

Reconstructing unsteady flows from sparse, noisy measurements with a physics-constrained convolutional neural network

Yaxin Mo¹ and Luca Magri^{1,2,*}

¹*Department of Aeronautics, Imperial College London, SW7 2AZ, United Kingdom*

²*The Alan Turing Institute, London NW1 2DB, United Kingdom*



(Received 31 August 2024; accepted 4 February 2025; published 4 March 2025)

Data from fluid flow measurements are typically sparse, noisy, and heterogeneous, often from mixed pressure and velocity measurements, resulting in incomplete datasets. In this paper, we develop a physics-constrained convolutional neural network, which is a deterministic tool, to reconstruct the full flow field from incomplete data. We explore three loss functions, two of which are from machine learning literature and one designed by us: (i) the softly constrained loss, which allows the prediction to take any value; (ii) the snapshot-enforced loss, which constrains the prediction at the sensor locations; and (iii) the mean-enforced loss, which constrains the mean of the prediction at the sensor locations. The proposed methods do not require the full flow field during training, making it suitable for reconstruction from incomplete data. We apply the method to reconstruct a laminar wake of a bluff body and a turbulent Kolmogorov flow. First, we assume that measurements are not noisy and reconstruct both the laminar wake and the Kolmogorov flow from sensors located at fewer than 1% of all grid points. The snapshot-enforced loss reduces the reconstruction error of the Kolmogorov flow by $\sim 25\%$ compared to the softly constrained loss. Second, we assume that measurements are noisy and propose the mean-enforced loss to reconstruct the laminar wake and the Kolmogorov flow at three different signal-to-noise ratios. We find that, across the ratios tested, the loss functions with harder constraints are more robust to both the random initialization of the networks and the noise levels in the measurements. At high noise levels, the mean-enforced loss can recover the instantaneous snapshots accurately, making it the suitable choice when reconstructing flows from data corrupted with an unknown amount of noise. The proposed method opens opportunities for physical flow reconstruction from sparse, noisy data.

DOI: [10.1103/PhysRevFluids.10.034901](https://doi.org/10.1103/PhysRevFluids.10.034901)

I. INTRODUCTION

In many flow applications, particularly in experimental settings, estimating a flow field from incomplete data is a valuable capability. To address this, various data assimilation methods have been developed for flow field reconstruction, employing different strategies. Linear approaches often rely on modal decompositions combined with linear regression [1–3] or nonlinear optimization techniques [4] to estimate modal coefficients. Although computationally efficient, modal-based methods may not fully fulfill the governing equations and often depend on modes derived from

*Contact author: l.magri@imperial.ac.uk

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International](https://creativecommons.org/licenses/by/4.0/) license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

spatially complete data or mean flow data. To address spatial incompleteness, extensions of modal decomposition such as gappy proper orthogonal decomposition (POD) [5] and spectral POD [6] have been utilized. These techniques can reconstruct flows from incomplete data [6–9], although they may struggle when dealing with large spatial or temporal gaps in the data [6–8]. To incorporate nonlinearities and physics-based constraints, adjoint-based constrained optimization techniques, such as 4DVar [10], have been applied. These approaches reconstruct flow fields that fulfill the Navier-Stokes equations [11–16]. However, adjoint-based methods can be computationally expensive and memory-intensive [13], and they may become unstable when the optimization time window exceeds the chaotic timescales of the system [15,17,18]. Sequential filtering methods, such as the ensemble Kalman filter [19], offer another strategy for reconstructing flow fields from sensor measurements [11,20–23]. However, the quality of reconstruction using sequential filtering depends on the robustness and accuracy of the adopted reduced-order model (e.g., [22,24]).

Recently, neural networks, with their ability to handle a large amount of data and approximate nonlinear functions, have become a popular choice for performing flow reconstruction [25], which often involves estimating the velocity and/or pressure from sparse measurements of the flow field. Many network architectures have been developed for flow reconstruction when ground-truth flow fields during training are available. Autoencoders have been used to reconstruct various flows, such as bluff body wakes and isotropic turbulence from sparse measurements [26,27]. Maulik *et al.* [28] reconstructs the ocean temperature, a bluff body wake, and an aerofoil with a mixture density network. Generative adversarial networks have been used to reconstruct rotating turbulence [29,30] and channel flows [31–33]. Other generative networks have also been used to recover turbulence statistics [34]. Convolutions are a common feature among the networks used for reconstruction and have been used to perform tasks such as recovering high-resolution flow fields from lower-resolution flow fields [35–38], reconstructing flows from sparse measurements [39], and obtaining the flow fields from images of geometries of porous media [40].

Fewer works are available when ground-truth data (i.e., the full flow field) is not available during the training of the networks. Güemes *et al.* [38] reconstructed turbulent flows from noisy point measurements of the flow, in which the points are randomly placed and moving using a generative adversarial network. Their network requires the point measurements to change locations snapshot to snapshot, making it suitable for reconstructing flows from results of particle image velocimetry where a large amount of moving point measurements are generated.

When the ground truth is unavailable and measurements are sparse, incorporating prior knowledge into machine learning allows the reconstruction of flow fields from limited measurements. Physics-informed neural networks (PINNs) can infer the unknowns at unseen locations and have been employed to reconstruct a range of different flows [41–45]. Convolutional neural networks (CNNs) allow an image-based approach, which is suitable for reconstructing high-dimensional flows. By incorporating physical knowledge such as the Navier-Stokes equations, CNNs have been employed to reconstruct steady flows [46] and turbulent flows [47,48] from limited measurements.

Depending on the level of understanding that we have of the problem at hand, there are different ways to incorporate physics into training. Hard-coded constraints such as sine activation function or Fourier modes are used to ensure periodicity in the output [49–51]. However, the most common way to enforce physics is to introduce the governing equations as a penalty term in the loss function to be minimized [41,42,46,47,52]. In most physics-constrained neural networks, the loss function to be minimized in training is a combination of data loss and physics loss, taking the form

$$\text{Loss}_{\text{network}} = \lambda_{\text{data}} \text{Loss}_{\text{data}} + \lambda_{\text{equation}} \text{Loss}_{\text{equation}},$$

where λ is the coefficient, the data loss, $\text{Loss}_{\text{data}}$, is a function of only prediction and observations, and physics loss, $\text{Loss}_{\text{equation}}$, is a function of the prediction derived from the governing equation [52]. The coefficients are hyperparameters, which need to be tuned. By introducing the governing equations as a penalty term in the loss function, we place a soft constraint on the physics of the prediction.

Although with their excellent capabilities, soft physics constraints may produce trivial solutions [49]. Since the importance of the physics depends on the coefficients, the physics loss is “competing” with the data loss. Sometimes networks learn a trivial solution, which satisfies the governing equations without satisfying the data loss, e.g., flows where the velocity is constant everywhere. Trivial solutions occur because a physics loss of zero with a larger data loss gives a better overall loss than a small physics and a small data loss.

The snapshot-enforced loss, proposed by Gao *et al.* [46], places a harder constraint on the data loss. The idea of the strictly enforced data loss is to force the prediction to be equal to the observed values at observed locations so that only the physics loss is minimized during training. Gao *et al.* [46] successfully reconstructed steady flows using the snapshot-enforced loss. However, the snapshot-enforced loss is not robust to noisy observations.

When measurements are taken from experiments, they are often noisy. Different network structures have been proposed for reconstructing flows from noisy measurements, such as placing restrictions on the shape of the solution [50] or using probabilistic networks [28]. Some information on the flow fields can also be recovered from noisy partial observations by reconstructing the mean instead of the instantaneous flow fields [53,54]. Shokar *et al.* [55] in particular, used forced noise and an ensemble network to study the long-term statistics of beta-plane turbulence.

Reconstruction of instantaneous turbulent flows from noisy partial observations has been performed with CNN-based networks in [29,31,33] as part of a robustness study. However, these studies require knowledge of the reference data (i.e., the full flow field) during training.

In this paper, we reconstruct the unsteady wake of a triangular cylinder and a turbulent Kolmogorov flow from both clean and noisy sparse measurements to infer the entire flow field. Kolmogorov flow and circular cylinder wakes are common test cases in flow reconstruction literature [26,28,47], with the wake of a triangular cylinder being important in flame holders in combustion chamber design [56,57]. First, we investigate the effect of hard and soft constraints placed on the instantaneous measurements in the loss functions on flow reconstruction with a physics-constrained neural network. Then we propose a new loss function for reconstructing flows from noisy measurements. Finally, we compare the loss functions at different signal-to-noise ratios.

The paper is organized as follows. First, we introduce two datasets—a laminar wake and a turbulent Kolmogorov flow in Sec. II. Second, we present our methodology, including taking measurements for collocation points, our neural network, and different loss functions, in Sec. III. Third, we reconstruct the laminar wake from both noisy and non-noisy measurements in Sec. IV and reconstruct the Kolmogorov flow from both noisy and non-noisy measurements in Sec. V. Finally, we present our conclusion in Sec. VI.

II. DATA AND PREPROCESSING

In this study, we reconstruct an unsteady laminar wake of a triangular body and a turbulent Kolmogorov flow. Details on the data generation and preprocessing are given in Sec. II A for the laminar dataset and Sec. II B for the turbulent dataset.

A. Laminar unsteady wake

The incompressible Navier-Stokes equations are cast in a residual fashion for the purpose of this study,

$$\begin{aligned} \nabla \cdot \mathbf{u} &= \mathcal{R}_d(\mathbf{u}) \\ \frac{\partial \mathbf{u}}{\partial t^*} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p - \frac{1}{\text{Re}} \Delta \mathbf{u} + \mathbf{f} &= \mathcal{R}_m(\mathbf{u}, p), \end{aligned} \quad (1)$$

where $\mathbf{u}(\mathbf{x}^*, t^*) \in \mathbb{R}^{N_u}$ and $p(\mathbf{x}^*, t^*) \in \mathbb{R}$ are the velocity and pressure at location $\mathbf{x}^*$ and time t^* , N_u is the number of velocity components, and $\mathbf{f}$ is a forcing term applied to the flow. The unsteady wake of a 2D triangular body at Reynolds number $\text{Re} = 100$ is generated by direct numerical simulation

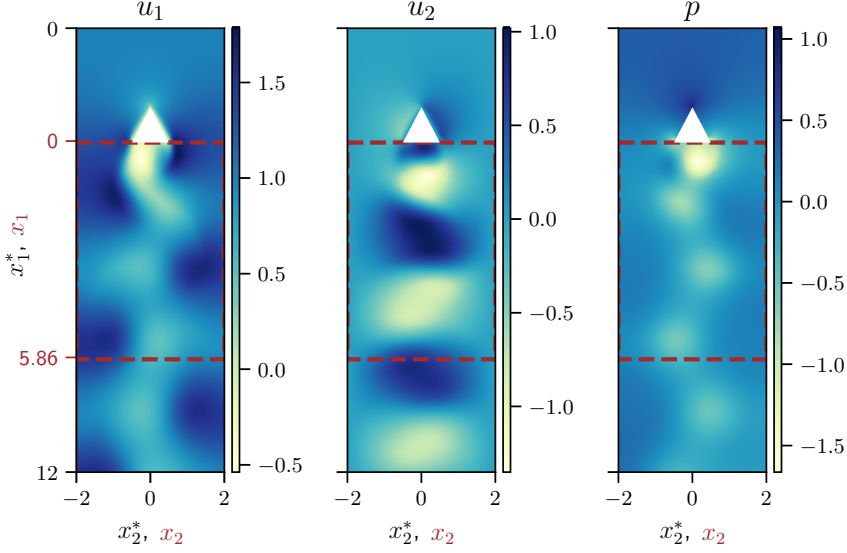


FIG. 1. Snapshots of streamwise (left) and wallnormal (center) velocity and pressure (right) at $t = 2.5$, where $t = 0$ is the end of the transient period discarded when generating the dataset. The area indicated by the brown box is the area considered by the neural networks, with its coordinates $\mathbf{x} = [x_1^* - 3.0, x_2^*]^T$ shown in brown.

using Xcompact3D¹ [58]. The residuals $\mathcal{R}_d(\mathbf{u})$ and $\mathcal{R}_m(\mathbf{u}, p)$ are zero when the equations are exactly solved. For the unsteady wake, $N_u = 2$ and $\mathbf{f} = \mathbf{0}$. The computational domain has size $L_1 = 12$ and $L_2 = 4$ in streamwise and wallnormal directions, with 513 and 129 uniformly spaced grid points in each direction, respectively. The center of the bottom edge of the equilateral triangle is placed at $(3.0, 2.0)$ oriented such that the base is perpendicular to the streamwise velocity. Lengths are nondimensionalized by the length of the side of the equilateral triangle. Zero gradients are applied at $x_1^* = 12$, and slip walls are applied at $x_2^* = -2$ and $x_2^* = 2$; the time step is $\Delta t^* = 0.0002$. Equation (1) are solved using the third-order Adams-Bashforth scheme in time and the sixth-order compact scheme in space [59]. When solved numerically, the simulated flow has near zero residuals, $[\mathcal{R}_d, \mathcal{R}_m]^T \approx \mathbf{0}$. The transient period from $t^* = 0$ to 100 is discarded to ensure that the final dataset consists of only periodic vortex shedding [60] at a statistically stationary regime. From here on, $t = 0$ denotes the time of the first snapshot of the final dataset, i.e., $t = t^* - 100$. A snapshot of the flow is saved every 625 steps, resulting in a time step $\Delta t = 0.125$ for the final dataset, i.e., over 40 snapshots per vortex shedding period. Figure 1 shows the data at $t = 2.5$ on the plane $x_3^* = 0$. Only data in the brown box of Fig. 1 are considered by the neural networks for computational efficiency, capturing the near wake and the parallel vortex shedding. For constructing the final dataset, a snapshot of a flow (area within the brown box) contains the grid points on the plane at $x_3^* = 0$ bounded by $x_1^* = 3.0$, $x_1^* = 8.86$, $x_2^* = -2$, and $x_2^* = 2$. For simplicity, we define the coordinates of the dataset to be $\mathbf{x} = [x_1^* - 3.0, x_2^*]^T$, shown in Fig. 1 in brown.

The snapshots are organized into snapshot matrices $\mathbf{U} \in \mathbb{R}^{N_t \times N_1 \times N_2 \times N_u}$ and $\mathbf{P} \in \mathbb{R}^{N_t \times N_1 \times N_2 \times 1}$ for velocity and pressure, where N_t is the number of snapshots, $N_1 = 250$ is the number of grid points in the x_1 direction, and $N_2 = 129$ is the number of grid points in the wallnormal direction. The dataset $\mathbf{D}^T = [\mathbf{U}^T, \mathbf{P}^T]$ is the snapshot matrix of the full state. Each full state vector is $\mathbf{D}(\mathbf{x}, t)^T = [\mathbf{u}^T, p]$,

¹Because Xcompact3D accepts only 3D domains, periodic boundary conditions are applied to the spanwise direction to simulate an infinitely long cylinder, and the data are a plane taken from the resulted simulations.

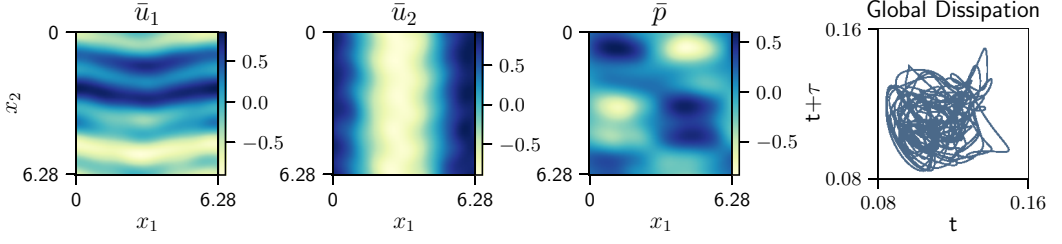


FIG. 2. The time-averaged velocity in x_1 direction (u_1), x_2 direction (u_2), pressure (p), and global dissipation with a time delay of $\tau \approx 5$ (left to right).

where $\mathbf{u}^T = [u_1(\mathbf{x}, t), u_2(\mathbf{x}, t)]$ is the 2D velocity vector and $p = p(\mathbf{x}, t)$ is the pressure. The laminar dataset contains 600 snapshots, $N_t = 600$, which cover approximately 15 vortex shedding periods.

B. Turbulent Kolmogorov flow

KolSol [61], a pseudospectral solver, is used to generate Kolmogorov flows by solving (1) with the forcing term $\mathbf{f} = \mathbf{e}_1 \sin k\mathbf{x}$, where $\mathbf{e}_1$ is the unit vector $[1, 0]^T$, in a 2D periodic box with length 2π on both sides. The Kolmogorov flows are generated at $\text{Re} = 34$, with 32 wave numbers, the forcing frequency $k = 4$ and the time step $\Delta t^* = 0.01$, resulting in weakly turbulent flows [62]. The flow in the frequency domain is then converted into the physical domain on a 128×128 grid. A snapshot is saved every $10 \Delta t^*$, resulting in $\Delta t = 0.1$ for the datasets. The end of the transient period is $t = 0$. Two datasets are generated by using different initial random seeds—one for the grid sensitivity study in Sec. A 1 and one for the results in Secs. V A and V B. Each Kolmogorov dataset, $\mathbf{D} = [\mathbf{U}^T, \mathbf{P}^T]^T \in \mathbb{R}^{N_t \times N \times N \times (N_u + 1)}$, is the snapshot matrix of the full state with velocity and pressure, where $N_t = 6000$, $N = 128$, and $N_u = 2$. Figure 2 shows an overview of a Kolmogorov flow dataset. The global dissipation is the local dissipation $\|\nabla \mathbf{u}\|^2 / \text{Re}$ averaged over the domain.

III. METHODOLOGY

In this section, we first present how the measurements used in the reconstruction are selected from our simulated datasets (Sec. III A). We then present the physics-constrained dual-branch convolutional neural network (Sec. III B). Finally, we present the different formulations of loss functions we will investigate (Sec. III C).

A. Taking measurements

From each dataset, we select a set of sparse measurements from which we will reconstruct the flows. This set of sparse measurements is the only data-related information available for the network in training, i.e., no high-resolution data are needed in the methods that we propose. Measurements of all velocity components and pressure are taken at sensor locations $\mathbf{x}_s$ (general sensors). Additional pressure measurements are taken at input locations $\mathbf{x}_{in}$. The separate input pressure measurements simulate a common situation in bluff body wake experiments, e.g., [63]. For the bluff body wake, $\mathbf{x}_{in}$ is selected to be immediately behind the triangular body; for the Kolmogorov flow, $\mathbf{x}_{in}$ is randomly selected. The set of measurements $\xi(\mathbf{D})$ is a collection of the input pressure and the velocities and pressure measured at the sensor locations, $\xi(\mathbf{D}) = \{\mathbf{U}(\mathbf{x}_s), \mathbf{P}(\mathbf{x}_s), \mathbf{P}(\mathbf{x}_{in})\}$. The measurements are taken at different locations for different tests. The exact locations will be shown together with the test results in Secs. IV and V.

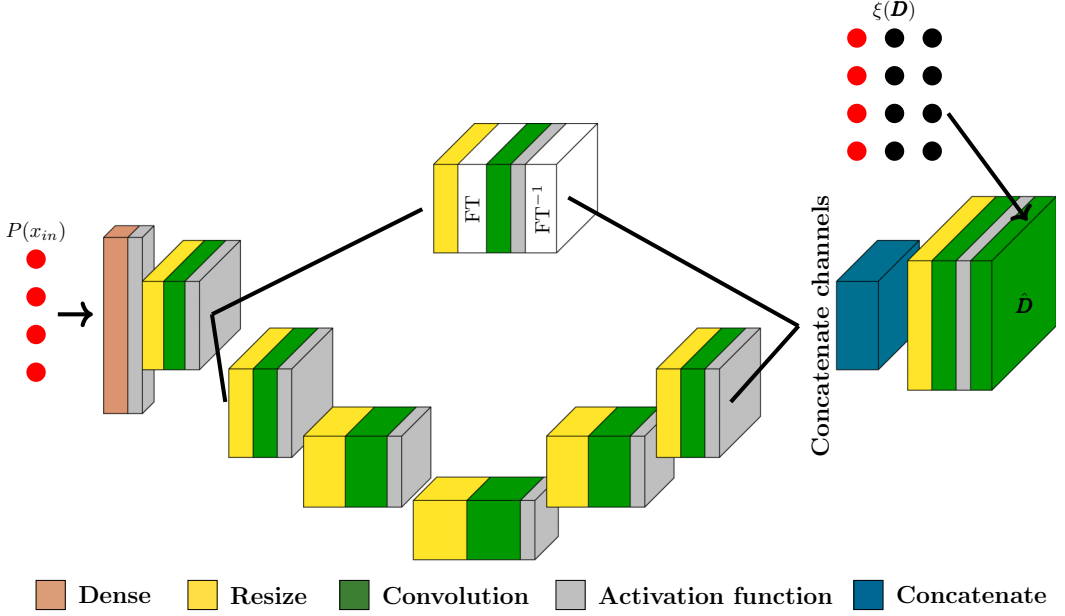


FIG. 3. Schematic of the two-branch Fourier convolutional neural network. The number and size of layers are for illustration. The Fourier transform layer (FT) and the inverse Fourier transform layer (FT⁻¹) are optional and are activated (or deactivated) for different loss functions.

B. Physics-constrained dual-branch convolutional neural network

We develop a physics-constrained dual-branch convolutional neural network (PC-DualConvNet) for the reconstructions, inspired by [47,55,64,65]. The advantage of the PC-DualConvNet is that it does not require the full state to be known at all grid points during training, and the measurements do not need to be in a regular grid. The goal of the network is to infer unseen spatial information for a given set of measurements.

PC-DualConvNet aims to produce the flow field $\hat{\mathbf{D}}$ to approximate the data $\mathbf{D}$ from pressure input $\mathbf{P}_{in}$, using sparse measurements $\xi(\mathbf{D})$ as collocation points. PC-DualConvNet consists of a U-Net style [64] convolutional branch, which is common in image processing (lower branch in Fig. 3), and an optional Fourier branch for reducing the spatial aliasing of the output (top branch in Fig. 3). The Fourier transform is performed over both temporal and spatial dimensions of the dataset. During training, we find that the Fourier transform in the upper branch is particularly helpful in improving the robustness of the snapshot-enforced loss $\mathcal{L}^s$ (Sec. III C). The benefit of activating the Fourier layer varies with different datasets and is considered as a hyperparameter. Depending on the test case, as explained in the later sections, the top branch may perform Fourier transform and inverse Fourier transform immediately before and after the convolution layers. The output of the top and lower branches are stacked channel-wise before the final convolution layers, which produce the output $\hat{\mathbf{D}}$.

The PC-DualConvNet's parameters $\mathbf{w}$ are the (local) minimizers of a loss function $\mathcal{L}$, which measures the error between $\hat{\mathbf{D}}$ and $\mathbf{D}$. The optimization process is

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \mathcal{L}(\mathbf{P}(\mathbf{x}_{in}); \mathbf{w}), \quad (2)$$

in which the design of the loss formulation $\mathcal{L}$ is a goal of this paper and will be discussed in detail in Sec. III C. The reconstruction error is the relative ℓ_2 error

$$\epsilon(\%) = \sqrt{\frac{\|\hat{\mathbf{D}} - \mathbf{D}\|_2^2}{\|\mathbf{D}\|_2^2}}. \quad (3)$$

C. Loss functions

The loss $\mathcal{L}$ is the quantity to be minimized when training a neural network (Eq. (2)). The governing equations are incorporated into the loss formulations for the physical reconstruction. We define the momentum and continuity loss terms of the dataset $\mathbf{D}$ to be the ℓ_2 norm of the residuals of the continuity and momentum equations, as defined in Eq. (1) at all grid points, as

$$\mathcal{L}_{\text{div}}(\mathbf{D}) = \|\mathcal{R}_d(\mathbf{U})\|_2^2, \quad (4)$$

$$\mathcal{L}_{\text{mom}}(\mathbf{D}) = \|\mathcal{R}_m(\mathbf{U}, \mathbf{P})\|_2^2. \quad (5)$$

The derivatives are computed with second-order finite difference schemes on the boundaries and fourth-order schemes at the interior points. A data-related loss is also needed for the output to minimize the error with the measurements selected from the reference datasets. We define the sensor loss to be the ℓ_2 norm of the difference of the output $\hat{\mathbf{D}}$ and the reference data $\mathbf{D}$ at observed locations as

$$\mathcal{L}_o(\hat{\mathbf{D}}, \mathbf{D}) = \|\hat{\xi}(\hat{\mathbf{D}}) - \xi(\mathbf{D})\|_2^2. \quad (6)$$

1. Softly constrained loss

A widely applied method to include physics into the network is to define the loss function to be the weighted sum of a sensor loss and a physics loss, applied in various superresolution convolutional neural networks and physics-informed neural networks [41,42,47,52]. We define the softly constrained loss $\mathcal{L}^c$ to be

$$\mathcal{L}^c(\mathbf{P}_{in}; \mathbf{w}) = \lambda_o \mathcal{L}_o(\hat{\mathbf{D}}, \mathbf{D}) + \lambda_{\text{div}} \mathcal{L}_{\text{div}}(\hat{\mathbf{D}}) + \lambda_{\text{mom}} \mathcal{L}_{\text{mom}}(\hat{\mathbf{D}}), \quad (7)$$

where λ denotes the non-negative regularization factor. Both the physics of the output and the values of the output at the sensor locations are included in the loss function as regularization terms, which means $\mathcal{L}_o$ and $\mathcal{L}_p$ will be minimized, but not guaranteed to be a specific value, hence its name “softly constrained.”

The components of the softly constrained loss can be grouped into the sensor loss $\mathcal{L}_o$ and the physics loss $\mathcal{L}_p$, which is the combined continuity and momentum loss $\mathcal{L}_{\text{div}} + \mathcal{L}_{\text{mom}}$. A trivial solution is a solution which has $\mathcal{L}_p = 0$ but a large $\mathcal{L}_o$, meaning that the network finds another solution to the governing equation that is different from the reference datasets. An example of a trivial solution is a flow field that is constant everywhere. Trivial solutions may develop when both $\mathcal{L}_p$ and $\mathcal{L}_o$ cannot, theoretically, be 0 at the same time, which can be caused by noise, either experimental or numerical. When reconstructing a synthetic dataset, the source of the noise could come from the use of different numerical schemes when estimating the derivatives. For example, the residuals in $\mathcal{L}_p$ for the reconstructed laminar dataset, defined in (5), are computed with a mix of second- and fourth-order finite differences. However, the laminar dataset is generated with a sixth-order scheme (see Sec. II A). Therefore, if the measurements at sensor locations from the reference and the reconstructed datasets are the same, meaning $\mathcal{L}_o(\hat{\mathbf{D}}, \mathbf{D}) = 0$, and the reference dataset has $\mathcal{L}_p(\hat{\mathbf{D}}) = 0$ when using the sixth-order scheme, then $\mathcal{L}_p(\hat{\mathbf{D}})$ cannot be zero when using the mix of second- and fourth-order scheme. The possible scenarios are summarized in Table I. Scenario 1 is not possible when different numerical schemes are used in data generation and training, or when the reference data are collected from experiments. For scenarios 2 to 4, the sensor and physics losses are in conflict with each other, meaning that they both have a theoretical minimum of 0 if considered independently, but their sum (the total loss value) cannot reach 0. Depending on the sensor placements, scenario 2 may have a theoretical sensor loss close to 0 (when using a few sensors) or much larger than 0 (large numbers of sensors or good sensor locations). Scenarios 3 and 4 are acceptable reconstructions, but their theoretical minimum total losses are also not 0. Therefore, the network may produce trivial solutions instead of the correct reconstructed flows. In other words, given that the ideal scenario 1 is not possible, we want to be in scenario 3 or 4, but they have the same nonzero theoretical minimum total loss as the unwanted scenario 2.

TABLE I. Possible solutions and their corresponding theoretical loss values when training with the softly constrained loss. The network needs to minimize both physics loss and sensor loss. Scenario 1 has a theoretical minimum loss of 0 and is ideal, but is also unlikely when different numerical schemes are used in data generation and loss computation, or if data are experimental. Scenario 2 is the trivial solution, where the outputs satisfy the governing equations but not the measurements. Depending on the sensor placements, the minimum $\mathcal{L}_o$ could be either approximately 0 or much larger than 0. Scenarios 3 and 4 are desired, but they also have a theoretical minimum total loss larger than 0.

	Total loss value	Sensor loss	Physics loss	Numerical scheme	Reconstructed flow
1	0	0	0	Same	$\hat{\mathbf{D}} = \mathbf{D}$
2	>0 or ≈ 0	>0 or ≈ 0	0	Different	Trivial solution
3	≈ 0	0	≈ 0	Different	$\hat{\mathbf{D}} = \mathbf{D}$
4	≈ 0	≈ 0	≈ 0	Different	$\hat{\mathbf{D}} \approx \mathbf{D}$

In principle, by carefully tuning the ratio between λ_o , λ_{div} and λ_{mom} in Eq. (7), we can penalize the occurrence of trivial solutions. In practice, good sets of hyperparameters become difficult to find as the measurements become noisier or more sparse, and trivial solutions are not effectively prevented, as shown in Sec. V C.

2. Snapshot-enforced loss

To eliminate trivial solutions and improve the training, we strictly enforced the measurements and minimize physics loss. The snapshot-enforced loss for steady flows was proposed by [46]. The snapshot-enforced loss minimizes only the physics loss while the sensor loss is enforced to be 0, providing a harder constraint on the predicted flow at sensor locations compared to the softly constrained. By enforcing a harder constraint on the measurements and minimizing only the physics loss, we eliminate trivial solutions. In this paper, we define the snapshot-enforced loss $\mathcal{L}^s$ as

$$\mathcal{L}^s = \lambda_{\text{div}} \mathcal{L}_{\text{div}}(\Phi) + \lambda_{\text{mom}} \mathcal{L}_{\text{mom}}(\Phi), \quad (8)$$

where $\Phi^T = [\Phi_u^T, \Phi_p^T]$ is defined as

$$\Phi_u(\mathbf{x}) = \begin{cases} \mathbf{U}(\mathbf{x}) & \text{where } \mathbf{x} \in \mathbf{x}_s, \\ \hat{\mathbf{U}}(\mathbf{x}) & \text{otherwise.} \end{cases} \quad \Phi_p(\mathbf{x}) = \begin{cases} \mathbf{P}(\mathbf{x}) & \text{where } \mathbf{x} \in \{\mathbf{x}_s, \mathbf{x}_{in}\}, \\ \hat{\mathbf{P}}(\mathbf{x}) & \text{otherwise.} \end{cases} \quad (9)$$

Practically, we start with the reconstructed flow field, $[\hat{\mathbf{U}}, \hat{\mathbf{P}}]$, then replace the reconstructed velocities and pressure at the sensor locations with sensor measurements to obtain the processed reconstructed flow field, Φ . In regions without sensors, we keep the reconstructed values. Finally, we compute the physics loss of Φ , which is minimized during training. In short, the network is not allowed to modify the output values at the sensor locations. Figure 4 is a graphical representation of training a PC-DualConvNet with the snapshot-enforced loss $\mathcal{L}^s$. A pseudoalgorithm outlining the practical steps to compute the strictly enforced loss for each batch of training data is shown in Algorithm 1.

The snapshot-enforced loss may struggle with noisy measurements. Because the noisy measurements are enforced at the sensors' locations, the network's outputs and gradients at the sensors' locations are not updated by training. As a result, regularizing the loss function is not effective in mitigating the impact of noise at the sensors' locations.

3. Mean-enforced loss

The snapshot-enforced loss $\mathcal{L}^s$ ensures the reconstructed flow fits the measurements, but it also enforces the noise on the measurements if the measurements are noisy. Gao *et al.* [46] suggests using

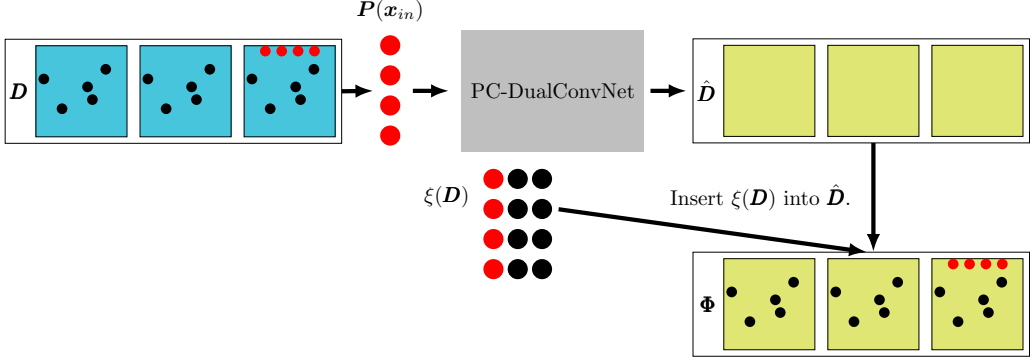


FIG. 4. Computing the snapshot-enforced loss, which involves enforcing measurements $\xi(D)$ into $\hat{D}$ to obtain Φ . The snapshot-enforced loss is the physics loss of Φ .

the softly constrained loss for noisy measurements. In this paper, to deal with noisy measurements, we propose the mean-enforced loss. The mean-enforced loss $\mathcal{L}^m$ combines the advantages of the softly constrained and the snapshot-enforced loss, defined as

$$\mathcal{L}^m = \lambda_o \mathcal{L}_o(\hat{D}, D) + \lambda_{\text{div}} \mathcal{L}_{\text{div}}(\Phi) + \lambda_{\text{mom}} \mathcal{L}_{\text{mom}}(\Phi), \quad (10)$$

where $\Phi^T = [\Phi_u^T, \Phi_p^T]$ is

$$\Phi_u(x) = \begin{cases} \bar{U}(x) + \hat{U}'(x) & \text{where } x \in x_s, \\ \hat{U} & \text{otherwise.} \end{cases} \quad \Phi_p(x) = \begin{cases} \bar{P}(x) + \hat{P}'(x) & \text{where } x \in \{x_s, x_{in}\}, \\ \hat{P} & \text{otherwise.} \end{cases} \quad (11)$$

The symbols $\bar{*}$ and $*'$ denote the time-averaged and fluctuating components of any variable $*$, respectively. In the mean-enforced loss, we split the reconstructed flow into mean and fluctuating components at the sensors' locations to obtain Φ . In regions without sensors, we keep the reconstructed values, whereas at the sensors' locations, we keep the fluctuating components of the reconstructed values while enforcing the mean of the sensors' measurements. Hence, we compute the physics loss of Φ and the sensor loss $\mathcal{L}_o(\hat{D}, D)$, which are both minimized during training. In short, the network can modify the fluctuating components of the output values at the sensor locations, but not the mean. A pseudoalgorithm is shown in Algorithm 2. Figure 5 is a graphical representation of $\mathcal{L}^m$. By changing the ratios of the coefficients λ , we can adjust how much smoothing to apply when reconstructing from noisy observation. The smaller the λ_o compared to λ_{div} and λ_{mom} , the stronger the smoothing.

ALGORITHM 1. Computing the snapshot-enforced loss.

Input: $\hat{D}$: the output of the PC-DualConvNet

x_s, x_{in} : the set of grid indexes where the sensors are located

$\xi(D) = \{D(x_s), P(x_{in})\}$: the measurements

$\lambda_{\text{div}}, \lambda_{\text{mom}}$: regularization factors

Output: $\mathcal{L}^s$: the strictly enforced loss

$\Phi \leftarrow \hat{D}$ // start with the output of the network

$\Phi(x_s) \leftarrow D(x_s)$ // insert the velocities and pressure at general sensor locations x_s with the measured

$\Phi_p(x_{in}) \leftarrow P(x_{in})$ // insert the pressure at input sensor locations x_{in} with the measured pressure inputs

return $\lambda_{\text{div}} \mathcal{L}_{\text{div}}(\Phi) + \lambda_{\text{mom}} \mathcal{L}_{\text{mom}}(\Phi)$

ALGORITHM 2. Computing the mean-enforced loss.

Input: $\hat{D}$: the output of the PC-DualConvNet
 $\mathbf{x}_s, \mathbf{x}_{in}$: the set of grid indexes where the sensors are located
 $\xi(\mathbf{D}) = \{\mathbf{D}(\mathbf{x}_s), \mathbf{P}(\mathbf{x}_{in})\}$: the measurements
 $\lambda_o, \lambda_{div}, \lambda_{mom}$: regularization factors
Output: $\mathcal{L}^m$: the mean-enforced loss
 $\Phi \leftarrow \hat{D}$ // start with the output of the network
 $*$ are the time-averaged components over a training batch, $*$ ' are the fluctuating components
 $\Phi(\mathbf{x}_s) \leftarrow \bar{\mathbf{D}}(\mathbf{x}_s) + \hat{\mathbf{D}}'(\mathbf{x}_s)$ // Insert the time-averaged components of the measurements
 $\Phi_p(\mathbf{x}_{in}) \leftarrow \bar{\mathbf{P}}(\mathbf{x}_{in}) + \hat{\mathbf{P}}'(\mathbf{x}_{in})$
return $\lambda_o \mathcal{L}_o(\xi(\hat{D}), \xi(\mathbf{D})) + \lambda_{div} \mathcal{L}_{div}(\Phi) + \lambda_{mom} \mathcal{L}_{mom}(\Phi)$

IV. RECONSTRUCTING THE LAMINAR WAKE

In this section, we reconstruct the laminar wake from both non-noisy measurements and noisy measurements. We first reconstruct the laminar wake from 18 sparse sensors placed in the domain in Sec. IV A. We then compare the three loss functions introduced in Sec. III C in reconstructing the wake from sparse measurements contaminated with white noise.

A. Reconstructing the laminar wake from non-noisy measurements

For all reconstructions of the laminar wake, pressure sensors are placed on all grid points immediately behind the bluff body (31 in total) at $\mathbf{x}_{in}$. These pressure measurements $\mathbf{P}(\mathbf{x}_{in})$ are used as the input to the network (red dots in Fig. 6). When reconstructing from non-noisy measurements, 12 general sensors (black dots in Fig. 6) are placed at the maximum and minimum of the first two leading POD modes of the pressure and velocity components [66]. A further six general sensors are

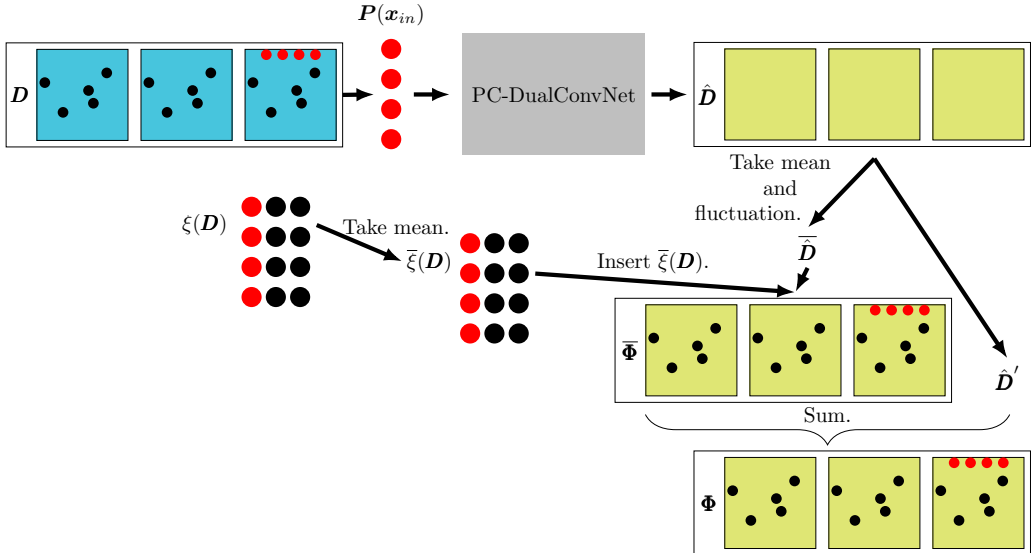


FIG. 5. Computing the mean-enforced loss. The mean is taken over a batch of inputs during training, which involves enforcing the measurements' mean $[\bar{\xi}(\mathbf{D})]$ into $\hat{D}$, and adding the fluctuating field $\hat{D}'$ to obtain Φ . The snapshot-enforced loss is the weighted sum of the physics loss of Φ and the data loss $\mathcal{L}_o(\hat{D}, D)$.

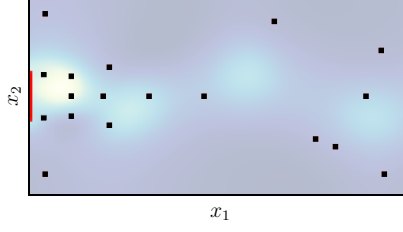


FIG. 6. Sensor placement for laminar wake, non-noisy measurements. The measurements consists of the pressure inlet measurement at $\mathbf{x}_{in}$ (red) and measurements of all components at $\mathbf{x}_s$ (black).

placed randomly near the edge of the domain. The sensor locations $\mathbf{x}_s$ account for only $\sim 0.06\%$ of all grid points.

The softly constrained loss $\mathcal{L}^c$ and the snapshot-enforced loss $\mathcal{L}^s$ are tested for their ability to reconstruct the wake. For each loss function, a hyperparameter optimization is carried out² and the hyperparameters that minimize total loss $\mathcal{L}_p + \mathcal{L}_o$ are selected. (There is a significant difference in the reconstruction results between $\mathcal{L}^c$ and $\mathcal{L}^s$ when applied to reconstruct the laminar wake from non-noisy measurements.)

The model with the lowest total loss in this case uses $\mathcal{L}^c$ and is used to generate the results in this section (see Table IV for the network structures and hyperparameters). A snapshot of the reconstructed flow is compared with the reference and Interpolated snapshot in Fig. 7(a), which shows that the reconstructed flow resembles closely to the reference data and has a smaller error. Vorticity $\mathbf{v}$ is the cross product of the velocity $\nabla \times \mathbf{u}$. The probability density of the fluctuating components of the reconstructed flow also matches more closely the reference data than the interpolated (Fig. 7(b)).

Table II compares the relative error and the physics loss of the reconstructed flow with the reference data and the interpolated flow. Interpolation of the laminar wake is performed per snapshot using a thin-plate spline to handle the very sparse and irregular data points. The physics loss of the reference data is nonzero because the physics loss in training is computed with a mix of second- and fourth-order numerical schemes when a different numerical scheme is used in data generation. Detailed explanations on the effect of using different numerical schemes are given in Secs. II A and III C. The small errors numerically show that the network has successfully reconstructed the laminar wake from sparse measurements.

B. Reconstructing the laminar wake from noisy observations

For reconstructing the laminar wake from noisy measurements, 250 general sensors are randomly placed in the domain (Fig. 8). The sensor locations $\mathbf{x}_s$ account for only $\sim 0.8\%$ of all grid points. The noise $\mathbf{e} \in \mathbb{R}^{N_u+1}$ for the full state $\mathbf{D}(\mathbf{x}, t)$ at each discrete grid point is generated from a Gaussian distribution $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma_e)$, where the components of σ_e correspond to the components of the flows $[\mathbf{u}, p]$. The level of noise is measured by the signal-to-noise ratio (SNR), defined as $\text{SNR} = 10 \log(\sigma_i^2 / \sigma_{e_i}^2)$, where σ_i and σ_{e_i} denote the standard deviation of the i th component of the flows. The noisy data are $\mathbf{D}_n(\mathbf{x}, t) = \mathbf{D}_n(\mathbf{x}, t) + \mathbf{e}$ ($\mathbf{e}$ is different at each discrete grid point); the noisy measurements are $\xi(\mathbf{D}_n)$; the inputs to the network are $\mathbf{P}_n(\mathbf{x}_{in})$. Three different SNRs are considered in this paper: SNR = 20, 10, and 5.

For consistency of the results, the same random seed is used to generate the noise for all test cases shown in this section. The hyperparameters for the networks are shown in Table V–VII. Figure 9 summarizes the effect of different loss functions on reconstructing the laminar wake from noisy,

²Using Bayesian hyperparameter optimization provided by Weights and Biases [67].

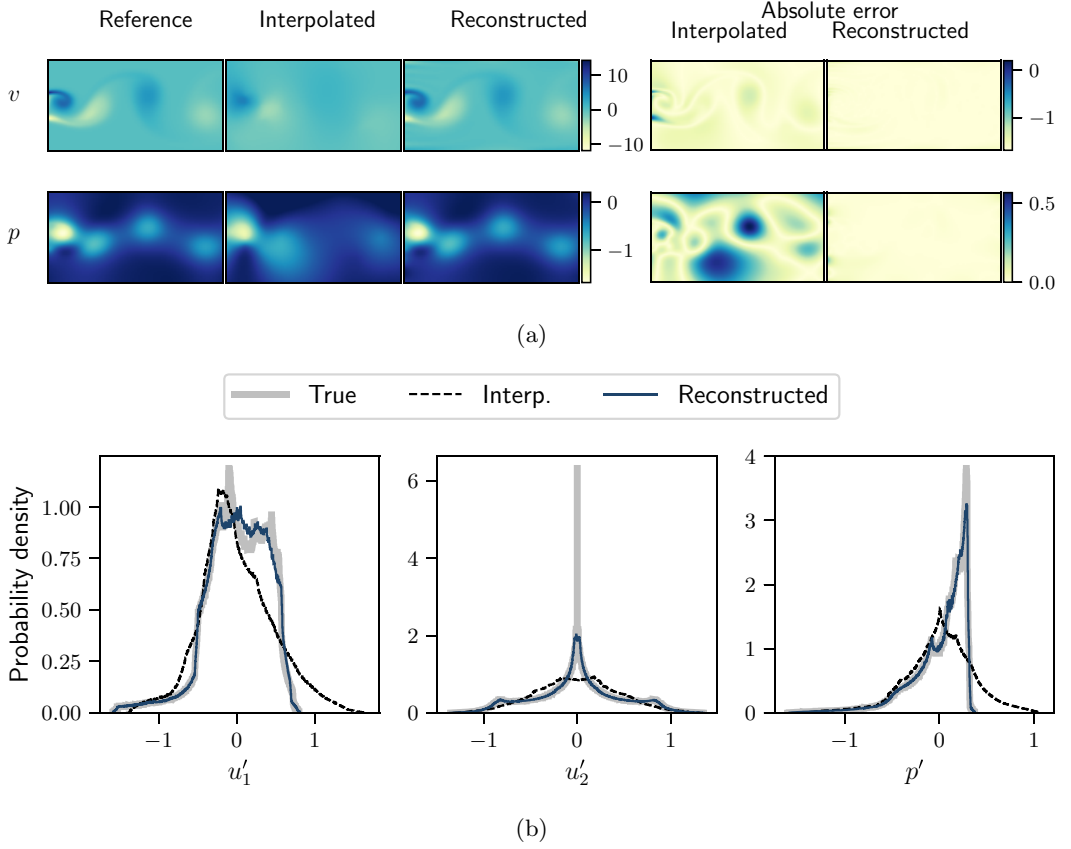


FIG. 7. Reconstructed laminar wake from non-noisy measurements. (a) Snapshots of the reconstructed laminar wake compared with the reference and interpolated snapshot. The last two columns show the instantaneous absolute error of the interpolated snapshot and the reconstructed snapshot, respectively. (b) The probability density of the fluctuating components of the entire flow. Both (a) and (b) show that the reconstructed flow is a closer match with the reference data than mere interpolation.

sparse measurements. At $\text{SNR} = 20$, all loss functions result in reconstruction errors under 10% (Fig. 9(a)), showing that all loss functions can lead to good reconstruction with small levels of noise. The reconstruction error of the softly constrained loss reaches above 10% by $\text{SNR} = 10$, while the errors of the strictly enforced and the mean-enforced loss reach only above 10% at $\text{SNR} = 5$. The slower rate of increase of error as the measurements become noisier shows that losses with harder constraints are less sensitive to noise.

TABLE II. Summary of the losses for the reconstructed laminar wake from non-noisy measurements. Values are shown in the format *mean* $\pm$ *standard deviation* calculated over five runs initiated with different random weights.

	ϵ (%)	$\mathcal{L}_p$
Reference data	n/q	0.04338 ± 0.00000
Interpolated	43.26 ± 0.00	1.06721 ± 0.00000
Reconstructed	3.34 ± 0.18	0.00028 ± 0.00002

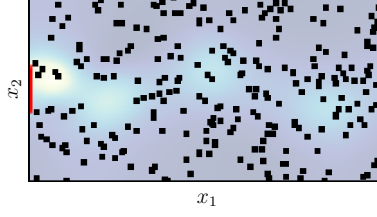


FIG. 8. Sensor placement for laminar wake, noisy measurements. The measurements consist of the pressure inlet measurement at x_{in} (red) and measurements of all components at x_s (black).

Figure 9(b) shows a comparison of the physics loss $\mathcal{L}_p$ for the networks using the different loss functions. At any level of noise, $\mathcal{L}^s$ leads to the highest $\mathcal{L}_p$ among the three loss functions tested. $\mathcal{L}^m$ also has a higher physics loss than $\mathcal{L}^c$, but close to the reference physics loss. The harder constraint on the instantaneous measurements in $\mathcal{L}^s$ does not allow for denoising, thus leading to a higher $\mathcal{L}_p$. Since $\mathcal{L}^m$ is specifically designed to avoid placing hard constraints on the instantaneous measurements, the impact of noise on the reconstruction error of $\mathcal{L}^m$ is smaller compared to $\mathcal{L}^s$. The effect of noise can be appreciated in Fig. 10, in which snapshots of the instantaneous and the mean reconstructed wake are compared with the reference and the interpolated. At SNR = 10 (Fig. 10(b)), the wake reconstructed with $\mathcal{L}^c$ shows the same flow features as the reference, but with a reduced variation of the flow field (the difference between the maximum and minimum value is smaller than the reference). A reduced variation of the predicted flow field is the first sign of a network breaking down. In comparison, the wake reconstructed with $\mathcal{L}^m$ and $\mathcal{L}^s$ at SNR = 10 and at SNR = 20 shows no visible difference. At both SNRs, the wake reconstructed with $\mathcal{L}^s$ shows visible noise in the snapshots, which is not present in the snapshots reconstructed with $\mathcal{L}^m$. $\mathcal{L}^m$ and $\mathcal{L}^s$ also start to show a reduced variation of the flow field at SNR = 5, even though the reduction is stronger for $\mathcal{L}^m$. At SNR = 5, the mean of the interpolated data is more similar to the reference than the reconstructed, particularly in the near wake. However, at all SNRs, the instantaneous snapshots reconstructed wake resemble the reference more closely than the interpolated, showing that regardless of noise, the PC-DualConvNet can reconstruct the flow with a lower relative error than interpolation. Both the quantitative (Fig. 9) and qualitative comparison of the loss functions (10) show that applying a harder constraint on the measurements delays the breakdown of the networks, so the networks are more robust to noise. At SNR = 20 and 10, $\mathcal{L}^m$ is able to achieve the same level of denoising as $\mathcal{L}^c$ while being more robust to noise, making it more suitable for medium levels of noise in the measurements.

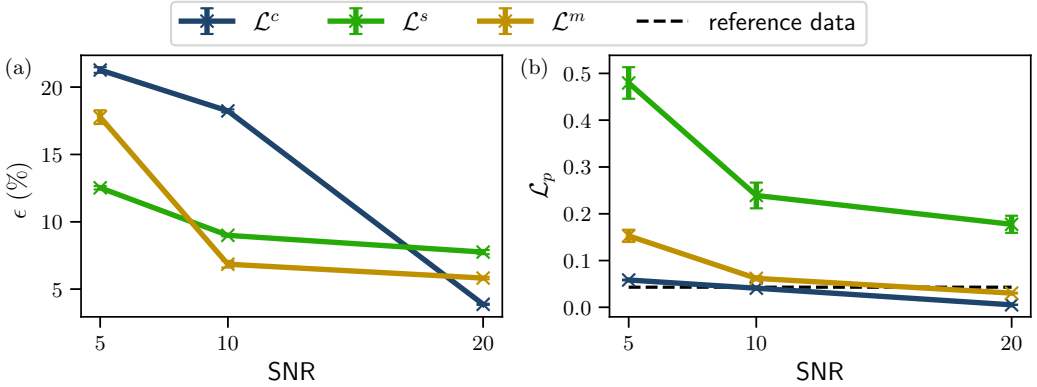


FIG. 9. Comparison of loss functions for different SNR. The reference physics loss is the residual of the Navier-Stokes equations solved numerically. The mean (markers) and standard deviation (error bars) are calculated over five repeats with randomly initialized weights.

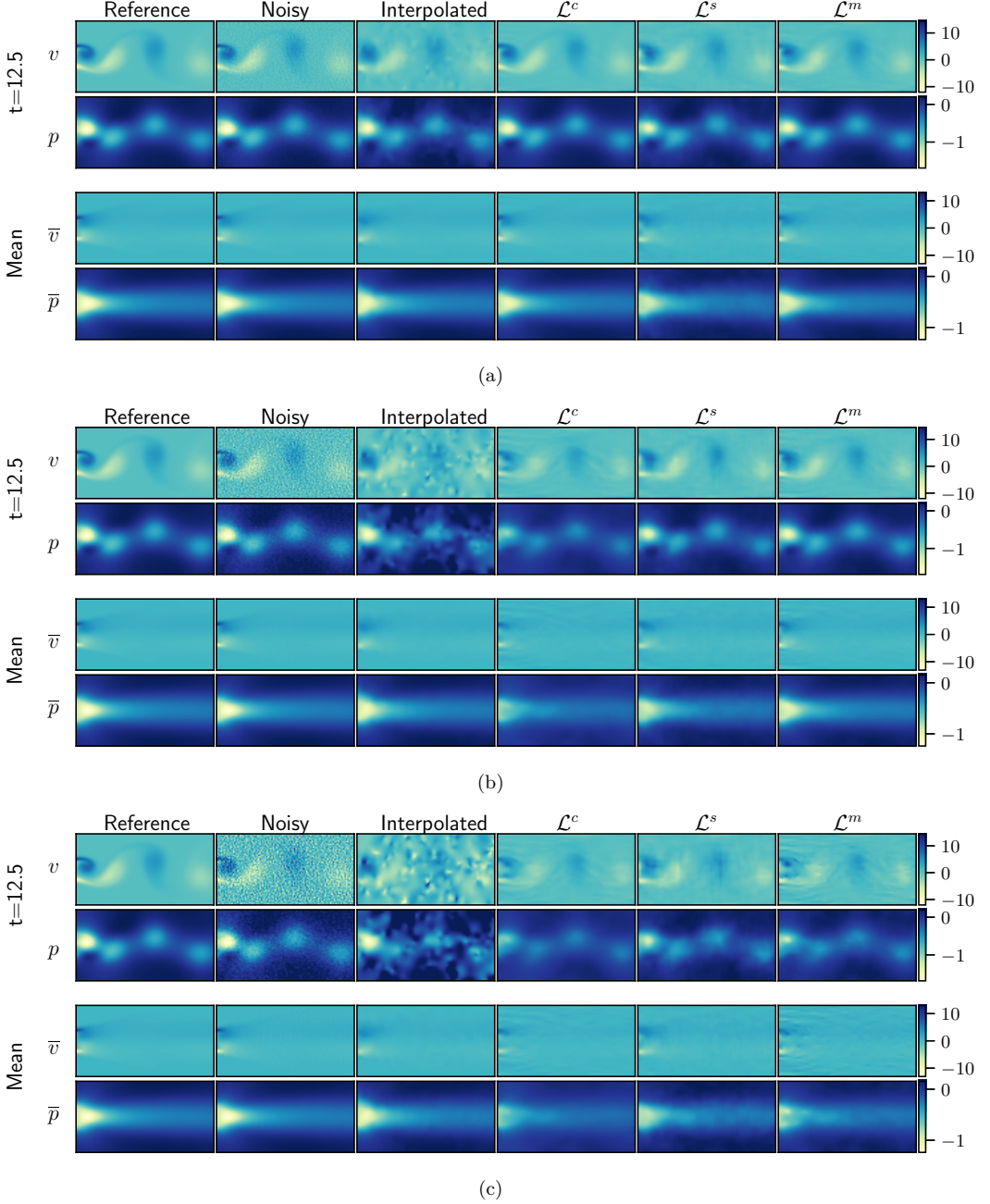


FIG. 10. Reconstructed laminar wake from noisy measurements at SNR = 20 (a), 10 (b), and 5 (c).

V. RECONSTRUCTING THE TURBULENT KOLMOGOROV FLOW

In this section, we apply and compare the three loss functions described in Sec. III C to reconstruct 2D turbulent Kolmogorov flows from sparse measurement. We also apply the three loss functions to reconstruct the Kolmogorov flow in a realistic scenario in Sec. V C with 24 sensors with noise.

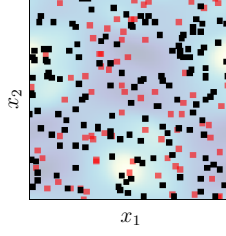


FIG. 11. Random sensors used to reconstruct turbulent Kolmogorov flow from non-noisy measurements, showing the input sensors (red) and general sensors (black).

A. Reconstructing from non-noisy measurements

In this section, we reconstruct the Kolmogorov flow from 80 input sensors and 150 general sensors placed randomly in the domain. The general sensors account for approximately 0.9% of all grid points. The sensor placement is shown in Fig. 11. The number of inputs and general sensors are determined through a grid sensitivity study, which has been included in Appendix A 1. The network parameters for both the current section (Sec. V A) and the next section (Sec. V B) are found in Tables VIII–X.

We investigate different loss functions for reconstructing the Kolmogorov flow. The same as the reconstruction of the laminar wake (Sec. IV A), all loss functions can achieve reconstruction errors of less than 10%. However, unlike the laminar wake, the loss functions have a stronger influence on the reconstruction of the Kolmogorov flow due to its chaotic nature. Table III shows the mean standard deviation of the relative error computed over five trainings with different random weight initialization. The mean relative error of networks trained with the snapshot-enforced loss $\mathcal{L}^s$ is over 20% lower than the networks trained with $\mathcal{L}^c$ and $\mathcal{L}^m$, while the mean relative errors of the networks trained with $\mathcal{L}^c$ and $\mathcal{L}^m$ are within 3% of each other. The standard deviations of the flow reconstructed with loss functions with harder constraints ($\mathcal{L}^s$ and $\mathcal{L}^m$) are approximately half of the flow reconstructed with the softly constrained loss $\mathcal{L}^c$. This shows that, as the dynamics become more nonlinear, loss functions with harder constraints on the measurements improve the robustness towards network initialization and the accuracy of the output. Figure 12 shows that the reconstructed flow has a smaller instantaneous absolute error.

We also tested how the PC-DualConvNet behaves in extreme situations by randomly removing the sensors in Fig. 11 until only 10 input sensors and ten general sensors remain. The reconstructed flow using 10 sensors has a reconstruction error of 73.1%, higher than the 69.9% of the interpolated flow. However, the instantaneous and mean vorticity field (Fig. 13) shows that the reconstructed flow is physical and shows a closer resemblance to the reference dataset. The reconstructed flow has also recovered the turbulence energy spectrum better compared to the interpolated flow (Fig. 13(b)). The results with 10 sensors highlight the limitation of measuring the quality of reconstruction using only the ℓ_2 error. Even if the interpolated flows have a lower relative error compared to the reconstructed flow, it clearly failed to reconstruct the flow by both the TKE and visual inspection. There are clear discrepancies between the reconstructed and the reference flow in Fig. 13, showing that more sensors are necessary to achieve a good reconstruction.

TABLE III. The snapshot-enforced loss leads to the lowest relative error (mean $\pm$ standard deviation) when reconstructing the turbulent Kolmogorov flow from non-noisy measurements. The relative error is measured over five repeated trainings with different random weight initialization and sensor locations.

	$\mathcal{L}^c$	$\mathcal{L}^s$	$\mathcal{L}^m$	Interpolated
$\epsilon(\%)$	7.19 ± 0.73	5.51 ± 0.34	7.38 ± 0.33	16.46 ± 0.00

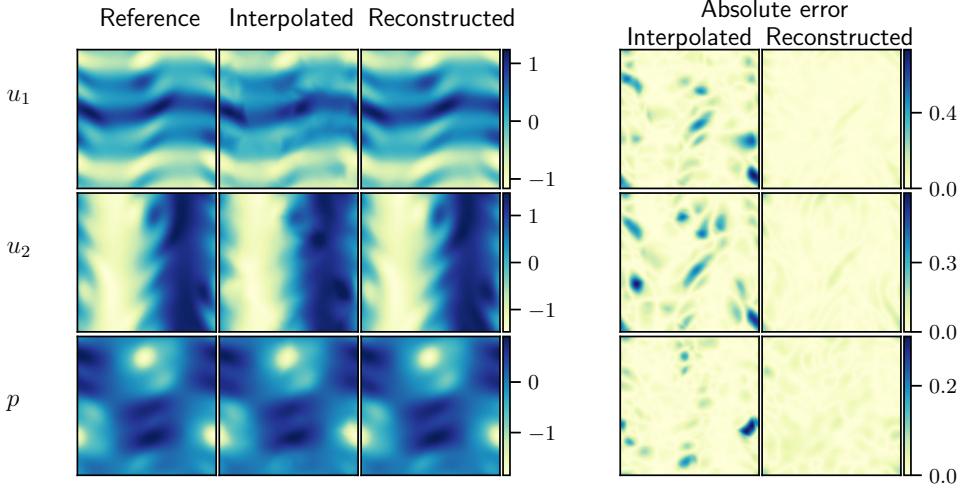


FIG. 12. Reconstructed turbulent Kolmogorov flow from non-noisy measurements with the snapshot-enforced loss, at $t = 1.7$.

B. Reconstructing from noisy measurements

In this section, we reconstruct the turbulent Kolmogorov flow from sparse measurements corrupted with white noise at $\text{SNR} = 20, 10$, and 5 . The same number of sensors (80 inputs, 150 general) are placed at random locations, shown in Fig. 14(a). The relative errors and the physics loss of the flow reconstructed with $\mathcal{L}^c$, $\mathcal{L}^s$, and $\mathcal{L}^m$ are summarized in Fig. 14(b). The physics loss shows a similar trend to Sec. IV B—harder constraints on the measurements lead to higher physics losses as the noise in the measurements increases. At a low level of noise ($\text{SNR} = 20$) $\mathcal{L}^s$ leads to the lowest reconstruction error, although all loss functions achieve a relative error lower than 10%. Unlike when reconstructing the laminar wake, the softly constrained loss is able to reconstruct the Kolmogorov flow from measurements at $\text{SNR} = 10$, suggesting that some variation of performance is the result of hyperparameter selection when the measurements are corrupted with a medium amount of noise. However, hyperparameters may be cumbersome to tune, and a (globally) optimal set is rarely found. On the other hand, $\mathcal{L}^m$ has achieved consistent reconstructions of both the

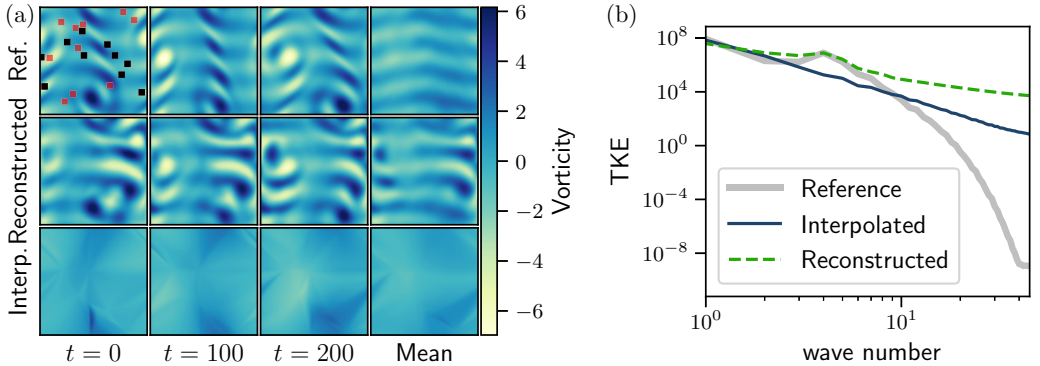


FIG. 13. The flow reconstructed from ten input sensors and 10 general sensors. The sensor placement is shown in the top left-hand panel. The reconstruction has recovered the instantaneous vorticity fields and the turbulent kinetic energy spectrum.

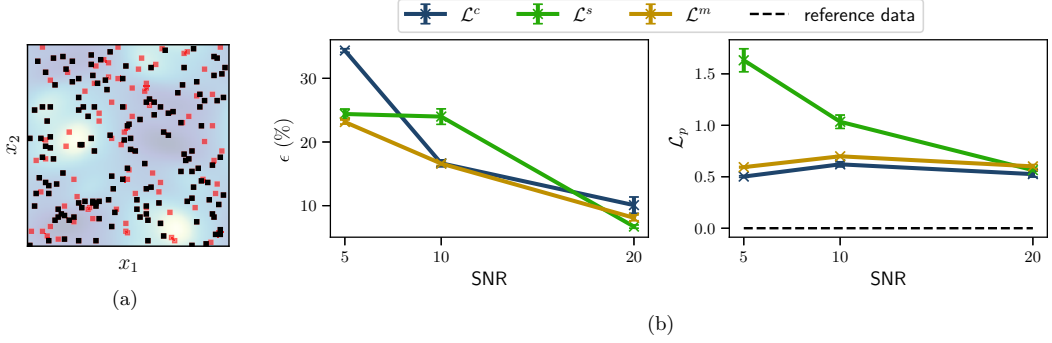


FIG. 14. (a) Sensor placement for reconstructing the turbulent Kolmogorov flows from noisy measurements. (b) Mean and standard deviation of the reconstructed Kolmogorov flow from noisy measurements at different SNR using different loss functions.

laminar flow (Sec. IV B) and the turbulent flow from noisy measurements. As the signal-to-noise ratio decreases further, the reconstruction error of all loss functions increases. At SNR = 5 for $\mathcal{L}^m$ and $\mathcal{L}^c$, the physics loss decreased compared to SNR = 10, showing that the reconstruction is favoring satisfying the physics over the noisy measurements. At SNR = 5, the reconstruction error of the flow reconstructed with $\mathcal{L}^c$ increased to 34.4%, higher than both the snapshot-enforced loss and the mean-enforced loss, showing again that harder constraints improve the robustness of the PC-DualConvNet to noise. The strictly enforced mean loss achieves a reconstruction error of 23.1%, similar to the error of the snapshot-enforced loss, but with much lower physics loss. For all SNRs, the reconstructed flows have recovered the correct energy spectrum (Fig. 15) up to wave number 1, showing that the reconstructed flow is more physical than the interpolated flow even when starting from noisy measurements.

Figure 16 compares the instantaneous snapshots and the mean of the reconstructed flows. At SNR = 20 (Fig. 16(a)) there is no visible difference between the reconstructed flows using different loss functions, even though the relative error shows that $\mathcal{L}^c$ is slightly worse than others. The

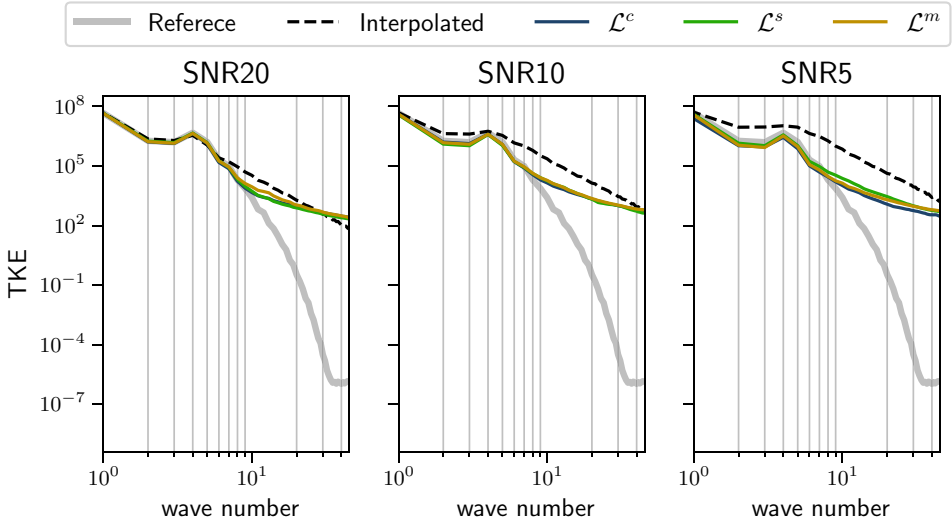


FIG. 15. The turbulent kinetic energy (TKE) spectrum.

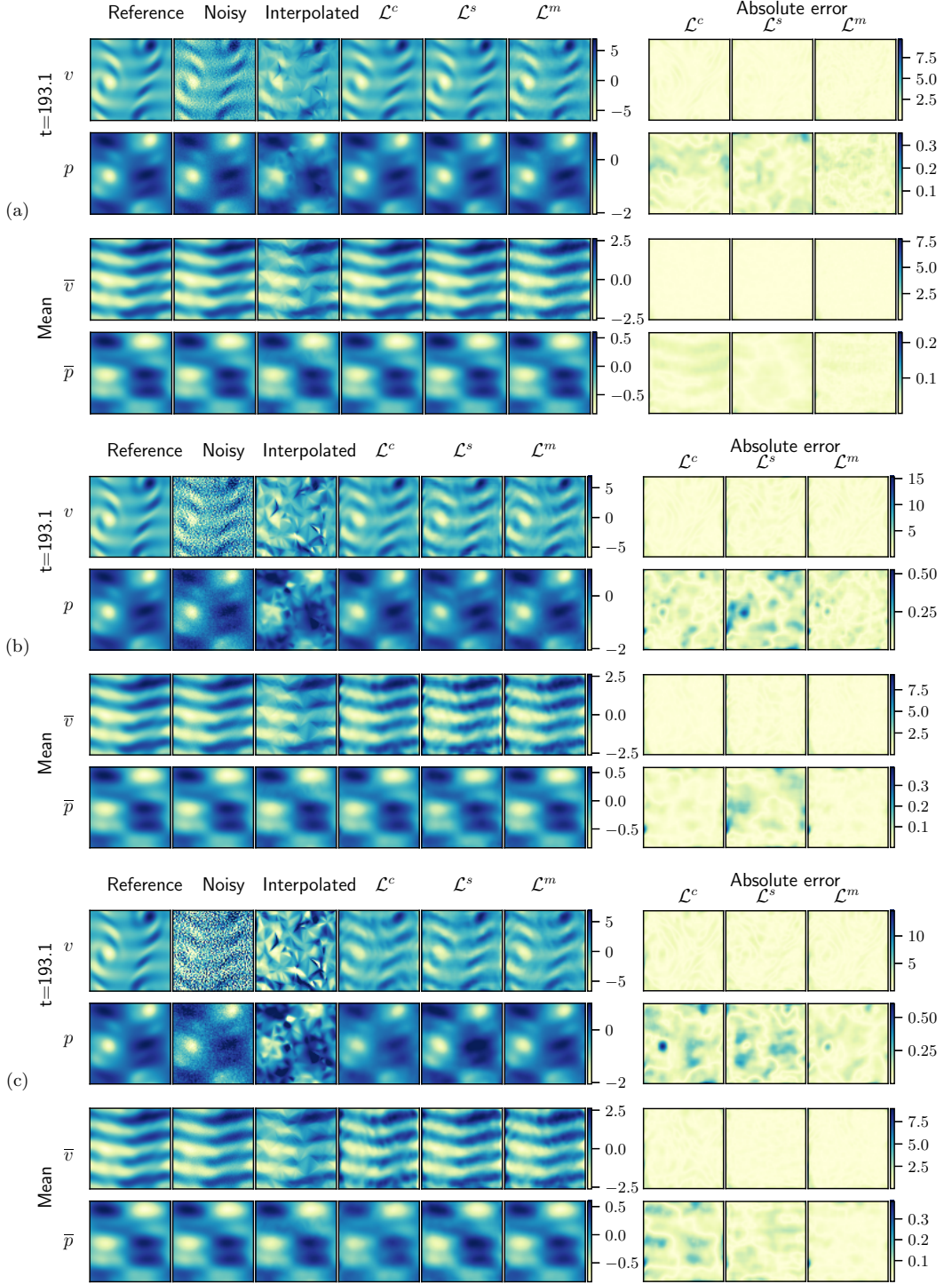


FIG. 16. Reconstructed Kolmogorov flow from noisy measurements at SNR = 20 (a), 10 (b), and 5 (c).

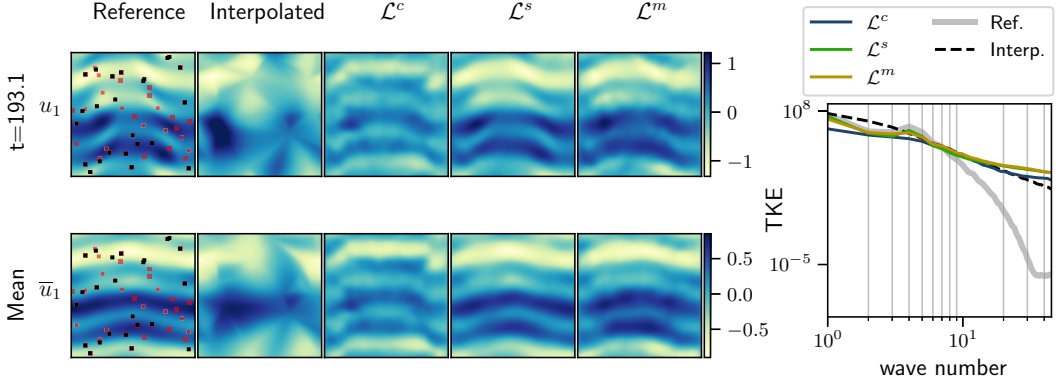


FIG. 17. Reconstructed Kolmogorov flows from sparse noisy measurements at SNR = 10. Only 24 input sensors and general sensors are employed (the sensors are shown in the left panels). The relative errors for each reconstruction case are 70% (interpolated), 75% ($\mathcal{L}^c$), 30% ($\mathcal{L}^s$), and 45% ($\mathcal{L}^m$).

flow reconstructed with $\mathcal{L}^s$ starts showing visible noise from SNR = 10 (Figs. 16(b) and 16(c)). At SNR = 5, the flow reconstructed with $\mathcal{L}^c$ shows a visible reduction in the variation of the flow field, while the flow reconstructed with $\mathcal{L}^m$ and $\mathcal{L}^s$ is not affected. At all SNRs using any loss function, the instantaneous reconstructed flow shows a closer resemblance to the reference data than the interpolated flow. Considering the relative error, the physics loss (Fig. 14(b)) and the snapshots (Fig. 16), $\mathcal{L}^m$ provides the most consistent performance at different levels of noise, providing a small relative error while keeping the physics loss close to the reference physics loss at high levels of noise.

C. Reconstructing from sparse noisy measurements

In this section, we present a case in which the sensors are sparse and noisy. From the sensors' locations of Sec. VB, we randomly select 24 general sensors and 24 input sensors. In the current setup, the general sensors account for 0.14% of all grid points. The measurements contain white noise at SNR = 10.

The reconstructed flows are shown in Fig. 17. (The hyperparameters can be found in the Appendix A2.) The flows reconstructed with the snapshot-enforced loss and the mean-enforced

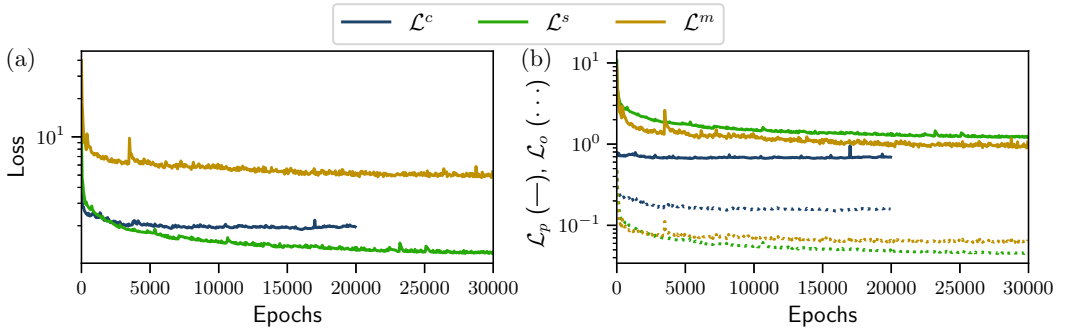


FIG. 18. Learning curves for sparse and noisy measurements: (a) the training loss; (b) the physics loss (solid line) and the data loss (dotted line). When using either the snapshot-enforced or the mean-enforced loss, the physics and data losses decrease together. On the other hand, when using the softly constrained loss, the data loss decreases while the physics loss increases.

loss closely match the reference energy spectrum up to a wave number ≈ 7 (Fig. 17 right panel). In contrast, both interpolation and the softly constrained loss have larger reconstruction errors. For the network trained with the softly constrained loss, the data loss decreases while the physics loss increases during training (Fig. 18(b)), ultimately causing the network to terminate training prematurely (Fig. 18(a)). This difficulty in simultaneously optimizing the physics and data loss terms forces the network to prioritize one over the other, which can result in trivial solutions (see also Sec. III C 1).

When measurements are sparse, the primary challenge in reconstructing flows with neural networks is finding a solution that fits the limited data, rather than filtering out noise as in Sec. V B. The snapshot-enforced loss prioritizes fitting the measurements more than the other loss functions, making it the most effective choice when the availability of data is limited and denoising is not the primary objective.

VI. CONCLUSION

In this paper, we reconstruct flow fields from sparse, noisy, and heterogeneous measurements (incomplete data) with a physics-constrained dual-branch convolutional neural network (PC-DualConvNet). The network is trained using only incomplete data. To recover the missing information and infer the full flow field, the governing equations of the flow are embedded into the loss function. We investigate three loss functions, i.e., the softly constrained loss, which is common in the flow reconstruction literature [41]; the snapshot-enforced loss [46] for steady flows; and the mean-enforced loss (proposed here). First, we reconstruct both the laminar wake and the turbulent Kolmogorov flow from non-noisy measurements from a number of sensors of $\sim 1\%$ of all grid points. When reconstructing the laminar wake, we find no significant difference between the softly constrained and the snapshot-enforced losses. However, when reconstructing the turbulent Kolmogorov, we find that the snapshot-enforced loss leads to a $\sim 25\%$ reduction in the reconstruction error compared to the softly constrained loss due to the flow being chaotic. Second, we tested the loss functions on measurements corrupted with white noise at signal-to-noise ratios (SNR) 20, 10, and 5. We find that the Fourier layer in the PC-DualConvNet improves the robustness of the snapshot-enforced loss to noise, which is designed for non-noisy measurements. At $\text{SNR} < 10$, the snapshot-enforced loss leads to noisy reconstructed flows because the hard constraint on the instantaneous measurements prevents the network from denoising. On the other hand, the softly constrained loss leads to a smaller variability in the reconstructed flow dynamics because the network favors minimizing the physics loss over the data loss. Finally, combining the benefit of the softly constrained and the snapshot-enforced losses, we propose the mean-enforced loss. At large levels of noise, the flow reconstructed with the mean-enforced loss is physical and does not spuriously attenuate the flow variations, while achieving similar reconstruction errors to the snapshot-enforced loss. The reconstruction errors show that hard constraints in the loss functions make the PC-DualConvNet more generalisable. In conclusion, we find the snapshot-enforced loss is suitable for reconstructing flows from non-noisy measurements, and the mean-enforced loss is suitable for reconstructing flows in more general situations, i.e., when the measurements are noisy.

ACKNOWLEDGMENTS

We acknowledge funding from the Engineering and Physical Sciences Research Council, UK and financial support from the ERC Starting Grant PhyCo 949388. L.M. is also grateful for the support from the grant EU-PNRR YoungResearcher TWIN ERC-PI_0000005. Y.M. acknowledges support from the Department of Aeronautics, Imperial College.

DATA AVAILABILITY

Codes are available at [68].

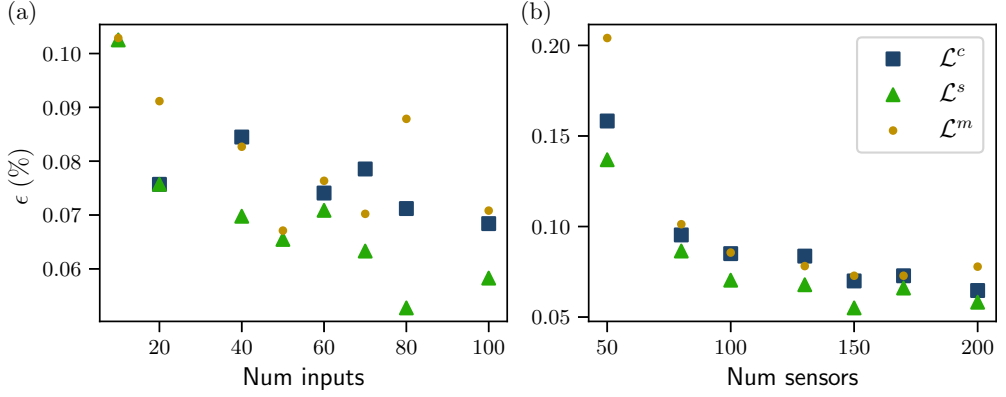


FIG. 19. Reconstruction error in response to changing the number of (a) pressure inputs P_{in} while holding the number of sensors constant at 200 and (b) sensors $D(x_s)$ while holding the number of pressure inputs constant at 80. This grid sensitivity test is performed on a Kolmogorov flow simulated with a different initial random seed.

APPENDIX

1. Grid sensitivity study for the reconstruction of the Kolmogorov flow

We use a Kolmogorov flow dataset generated with a different initial random seed than the dataset used in the rest of Sec. V to determine the number of sensors to place in the domain. A different random seed is used to eliminate possible bias in sensor placement. In order words, we want to make sure the reconstruction results presented in the main text come from a previously unseen dataset. We followed the procedure below to determine the number of sensors:

- (1) Conduct a hyperparameter search for all loss functions $\mathcal{L}^c$, $\mathcal{L}^s$, and $\mathcal{L}^m$ using 80 input sensors (x_{in}) and 200 general sensors (x_s) randomly placed in the domain.
- (2) Randomly remove the input pressure sensors while holding the number of general sensors constant at 200, until the relative error starts to increase (Fig. 19(a)). We find that the relative error of the reconstructed flow starts to increase when the number of input sensors is smaller than 80.
- (3) Randomly remove the general sensors while holding the number of input sensors constant at 80, until the relative error starts to increase (Fig. 19(b)). We find that the relative error starts to increase when using fewer than 150 sensors.

For Secs. V A and V B, we use 80 input sensors and 150 general sensors randomly placed in the domain to reconstruct the flow.

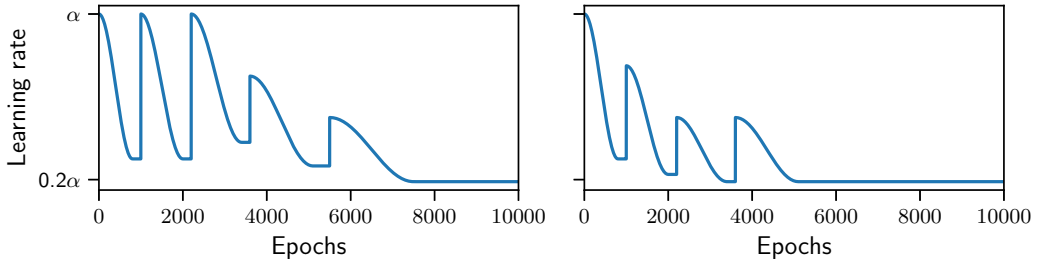


FIG. 20. The cyclic decay learning rate schedule applied to reconstruct the laminar flow (left) and the Kolmogorov flow (right), where α is the initial learning rate.

TABLE IV. The network structure and hyperparameters used to reconstruct the laminar wake from non-noisy measurements.

Bottleneck image dimension	(64,16)
Convolution filter size (lower branch)	(5,5)
Convolution filter size (all other branches)	(3,3)
Input branch channels	[3,]
Upper branch channels	[8,]
Lower branch channels	[8,16,8]
Output branch channels	[4,3]
FFT	off
Batch size	[35]
λ_{div}	1.0
λ_{mom}	1.0
λ_{o}	40.0
Initial learning rate (α)	0.0023
Regularization	0.0
Dropout rate	0.0

TABLE V. The structure and hyperparameters of the network using the softly constrained loss, from noisy measurements, showing the difference from Table IV only.

	SNR = 20	SNR = 10	SNR = 5
Upper branch channels	[4,4]	[4,4]	[4,4]
Initial learning rate (α)	0.0013	0.0023	0.0012
Batch size	46	67	67
Regularization	0.007	0.05	0.022
λ_{div}	2.0	2.1	3.8
λ_{o}	12.0	11.0	11.0
Dropout rate	0.05%	0.5%	0.5%

TABLE VI. The structure and hyperparameters of the network using the snapshot-enforced loss, from noisy measurements, showing the difference from Sec. IV only.

	SNR = 20	SNR = 10	SNR = 5
Upper branch channels	[1,]	[1,]	[1,]
FFT	On	On	On
Initial learning rate (α)	0.014	0.0068	0.0088
Batch size	37	46	46
Regularization	0.095	0.056	0.070
λ_{div}	1.0	1.0	1.0
λ_{o}	n/a	n/a	n/a
Dropout rate	0.48%	0.55%	0.6%

TABLE VII. The structure and hyperparameters of the network using the mean-enforced loss, from noisy measurements, showing the difference from Table IV only.

	SNR=20	SNR=10	SNR=5
Upper branch channels	[1,]	[4,4]	[8,]
Initial learning rate (α)	0.003	0.0023	0.0046
Batch size	50	200	120
Regularization	0.076	0.09	0.042
λ_{div}	2.5	2.7	2.6
λ_{o}	31.5	48.5	24.0
Dropout rate	0.22%	0.25%	0.97%

TABLE VIII. The structure and hyperparameters of the network used to reconstruct the Kolmogorov flow with softly constrained loss. The empty cells in columns 2 through 4 represent parameters that are the same as in column 1. The very sparse case presented in Sec. VC is trained with the hyperparameters in column 1.

	Non-noisy	SNR = 20	SNR = 10	SNR = 5
Bottleneck image dimension	(16,16)			
Convolution filter size (upper branches)	(5,5)			
Convolution filter size (lower branches)	(5,5)		(3,3)	(3,3)
Convolution filter size (all other)	(3,3)			
Input branch channels	[3,]			
Upper branch channels	[4,4]			
Lower branch channels	[4,8,16,8,4]			
Output branch channels	[4,3]			
FFT	off			
Batch size	182		214	240
λ_{div}	1.0			
λ_{mom}	1.0			
λ_{o}	8.0		26.1	7.6
Initial learning rate (α)	0.004		0.00034	0.00018
Regularization	0.0		0.0012	0.0037
Dropout rate	0.0%		0.24%	0.61%

TABLE IX. The structure and hyperparameters of the network used to reconstruct the Kolmogorov flow with snapshot-enforced loss. The empty cells in columns 2 through 4 represent parameters that are the same as in column 1. The very sparse case presented in Sec. VC is trained with the hyperparameters in column 1.

	Non-noisy	SNR = 20	SNR = 10	SNR = 5
Bottleneck image dimension	(16,16)			
Convolution filter size (upper branch)	(5,5)			
Convolution filter size (lower branch)	(5,5)			(3,3)
Convolution filter size (all other)	(3,3)			
Input branch channels	[3,]			
Upper branch channels	[1,]			
Lower branch channels	[4,8,16,8,4]			
Output branch channels	[4,3]			
FFT	on			
Batch size	158		214	214
λ_{div}	1.0			
λ_{mom}	1.0			
Initial learning rate (α)	0.0008		0.0028	0.0088
Regularization	0.0		0.008	0.0056
Dropout rate	0.0%		0.59%	0.17%

TABLE X. The structure and hyperparameters of the network used to reconstruct the Kolmogorov flow with mean-enforced loss. The empty cells in columns 2 through 4 represent parameters that are the same as in column 1. The very sparse case presented in Sec. VC is trained with the hyperparameters in column 1.

	Non-noisy	SNR = 20	SNR = 10	SNR = 5
Bottleneck image dimension	(16,16)			
Convolution filter size (upper branch)	(5,5)			
Convolution filter size (lower branch)	(5,5)		(3,3)	(3,3)
Convolution filter size (all other)	(3,3)			
Input branch channels	[3,]			
Upper branch channels	[4,]			
Lower branch channels	[4,8,16,8,4]			
Output branch channels	[4,3]			
FFT	off			
Batch size	600		857	1200
λ_{div}	1.0			
λ_{mom}	1.0			
λ_0	64.0		34.6	5.9
Initial learning rate (α)	0.0047		0.0013	0.0013
Regularization	0.0		0.0007	0.0039
Dropout rate	0.0%		0.21%	0.23%

2. Model structures and hyperparameters

This section contains the training- and network-related parameters for all examples shown in this paper. Figure 20 shows the learning rate schedules employed in the reconstruction of the laminar wake and the Kolmogorov flow. Tables IV–VII contains the network and training parameters for

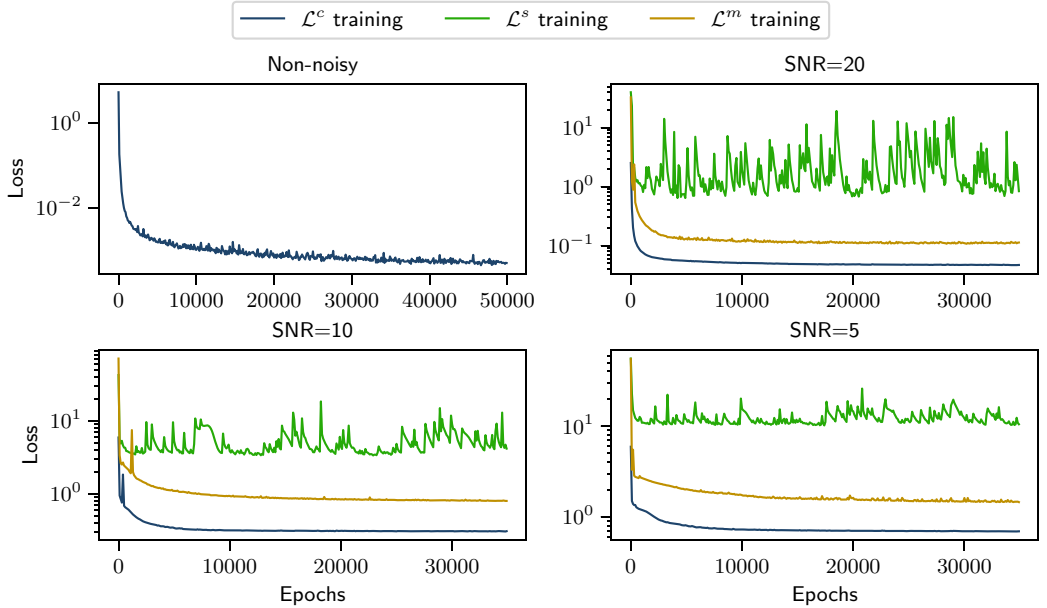


FIG. 21. Learning curves of the reconstruction of the laminar wake dataset. Learning curves are shown for the test runs plotted in Figs. 7 and 10.

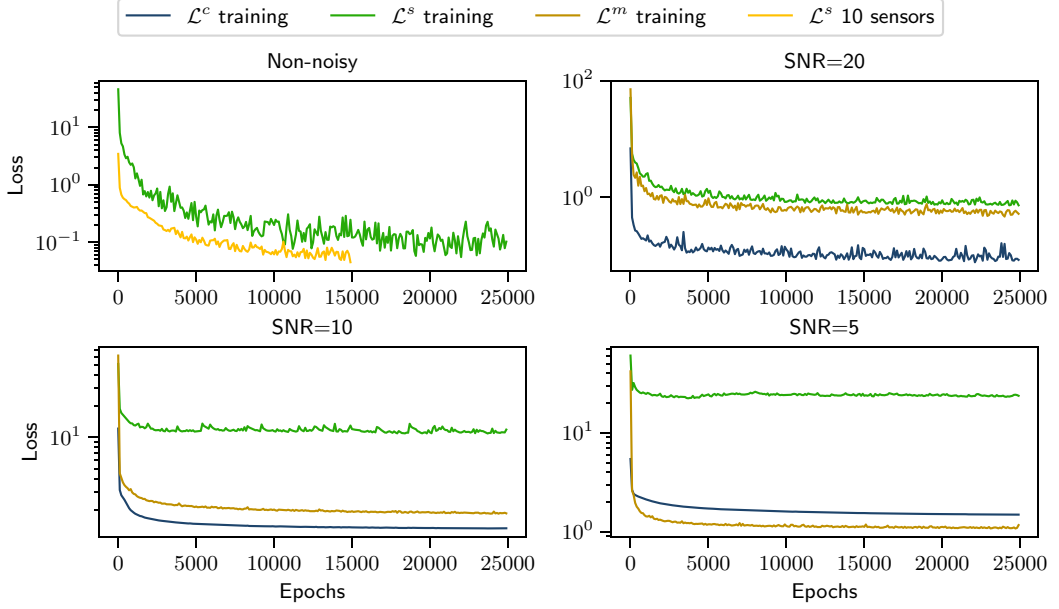


FIG. 22. Learning curves of the reconstruction of the turbulent Kolmogorov flow. Learning curves are shown for the test runs plotted in Figs. 12, 13, and 16.

the laminar wake test cases. Tables VIII–X contains the network and training parameters for the Kolmogorov flow test cases. The parameter “bottleneck image dimension” refers to the smallest image dimension achieved through the resizing layers in the lower branch of the PC-DualConvNet.

3. Learning curves

Figures 21 and 22 show the learning curves for the laminar wake and Kolmogorov examples shown in this paper.

-
- [1] S. Beneddine, R. Yegavian, D. Sipp, and B. Leclaire, Unsteady flow dynamics reconstruction from mean flow and point sensors: An experimental study, *J. Fluid Mech.* **824**, 174 (2017).
 - [2] J. L. Callaham, K. Maeda, and S. L. Brunton, Robust flow reconstruction from limited measurements via sparse representation, *Phys. Rev. Fluids* **4**, 103907 (2019).
 - [3] P. Chandramouli, E. Memin, D. Heitz, and L. Fiabane, Fast 3D flow reconstructions from 2D cross-plane observations, *Exp. Fluids* **60**, 30 (2019).
 - [4] L. Franceschini, D. Sipp, and O. Marquet, Mean- and unsteady-flow reconstruction with one or two time-resolved measurements, *arXiv:2102.03839*.
 - [5] R. Everson and L. Sirovich, Karhunen–Loève procedure for gappy data, *J. Opt. Soc. Am. A* **12**, 1657 (1995).
 - [6] A. Nekkanti and O. T. Schmidt, Gappy spectral proper orthogonal decomposition, *J. Comput. Phys.* **478**, 111950 (2023).
 - [7] D. Venturi and G. Karniadakis, Gappy data and reconstruction procedures for flow past a cylinder, *J. Fluid Mech.* **519**, 315 (1999).
 - [8] H. Gunes, S. Sirisup, and G. E. Karniadakis, Gappy data: To Krig or not to Krig? *J. Comput. Phys.* **212**, 358 (2006).

- [9] S. G. Raben, J. J. Charonko, and P. P. Vlachos, Adaptive gappy proper orthogonal decomposition for particle image velocimetry data reconstruction, *Meas. Sci. Technol.* **23**, 025303 (2012).
- [10] F.-X. Le Dimet and O. Talagrand, Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects, *Tellus A: Dyn. Meteorol. Oceanogr.* **38**, 97 (1986).
- [11] V. Mons, J. C. Chassaing, T. Gomez, and P. Sagaut, Reconstruction of unsteady viscous flows using data assimilation schemes, *J. Comput. Phys.* **316**, 255 (2016).
- [12] S. Symon, N. Dovetta, B. J. McKeon, D. Sipp, and P. J. Schmid, Data assimilation of mean velocity from 2D PIV measurements of flow over an idealized airfoil, *Exp. Fluids* **58**, 61 (2017).
- [13] L. Franceschini, D. Sipp, and O. Marquet, Mean-flow data assimilation based on minimal correction of turbulence models: Application to turbulent high Reynolds number backward-facing step, *Phys. Rev. Fluids* **5**, 094603 (2020).
- [14] M. Wang and T. A. Zaki, State estimation in turbulent channel flow from limited observations, *J. Fluid Mech.* **917**, A9 (2021).
- [15] T. A. Zaki and M. Wang, From limited observations to the state of turbulence: Fundamental difficulties of flow reconstruction, *Phys. Rev. Fluids* **6**, 100501 (2021).
- [16] A. S. Cato, P. S. Volpiani, V. Mons, O. Marquet, and D. Sipp, Comparison of different data-assimilation approaches to augment RANS turbulence models, *Comput. Fluids* **266**, 106054 (2023).
- [17] Q. Wang, R. Hu, and P. Blonigan, Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations, *J. Comput. Phys.* **267**, 210 (2014).
- [18] F. Huhn and L. Magri, Stability, sensitivity and optimisation of chaotic acoustic oscillations, *J. Fluid Mech.* **882**, A24 (2020).
- [19] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter* (Springer-Verlag, Berlin, 2009).
- [20] C. H. Colburn, J. B. Cessna, and T. R. Bewley, State estimation in wall-bounded flow systems. Part 3. The ensemble Kalman filter, *J. Fluid Mech.* **682**, 289 (2011).
- [21] Z. Deng, C. He, X. Wen, and Y. Liu, Recovering turbulent flow field from local quantity measurement: Turbulence modeling using ensemble-Kalman-filter-based data assimilation, *J. Visualization* **21**, 1043 (2018).
- [22] T. Suzuki and Y. Hasegawa, Estimation of turbulent channel flow at based on the wall measurement using a simple sequential approach, *J. Fluid Mech.* **830**, 760 (2017).
- [23] X.-L. Zhang, H. Xiao, G.-W. He, and S.-Z. Wang, Assimilation of disparate data for enhanced reconstruction of turbulent mean flows, *Comput. Fluids* **224**, 104962 (2021).
- [24] A. Nóvoa, A. Racca, and L. Magri, Inferring unknown unknowns: regularized bias-aware ensemble Kalman filter, *Comput. Methods Appl. Mech. Eng.* **418**, 116502 (2024).
- [25] S. L. Brunton and J. N. Kutz, Promising directions of machine learning for partial differential equations, *Nat. Comput. Sci.* **4**, 483 (2024).
- [26] N. B. Erichson, L. Mathelin, Z. Yao, S. L. Brunton, M. W. Mahoney, and J. N. Kutz, Shallow neural networks for fluid flow reconstruction with limited sensors, *Proc. R. Soc. A* **476**, 20200097 (2020).
- [27] P. Dubois, T. Gomez, L. Planckaert, and L. Perret, Machine learning for fluid flow reconstruction from limited measurements, *J. Comput. Phys.* **448**, 110733 (2022).
- [28] R. Maulik, K. Fukami, N. Ramachandra, K. Fukagata, and K. Taira, Probabilistic neural networks for fluid flow surrogate modeling and data recovery, *Phys. Rev. Fluids* **5**, 104401 (2020).
- [29] T. Li, M. Buzzicotti, L. Biferale, F. Bonaccorso, S. Chen, and M. Wan, Multi-scale reconstruction of turbulent rotating flows with proper orthogonal decomposition and generative adversarial networks, *J. Fluid Mech.* **971**, A3 (2023).
- [30] M. Buzzicotti, F. Bonaccorso, P. C. Di Leoni, and L. Biferale, Reconstruction of turbulent data with deep generative models for semantic inpainting from TURB-Rot database, *Phys. Rev. Fluids* **6**, 050503 (2021).
- [31] H. Kim, J. Kim, S. Won, and C. Lee, Unsupervised deep learning for super-resolution reconstruction of turbulence, *J. Fluid Mech.* **910**, A29 (2021).
- [32] L. Yu, M. Z. Yousif, M. Zhang, S. Hoyas, R. Vinuesa, and H.-C. Lim, Three-dimensional ESRGAN for super-resolution reconstruction of turbulent flows with tricubic interpolation-based transfer learning, *Phys. Fluids* **34**, 125126 (2022).

- [33] M. Z. Yousif, L. Yu, S. Hoyas, R. Vinuesa, and H. Lim, A deep-learning approach for reconstructing 3D turbulent flows from 2D observation data, *Sci. Rep.* **13**, 2529 (2023).
- [34] M. Sardar, A. Skillen, M. J. Zimón, S. Draycott, and A. Revell, Spectrally decomposed diffusion models for generative turbulence recovery, *arXiv:2312.15029*.
- [35] K. Fukami, K. Fukagata, and K. Taira, Super-resolution reconstruction of turbulent flows with machine learning, *J. Fluid Mech.* **870**, 106 (2019).
- [36] K. Fukami, K. Fukagata, and K. Taira, Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows, *J. Fluid Mech.* **909**, A9 (2021).
- [37] M. Matsuo, K. Fukami, T. Nakamura, M. Morimoto, and K. Fukagata, Reconstructing three-dimensional bluff body wake from sectional flow fields with convolutional neural networks, *SN Comput. Sci.* **5**, 306 (2024).
- [38] A. Güemes, C. Sanmiguel Vila, and S. Discetti, Super-resolution generative adversarial networks of randomly-seeded fields, *Nat Mach Intell* **4**, 1165 (2022).
- [39] A. G. Özbay and S. Laizet, Deep learning fluid flow reconstruction around arbitrary two-dimensional objects from sparse sensors using conformal mappings, *AIP Adv.* **12**, 045126 (2022).
- [40] X.-H. Zhou, J. E. McClure, C. Chen, and H. Xiao, Neural network-based pore flow field prediction in porous media using super resolution, *Phys. Rev. Fluids* **7**, 074302 (2022).
- [41] M. Raissi, P. Perdikaris, and G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *J. Comput. Phys.* **378**, 686 (2019).
- [42] M. Raissi, A. Yazdani, and G. E. Karniadakis, Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, *Science* **367**, 1026 (2020).
- [43] A. Arzani, J.-X. Wang, and R. M. D’Souza, Uncovering near-wall blood flow from sparse data with physics-informed neural networks, *Phys. Fluids* **33**, 071905 (2021).
- [44] J. Zhang and X. Zhao, Three-dimensional spatiotemporal wind field reconstruction based on physics-informed deep learning, *Appl. Energy* **300**, 117390 (2021).
- [45] H. Eivazi, Y. Wang, and R. Vinuesa, Physics-informed deep-learning applications to experimental fluid mechanics, *Meas. Sci. Technol.* **35**, 075303 (2024).
- [46] H. Gao, L. Sun, and J.-X. Wang, Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels, *Phys. Fluids* **33**, 073603 (2021).
- [47] D. Kelshaw and L. Magri, Physics-constrained convolutional neural networks for inverse problems in spatiotemporal partial differential equations, *DCE* **5**, e43 (2024).
- [48] A. Subramaniam, M. L. Wong, R. D. Borker, S. Nimmagadda, and S. K. Lele, Turbulence enrichment using physics-informed generative adversarial networks, *arXiv:2003.01907*.
- [49] J. C. Wong, C. Ooi, A. Gupta, and Y.-S. Ong, Learning in sinusoidal spaces with physics-informed neural networks, *IEEE Trans. Artificial Intell.* **5**, 985 (2022).
- [50] G. Raynaud, S. Houde, and F. P. Gosselin, ModalPINN: An extension of physics-informed neural networks with enforced truncated Fourier decomposition for periodic flow reconstruction using a limited number of imperfect sensors, *J. Comput. Phys.* **464**, 111271 (2022).
- [51] D. E. Ozan and L. Magri, Hard-constrained neural networks for modeling nonlinear acoustics, *Phys. Rev. Fluids* **8**, 103201 (2023).
- [52] K. Fukami, K. Fukagata, and K. Taira, Super-resolution analysis via machine learning: A survey for fluid flows, *Theor. Comput. Fluid Dyn.* **37**, 421 (2023).
- [53] L. Sun and J.-X. Wang, Physics-constrained Bayesian neural network for fluid flow reconstruction with sparse and noisy data, *Theor. Appl. Mech. Lett.* **10**, 161 (2020).
- [54] L. Sliwinski and G. Rigas, Mean flow reconstruction of unsteady flows using physics-informed neural networks, *Data-Centric Eng.* **4**, e4 (2023).
- [55] I. J. S. Shokar, R. R. Kerswell, and P. H. Haynes, Stochastic latent transformer: Efficient modeling of stochastically forced zonal jets, *J. Adv. Model. Earth Syst.* **16**, e2023MS004177 (2024).
- [56] N. Agrawal, S. Dutta, and B. K. Gandhi, An experimental study on flow past an equilateral triangular prism at intermediate Reynolds number and the effect of its orientation, in *Vol. 1B, Symposia: Fluid Machinery; Fluid Power; Fluid-Structure Interaction and Flow-Induced Noise in Industrial Applications*;

- Flow Applications in Aerospace; Flow Manipulation and Active Control: Theory, Experiments and Implementation; Fundamental Issues and Perspectives in Fluid Mechanics* (American Society of Mechanical Engineers, Incline Village, NV, 2013).
- [57] A. Ghani, T. Poinso, L. Gicquel, and G. Staffelbach, LES of longitudinal and transverse self-excited combustion instabilities in a bluff-body stabilized turbulent premixed flame, *Combust. Flame* **162**, 4075 (2015).
 - [58] P. Bartholomew, S. Laizet, F. N. Schuch, R. A. S. Frantz, S. Rolfo, A. Hamzehloo, C. Flageul, G. Y. Deskos, N. Bempedelis, K. Schäfer, nasos94, H. Jing, vc218385, and L. R. Monteiro, Xcompact3d/Incompact3d, Zenodo (2022), <https://zenodo.org/record/5870206>.
 - [59] S. Laizet and E. Lamballais, High-order compact schemes for incompressible flows: A simple and efficient method with quasi-spectral accuracy, *J. Comput. Phys.* **228**, 5989 (2009).
 - [60] S. Ganga Prasath, M. Sudharsan, V. Vinodh Kumar, S. Diwakar, T. Sundararajan, and S. Tiwari, Effects of aspect ratio and orientation on the wake characteristics of low Reynolds number flow over a triangular prism, *J. Fluids Struct.* **46**, 59 (2014).
 - [61] D. Kelshaw, MagriLab/KolSol: Pseudospectral Kolmogorov Flow Solver (2024), <https://github.com/MagriLab/KolSol>.
 - [62] A. Racca, Neural networks for the prediction of chaos and turbulence, Ph.D. thesis, University of Cambridge, 2023.
 - [63] Rowan D. Brackston, Feedback control of three-dimensional bluff body wakes for efficient drag reduction, Ph.D. thesis, Imperial College London, 2017.
 - [64] O. Ronneberger, P. Fischer, and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015*, edited by N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi (Springer International Publishing, Cham, 2015), Vol. 9351, pp. 234–241.
 - [65] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, Fourier neural operator for parametric partial differential equations, [arXiv:2010.08895](https://arxiv.org/abs/2010.08895).
 - [66] K. Cohen, S. Siegel, and T. McLaughlin, Sensor placement based on proper orthogonal decomposition modeling of a cylinder wake, in *33rd AIAA Fluid Dynamics Conference and Exhibit* (American Institute of Aeronautics and Astronautics, Orlando, FL, 2003).
 - [67] <http://wandb.ai/>.
 - [68] <https://github.com/MagriLab/FlowReconstructionFromExperiment>.