Experiments in Fluids (2023) 64:95
https://doi.org/10.1007/s00348-023-03629-4

RESEARCH ARTICLE

Reconstructing turbulent velocity and pressure fields from under-resolved noisy particle tracks using physics-informed neural networks

Patricio Clark Di Leoni[1] · Karuna Agarwal[2] · Tamer A. Zaki[2] · Charles Meneveau[2] · Joseph Katz[2]

Received: 10 October 2022 / Revised: 8 March 2023 / Accepted: 3 April 2023 / Published online: 2 May 2023

Abstract
Volume-resolving imaging techniques are rapidly advancing progress in experimental fluid mechanics. However, reconstructing the full and structured Eulerian velocity and pressure fields from under-resolved and noisy particle tracks obtained experimentally remains a significant challenge. We adopt and characterize a method based on Physics-Informed Neural Networks (PINNs). In this approach, the network is regularized by the Navier–Stokes equations to interpolate the velocity data and simultaneously determine the pressure field. We compare this approach to the state-of-the-art Constrained Cost Minimization method Agarwal et al. (2021). Using data from direct numerical simulations and various types of synthetically generated particle tracks, we show that PINNs are able to accurately reconstruct both velocity and pressure even in regions with low particle density and small accelerations. We analyze both the root-mean-square error of the reconstructions as well their energy spectra. PINNs are also robust against increasing the distance between particles and the noise in the measurements, when studied under synthetic and experimental conditions. Both the synthetic and experimental datasets used correspond to moderate Reynolds number flows.

1 Introduction

Unstructured particle trajectory information measured in a fluid flow experiment through particle tracking techniques (Dabiri and Pecora 2019), such as the Shake-the-Box method (Schanz et al. 2016), can be used to calculate velocity statistics and Lyapunov exponents, find Lagrangian coherent structures (Haller 2015), or estimate helicity (Angriman et al. 2021), for example. More challenging, however, is the calculation of spatial gradients or reconstruction of the pressure field. Procedures to infer the pressure usually involve first interpolating the velocity field and then solving the pressure-Poisson equation (Ghaemi et al. 2012; de Kat and Ganapathisubramani 2012; van Oudheusden 2013; Villegas and Diez 2014). Other methods rely on calculating the material acceleration, either through a pseudo-Lagrangian (Jensen and Pedersen 2004; Liu and Katz 2006, 2013) or Eulerian approach (Violato et al. 2011) from the interpolated velocity fields, or directly from the tracks (Novara and Scarano 2013; Schanz et al. 2016), and then integrating the pressure gradients (Baur and Kongeter 1999; Dabiri et al. 2014) with, for example, an omni-directional method (Liu and Katz 2006; Wang et al. 2019).

The problem of reconstructing flow fields from partial measurements can be approached using a variety of data assimilation techniques (Zaki and Wang 2021). Ensemble- and adjoint-variational methods (Mons et al. 2019; Buchta and Zaki 2021; Buchta et al. 2022; Wang et al. 2019, 2019) and nudging approaches (Clark Di Leoni et al. 2018, 2020; Wang and Zaki 2022), can mix Eulerian and Lagrangian data, and work under turbulent conditions, but usually require numerically solving sets of partial differential equations on a grid. On the other hand, new machine learning methods are proving to be extremely successful in many related tasks. Convolutional neural networks (Fukami et al. 2019; Callaham et al. 2019; Cai et al. 2020; Lagemann et al. 2021), generative adversarial networks (Xie et al. 2018; Buzzicotti et al. 2021; Oh et al. 2022), and deep operator networks (Cai et al. 2021; Mao et al. 2021; Clark Di Leoni et al. 2023d), have been applied to the reconstruction of

✉ Patricio Clark Di Leoni
pclarkdileoni@udesa.edu.ar

[1] Departamento de Ingeniería, Universidad de San Andrés, Victoria, Buenos Aires, Argentina

[2] Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218, USA

two-dimensional images and low-dimensional flow fields. In the context of PIV and PTV measurements, deep neural networks have been used to process and enhance PIV measurements of the velocity field (Cai et al. 2020; Lagemann et al. 2022, 2021; Liang et al. 2020) as well as track particles in PTV (Mallery et al. 2020; Liang et al. 2022).

In this paper we focus on Physics-Informed Neural Networks (Raissi et al. 2019), a class of artificial neural networks designed to approximate physical fields and whose training is informed by the physics of the problem. The physics-information comes from adding the residuals of the equations of motion of the problem, calculated directly from the network using automatic differentiation (Goodfellow et al. 2016), to the loss function as a regularization term and therefore weakly enforcing the governing equations. The network architecture can also be designed to enforce hard constraints, for example predictions of solenoidal fields (Du and Zaki 2021). Two notable advantages are that PINNs do not require data on a regular grid and do not require information on the material acceleration. PINNs have been shown to be effective in inverse problems in fluid and solid mechanics (Raissi et al. 2020; Shukla et al. 2020) and have been used to reconstruct velocity and pressure fields in Tomographic Background Oriented Schilieren measurements at low Reynolds number (Cai et al. 2021), as well as enhance PIV reconstructions (Hasanuzzaman et al. 2022). A detailed comparison of PINN and adjoint-variational data assimilation (4DVar) in a minimal turbulent channel flow was performed in Du et al. (2023), where it was shown that while PINNs are robust to measurement noise, 4DVar is more accurate for sparse data. Also, two detailed reviews on PINNs have been published recently, one with a general focus (Cuomo et al. 2022), and one specific to fluid mechanics applications (Cai et al. 2021).

We apply the PINN technique to both a synthetic dataset and experimental datasets and compare to the state-of-the-art Constrained Cost Minimization (CCM) technique (Agarwal et al. 2021). The synthetic datasets are from Direct Numerical Simulation of a turbulent channel flow at $\mathrm{Re}_\tau = 1000$, where the particle trajectories are either obtained directly from the simulation or via a synthetic tomography procedure (Schanz et al. 2016) designed to mimic error sources in experimental measurements. The experimental dataset consists of velocity measurements in the shear layer that develops behind a backward-facing step at $\mathrm{Re}_\tau = 800$. We compare errors and correlations in the streamwise velocity field, the pressure and their respective gradients, as well as calculate the temporally resolved energy spectra, and provide quantitative comparisons with CCM.

2 Physics-Informed Neural Networks

Physics-Informed Neural Networks (Weinan and Yu 2018; Raissi et al. 2020, 2019) are designed to describe a set of physical fields and make use of the partial differential equations that govern these fields to constrain and regularize the training process. In the present application the PINNs enforce the three-dimensional, incompressible Navier-Stokes equations, with the coordinates (x, y, z, t) as inputs and (u, v, w, p) as the outputs evaluated at the given coordinate. The architecture is a typical fully connected neural network with parameters $\boldsymbol{\theta}$ and where every hidden unit is passed through an activation function σ, as shown in the diagram in Fig. 1a. The PINN implementation used is available in Clark Di Leoni (2022). The specific scripts used to run the PINNs presented here are available in Clark Di Leoni et al. (2023a).

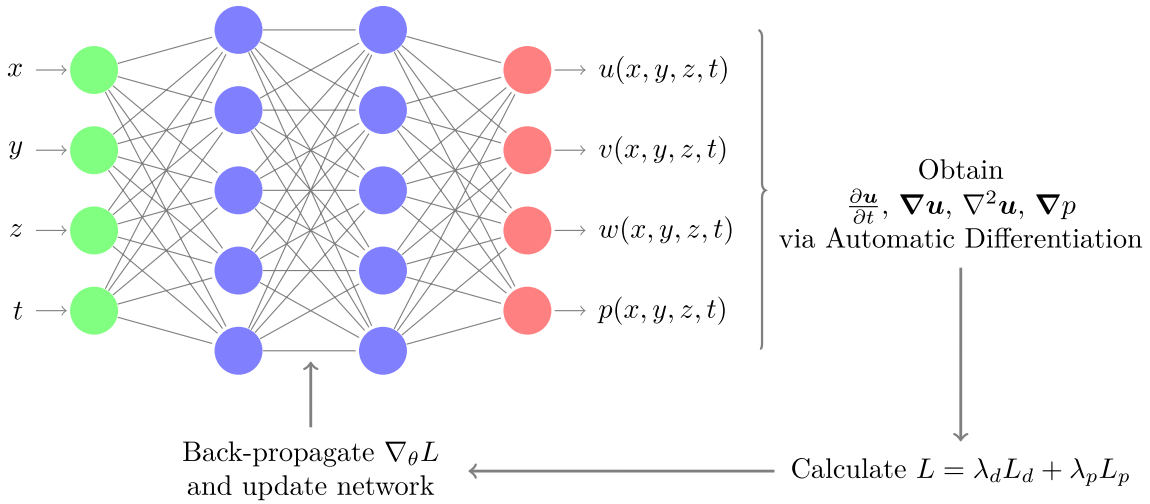

Fig. 1 Diagram of the PINN and its training procedure

The goal is to take a set of velocity-field measurements $\hat{\Omega}_d = \{x_j, y_j, z_j, t_j; \hat{u}_j, \hat{v}_j, \hat{w}_j\}_{i=1}^{N_d}$, and use the PINN to interpolate these data. The *Physics-Informed* property is achieved by applying automatic differentiation to calculate the derivatives of the outputs of the network with respect to its inputs and then evaluating the terms in the Navier–Stokes equations; the residual of the equations is included in the loss function of the network as a regularization term. The loss function is thus composed of two terms: The first compares the network predictions to the measurement data,

$$L_d = \frac{1}{N_d} \sum_{j=1}^{N_d} \left| \boldsymbol{u}_j - \hat{\boldsymbol{u}}_j \right|^2, \tag{1}$$

where $\Omega_d = \{x_j, y_j, z_j, t_j; u_j, v_j, w_j\}_{j=1}^{N_d}$ are the input–output pairs of the PINN and where the coordinates points (x_j, y_j, z_j, t_j) in Ω_d and $\hat{\Omega}_d$ coincide. The second loss function is associated with the physics, and is comprised of two contributions, one from the residual of the incompressibility condition,

$$L_i = \frac{1}{N_p} \sum_{j=1}^{N_p} \left| \boldsymbol{\nabla} \cdot \boldsymbol{u}_j \right|^2, \tag{2}$$

and the other from the residual of the momentum equations,

$$L_m = \frac{1}{N_p} \sum_{j=1}^{N_p} \left| \frac{\partial \boldsymbol{u}_j}{\partial t} + \boldsymbol{u}_j \cdot \boldsymbol{\nabla} \boldsymbol{u}_j + \frac{1}{\rho} \boldsymbol{\nabla} p_j - \nu \nabla^2 \boldsymbol{u}_j \right|^2. \tag{3}$$

Note that the set $\Omega_p = \{x_j, y_j, z_j, t_j; u_j, v_j, w_j, p_j\}_{j=1}^{N_p}$ does not necessarily have to coincide or overlap with Ω_d. Finally, the total loss function takes the form,

$$L = \underbrace{\lambda_d L_d}_{\text{data part}} + \underbrace{\lambda_i L_i + \lambda_m L_m}_{\text{physics part}}, \tag{4}$$

where λ_d, λ_i and λ_m are, in general, independent hyper-parameters used to balance each term of the loss function. Since the loss can be arbitrarily normalized, one of these hyper-parameters can be set to unity. For simplicity, we refer to the physics part as $L_p = \lambda_i L_i + \lambda_m L_m$. Further details on how to choose these hyper-parameters are given below.

It is important to remark that all the derivatives in L_p are calculated through automatic differentiation on the network. Automatic differentiation is the process by which the derivative of a function composed of combinations and concatenations of known elementary functions (the activation functions for the case of neural networks) is calculated using the chain rule. It is the same process used to calculate the gradients of the loss function with respect to the weights of the network when training a normal neural network, but in PINNs it is also applied to the inputs of the network. All deep learning frameworks, such as Pytorch and Tensorflow, provide functionality to perform automatic differentiation with respect to the input variables in a straightforward fashion.

2.1 Balancing the terms of the loss function

Including a regularization term in the loss function of a neural network introduces additional weights, or hyper-parameters, to be prescribed. Recent works have presented different strategies: some based on the analysis of the Hessian of the loss function (Wang et al. 2021), others based on Neural Tangent Kernel Theory (Wang et al. 2022), and others that incorporate the weighting hyperparameters into the trainable parameters (McClenny and Braga-Neto 2020; Xiang et al. 2021). In this work we use the first of these methods, but aided by our knowledge of the physical problem and its governing equations. The flows we present here have a mean component whose magnitude is on the order of U along direction $\hat{x}$, so if we recognize the streamwise advection term to be the leading term in the momentum equation, we obtain the following scalings for each term in the loss function,

$$L_d \sim U^2, \tag{5}$$

$$L_i \sim \left(\frac{\partial u}{\partial x} \right)^2, \tag{6}$$

$$L_m \sim U^2 \left(\frac{\partial u}{\partial x} \right)^2. \tag{7}$$

In highly anisotropic flows it is reasonable to separate the data and momentum terms into the three different components. While we do not separate them since in Eq. 3 the norm includes all components, we do apply input and output normalization layers separately for each velocity component (introduced in the following section) to alleviate the effects of anisotropy.

Since in all the flows that we present $U \approx 1$, L_i and L_m are of the same order, we therefore take $\lambda_i = \lambda_m = 1$ and balance the loss function by varying only λ_d, which according to our analysis should be of the order,

$$\lambda_d \sim \frac{L_p}{L_d} \sim \left(\frac{\partial u}{\partial x} \right)^2, \tag{8}$$

in order for every term of the loss function to be balanced. This gradient can be estimated very approximately using Kolmogorov's turbulence theory. At scale ℓ the gradient magnitude is on the order

$$\frac{\partial u}{\partial x} \sim \epsilon^{1/3} \ell^{-2/3}, \tag{9}$$

where ϵ is the rate of energy dissipation. This estimate takes its largest value at the smallest scale in the flow, i.e., the Kolmogorov scale $\eta \sim (\nu^3/\epsilon)^{1/4}$. Thus, we obtain

$$\frac{\partial u}{\partial x} \sim \epsilon^{1/2}\nu^{-1/2}. \tag{10}$$

Since the convergence of the training procedure is ultimately dictated by the respective gradients of different terms in the loss function (Wang et al. 2021, 2022), we cannot set the values of the weighting hyper-parameters solely based on the above scaling alone. For this reason, we factorize λ_d into $\lambda_d = \lambda_d^0 \hat{\lambda}_d$, where λ_d^0 is a fixed part whose value is inspired by Eqs. (8) and (10), and $\hat{\lambda}_d$ is a varying part set by the algorithm presented in Wang et al. (2021), which updates $\hat{\lambda}_d$ at epoch n according to,

$$\hat{\lambda}_d^n = (1-\alpha)\hat{\lambda}_d^{n-1} + \alpha \frac{\langle|\nabla_\theta L_p|\rangle_\theta}{\langle|\lambda_d^0 \nabla_\theta L_d|\rangle_\theta}, \tag{11}$$

where the superscripts n and $n-1$ denote the value of $\hat{\lambda}_d$ at the nth and $(n-1)$th iteration, α is a new free hyperparameter usually set to 0.1, and $\langle|\nabla_\theta \cdot|\rangle_\theta$ is the mean of the absolute value of the gradients of each quantity with respect to the network parameters θ.

2.2 Normalization layers

Due to the nature of activation functions and in order to mitigate the differences in the orders of magnitudes of various inputs and also outputs, we added an input and an output normalization layer (LeCun et al. 2012). Both layers aim to ensure that the values for each input or output channel are within the range $(-1, 1)$ and are nearly symmetrized.

For the input part we use min-max normalization, which takes the minimum and maximum value for each coordinate and rescales the coordinate between -1 and 1, namely (for coordinate x for example)

$$x \leftarrow 2\frac{x - x_{\min}}{x_{\max} - x_{\min}} - 1, \tag{12}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values of x in our domain, respectively. As the flow domain is known, all the respective maximums and minimums are also known.

For the output part we use z-score normalization, which centers and standardizes each output, for example

$$u \leftarrow \sigma_u u + \mu_u, \tag{13}$$

where μ_u and σ_u are the mean and standard deviation of u, respectively, and the u on the RHS is the output of the last trainable layer. As we use velocity measurements to train the networks, the values of μ_u, μ_v, μ_w, σ_u, σ_v, and σ_w can be easily estimated. Since no data is assumed to be known regarding pressure, the values of μ_p and σ_p are left as free hyperparameters of the network. All cases were trained on NVIDIA Tesla K80 GPUs.

3 Methods and dataset description

We analyze three different cases: the first two use data from a direct numerical simulation (DNS) of a turbulent channel flow at $Re_\tau = 1000$ available through the Johns Hopkins Turbulence Database (JHTDB) (Li et al. 2008; Graham et al. 2016; [1]), and the third case studies experimental measurements of a turbulent shear layer (Agarwal et al. 2021). While the sources and acquisition methods differ between cases, in all tests the data are particle tracks, as shown in Fig. 2. We define x, y, z as the streamwise, vertical and spanwise directions, respectively. All volumes expressed below are in $\{x, y, z\}$ order. Below we discuss each dataset and detail the PINN hyperparameters used for each case.

3.1 Case 1: synthetic particle tracks in DNS of turbulent channel flow

The goal of the first case is to study the effects of particle spacing and noise level on the accuracy of flow reconstruction. All quantities are given in terms of the friction velocity u_τ and the viscous length scale $\delta_\nu = \nu/u_\tau$ where ν is the fluid's kinematic viscosity. To generate the dataset we randomly seeded a volume of size $235\delta_\nu \times 215\delta_\nu \times 185\delta_\nu$, in the streamwise, vertical and spanwise directions, respectively, approximately 10^{-5} of the total DNS volume, located at the bottom wall of the channel. The interrogation volume has dimensions $195 \times 195 \times 145$ wall units and is slightly smaller than the seeded one so as to avoid edge effects. The DNS has a constant horizontal resolution of $12.3\delta_\nu$ in the streamwise direction and $6.1\delta_\nu$ in the spanwise direction, while its vertical resolution varies along the height of the channel, it is close to $0.02\delta_\nu$ at the bottom of the test volume and close to $3.7\delta_\nu$ at the top. The results are available every 5 frames of the original DNS times steps. The JHTDB uses fourth order Lagrange polynomials in space and a piecewise-cubic interpolation scheme in time to interpolate data from grid locations to the spatio-temporal position of interest. The seeded particles are treated as Lagrangian tracers and the synthetic tracks are thus generated by tracking the particles' evolution in the volume. Several sets of tracks were generated, each composed of nine snapshots (exposures), centered around a target time and with timestep $dt = 0.00325$ (or 2.5 times the DNS timestep). The velocity and acceleration of

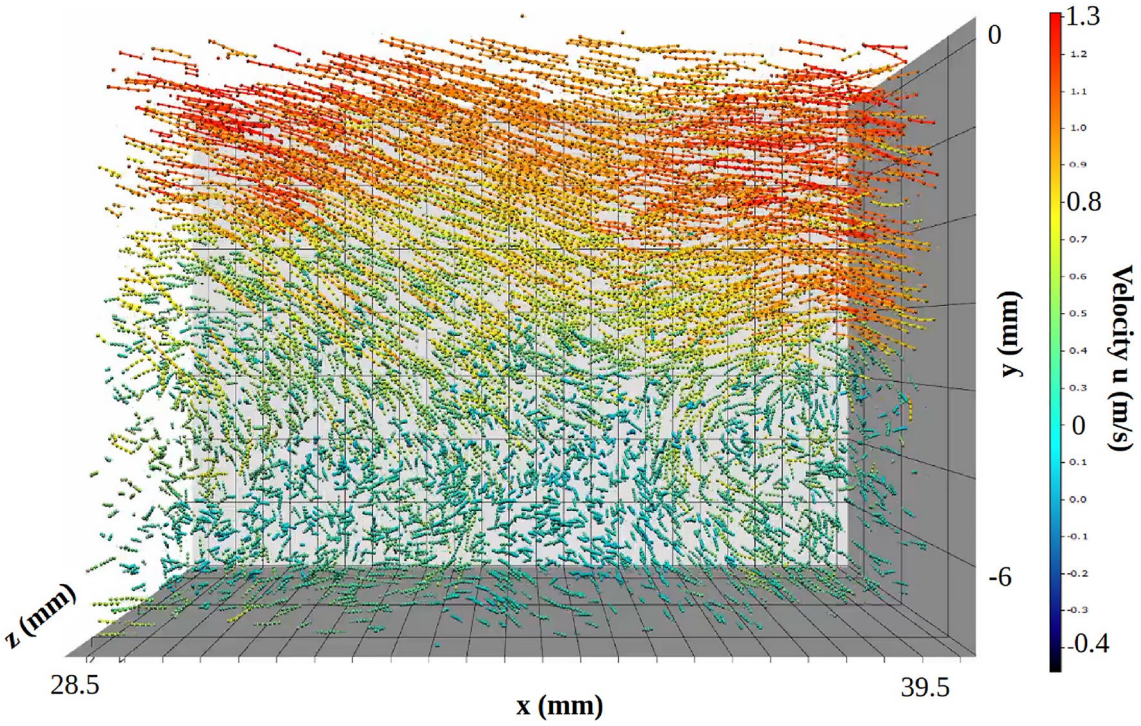

Fig. 2 Particle tracks obtained from Case 3

each particle were then calculated by fitting a second order polynomial around the target time. All nine snapshots were used for the reconstruction using PINNs, while only three snapshots (the target time plus and minus one time step) were used when using CCM. Noise was introduced to the measurements by adding a random Gaussian fluctuation to each particle position along the tracks. The flow reconstruction was performed in a total 39 different and independent target times at various values of particle spacing and noise levels. The particle spacing was controlled by varying the number of particles in the flow and it ranged from $3.4\delta_v$ (when using 60, 000 particles) to $8.4\delta_v$ (when using 4, 000 particles). The noise levels ranged from zero (no noise) to 0.8 pixels (equivalent to $0.32\delta_v$).

The PINNs used to reconstruct this case were eight layers deep and 200 units wide. Their initial learning rate was set to 10^{-3} and then followed an exponential decay schedule with rate 0.9 and 100 epochs characteristic time. The pressure was scaled with $\sigma_p = 0.1$ and λ_d^0 was set to unity. The sets of points used to enforce the data and physics parts of the loss function, Ω_d and Ω_p, respectively, coincided. The networks were trained for 750 epochs. The scripts used to run these PINNs, as well as those used to download the JHTDB data, are available in Clark Di Leoni et al. (2023a). The particle trajectories used to train the PINNs in this case are available in Clark Di Leoni et al. (2023b).

3.2 Case 2: synthetic tomographic images from DNS of channel flow

In this case we use the same flow configuration and fields as in the previous one, i.e., the turbulent channel flow data from the JHTDB. We generate synthetic tomographic images of the particle fields using the EUROPIV Synthetic Image Generator (Lecordier and Westerweel 2004) and apply the Shake-the-box algorithm to reconstruct the tracks (Schanz et al. 2016) instead of using the particle positions directly. This procedure is a surrogate for all the imaging, calibration and reconstruction errors encountered experimentally. The synthetic tomography is generated by projecting the particle positions onto four views, with all views aligned in the wall-normal direction and forming angles of $\pm 15°$ and $\pm 30°$ in the spanwise direction. The images are then processed with LaVision's DaVis 10 software to retrieve particle positions. Further details can be found in Agarwal et al. (2021). Contrary to the previous case, we now reconstruct an extensive flow history consisting of 2,000 timesteps with $dt = 0.00325$. The sample volume has dimensions $95\delta_v \times 95\delta_v \times 45\delta_v$ and the mean particle spacing is $5\delta_v$, which is maintained constant by re-feeding particles into the flow over a test volume larger than the sample volume. The velocity (and accelerations for the CCM) were calculated by fitting a second order polynomial on 17 exposures.

Due to the size of the dataset, two PINNs were used to reconstruct the full 2, 000 frame long time window, each tasked with 1, 000 non-overlapping frames. As before, the PINNs used were eight layers deep and 200 units wide. Their initial learning rate was set to 10^{-3} and then followed an exponential decay schedule with rate 0.9 and 100 epochs characteristic time. The pressure was scaled with $\sigma_p = 0.1$ and λ_d^0 was set to unity. The set of points used to enforce the physics, Ω_p contained all of the points in Ω_d as well as N_d additional points randomly selected throughout the spatio-temporal domain. The networks were trained for 1,000 epochs. As a way of showing the effects an imbalanced can have on the results, this case was also trained with a fixed $\hat{\lambda}_d = 1$. Unless noted, all results shown correspond to the variable $\hat{\lambda}_d$ setup. The scripts used to run these PINNs, as well as those used to download the JHTDB data, are available in Clark Di Leoni et al. (2023a). The particle trajectories used to train the PINNs in this case are available in Clark Di Leoni et al. (2023c).

3.3 Case 3: experimental data measured in a turbulent shear layer

For the third case we use experimental data from a flow behind a step in a small water tunnel. Figure 3 shows the configuration. The Reynolds number of the boundary layer upstream of separation is $Re_\tau = 800$ and the flow presents strong vortical structures.

The test section is $405 \times 63 \times 51$ mm^3 and the step height h is equal to 10 mm. Four Pco.dimax cameras were used to record the images of size 624×380 pixel at 14925 Hz over a field of view of $12.5 \times 7.5 \times 4.5$ mm^3 located in a region behind the step. The free-stream velocity is 5.3 m/s, which leads to high acquisition frequencies and low spatial resolution of images. As a result relatively sparse particle spacing of ~ 300 µ m are obtained. The time window chosen for reconstruction is 900 frames long.

Similar to the synthetic camera settings adopted in Case 2, the four cameras were at $\pm 15°$ and $\pm 40°$ angles in the spanwise direction. A Photonics DM60-527 Nd:YLF laser was used to illuminate the flow field. The particles were 13 µm silver-coated hollow glass spheres. As in Case 2, the tomographic PTV data were processed with the Shake-the-Box algorithm from DaVis 10. For more details on the experimental setup see Gopalan and Katz (2000), and for more details on these particular measurements see Agarwal et al. (2021).

Due to the size of the dataset, nine PINNs were used to reconstruct the full 900 frame long time window, each tasked with 100 non-overlapping frames. The PINNs used were six layers deep and 350 units wide. Their initial learning rate was set to 10^{-3} and then followed an exponential decay schedule with rate 0.9 and 50 epochs characteristic time. The pressure was scaled with $\sigma_p = 1$ and λ_d^0 was set to 10^7. The set of points used to enforce the physics, Ω_p, contained all of the points in Ω_d plus N_d additional points randomly selected throughout the whole spatiotemporal domain. The networks were trained for 2,000 epochs.

4 Results

We now present results from the three test cases. To present quantitative comparisons between DNS and measured values for Cases 1 and 2 (where we have the "truth" from the DNS data), we use the root-mean-square error (RMSE) and correlation coefficients. The root-mean-square error has the following definition:

$$\epsilon_u = \frac{\sqrt{\langle (u - u^{DNS})^2 \rangle}}{u_\tau}, \qquad \epsilon_p = \frac{\sqrt{\langle (p - p^{DNS})^2 \rangle}}{u_\tau^2}, \tag{14}$$

where the averaging operation $\langle \cdot \rangle$ is performed over the entire spatiotemporal domain. In cases where the averaging

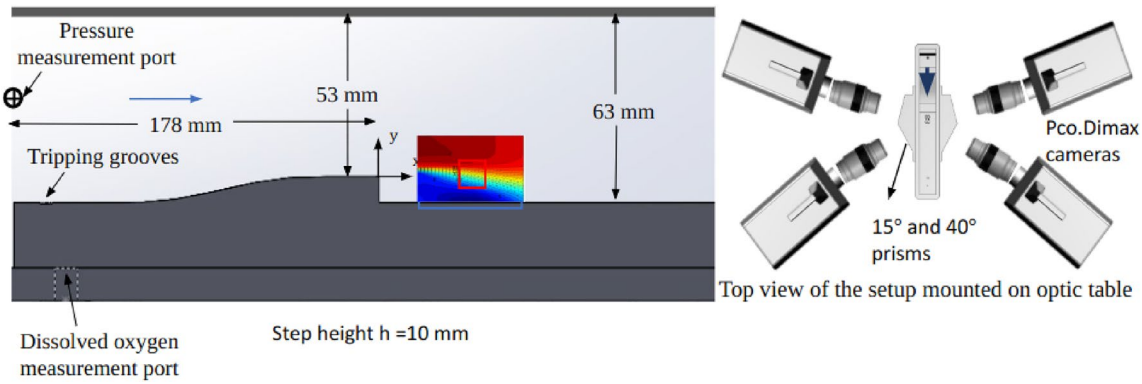

Fig. 3 Experimental setup used for Case 3. The area of interest is marked with the red square. The colored square shows a 2D PIV visualization of the streamwise velocity

operation is not performed over a particular dimension, this will be stated explicitly, for example $\epsilon_u(y^+)$ is the vertical profile of the RMSE of u which is not averaged in the vertical dimension. The correlation coefficients between DNS and measured data are defined by

$$\rho_u = \frac{\langle (u - \langle u \rangle)(u^{DNS} - \langle u^{DNS} \rangle) \rangle}{\langle (u - \langle u \rangle)^2 \rangle \langle (u^{DNS} - \langle u^{DNS} \rangle)^2 \rangle}, \tag{15}$$

and ρ_p follows the same definition as ρ_u.

4.1 Case 1: synthetic particle tracks in DNS of turbulent channel flow

We start by considering the results from Case 1. In Fig. 4 we show visualizations of the streamwise velocity field u and the pressure p of the true data, the CCM-reconstructed field and the PINN-reconstructed field at one particular instant. The visualizations show only half the volume in order to demonstrate the quality of the reconstruction within the bulk of the volume. Both CCM and PINNs are in good qualitative agreement with the true data, with PINNs producing slightly smoother fields than CCM. In Fig. 5 we show the evolution of the weighted data loss $\lambda_d L_d$ and the physics loss L_p during the training of the PINN. Both losses are minimized by the training procedure and the balancing term λ_d helps keep both terms of the same order.

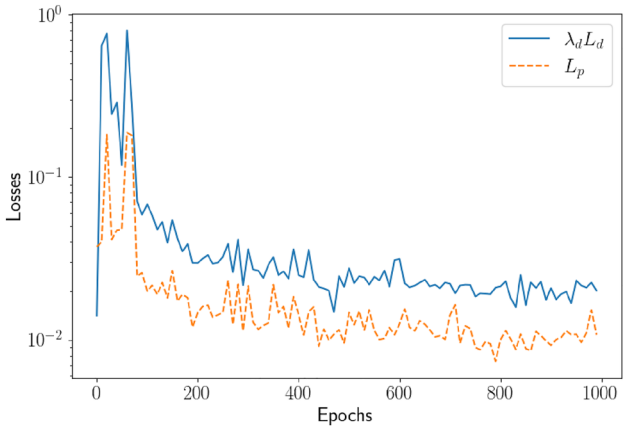

Fig. 5 Evolution of the data and physics losses during the training of the PINN that produced the results of Fig. 4

For a quantitative assessment, we examine the accuracy of the reconstruction as a function of the particle spacing. In Fig. 6 we show ϵ_u as function of the particle spacing in absence of any measurement noise, at three different heights, and in Fig. 7 we show the same for ϵ_p. Results are averaged over the 39 independently reconstructed snapshots, and all error bars are equal to the calculated standard deviations. Both CCM and PINNs techniques have commensurate success in reconstructing the true flow, with PINNs being slightly more robust to particle spacing for reconstructing u

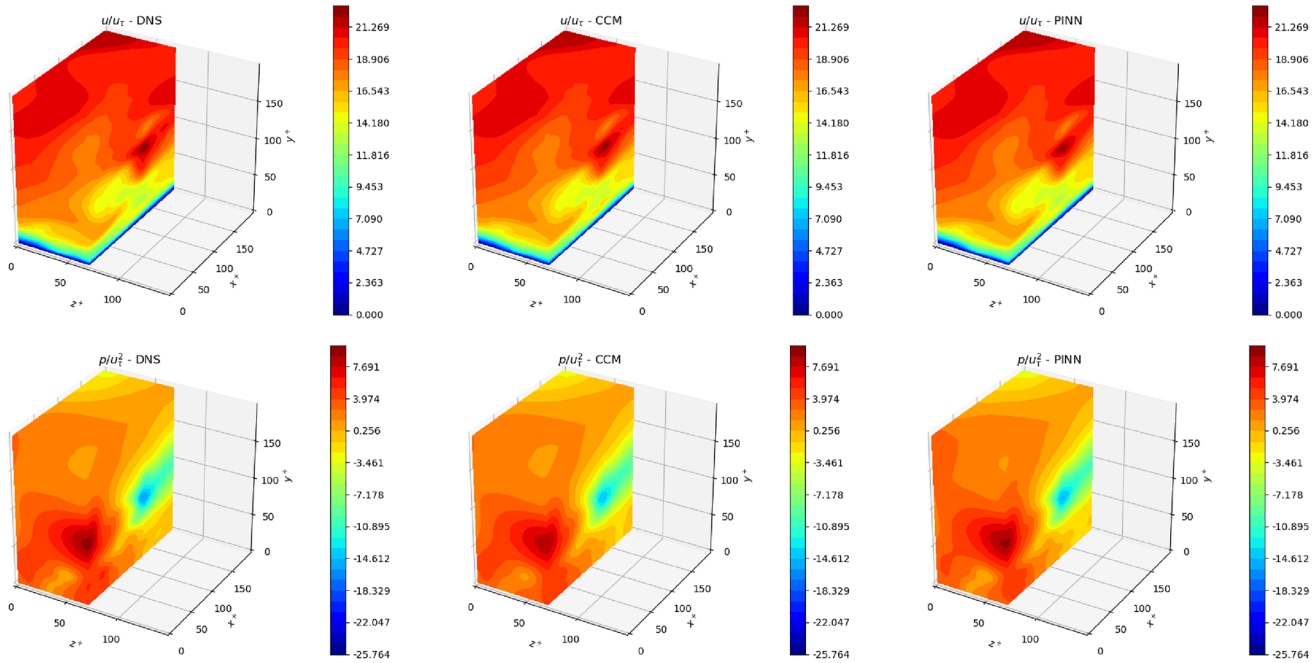

Fig. 4 Instantaneous visualization of the data and the CCM and PINN reconstruction for Case 1. Top row: streamwise velocity field u. Bottom row: pressure p. Only half of the reconstructed volume is shown

and CCM yielding slightly smaller errors for p near the top boundary (the omni-directional integration used by CCM is well-suited for boundaries since it iteratively solves for boundary pressures to match with interior material acceleration information).

In Figs. 8 and 9 we again show ϵ_u and ϵ_p, respectively, at different locations, but this time with an added noise of 0.4 pixels (equivalent to $0.16\delta_v$). Right away we can appreciate how PINNs are more robust than CCMs in noisy systems. To take this point further, in Figs. 10 and 11 we show ϵ_u and ϵ_p, respectively, at three different heights but this time at a different noise levels, all with constant particle spacing $r^+ = 5.4$. The errors in the reconstruction generated with PINNs increase more slowly with noise level than in the case of CCM, especially for pressure. This is because the data is projected onto the space of solutions of the PDE by the physics loss, which is an effective way of filtering errors (Wang and Zaki 2021).

4.2 Case 2: synthetic tomographic images from DNS of channel flow

We now turn to Case 2, where we analyze synthetic tomographic images generated using the turbulent channel flow data from the JHTDB. Contrary to Case 1 where we performed a reconstruction of an instantaneous field within a very narrow time window (nine snapshots for PINNs and three for CCM), we now reconstruct the time evolution of

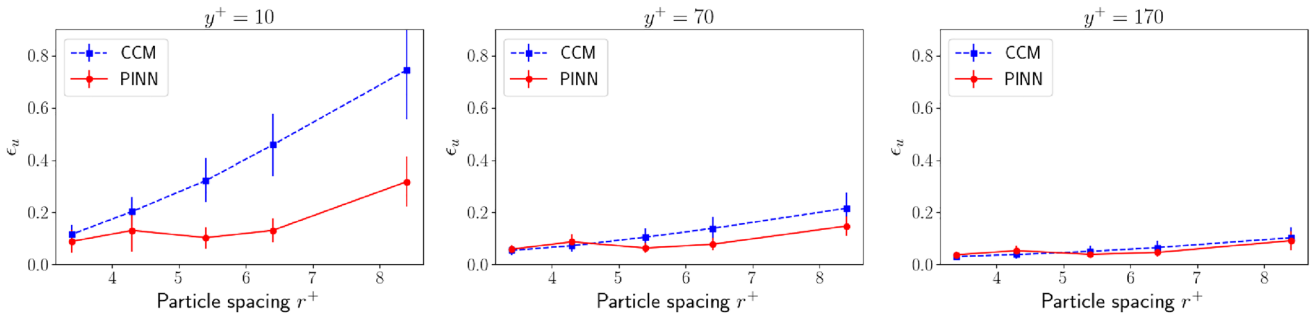

Fig. 6 RMSE ϵ_u in Case 1 as a function of particle spacing without any added noise at three different heights

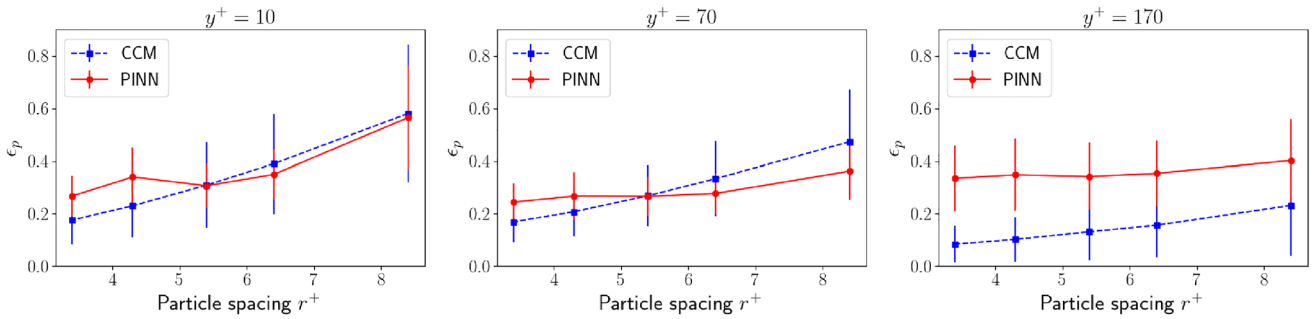

Fig. 7 RMSE ϵ_p in Case 1 as a function of particle spacing without any added noise at three different heights

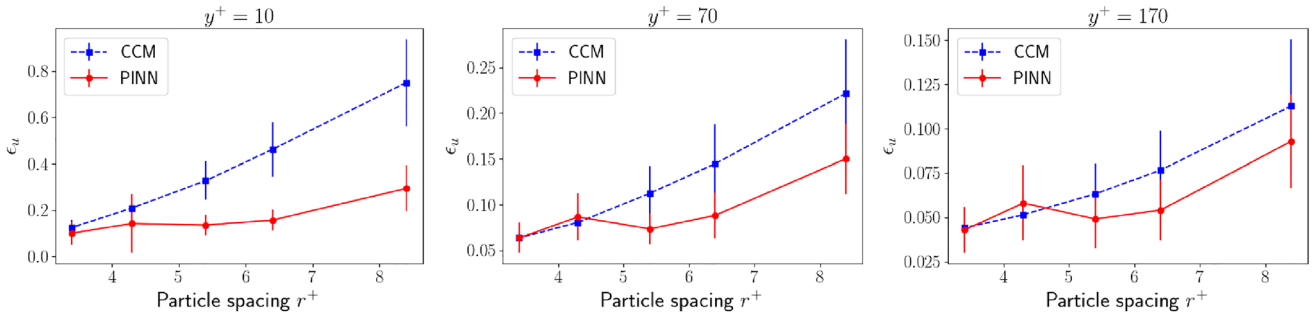

Fig. 8 RMSE ϵ_u in Case 1 as a function of particle spacing with 0.4 px noise added at three different heights

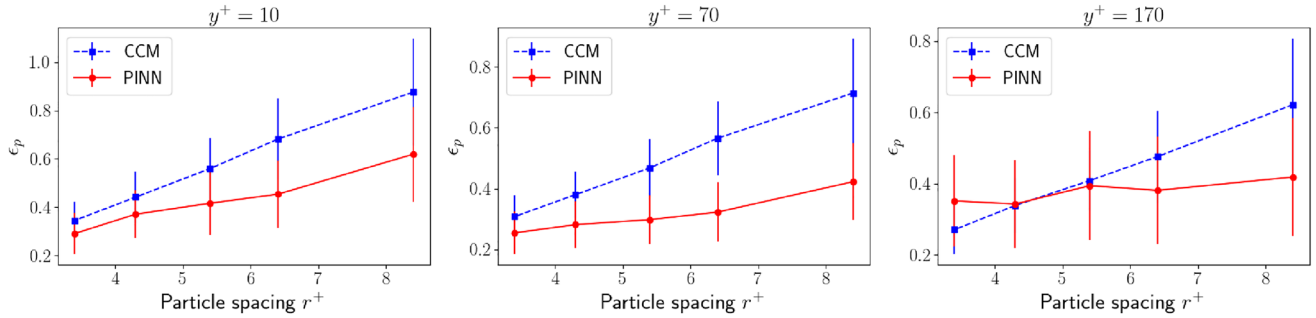

Fig. 9 RMSE ϵ_p in Case 1 as a function of particle spacing with 0.4 px noise added at three different heights

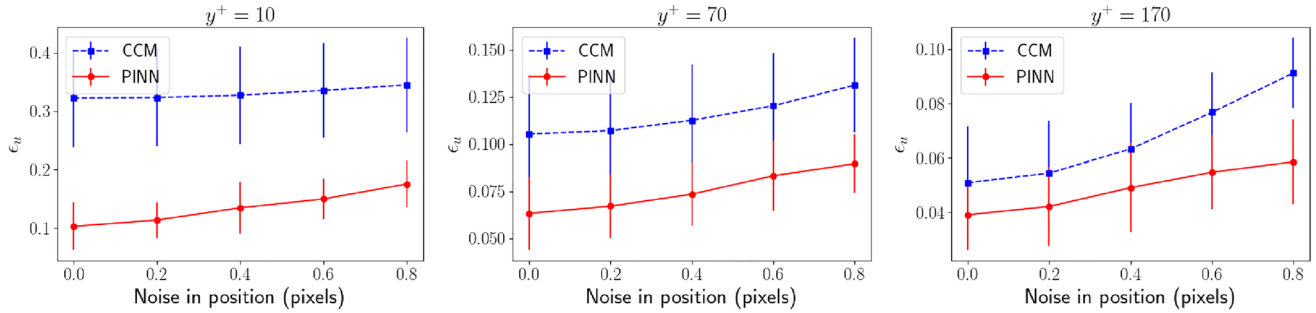

Fig. 10 RMSE ϵ_u in Case 1 as a function of noise added with a particle spacing of 5.4 δ_v at three different heights

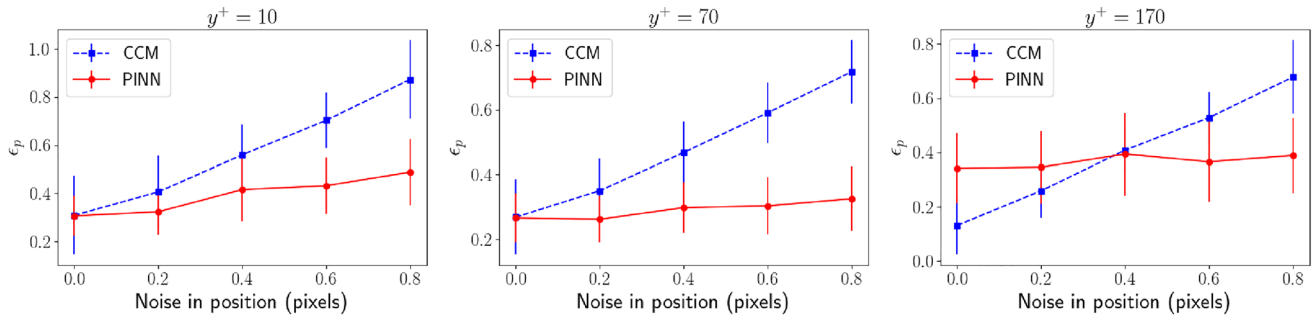

Fig. 11 RMSE ϵ_p in Case 1 as a function of noise added with a particle spacing of 5.4 δ_v at three different heights

the flow within the volume. In Fig. 12 we show visualizations of the streamwise velocity field u and the pressure p of the data, the CCM-reconstructed field and the PINN-reconstructed field at one particular instant. The visualizations show only half the volume in order to expose the interior of the field. Both CCM and PINNs are in good qualitative agreement with the true data and, similar to the results from Case 1, PINNs produce smoother fields. This quality is highlighted in Fig. 13, where we show profiles of instantaneous u and p along the streamwise direction x. Especially for u, the difference in the level of small-scale structure between both techniques is noticeable.

For a quantitative comparison, in Fig. 14 we show the RMSEs of the fields as a function of height above the wall, $\epsilon_u(y)$ and $\epsilon_p(y)$, while in Fig. 15 we show vertical profiles of RMSEs of the spanwise gradients of the fields, $\epsilon_{\partial_z u}$ and $\epsilon_{\partial_z p}$. The error in the reconstruction of u is similar for both techniques, and is on the order of 0.1 u_τ, except near the wall where the velocity should vanish. PINNs achieve lower errors in the reconstruction of the pressure, especially away from the wall, where the error for the PINNs is approximately half of that of he CCM. As expected from Fig. 13, the reconstruction errors of the spanwise gradients, of both u and p, are significantly lower for the PINNs.

In order to investigate the temporal behavior of the reconstruction, in Fig. 16 we show the correlation coefficients ρ_u and ρ_p between the true fields and the reconstructed ones, as function of time. The reconstructed velocity fields exhibit

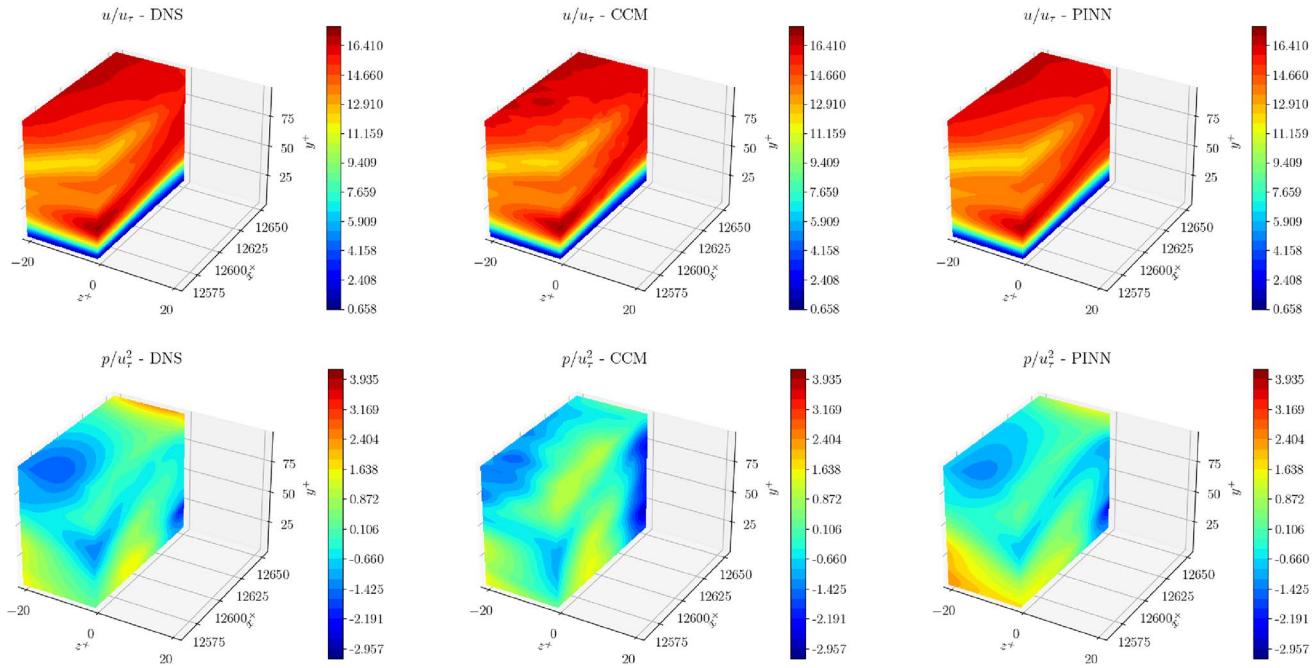

Fig. 12 Instantaneous visualization of the data and the CCM and PINN reconstruction for Case 2. Top row: streamwise velocity field u. Bottom row: pressure p. Only half of the reconstructed volume is shown

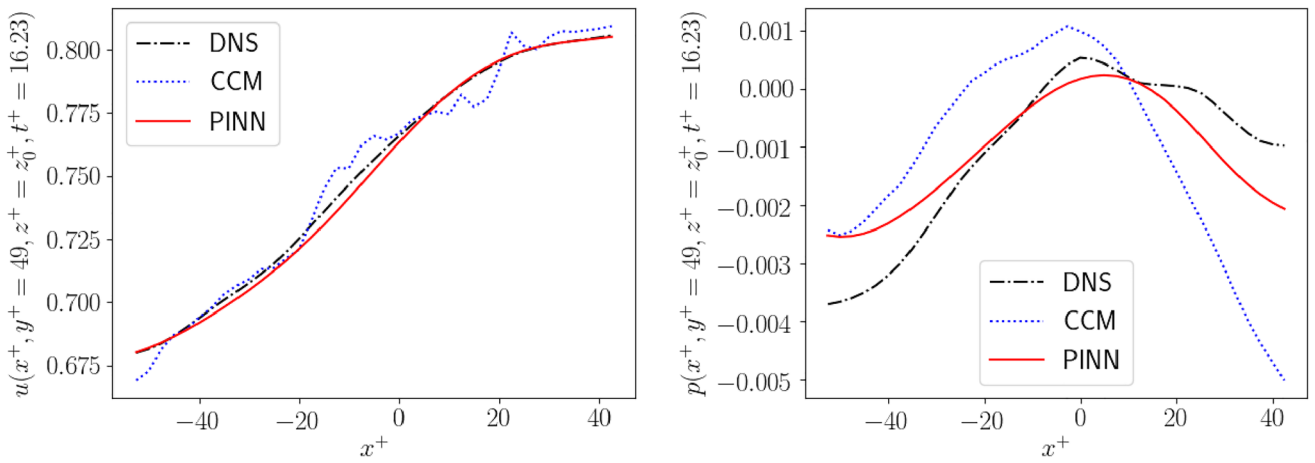

Fig. 13 Streamwise profiles extracted from Fig. 12

very high correlations with the true flow throughout the time window, while the reconstructed pressure fields show mostly high accuracy with some instantaneous reductions in the correlations. These instantaneous reductions in ρ_p are observed in both CCM and PINNs around the same instants, and the effect is more pronounced in CCM. In Fig. 17 we show the frequency spectra of the true and reconstructed velocity and pressure, evaluated at $y = 49\delta_v$. A Hanning window was applied to the time signals in order to calculate the spectra, which were then averaged in the horizontal directions. Three vertical lines are marked on the figure: The dashed line indicates the Nyquist frequency at the height where the spectra are evaluated; this frequency is associated with the timescale, $dx/U(y = 49\delta_v)$ where dx is the streamwise DNS grid size and $U(y = 49\delta_v)$ is the mean streamwise velocity at this particular height; The dotted line indicates the frequency associated with the field of view: $L/U(y = 49\delta_v)$, where L is the streamwise length of the field of view; The dash-dotted line indicates the frequency associated with the temporal resolutions of the JHTDB. Both the

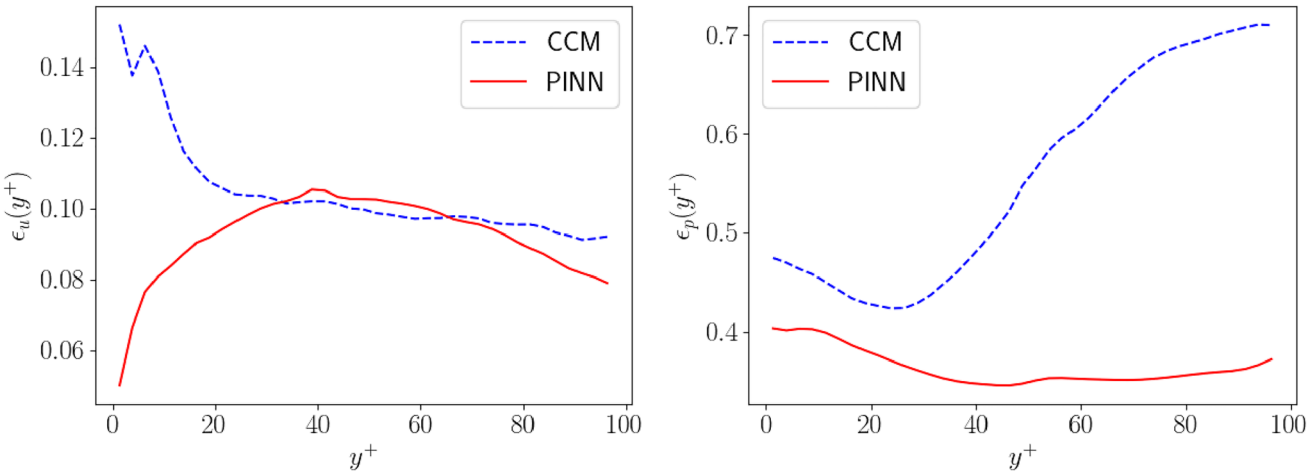

Fig. 14 RMSE of streamwise velocity (ϵ_u) and pressure (ϵ_p) as function of height above the wall, for Case 2

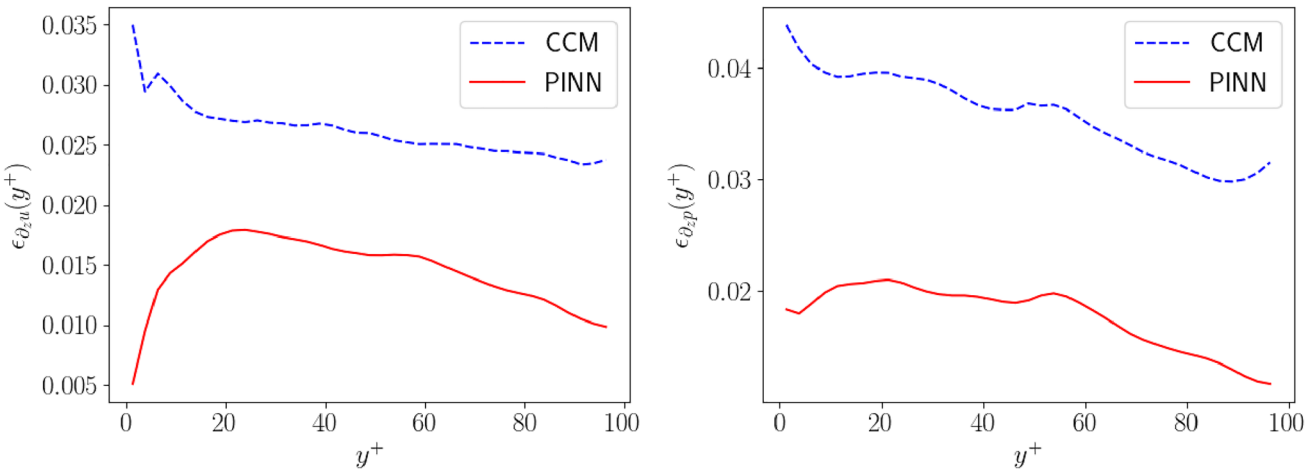

Fig. 15 RMSE of spanwise gradient of u-velocity ($\epsilon_{\partial_z u}$) and of pressure ($\epsilon_{\partial_z p}$) as function of height above the wall, for Case 2

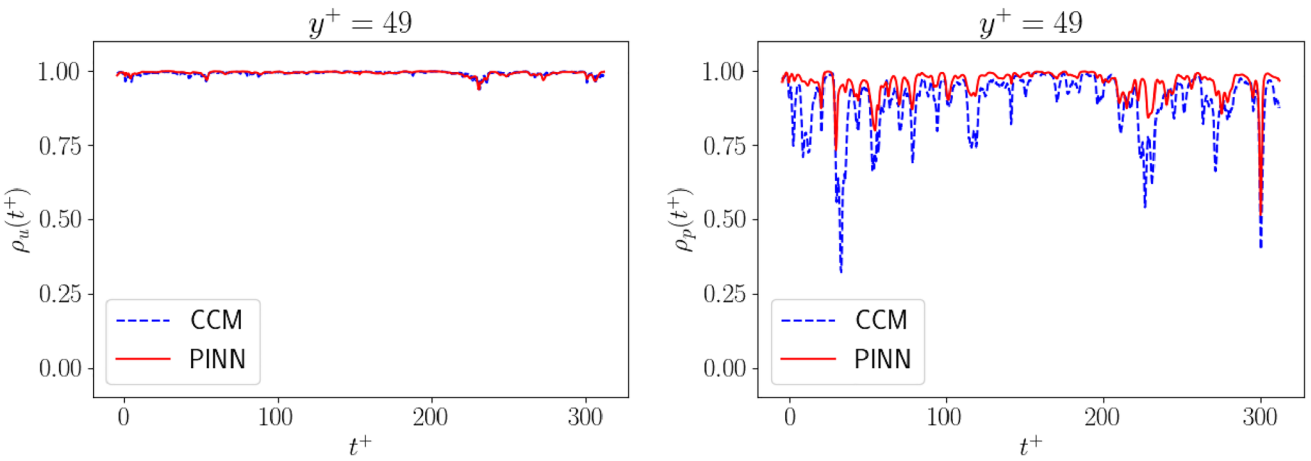

Fig. 16 Correlation coefficient between DNS and measured (PINNs and CCM) velocity and pressure, ρ_u and ρ_p, respectively, for Case 2 plotted as function of time

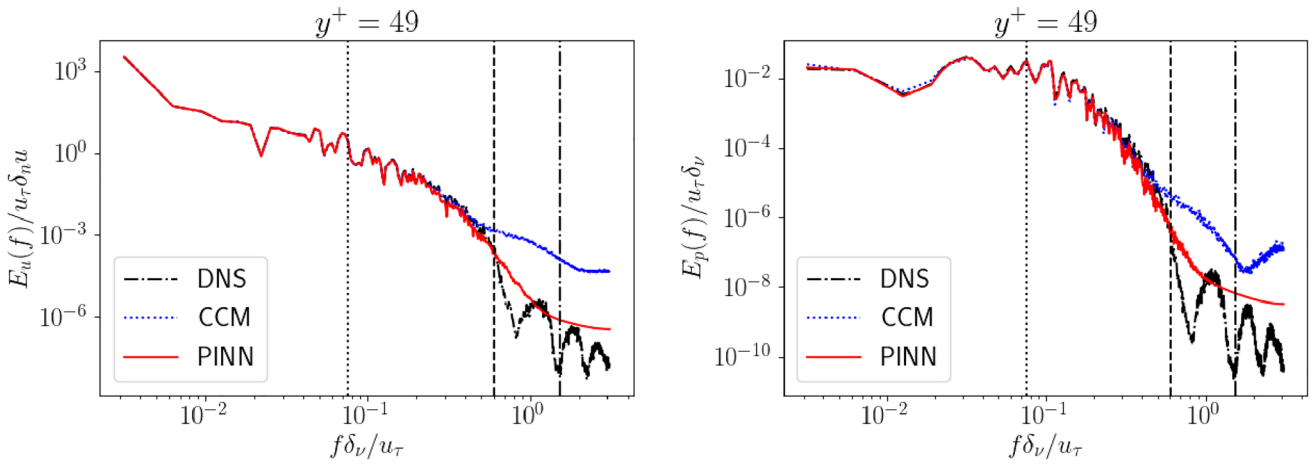

Fig. 17 Frequency spectra of velocity and pressure, $E_u(f)$ and $E_p(f)$, respectively, for Case 2 from DNS and obtained from the PINN and CCM methods

CCM and PINNs techniques perform very well within the range of timescales of interest. The lobes present in the DNS spectra beyond the Nyquist frequency of the time series are due to the temporal interpolation (based on Piecewise Cubic Hermite Interpolating Polynomial) performed by the JHTDB. While PINNs are able to smooth out those last frequencies, CCM retains a higher energy content, which is consistent with the observed small-scale fluctuations that can be appreciated in Figs. 12 and 13.

In order to demonstrate the potential impact of an imbalanced loss function, in Fig. 18 we compare the velocity and pressure spectra obtained from the appropriately weighted PINN loss (reproduced from Fig. 17) to a new PINN prediction with $\hat{\lambda}_d = 1$ throughout the training. In the former case, $\hat{\lambda}_d$ oscillated around 200 after an initial transient. Thus, the resulting λ_d in both cases is about two orders of magnitude different, with the fixed $\hat{\lambda}_d = 1$ case decreasing the influence of the data term in the loss function. The difference between the obtained velocity field spectra is more noticeable at higher frequencies, with the unbalanced case clearly damping the flow. On the other hand, the reconstructed pressure spectra show differences at all scales. The RMSEs obtained in the fixed $\hat{\lambda}_d$ (not shown) case are about 4 times as large as those obtained in the variable case. Thus, as the influence of the data term is decreased, the velocity field is not correctly learnt from the available data, this in turn impacts the reconstruction of the pressure field. Correctly setting λ_d (either manually or through an algorithm) is therefore crucial when setting up a PINN (Cuomo et al. 2022).

Finally, we report on the computational costs of each technique. As stated above, the PINNs were trained on NVIDIA Tesla K80 GPUs. Each PINN in this case used

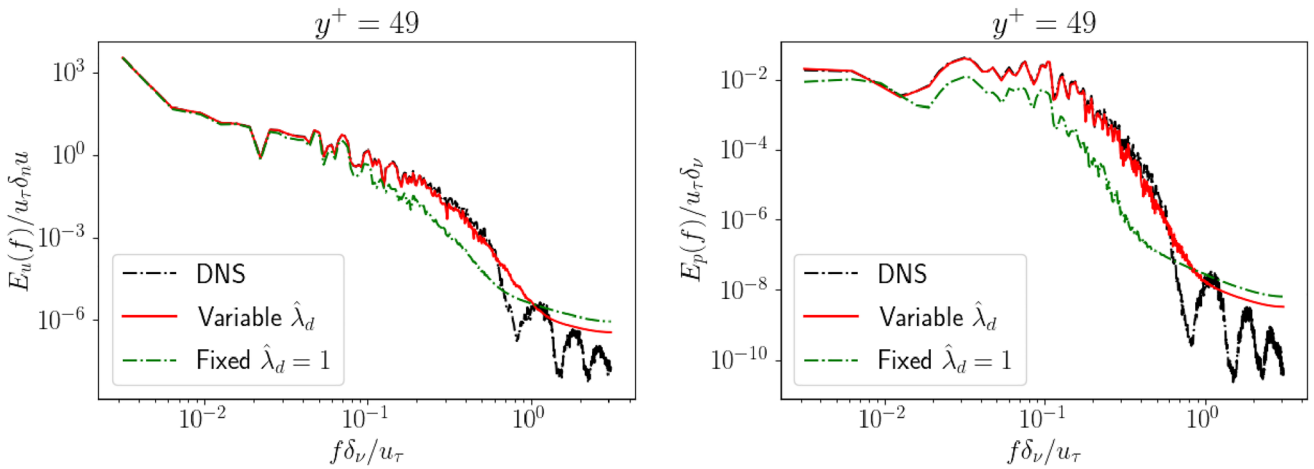

Fig. 18 Frequency spectra of velocity and pressure, $E_u(f)$ and $E_p(f)$, respectively, for Case 2 from DNS, obtained via the PINN with variable $\hat{\lambda}_d$ (same as Fig. 17) and via PINN with fixed $\hat{\lambda}_d = 1$

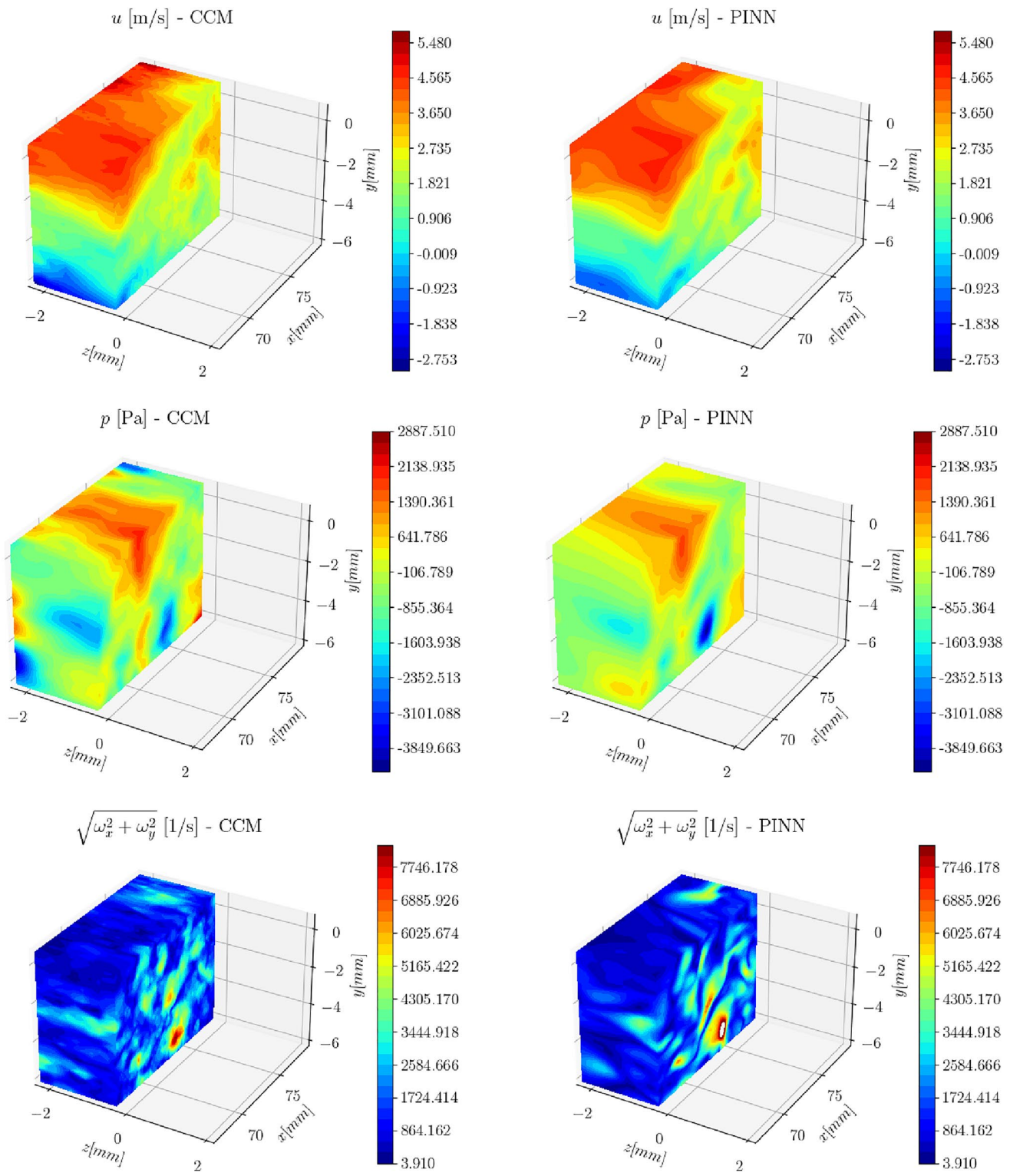

Fig. 19 Instantaneous visualization of the CCM and PINN reconstruction for Case 3. Top row: streamwise velocity field u. Bottom row: pressure p. Only half of the reconstructed volume is shown

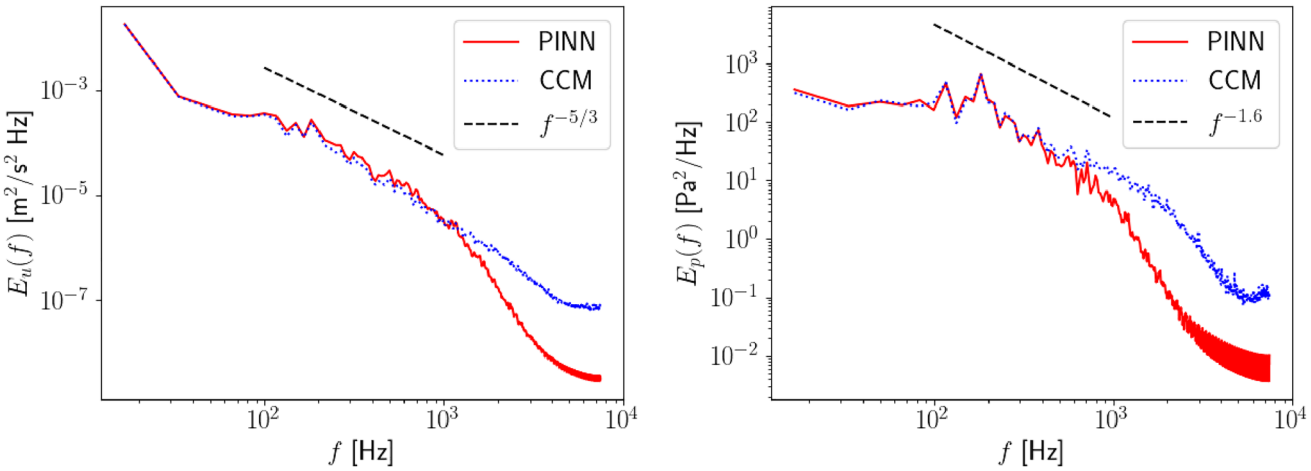

Fig. 20 Time-resolved spectra $E_u(f)$ and $E_p(f)$ for Case 3

data that spans 1, 000 snapshots, and required 40 h of training. On the other hand, CCM first performs a CPU-based SVD computation and then performs the omni-directional integration step in a GPU. Using a i7 CPU and a GTX 1080 Ti GPU, the procedure takes approximately 60 s per snapshot, totalling 16 h for 1000 snapshots. It is important to note that both can be parallelized and optimized for faster computations.

4.3 Case 3: experimental data in a turbulent shear layer

In this section we present the results of the reconstruction of the flow behind a step. In Fig. 19 we show instantaneous visualizations of u, p and $\sqrt{\omega_x^2 + \omega_y^2}$ from the CCM and PINN approaches. Again only half the volume is visualized in order to expose the interior of the volume. Similar to the previous cases, both techniques produce qualitatively similar results, with the locations and shapes of the main structures matching between the two, and with the regions of high vorticity coinciding with the low pressure areas. PINNs produce smoother fields as can be expected based on the earlier tests. In Fig. 20 we show the frequency spectra $E_u(f)$ and $E_p(f)$, with reference power-laws $f^{-5/3}$ and $f^{-1.6}$, respectively, as per the literature (Tsuji et al. 2007). The reconstructed spectra match over most frequencies of interest, diverging only at high frequencies as seen in Fig. 17. The spectra further confirm the trend that PINNs generate smoother results compared to CCM.

5 Conclusions

Machine learning has achieved significant advancements in the past years. In the context of fluid mechanics, physics-informed neural networks (PINNs) have proved very successful at tackling inverse problems. In this work we assess PINNs' capabilities to reconstruct the velocity and pressure fields of a flow from particle-tracks measurements. We test the approach in three cases, two based on synthetic data from Direct Numerical Simulations (DNS) and one based on experimental measurements, all of them at moderate Reynolds number and compare the PINN results to those obtained by the state-of-the-art Constrained Cost Minimization (CCM) method. PINNs are able to successfully reconstruct the velocity and pressure fields in all cases, achieving errors equal to or smaller than those of CCM. PINNs can reproduce the energy spectra of the flow accurately and are also shown to be robust against an increase in both the noise level and sparsity (particle spacing) in the data. Compared to CCM, PINNs produce smoother fields, devoid of small scale noise and thus leading to more reliable predictions of the gradients. PINNs enforce the Navier–Stokes equations in a weak sense (L2 norm over space and time); CCM enforces the incompressiblity condition on the velocity fields and the curl-free condition on the material acceleration, although in a strong sense. These two approaches can be contrasted to 4DVar which enforces the full Navier-Stokes equations in a strong sense (Du et al. 2023).

We emphasized a training methodology that can be generalized, characterized by the use of normalization layers and of physical estimates for the weighting parameters to

balance the equations. This methodology can be easily applied to other flows or problems, with the proper care and user expertise any reconstruction task requires. The PINN results are robust under small changes in hyperparameters, but an extensive exploration of the effects of changing hyperparameters is left for future work. In Cases 2 and 3 the reconstructed domains were split along the temporal dimension in order to ease training and avoid having to use overly large networks. Such limitations can be mitigated with the use of extended domain PINNs which incorporate domain splitting and training parallelization into their architecture (Jagtap and Karniadakis 2020).

Acknowledgements The authors acknowledge financial support from the Defense Advance Research Projects Agency (AIRA HR00111990025, CompMods HR00112090062), and from the Office of Naval Research (N00014-20-1-2715, N00014-21-1-2375). The authors thank George Em Karniadakis for useful discussions.

Author Contributions PC and KA wrote the main manuscript text. PC performed the neural network training and analysis. KA and JK generated the datasets. All authors contributed to the development of the technique, the analysis of the results and reviewed the manuscript.

Funding The authors acknowledge financial support from the Defense Advance Research Projects Agency (AIRA HR00111990025, CompMods HR00112090062), and from the Office of Naval Research (N00014-20-1-2715, N00014-21-1-2375).

Data availability The PINN implementation that was used in this work is available in Clark Di Leoni (2022). The data used in the synthetic cases is freely available at the Johns Hopkins Turbulence Database (Li et al. 2008; Graham et al. 2016; [1]). The specific scripts used to download the data and run the PINNs are available in Clark Di Leoni et al. (2023a). Processed particle trajectories used in Cases 1 and 2 are availabe in Clark Di Leoni et al. (2023b) and Clark Di Leoni et al. (2023c), respectively. Other codes and data can be made available upon request.

Declarations

Conflict of interest Not applicable.

References

Agarwal K, Ram O, Wang J, Lu Y, Katz J (2021) Reconstructing velocity and pressure from noisy sparse particle tracks using constrained cost minimization. Exp Fluids 62(4):1–20. https://doi.org/10.1007/s00348-021-03172-0

Angriman S, Cobelli PJ, Bourgoin M, Huisman SG, Volk R, Mininni PD (2021) Broken mirror symmetry of tracer's trajectories in turbulence. Phys Rev Lett 127(25):254502. https://doi.org/10.1103/PhysRevLett.127.254502

Baur T, Kongeter J (1999) PIV with high temporal resolution for the determination of local pressure reductions from coherent turbulence phenomena. In: Proceedings of 3rd International Workshop on PIV-Santa Barbara pp. 101–106

Buchta DA, Laurence SJ, Zaki TA (2022) Assimilation of wall-pressure measurements in high-speed flow over a cone. J Fluid Mech 947:R2

Buchta DA, Zaki TA (2021) Observation-infused simulations of high-speed boundary-layer transition. J Fluid Mech 916:A44. https://doi.org/10.1017/jfm.2021.172

Buzzicotti M, Bonaccorso F, Leoni P, Biferale L (2021) Reconstruction of turbulent data with deep generative models for semantic inpainting from TURB-Rot database. Phys Rev Fluids 6(5):050503. https://doi.org/10.1103/PhysRevFluids.6.050503

Cai S, Liang J, Gao Q, Xu C, Wei R (2020) Particle image velocimetry based on a deep learning motion estimator. IEEE Trans Instrum Meas 69(6):3538–3554. https://doi.org/10.1109/TIM.2019.2932649

Cai S, Mao Z, Wang Z, Yin M, Karniadakis GE (2021) Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta Mech Sinica 37(12):1727–1738. https://doi.org/10.1007/s10409-021-01148-1

Cai S, Wang Z, Fuest F, Jeon YJ, Gray C, Karniadakis GE (2021) Flow over an espresso cup: inferring 3-D velocity and pressure fields from tomographic background oriented Schlieren via physics-informed neural networks. J Fluid Mech. https://doi.org/10.1017/jfm.2021.135

Cai S, Wang Z, Lu L, Zaki TA, Karniadakis GE (2021) DeepM &Mnet: inferring the electroconvection multiphysics fields based on operator approximation by neural networks. J Comput Phys 436:110296. https://doi.org/10.1016/j.jcp.2021.110296

Callaham JL, Maeda K, Brunton SL (2019) Robust flow reconstruction from limited measurements via sparse representation. Phys Rev Fluids 4(10):103907. https://doi.org/10.1103/PhysRevFluids.4.103907

Clark Di Leoni P (2022) PINNs. https://github.com/PatricioClark/PINNs

Clark Di Leoni P, Agarwal K, Zaki TA, Meneveau C, Katz J (2023) Code repository. https://github.com/PatricioClark/ptv_pinns_paper

Clark Di Leoni P, Agarwal K, Zaki TA, Meneveau C, Katz J (2023) Dataset for case 1. https://doi.org/10.5281/zenodo.7688953

Clark Di Leoni P, Agarwal K, Zaki TA, Meneveau C, Katz J (2023) Dataset for case 2. https://doi.org/10.5281/zenodo.7688824

Clark Di Leoni P, Lu L, Meneveau C, Karniadakis GE, Zaki TA (2023) Neural operator prediction of linear instability waves in high-speed boundary layers. J Comput Phys 474:111793. https://doi.org/10.1016/j.jcp.2022.111793

Clark Di Leoni P, Mazzino A, Biferale L (2018) Inferring flow parameters and turbulent configuration with physics-informed data assimilation and spectral nudging. Phys Rev Fluids 3(10):104604. https://doi.org/10.1103/PhysRevFluids.3.104604

Clark Di Leoni P, Mazzino A, Biferale L (2020) Synchronization to big data: nudging the Navier-Stokes equations for data assimilation of turbulent flows. Phys Rev X 10(1):011023. https://doi.org/10.1103/PhysRevX.10.011023

Cuomo S, Di Cola VS, Giampaolo F, Rozza G, Raissi M, Piccialli F (2022) Scientific machine learning through physics-informed neural networks: where we are and what's next. J Sci Comput 92(3):88. https://doi.org/10.1007/s10915-022-01939-z

Dabiri D, Pecora C (2019) Particle Tracking Velocimetry. Institute of Physics Publishing. WGnLvAEACAAJ. Google-Books-ID

Dabiri JO, Bose S, Gemmell BJ, Colin SP, Costello JH (2014) An algorithm to estimate unsteady and quasi-steady pressure fields from velocity field measurements. J Exp Biol 217(3):331–336. https://doi.org/10.1242/jeb.092767

Du Y, Wang M, Zaki TA (2023) State estimation in minimal turbulent channel flow: a comparative study of 4DVar and PINN. Int J Heat Fluid Flow 99:109073

Du Y, Zaki TA (2021) Evolutional deep neural network. Phys Rev E 104:045303. https://doi.org/10.1103/PhysRevE.104.045303

Fukami K, Fukagata K, Taira K (2019) Super-resolution reconstruction of turbulent flows with machine learning. J Fluid Mech 870:106–120. https://doi.org/10.1017/jfm.2019.238

Ghaemi S, Ragni D, Scarano F (2012) PIV-based pressure fluctuations in the turbulent boundary layer. Exp Fluids 53(6):1823–1840. https://doi.org/10.1007/s00348-012-1391-4

Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press. Google-Books-ID: Np9SDQAAQBAJ

Gopalan S, Katz J (2000) Flow structure and modeling issues in the closure region of attached cavitation. Phys Fluids 12(4):895–911. https://doi.org/10.1063/1.870344

Graham J, Kanov K, Yang X, Lee M, Malaya N, Lalescu C, Burns R, Eyink G, Szalay A, Moser R et al (2016) A web services accessible database of turbulent channel flow and its use for testing a new integral wall model for LES. J Turbulence 17(2):181–215

Haller G (2015) Lagrangian coherent structures. Ann Rev Fluid Mech 47(1):137–162. https://doi.org/10.1146/annurev-fluid-010313-141322

Hasanuzzaman G, Eivazi H, Merbold S, Egbers C, Vinuesa R (2022) Enhancement of PIV measurements via physics-informed neural networks. Meas Sci Technol. https://doi.org/10.1088/1361-6501/aca9eb

Jagtap A, Karniadakis G (2020) Extended physics-informed neural networks (xpinns): a generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Commun Comput Phys 28(5):2002–2041. https://doi.org/10.4208/cicp.OA-2020-0164

Jensen A, Pedersen GK (2004) Optimization of acceleration measurements using PIV. Meas Sci Technol 15(11):2275–2283. https://doi.org/10.1088/0957-0233/15/11/013

Johns Hopkins turbulence database: Turbulent channel flow. https://doi.org/10.7281/T10K26QW. http://turbulence.pha.jhu.edu/Channel_Flow.aspx

de Kat R, Ganapathisubramani B (2012) Pressure from particle image velocimetry for convective flows: a Taylor's hypothesis approach. Meas Sci Technol 24(2):024002. https://doi.org/10.1088/0957-0233/24/2/024002

Lagemann C, Lagemann K, Mukherjee S, Schröder W (2021) Deep recurrent optical flow learning for particle image velocimetry data. Nat Mach Intell 3(7):641–651. https://doi.org/10.1038/s42256-021-00369-0

Lagemann C, Lagemann K, Mukherjee S, Schröder W (2022) Generalization of deep recurrent optical flow estimation for particle-image velocimetry data. Meas Sci Technol 33(9):094003. https://doi.org/10.1088/1361-6501/ac73db

Lecordier B, Westerweel J (2004) The EUROPIV synthetic image generator (S.I.G.). In: Stanislas M, Westerweel J, Kompenhans J (eds) Particle image velocimetry: recent improvements. Springer, Berlin, Heidelberg, pp 145–161. https://doi.org/10.1007/978-3-642-18795-7_11

LeCun YA, Bottou L, Orr GB, Müller KR (2012) Efficient BackProp. In: Montavon G, Orr GB, Müller KR (eds) Neural Networks: Tricks of the Trade: 2nd edn. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp 9–48

Li Y, Perlman E, Wan M, Yang Y, Meneveau C, Burns R, Chen S, Szalay A, Eyink G (2008) A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence. J Turbul 9:N31. https://doi.org/10.1080/14685240802376389

Liang J, Cai S, Xu C, Chu J (2020) Filtering enhanced tomographic PIV reconstruction based on deep neural networks. IET Cyber-Systems and Robotics 2(1):43–52. https://doi.org/10.1049/iet-csr.2019.0040. https://onlinelibrary.wiley.com/doi/abs/10.1049/iet-csr.2019.0040

Liang J, Xu C, Cai S (2022) GotFlow3D: recurrent graph optimal transport for learning 3d flow motion in particle tracking. https://doi.org/10.48550/arXiv.2210.17012. arXiv:2210.17012 [physics]

Liu X, Katz J (2006) Instantaneous pressure and material acceleration measurements using a four-exposure PIV system. Exp Fluids 41(2):227. https://doi.org/10.1007/s00348-006-0152-7

Liu X, Katz J (2013) Vortex-corner interactions in a cavity shear layer elucidated by time-resolved measurements of the pressure field. J Fluid Mech 728:417–457. https://doi.org/10.1017/jfm.2013.275

Mallery K, Shao S, Hong J (2020) Dense particle tracking using a learned predictive model. Expe Fluids 61(10):223. https://doi.org/10.1007/s00348-020-03061-y

Mao Z, Lu L, Marxen O, Zaki TA, Karniadakis GE (2021) DeepM &Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators. J Comput Phys 447:110698. https://doi.org/10.1016/j.jcp.2021.110698

McClenny L, Braga-Neto U (2020) Self-adaptive physics-informed neural networks using a soft attention mechanism. arXiv:2009.04544 [cs, stat]

Mons V, Wang Q, Zaki TA (2019) Kriging-enhanced ensemble variational data assimilation for scalar-source identification in turbulent environments. J Comput Phys 398:108856. https://doi.org/10.1016/j.jcp.2019.07.054

Novara M, Scarano F (2013) A particle-tracking approach for accurate material derivative measurements with tomographic PIV. Exp Fluids 54(8):1584. https://doi.org/10.1007/s00348-013-1584-5

Oh S, Lee S, Son M, Kim J, Ki H (2022) Accurate prediction of the particle image velocimetry flow field and rotor thrust using deep learning. J Fluid Mech. https://doi.org/10.1017/jfm.2022.135

van Oudheusden BW (2013) PIV-based pressure measurement. Meas Sci Technol 24(3):032001. https://doi.org/10.1088/0957-0233/24/3/032001

Raissi M, Perdikaris P, Karniadakis GE (2019) Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 378:686–707. https://doi.org/10.1016/j.jcp.2018.10.045

Raissi M, Yazdani A, Karniadakis GE (2020) Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations. Science 367(6481):1026–1030. https://doi.org/10.1126/science.aaw4741

Schanz D, Gesemann S, Schröder A (2016) Shake-the-box: Lagrangian particle tracking at high particle image densities. Exp Fluids 57(5):70. https://doi.org/10.1007/s00348-016-2157-1

Shukla K, Clark Di Leoni P, Blackshire J, Sparkman D, Karniadakis GE (2020) Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. J Nondestruct Eval 39(3):61. https://doi.org/10.1007/s10921-020-00705-1

Tsuji Y, Fransson JHM, Alfredsson PH, Johansson AV (2007) Pressure statistics and their scaling in high-Reynolds-number turbulent boundary layers. J Fluid Mech 585:1–40. https://doi.org/10.1017/S0022112007006076

Villegas A, Diez FJ (2014) Evaluation of unsteady pressure fields and forces in rotating airfoils from time-resolved PIV. Exp Fluids 55(4):1697. https://doi.org/10.1007/s00348-014-1697-5

Violato D, Moore P, Scarano F (2011) Lagrangian and Eulerian pressure field evaluation of rod-airfoil flow from time-resolved tomographic PIV. Exp Fluids 50(4):1057–1070. https://doi.org/10.1007/s00348-010-1011-0

Wang J, Zhang C, Katz J (2019) GPU-based, parallel-line, omni-directional integration of measured pressure gradient field to obtain the 3D pressure distribution. Exp Fluids 60(4):58. https://doi.org/10.1007/s00348-019-2700-y

Wang M, Wang Q, Zaki TA (2019) Discrete adjoint of fractional-step incompressible Navier-Stokes solver in curvilinear coordinates

and application to data assimilation. J Comput Phys 396:427–450. https://doi.org/10.1016/j.jcp.2019.06.065

Wang M, Zaki TA (2021) State estimation in turbulent channel flow from limited observations. J Fluid Mech. https://doi.org/10.1017/jfm.2021.268

Wang M, Zaki TA (2022) Synchronization of turbulence in channel flow. J Fluid Mech 943:A4

Wang Q, Hasegawa Y, Zaki TA (2019) Spatial reconstruction of steady scalar sources from remote measurements in turbulent flow. J Fluid Mech 870:316–352. https://doi.org/10.1017/jfm.2019.241

Wang S, Teng Y, Perdikaris P (2021) Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM J Sci Comput 43(5):A3055–A3081. https://doi.org/10.1137/20M1318043

Wang S, Yu X, Perdikaris P (2022) When and why PINNs fail to train: a neural tangent kernel perspective. J Comput Phys 449:110768. https://doi.org/10.1016/j.jcp.2021.110768

Weinan E, Yu B (2018) The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun Math Stat 6(1):1–12

Xiang Z, Peng W, Zheng X, Zhao X, Yao W (2021) Self-adaptive loss balanced physics-informed neural networks for the incompressible Navier–Stokes equations. https://doi.org/10.48550/arXiv.2104.06217. arXiv:2104.06217 [physics]

Xie Y, Franz E, Chu M, Thuerey N (2018) tempoGAN: a temporally coherent, volumetric GAN for super-resolution fluid flow. ACM Trans Graph 37(4):95:1-95:15. https://doi.org/10.1145/3197517.3201304

Zaki TA, Wang M (2021) From limited observations to the state of turbulence: fundamental difficulties of flow reconstruction. Phys Rev Fluids 6(10):100501. https://doi.org/10.1103/PhysRevFluids.6.100501