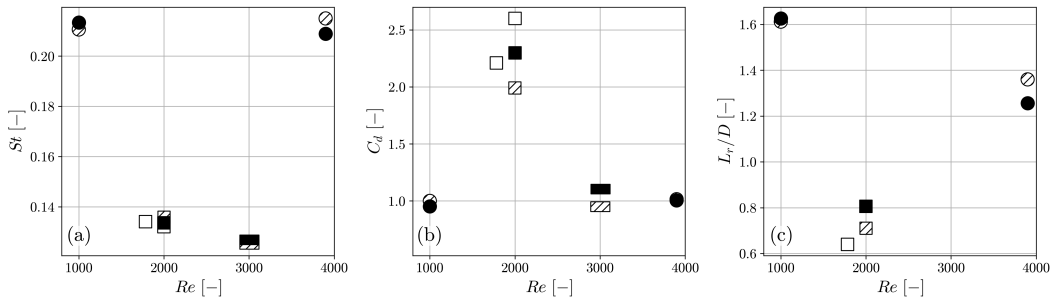


Figure 7: Comparison between present simulations (solid black markers) and validation data from the literature (open and hatched markers) for (a) Strouhal number, (b) drag coefficient, and (c) recirculation length. Validation data are taken at discrete Reynolds numbers from the following sources: Re=1000 (Jiang and Cheng, 2017); Re=1783 (Bai and Alam, 2018); Re=2000 (hatched) (Sharma and Dutta, 2022) and (open) (Brun et al., 2008); Re=3000 (Cimarelli et al., 2018); Re=3900 (Lehmkuhl et al., 2013).

observation windows that were too short to record both modes, leading to results that reflected either the short-bubble state (Lourenço, 1993) or the long-bubble state (Parnaudeau et al., 2008b).

Figure 8b displays the streamwise velocity along the centreline ($y = 0$) downstream of the cylinder. The three mesh resolutions yield nearly identical profiles in agreement with the DNS data, although a slightly shorter recirculation bubble is consistently observed. Figure 8a shows the pressure coefficient distribution over the cylinder surface, defined as $C_p = (p - p_\infty)/\frac{1}{2}\rho_\infty U_\infty^2$, where ∞ denotes free-stream conditions. Due to symmetry, the results from the upper and lower surfaces are averaged together. At $\theta = 0°$, the flow encounters the stagnation point where its velocity reduces to zero and C_p attains its maximum. For incompressible flow, as in Lehmkuhl et al. (2013), $C_p = 1$ is expected when the flow velocity is zero, consistent with Bernoulli's principle. In the present study, however, C_p slightly exceeds 1. This deviation can be attributed to compressibility effects, as the flow is computed at $Ma = 0.2$. Furthermore, minor discrepancies with the reference data are observed between 45 and 90 degrees, which is consistent with solutions that capture more of the short-bubble state.

Time-averaged wake velocities and their fluctuations at positions $x/D = 1.06, 1.54$ and 2.02 are presented in Figure 9 and Figure 10, respectively. Fluctuations are computed by subtracting the temporal mean from the velocity signal: $u' = u - \langle u \rangle$. The mean velocity profiles show an excellent agreement with Lehmkuhl et al. (2013). The only notable discrepancy occurs for the streamwise velocity at $x/D = 2.02$ where the simulations predict a slightly faster recovery of the velocity deficit, consistent with short-bubble solutions. The fluctuation profiles exhibit larger deviations from the reference data, a challenge commonly reported in the literature. For instance, Vermeire et al. (2017) observed over-predictions of similar magnitude and character when comparing second-order statistics to Lehmkuhl et al. (2013).

Comparing the different computational grids shows that the medium and fine grid produce equivalent results. In contrast, the coarse grid converges to a different solution for the pressure distribution in Figure 8b over the range $45° < \theta < 90°$ and across the fluctuation profiles in Figure 10, especially for $\langle v'v' \rangle$ at $x/D = 1.54$. Therefore, the medium grid is selected, as it ensures mesh-independent results at a reduced cost relative to the fine grid.

An identical mesh convergence procedure is performed for the $Re = 1000$ case using grids with 3.7M, 7.0M and 12.9M solution points. Due to the limited availability of reliable surface and wake validation data in the literature, detailed profile comparisons are omitted. Grid convergence is achieved on the the finest mesh (12.9M points) and its statistics are compared with Jiang and Cheng (2017). The Strouhal number, drag coefficient and recirculation length presented in Figure 7 demonstrate a strong agreement. Furthermore, close agreement is found for the lift fluctuations and base pressure coefficient. The present study yields $C_{l,\mathrm{rms}} = 0.097$ and $C_{pb} = -0.8388$, compared to 0.100 and -0.8148, respectively, reported by Jiang and Cheng (2017).

The Power Spectral Density (PSD) of the stream-wise and cross-stream velocity is used to further characterize the unsteady dynamics of the wake for the $Re = 3900$ case on the selected medium mesh.

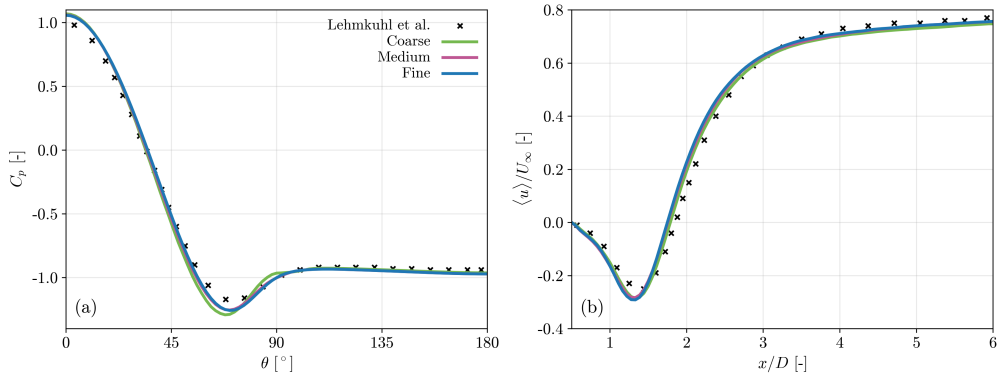


Figure 8: Results for the coarse, medium and fine computational grids: (a) pressure coefficient distribution over the cylinder surface as a function of the angular position θ, defined from the front stagnation point ($\theta = 0°$) and increasing along the cylinder surface in the clockwise direction; and (b) mean streamwise velocity along the centreline ($y = 0$) downstream of the cylinder. Validation data are taken from Lehmkuhl et al. (2013).

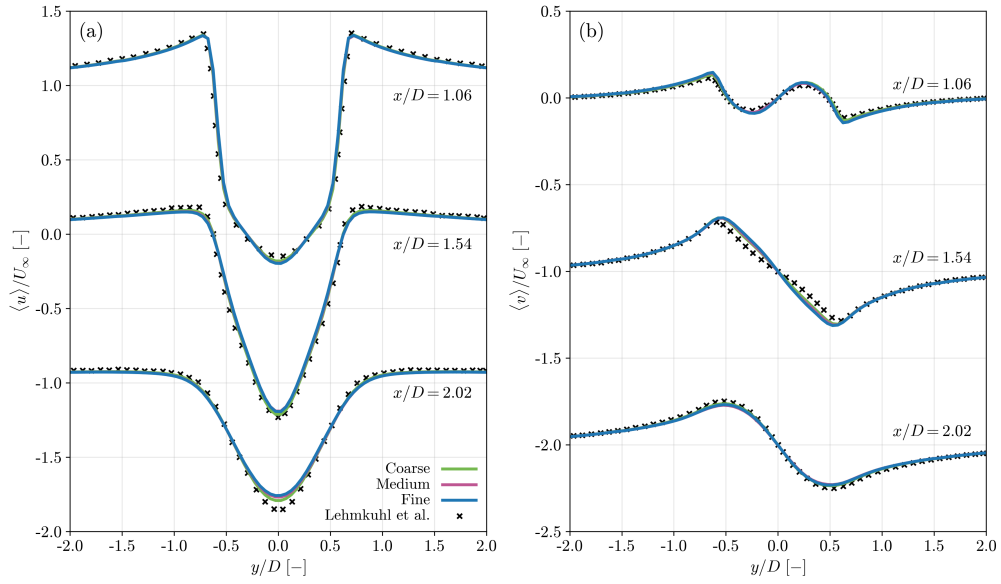


Figure 9: Results for the coarse, medium, and fine computational grids: mean wake velocity profiles of (a) streamwise component u and (b) transverse component v at downstream locations $x/D = 1.06$, 1.54, and 2.02. For clarity, profiles at $x/D = 1.54$ and 2.02 are shifted downward. Validation data are taken from Lehmkuhl et al. (2013).

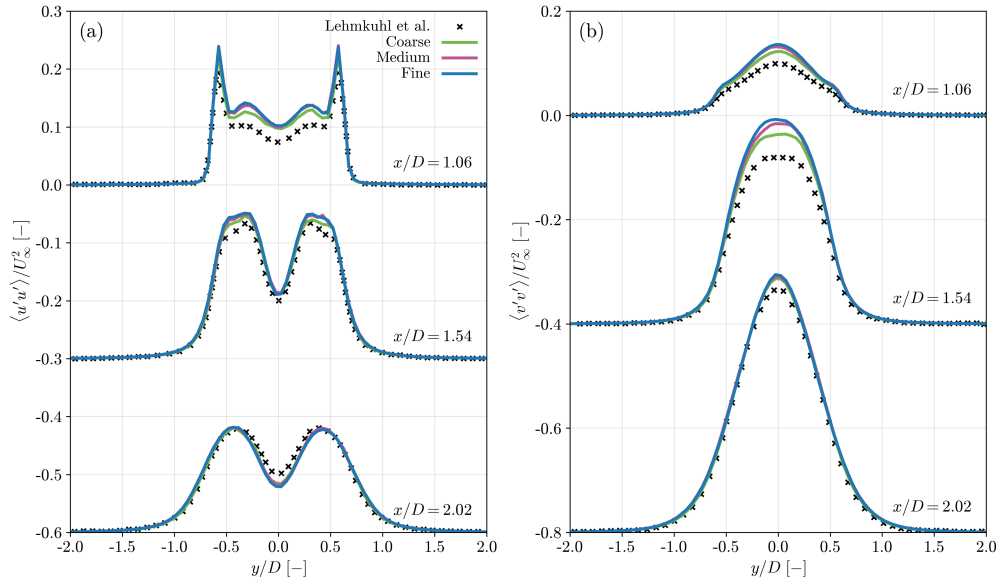


Figure 10: Results for the coarse, medium, and fine computational grids: mean fluctuation profiles of (a) streamwise component $u'u'$ and (b) transverse component $v'v'$ at downstream locations $x/D = 1.06$, 1.54, and 2.02. For clarity, profiles at $x/D = 1.54$ and 2.02 are shifted downward. Validation data are taken from Lehmkuhl et al. (2013).

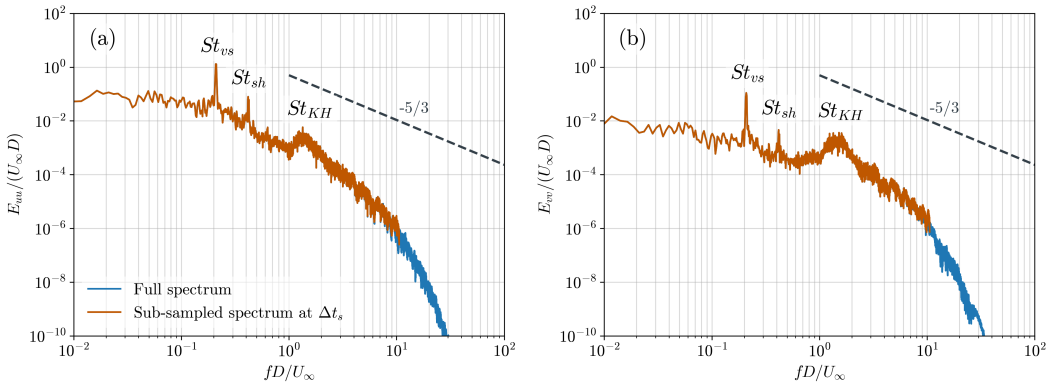


Figure 11: Comparison of velocity power spectra for (a) stream-wise and (b) cross-stream components at position $(x, y) = (0.71, 0.66)$. Spectra are shown for both the full temporal resolution and the sub-sampled resolution Δt_s. The characteristic frequencies St_{vs}, St_{sh}, and St_{KH} are annotated for reference alongside the theoretical $-5/3$ turbulence decay slope.

The PSD quantifies how the energy of velocity fluctuations is distributed across frequencies, thereby providing insight into dominant flow structures and their associated time scales. Figure 11 displays the PSD in the shear layer at $(x/D, y/D) = (0.71, 0.66)$, while Figure 12 shows the spectrum in the middle of the wake at $(x/D, y/D) = (3.0, 0.0)$. All subsequent discussions are based on the full spectrum result (blue line). The sub-sampled spectrum (orange line) is addressed separately in Appendix H, where the information loss caused by sub-sampling the solution, necessitated by storage constraints, is examined. Overall, the spectra are in good agreement with Lehmkuhl et al. (2013). The spectra exhibit the expected energy decay from low to high frequency scales, with an inertial range scaling in accordance with the theoretical $-5/3$ Kolmogorov law. Distinct peaks associated with characteristic flow phenomena are indicated in Figure 11. The vortex shedding peak at St_{vs}) represents the periodic release of vortices from the cylinder surface, while $St_{sh} = 2St_{vs}$ denotes the second harmonic of this process. Finally, the Kelvin-Helmholtz instability within the shear layer is observed at St_{KH}.

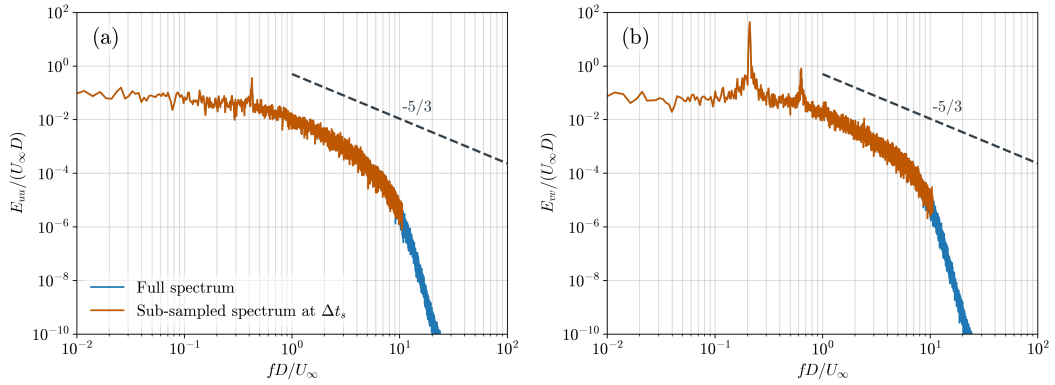


Figure 12: Comparison of velocity power spectra for (a) stream-wise and (b) cross-stream components at position $(x, y) = (3.0, 0.0)$. Spectra are shown for both the full temporal resolution and the sub-sampled resolution Δt_s. The theoretical $-5/3$ turbulence decay slope is indicated for reference.

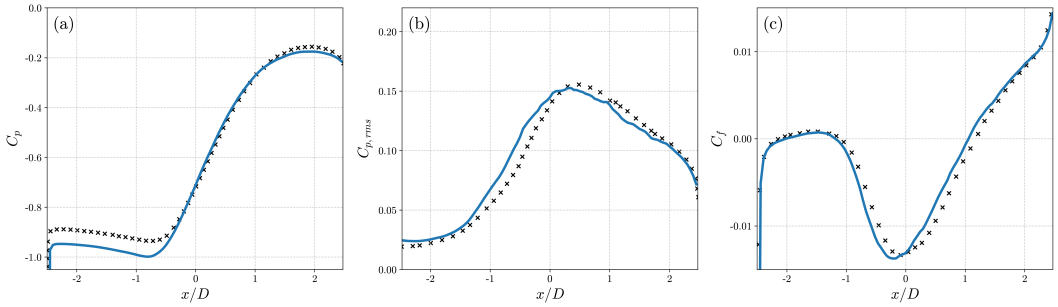


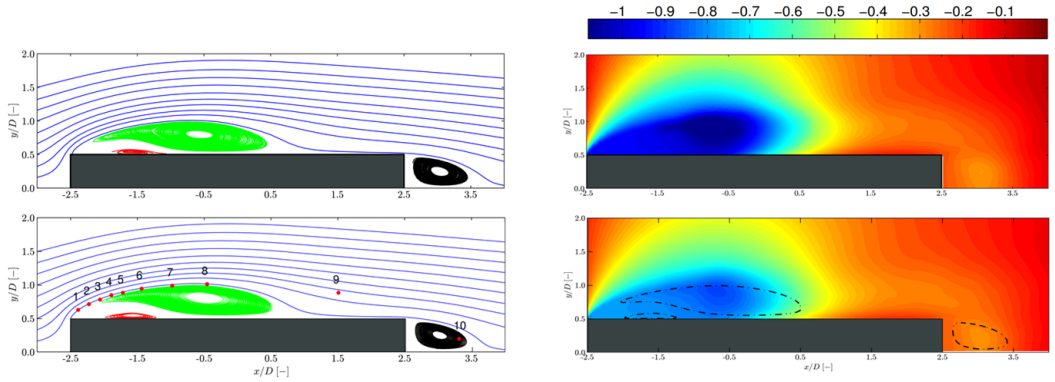
Figure 13: Comparison of (a) mean pressure coefficient distribution, (b) root-mean-square pressure coefficient fluctuations, and (c) skin-friction coefficient over the rectangle surface. Validation data are taken from Cimarelli et al. (2018).

Elongated rectangle ($a = 2.5$, $n = \infty$) **at** $Re = 3000$

The simulation setup and selected meshing parameters are further evaluated using the Benchmark on the Aerodynamics of a Rectangular 5:1 Cylinder (BARC). The reference data for this case are taken from Cimarelli et al. (2018), who performed DNS for this configuration. Although this specific rectangle is not included in the dataset, where elongations are restricted to $a \leq 1.5$, it provides a strict test case on the method's ability to handle object elongations. In contrast to the circular cylinder, the sharp corners induce fixed separation points while the extended flat top and bottom surfaces offer insight into the simulation's ability to capture near-wall dynamics. To this end, the mean and root-mean-square distributions of the surface pressure coefficient are presented in Figure 13, together with the skin friction coefficient, defined as $C_f = \tau_w/(\frac{1}{2}\rho u_\infty^2)$, where $\tau_w = \mu(\partial u/\partial y)_{\text{wall}}$. Overall, the profiles closely match with the DNS data, except for an under-prediction of the mean pressure over the first half of the rectangle. The mean pressure field in Figure 14b and streamlines of the time-averaged velocity in Figure 14a indicate this discrepancy is located in the region near the *primary vortex* and *secondary vortex*, for which the present simulations predict a more intense region of negative pressure. All other key phenomena, including the *wake vortex*, are accurately captured in both position and magnitude of the associated pressure field.

Square rectangle ($a = 0.5$, $n = \infty$) **at** $Re = 2000$

To further assess the performance of the numerical setup in the wake of sharp-edged geometries, simulations are conducted for the square cylinder at $Re = 2000$ and validated against experimental data from Sharma and Dutta (2022). Figure 15 provides a comparison of mean velocity and Reynolds stress profiles at various locations in the wake. Overall, the mean velocity deficit is accurately captured in the near wake and further downstream. A comparable agreement is observed for the



(a) (b)

Figure 14: Comparison between the present study (top row) and Cimarelli et al. (2018) (bottom row) of the mean flow field above the rectangular surface. Streamlines of the mean velocity field are shown in (a), and the mean pressure field in (b). In (a), green lines indicate the *primary vortex*, red lines the *secondary vortex*, and black lines the *wake vortex*.

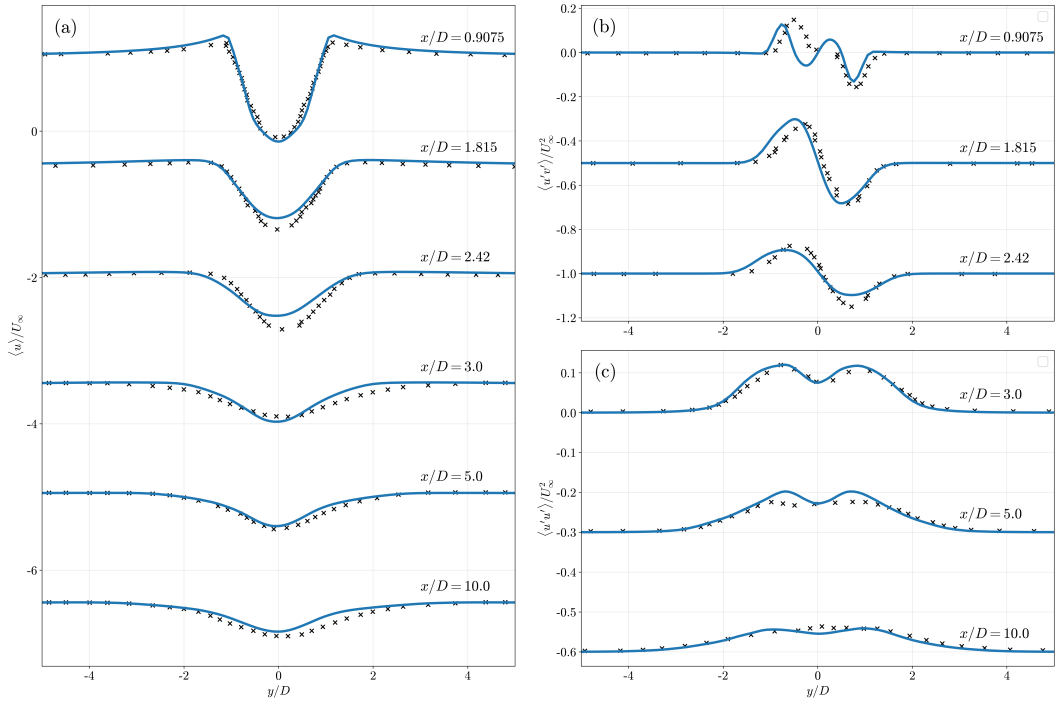


Figure 15: Mean velocity profiles in (a) from near to far downstream of the object, and Reynolds stress components: (b) $u'v'$ close to the cylinder and (c) $u'u'$ in the far wake region. Validation data are taken from Sharma and Dutta (2022).

Reynolds stresses close to the square. Notably, the simulation predicts a four-peak structure for $\langle u'v' \rangle$ at $x/D = 0.9075$. A comparable result is reported by Sharma and Dutta (2022) for a confined region around $x/D \approx 0.85$, beyond which the Reynolds stresses transition to the two-peak structure observed in Figure 15b. The present results therefore reproduce the four-peak behaviour, but its location is slightly shifted relative to the experiments. Finally, Reynolds stress profiles far downstream are accurately reproduced.

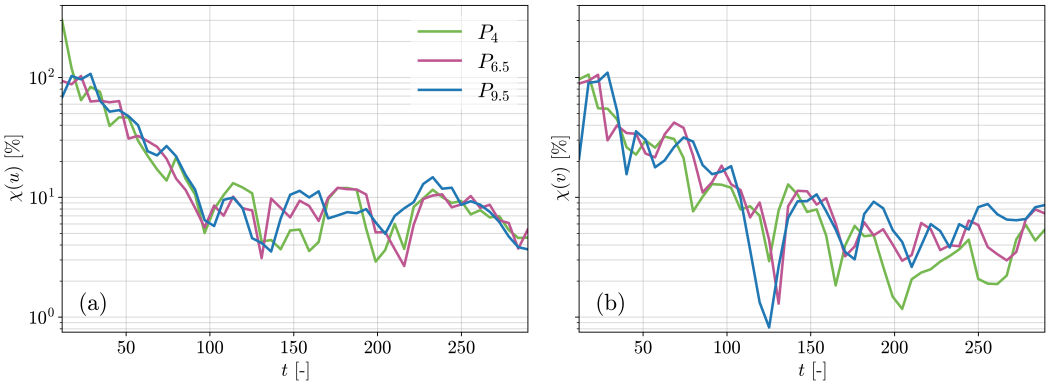


Figure 16: Numerical convergence criterion χ for (a) streamwise and (b) vertical velocity at probes P_4, $P_{6.5}$, and $P_{9.5}$ for the $[Re, \alpha, a, n] = [1796, 11, 0.424, 1]$ case. For v, χ is computed using standard deviation only. Stabilization indicates the evacuation of numerical transients and the onset of statistically steady flow.

H Data Processing

After initialization, the simulation contains an unphysical numerical transient. To ensure the solution trajectories are unaffected by this startup phase, a numerical convergence criterion is applied following Boudet et al. (2015). Measurement probes are placed in regions where the flow exhibits low frequency dynamics. In the present study, they are located along the centreline ($y/D = 0$) at streamwise distances 4, 6.5 and 9.5 behind the object, denoted by P_4, $P_{6.5}$ and $P_{9.5}$, respectively. These positions correspond to the wake region, which provides a consistent reference across all flow cases, whereas other regions of interest, such as the shear layer, change with geometry. Each probe measures a time-dependent signal, ϕ. At time t, the signal is cut into four equal segments. The final three segments are then compared using the following criterion:

$$\chi(\phi) = \max \left[\max_{i,j=2,4} \left(\left| \frac{\overline{\phi}^{(i)} - \overline{\phi}^{(j)}}{\overline{\phi}^{(4)}} \right| \right), \max_{i,j=2,4} \left(\left| \frac{\phi'^{(i)} - \phi'^{(j)}}{\phi'^{(4)}} \right| \right) \right]. \tag{9}$$

The superscripts indicate the temporal segment the data originates from, while $\overline{\phi}$ and ϕ' represent the spanwise-averaged mean and standard deviation of the signal, respectively. Once χ settles around a certain percentage, the flow's statistics have stabilized over the three most recent temporal quarters, implying the numerical transient was evacuated after the first quarter. Nevertheless, data sampling starts only after the fourth quarter to provide an additional margin of confidence.

The convergence criterion method is initially evaluated after four flow-through times (FTTs) to ensure enough samples are collected for a reliable assessment. If the signal variations have not yet reached a steady level, the simulation is extended for one FTT at a time until they have. For illustration, Figure 16 displays the metric for the u and v velocity components for the $[Re, \alpha, a, n] = [1796, 11, 0.424, 1]$ case. Note that for the vertical velocity, only the standard deviation is considered, as its mean approaches zero, which would cause the criterion to diverge.

Once the numerical transient has been evacuated, the solution fields are sampled at a non-dimensional time-step of $\Delta t_s = 0.04732$. While the solver requires much smaller temporal steps to maintain numerical stability and accuracy, saving the full resolution is computationally prohibitive due to storage requirements. Sub-sampling the simulation at a coarser time-step inevitably results in a loss of information. To quantify the extent of this loss, the energy spectrum is examined. Figure 11 and Figure 12 display the energy spectra for the circular cylinder flow case at $Re = 3900$, evaluated at $(x/D, y/D) = [(0.71, 0.66), (3.0, 0.0)]$, computed using the original u-velocity signal and the sub-sampled solution at $\Delta t_s = 0.04732$. The spectra match closely across most of the frequencies. In particular, all major characteristic frequencies are accurately recovered. Beyond the inertial subrange, however, the sub-sampled signal is unable to represent the highest frequencies due to its coarser

temporal resolution. This is deemed acceptable as this portion of the spectrum is primarily influenced by numerical dissipation rather than physical processes and contains relatively low energy.

I Baseline implementations

This section documents the architectures, hyperparameters, and training recipes used for the four neural-operator baselines reported in Section 5. We first describe the components shared across baselines, then provide a per-model architectural summary. A consolidated view of the hyperparameters appears in Table 5.

Shared training and inference setup. All baselines are trained on the same temporal task: predict the residual $\Delta u_{t+1} = u_{t+1} - u_t$ from a single context frame u_t, conditioned on the Reynolds number Re. Training uses a standard mean-squared-error loss on the residual. We sample one window per gradient step from a uniform-random subset of $524{,}288$ mesh nodes per trajectory, repeated for $100{,}000$ gradient steps with an effective batch size of 2 on a single H100 GPU (80 GB). The optimizer is AdamW with learning rate 10^{-4}, minimum learning rate 10^{-7}, weight decay 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.95$, and gradient-norm clipping at 1.0. The schedule is a linear warmup over the first $5{,}000$ steps followed by a cosine decay. PDE-parameter conditioning uses only Re; geometry parameters (α, a, n) are not exposed to the model and are absorbed into the spatial coordinates and field statistics. For all models except GINO we enable bfloat16 mixed precision and `torch.compile`. GINO's FNO backbone relies on complex-valued spectral convolutions that interact poorly with bfloat16 autocast and produce graph-breaking warnings under `torch.compile` via the `neuraloperator/tltorch` factorised-tensor indexing path. We therefore train GINO in fp32 without `torch.compile`, with a per-GPU batch size of one and gradient accumulation of two to keep the effective batch size at two. Activation checkpointing is enabled for all transformer backbones.

At inference time we use a batch size of one and a context window of one frame. The full mesh (3.1–9.2 million nodes per trajectory) is processed either by the model's encoder–decoder split (UPT, GINO, CALM-PDE) or by tile partition (Transolver++) with a chunk size of $524{,}288$ output points. Each test trajectory is unrolled for 399 autoregressive steps starting from the first ground-truth frame.

UPT (Alkin et al., 2024b). UPT is a perceiver-style transformer for unstructured meshes. The mesh is first pooled to $K = 16{,}384$ supernodes via a runtime k-nearest-neighbour gather ($k = 32$) and a small message-passing MLP. Two encoder transformer blocks then operate on the supernode tokens. A perceiver cross-attention compresses the supernodes to $L = 4{,}096$ learned latent tokens, on top of which we run four self-attention blocks. Two more transformer blocks act on the decoder side before a perceiver cross-attention queries the original mesh coordinates with sincos positional embeddings (max angular frequency 100) and produces the per-node residual. The hidden dimension is 384 with 6 attention heads, an MLP expansion ratio of 4, no dropout, and DiT-style adaLN-zero conditioning on Re. The trained model has 61.96 M parameters.

GINO (Li et al., 2023c). GINO consists of a graph-neural-operator (GNO) encoder that maps the irregular mesh onto a regular latent grid, an FNO operating on that grid, and a GNO decoder that queries the predicted residual at the original mesh nodes. The latent grid has resolution 12^3 in $[0, 1]^3$, with input coordinates min–max normalised to that cube. Both GNOs use a linear transform with neighbourhood radius 0.033 (about one cell at this resolution). The FNO has four spectral layers, $6{\times}6{\times}6$ Fourier modes, 64 hidden channels, and lifting/projection MLPs with channel ratios of 2 and 4 respectively. PDE-parameter conditioning is realised through an AdaIN layer (4 conditioning features). We use the official `neuraloperator` implementation with Open3D-accelerated GNO kernels (Kossaifi et al., 2025). The trained model has 3.10 M parameters; we found that the FNO backbone saturates well below the parameter budgets used for the other baselines, so this is the configuration reported in the main text.

Transolver++ (Luo et al., 2025a). Transolver++ is a monolithic transformer that aggregates N spatial tokens into a small set of learned physics-aware slices via Gumbel-softmax routing, applies self-attention on the slice tokens, then scatters back to the mesh. We use 8 transformer blocks with hidden dimension 512, 8 attention heads, MLP ratio 4, 128 slices per head, GELU activation, and no dropout. PDE-parameter conditioning is an additive bias on the encoder output. Because Transolver++ is monolithic (input cardinality equals output cardinality), its inference path uses a

Table 5: Summary of baseline hyperparameters. Counts are per model.

Hyperparameter	UPT	GINO	Transolver++	CALM-PDE
Trainable parameters	61.96 M	3.10 M	21.79 M	45.14 M
Hidden dim	384	64 (FNO)	512	384
Heads	6	–	8	6
MLP ratio	4	–	4	4
Depth	2/4/2 (enc/lat/dec)	4 (FNO)	8	8
Tokens / latent grid	K=16,384, L=4,096	12^3	128 slices	16^3=4,096
k-NN k	32	–	–	32
Conditioning	adaLN-zero (Re)	AdaIN (Re)	additive bias (Re)	adaLN-zero (Re)
Activation checkpointing	yes	–	yes	yes
Mixed precision (bf16)	yes	no	yes	yes
`torch.compile`	yes	no	yes	no
Steps	100,000			
Optimizer	AdamW (β_1=0.9, β_2=0.95, wd=10^{-3})			
Learning rate	$10^{-4} \to 10^{-7}$, cosine, 5,000 linear warmup			
Effective batch size	2 (GINO: bs=1, accum=2)			
Subsampled nodes / step	524,288			
Hardware	1× NVIDIA H100 80 GB, ∼20 h wall time			

tile partition rather than an encoder–decoder split: disjoint chunks of the output mesh are processed independently, losing cross-chunk context but bounding memory. The trained model has 21.79 M parameters.

CALM-PDE (Hagnberger et al., 2025; Koupaï et al., 2025). We adopt the continuous-convolution encoder/decoder of CALM-PDE combined with the scaling improvements introduced in ECHO; the implementation is registered as `CALM-PDE` in our codebase. The encoder uses a continuous convolution with k=32 neighbours and a 64-wide kernel MLP to lift features from the irregular mesh to a fixed $16^3 = 4{,}096$-point regular latent grid in $[0, 1]^3$, with global coordinate bounds computed once over the full dataset (267 trajectories, ∼1.4 B nodes) so that the latent grid is shared across batches. A UPT-style transformer with hidden dimension 384, 6 heads, MLP ratio 4, and 8 blocks acts as the time-stepping operator on the latent tokens, conditioned on Re via DiT adaLN-zero. The continuous convolution decoder maps the post-transformer latent back to the original mesh nodes, with output queries chunked at 32,768 points to bound peak memory. The trained model has 45.14 M parameters.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the three main contributions of TurtLES: (i) a 13 TB dataset of high-fidelity iLES trajectories of 3D turbulent cylinder wakes, (ii) benchmarking of state-of-the-art neural operators on long-horizon autoregressive rollouts, and (iii) identification of key failure modes including temporal instability, loss of small-scale energy, and challenges on large irregular meshes. These claims are directly supported by the dataset description in Section 3 and the experimental results in Section 5.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations are discussed in the conclusion (Section 6), including the restriction to cylinder wake geometries, a bounded Reynolds number range ($Re \in [1000, 3900]$), the use of iLES rather than DNS, and the use of compressible flow at $Ma = 0.2$. The conclusion also identifies open directions such as higher Reynolds numbers, richer boundary conditions, and additional geometries. Additionally, the benchmarking results explicitly identify current model limitations on this dataset (temporal instability, spectral inaccuracy, inference cost).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

   Justification: This paper does not include theoretical results or proofs. It is a dataset and benchmark contribution; all mathematical content consists of metric definitions and governing equations for the fluid simulations.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The benchmarking setup is described in Section 4, including the training objective (residual prediction), optimizer (Adam), learning rate (10^{-4}), weight decay (10^{-3}), batch size (2), number of training steps (100,000), warmup schedule (5,000 steps), cosine decay, and hardware (single H100 GPU, $\sim$20 hours). Evaluation metrics (Rel-L^2, VRMSE, spectral RMSE, latency) are fully defined with equations. The dataset generation pipeline, mesh sizes, physical parameters, and time-sampling procedure are described in Section 3 and the appendices.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The dataset is publicly available on Hugging Face at https://huggingface.co/turtlesprojectml. Code for dataset generation, benchmarking, and evaluation is available in an anonymized repository at https://anonymous.4open.science/r/turtlescode-DC26/README.md

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 4 specifies the optimizer (Adam), learning rate (10^{-4}), weight decay (10^{-3}), batch sizes for training (2) and inference (1), number of steps (100,000), warmup and cosine LR schedule, hardware (H100), and approximate training time ($\sim$20 hours per model). The inference protocol (full autoregressive rollout over 400 steps, iterating over subsets of mesh nodes) is also described.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: The results tables do not include error bars or confidence intervals. Each model is trained once due to the high computational cost ($\sim$20 hours per run on a single H100). The diversity of the test set (multiple trajectories across varying geometry, Reynolds number, and angle of attack) provides some implicit averaging, but formal statistical significance testing is not performed.

Guidelines:

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

8. **Experiments compute resources**

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

Answer: [Yes]

Justification: Section 4 states that all models are trained for approximately 20 hours on a single H100 GPU. Inference latency per frame is reported for each model in Table 3 (UPT: 835s, GINO: 4097s, Transolver++: 1148s, CALM-PDE: 1073s for a full trajectory). The dataset generation uses PyFR with GPU-based solvers; further details on simulation compute are provided in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: This work presents a fluid dynamics simulation dataset and benchmark for scientific machine learning. It involves no human subjects, no data scraping, no personally identifiable information, and poses no foreseeable risks of misuse.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The introduction discusses the positive societal impact of accelerating computational fluid dynamics via surrogate models, with applications in industrial design, forecasting, and control. The dataset targets industrial CFD regimes, with potential to reduce the computational cost of high-fidelity simulations. No significant negative societal impacts are foreseen for this type of scientific infrastructure contribution.

    Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: The dataset consists of numerical fluid dynamics simulations and poses no meaningful risk of misuse. No safeguards beyond standard open-data practices are required.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: The paper cites all third-party software used, including PyFR (Witherden et al., 2014) for fluid simulation and Gmsh (Geuzaine and Remacle, 2009) for mesh generation. The neural operator baselines (UPT, GINO, Transolver++, CALM-PDE) are cited with their original papers. Validation data sources from the literature are also fully cited.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: The TurtLES dataset is described in detail in Section 3 and the appendices, including the parameter space (Section 3 / Appendix A), simulation setup (Section 3 / Appendix C), meshing procedure (Appendix C), validation (Appendix D), and data processing (Appendix E). The data structure (coordinates + 400 solution snapshots per trajectory, fields $\{u, v, w, p, \rho\}$) is illustrated in Figure 3.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This paper does not involve human subjects. No IRB approval is required.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: LLMs are not used as part of the core methodology.

    Guidelines:

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