Physics-informed deep-learning applications to experimental fluid mechanics

Hamidreza Eivazi\textsuperscript{1}, Yuning Wang\textsuperscript{2} and Ricardo Vinuesa\textsuperscript{2,3,*}

\textsuperscript{1} Institute for Software and Systems Engineering, Clausthal University of Technology, 38678 Clausthal-Zellerfeld, Germany
\textsuperscript{2} FLOW, Engineering Mechanics, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden
\textsuperscript{3} Swedish e-Science Research Centre (SeRC), Stockholm, Sweden

E-mail: rvinuesa@mech.kth.se and he76@tu-clausthal.de

Received 13 January 2023, revised 10 April 2024
Accepted for publication 17 April 2024
Published 26 April 2024

Abstract

High-resolution reconstruction of flow-field data from low-resolution and noisy measurements is of interest due to the prevalence of such problems in experimental fluid mechanics, where the measurement data are in general sparse, incomplete and noisy. Deep-learning approaches have been shown suitable for such super-resolution tasks. However, a high number of high-resolution examples is needed, which may not be available for many cases. Moreover, the obtained predictions may lack in complying with the physical principles, e.g. mass and momentum conservation. Physics-informed deep learning provides frameworks for integrating data and physical laws for learning. In this study, we apply physics-informed neural networks (PINNs) for super-resolution of flow-field data both in time and space from a limited set of noisy measurements without having any high-resolution reference data. Our objective is to obtain a continuous solution of the problem, providing a physically-consistent prediction at any point in the solution domain. We demonstrate the applicability of PINNs for the super-resolution of flow-field data in time and space through three canonical cases: Burgers’ equation, two-dimensional vortex shedding behind a circular cylinder and the minimal turbulent channel flow. The robustness of the models is also investigated by adding synthetic Gaussian noise. Furthermore, we show the capabilities of PINNs to improve the resolution and reduce the noise in a real experimental dataset consisting of hot-wire-anemometry measurements. Our results show the adequate capabilities of PINNs in the context of data augmentation for experiments in fluid mechanics.

Keywords: physics informed neural networks, machine learning, deep learning, vortex shedding, channel flow

* Author to whom any correspondence should be addressed.

Original Content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
1. Introduction

Deep learning (DL) has brought new opportunities for modeling and forecasting the dynamics of complex multiscale systems due to its excellent performance in non-linear function approximation from examples. The emergence of various architectures, e.g. convolutional neural networks (CNNs) (Lecun et al 1998) and long short-term memory (Hochreiter and Schmidhuber 1997), has led to the development of a variety of DL-based frameworks for modeling complex physical systems by accounting for spatio-temporal dependencies in the predictions. DL has been employed in many scientific and engineering disciplines, e.g. astronomy (Waldmann and Griffith 2019), climate modeling (Ham et al 2019), solid mechanics (Mianroodi et al 2021), chemistry (Segler et al 2018) and sustainability (Vinuesa et al 2020, Larosa et al 2023). Fluid mechanics has been one of the active research topics for the development of innovative deep-learning-based approaches (Kutz 2017, Duraisamy et al 2019, Brunton et al 2020, Khan et al 2021a, 2021b, Vinuesa and Brunton 2022). Successful application of deep neural networks (DNNs) has been shown in the data-driven turbulence closure modeling (Ling et al 2016, Jiang et al 2021), prediction of temporal dynamics of low-order models of turbulence (Srinivasan et al 2019, Eivazi et al 2021), extraction of turbulence theory for two-dimensional decaying isotropic turbulence (Jiménez 2018), non-intrusive sensing in turbulent flows (Güemes et al 2021, Guastoni et al 2021), and active flow control through deep reinforcement learning (Rabault et al 2019, Tang et al 2020, Vinuesa et al 2022, Guastoni et al 2023).

The first example of learning in experimental fluid mechanics dates back to the early 1960s for drag reduction (Rechenberg 1964). In the early 1990s several approaches were developed based on neural networks with the application in trajectory analysis and classification for particle tracking velocimetry (Adamczyk and Rimai 1988) and particle image velocimetry (PIV) (Adrian 1984, Teo et al 1991, Grant and Pan 1995). In recent decades, image processing and computer vision have experienced a breakthrough with the introduction of DL methods, fostering the application of DL in processing and enhancing PIV measurements. The majority of the literature has been devoted to improving spatial resolution, recovering missing data, reconstruction of three-dimensional fields from sparse measurements, and post-processing.

Reconstruction of high-resolution images from low-resolution ones, the so-called super-resolution reconstruction, has been an active research topic in the field of computer science (Wang et al 2021). The success of DL in super-resolution tasks within computer science motivated its application to fluid mechanics and turbulence using CNNs (Fukami et al 2019, Liu et al 2020), super-resolution generative adversarial networks (Güemes et al 2021), and unsupervised learning based on cycle-consistent GANs (Kim et al 2021). DL has been also applied for super-resolution of four-dimensional flow magnetic-resonance imaging (MRI) data (Ferdian et al 2020). However, supervised deep-learning-based super-resolution approaches are limited by the need for high-resolution labels for training, i.e. a pair set of low and high-resolution fields are needed, which is difficult to obtain from experiments. Recently, Güemes et al (2021) showed the capability of GANs to generate high-resolution fields from PIV data without having a priori any high-resolution field.

Flow data estimation and reconstruction from incomplete data are other common problems in experimental fluid mechanics. Gappy proper orthogonal decomposition (POD) (Everson and Sirovich 1995) has been shown suitable for flow-field reconstruction from incomplete data (Bui-Thanh et al 2004); however, it can not be directly applied to incomplete original data because it requires the spatial coefficients as a basis from the reference data (Bui-Thanh et al 2003, Willcox 2006). Recently, (Morimoto et al 2021) proposed an autoencoder-type CNN for PIV data estimation considering missing regions that can overcome such limitations. Application of machine learning and DL has been also shown for flow-field reconstruction from sparse turbomachinery data (Akbari and Montazerin 2021). Another research direction for augmenting sparse data is to perform three-dimensional fluid data reconstruction from sectional flow fields (Chandramouli et al 2019). Stulov and Chertkov (2021) proposed a CNN-based architecture including two-dimensional and three-dimensional convolutions to reconstruct three-dimensional data from a few numbers of two-dimensional sections in a computationally friendly manner. Moreover, the application of DL for performing end-to-end PIV has been shown in the experimental-fluid-mechanics community by Rabault et al (2017) and motivated by the progress in computer vision regarding the motion-estimation problem (Hui et al 2018, Yang and Ramanan 2019, Cai et al 2020, Stulov and Chertkov 2021).

Deep-learning methods are powerful modeling tools for data-rich domains such as vision, language, and speech. However, extracting interpretable information and learning generalizable knowledge is still a challenge, especially for domains with limited data available (Rudin 2019, Vinuesa and Sirmacek 2021). Purely data-driven techniques require large datasets for training that may not be available for many engineering problems. Moreover, these models may fit the observational data very well but lack in complying with the fundamental physical laws. Therefore, there is a pressing need for integrating governing physical laws and domain knowledge into the model training, which can act as an informative prior to informing the model about the empirical, physical, or mathematical understanding of the system besides the observational data. Physics-informed neural networks (PINNs), which were introduced by Raissi et al (2019), provide a framework to integrate data and physical laws in the form of governing partial differential equations (PDEs) for learning. This can lead to models that are robust in the presence of imperfect data, e.g. missing or noisy values, and can provide accurate and physically-consistent predictions (Karniadakis et al 2021).
PINNs are well suited for ill-posed and inverse problems related to several different types of PDEs. PINNs were applied to simulate vortex-induced vibrations (Raissi et al. 2019) and to tackle ill-posed inverse fluid-mechanics problems (Raissi et al. 2020). Jin et al. (2021) showed the applicability of PINNs for the simulation of turbulence directly, where good agreement was obtained between the direct numerical simulation (DNS) results and the PINNs simulation results. Eivazi et al. (2022) employed PINNs for solving Reynolds-averaged Navier–Stokes (RANS) equations without any specific model or assumption for turbulence and using data as the turbulence closure. PINNs were employed also for super-resolution and denoising of cardiovascular-flow MRI data (Fathi et al. 2020, Gao et al. 2021) and the prediction of near-wall blood flow from sparse data (Arzani et al. 2021).

Recently, Cai et al. (2021b) proposed a method based on PINNs to infer the full continuous three-dimensional velocity and pressure fields from snapshots of three-dimensional temperature fields obtained by tomographic background-oriented Schlieren imaging. A detailed discussion on the prevailing trends in embedding physics into machine-learning algorithms and diverse applications of PINNs can be found in the work by Karniadakis et al. (2021), and a review of PINNs applications to fluid mechanics was provided by Cai et al. (2021a).

In this study, we employ PINNs for the super-resolution of flow fields both in time and space from a limited set of measurements without having any high-resolution targets. This is a very important problem that can enhance the accuracy of experimental data in fluid-mechanics studies. Our objective is to learn a functional mapping parameterized by a neural network from spatial $(x)$ and temporal $(t)$ coordinates to the flow parameters, i.e. velocity components and pressure, which provides a continuous solution to the problem. We first show the applicability of PINNs for super-resolution tasks in time by solving an ill-posed problem regarding the canonical case of Burgers’ equation. Next, we examine the performance of PINNs in the super-resolution of flow-field data in the presence of noise for the two-dimensional vortex shedding behind a circular cylinder at a Reynolds number of $Re_D = U_c D / \nu = 100$ based on the cylinder diameter $D$ and the free-stream velocity $U_c$, where $\nu$ is the kinematic viscosity. Then, we employ PINNs for the super-resolution of noisy data with different resolutions in time and space for a turbulent minimal channel flow at a Reynolds number based on the channel half-height $h$ and the laminar centerline-velocity $U_c$ of $Re_D = U_c h / \nu = 5000$. Finally, we show the feasibility of using PINNs for super-resolution and de-noising of a real experimental dataset consisting of hot-wire-anemometry measurements.

One important novelty of this study is the fact that it does not require high-resolution targets to improve the resolution of the measurements, as opposed to computer-vision approaches such as GANs (Güemes et al. 2021). Furthermore, computer-vision-based methods may not necessarily fulfill the underlying governing equations, whereas, in the present framework, the governing equations are part of the loss function. This ensures the compliance of the high-resolution data with the physics of the problem. Another important novel aspect of this work is the treatment of unsteady problems in PINNs. While recent work has documented successful reconstruction of the mean quantities based on unsteady data (Sliwinski and Rigas 2023), in this work we perform the reconstruction of the instantaneous flow fields, based on low-resolution and noisy data. Note that this is done in a turbulent channel, which is a very challenging flow case. Finally, while there is work in the literature discussing de-noising strategies in fluid mechanics using data-driven methods, such as POD (Méndez et al. 2017), the advantage of the current approach is the fact that it requires a lower amount of snapshots, and the data in those snapshots can be incomplete in space. Therefore, the present framework can significantly help to improve experimental measurements in fluid mechanics.

This article is organized as follows: in section 2 we provide an overview of the PINNs and discuss the theoretical background; in section 3 we discuss the application of PINNs for super-resolution of flow-field data in both time and space without having any high-resolution targets; in section 4 we discuss the application of the proposed methods to an actual experimental dataset; and finally, in section 5 we provide a summary and the conclusions of the study.

2. Methodology

2.1. DNNs

DNNs are universal approximators of continuous functions, obtained by composing simple but non-linear transformations at each level (the so-called layer). By the composition of enough such non-linear transformations and having a large training dataset available, a DNN can learn very complex functions using the backpropagation algorithm (LeCun et al. 2015). Multilayer perceptrons (MLPs) (Rumelhart and McClelland 1987) are the most basic type of artificial neural network comprised of two or more fully-connected layers of neurons, where each node is connected to all nodes in the preceding and succeeding layers. Let us consider $X$ and $\Psi$ as the input and output spaces and each pair of vectors $(\chi, \psi)$ as a training example or sample. The neural network aims to find the mapping $f: X \rightarrow \Psi$ using the training set such that a loss function $L(f(\chi); \psi)$ is minimized. An MLP approximates the function $f$ using the composition of simple transfer functions, i.e. linear matrix operations followed by an element-wise nonlinear function called activation function, e.g. sigmoidal or hyperbolic tangent. For the layer $i$, the weight matrix $W_i$ and the bias $b_i$ perform the linear matrix transformation from layer $i - 1$ to the next; then the operation of the activation function $g_i$ leads to the output of the layer as $z_i = g_i(W_i z_{i-1} + b_i)$. The goal of training an MLP is to determine $W_i$ and $b_i$ using training data for the given activation functions.
2.2. PINNs

2.2.1. Model architecture. PINNs are deep-learning-based frameworks for solution or discovery of PDEs (Raissi et al 2019). The goal of PINNs is to include the information from the governing physical laws into the training process of a DNN. A PINN comprises two parts: an MLP and a so-called residual network, which calculates the residual of the governing equations. In a general set-up dealing with spatio-temporal PDEs, the temporal and the spatial coordinates \((t, x)\) are the inputs of the MLP and the solution vector of the PDE system \(u = f(t, x)\) is the output where the function \(f\) is parameterized by the MLP. Automatic differentiation (AD) (Baydin et al 2018) is utilized to differentiate the outputs \(u\) with respect to the inputs \((t, x)\) and formulate the governing equations. AD can be implemented directly from the deep-learning framework since it is used to compute the gradients and update the network parameters, i.e. weights \(W\) and biases \(b\), during the training. We consider PDEs of the general form:

\[
\mathbf{u}_t + \mathcal{N} [\mathbf{u}] = 0, \quad x \in \Omega, t \in [0, T],
\]

where \(\mathbf{u}(t, x)\) denotes the solution vector of the PDE system, \(\mathbf{u}_t\) is its derivative with respect to time \(t\) in the period \([0, T]\), \(\mathcal{N}[\cdot]\) is a nonlinear differential operator and \(x\) is the vector of spatial coordinates defined over the domain \(\Omega\). Let us consider the residual of the PDEs \(e(t, x)\) as the left-hand-side of equation (1):

\[
e := \mathbf{u}_t + \mathcal{N} [\mathbf{u}],
\]

while the MLP approximates \(u = f(t, x)\). The function \(f\) is a composition of simple differentiable operations, which maps the coordinates to the solution, i.e. \(f: (t, x) \rightarrow u\). Therefore, it is possible to obtain derivatives of the outputs with respect to the inputs simply using the chain rule on \(f\): \(\partial u / \partial x = \mathcal{G}(\mathbf{u}, x)\) and \(\partial u / \partial t = \mathcal{G}(\mathbf{u}, t)\) where \(\mathcal{G}\) indicates the chain-rule operation. Therefore, \(\mathbf{u}_t\) and \(\mathcal{N}[u]\) in equation (1) can be obtained by applying the chain rule for differentiating compositions of functions in the MLP using AD, thus yielding the residual of the governing equations. The aforementioned process for computation of the residuals through the implementation of AD is known as the residual network. We use the open-source deep-learning framework TensorFlow (Abadi et al 2016) to develop our PINN models. TensorFlow provides the ‘tf.GradientTape’ application programming interface (API) for AD by computing the gradient of a computation with respect to inputs. TensorFlow ‘records’ the computational operations executed inside the context of a tf.GradientTape onto a so-called ‘tape’, and then uses that tape to compute the gradients of a recorded computation using reverse-mode differentiation.

2.2.2. Loss function. The training process of a PINN consists of both supervised and unsupervised parts. We refer to supervised learning for those samples for which the target solution is available and unsupervised learning for the samples for which the solution is unknown, while the residual of the governing equations is considered as the loss. Let us consider \(N_s\) as the number of points for which we have the solution, e.g. initial and boundary conditions or sparse measurements, and \(N_e\) as the number of collocation points for which we calculate the unsupervised loss for training. The unsupervised loss can be defined as:

\[
L_e = \frac{1}{N_e} \sum_{i=1}^{N_e} \left| e (t_i, x_i') \right|^2,
\]

where \(e(t_i, x_i')\) is the residual of the governing equation at each point \(x_i'\) and instant \(t_i\), and the supervised loss as:

\[
L_s = \frac{1}{N_s} \sum_{i=1}^{N_s} \left| u_i - u (t_i, x_i) \right|^2,
\]

where the subscript \(s\) denotes known values. Furthermore, we define the total loss as:

\[
L = \alpha L_s + \beta L_e,
\]

governing equations is considered as the loss. Let us consider \(N_s\) as the number of points for which we have the solution, e.g. initial and boundary conditions or sparse measurements, and \(N_e\) as the number of collocation points for which we calculate the unsupervised loss for training. The unsupervised loss can be defined as:

\[
L_e = \frac{1}{N_e} \sum_{i=1}^{N_e} \left| e (t_i, x_i') \right|^2,
\]

where \(e(t_i, x_i')\) is the residual of the governing equation at each point \(x_i'\) and instant \(t_i\), and the supervised loss as:

\[
L_s = \frac{1}{N_s} \sum_{i=1}^{N_s} \left| u_i - u (t_i, x_i) \right|^2,
\]

where the subscript \(s\) denotes known values. Furthermore, we define the total loss as:

\[
L = \alpha L_s + \beta L_e,
\]

where \(\alpha\) and \(\beta\) are the weighting coefficients. The shared parameters between the neural network and the residual network can be learned in a physics-informed manner by minimizing the total loss \(L\). As discussed by Eivazi et al (2022), although the PINNs approach is slower than a regular numerical integration of the governing equations, PINNs enable using coarser meshes due to the AD. Therefore, the computational cost is similar to that of a numerical solver, with better performance when turbulence is present, e.g. for RANS simulations. A schematic representation of the proposed method is shown in figure 1.

2.2.3. Training. Following the work by Raissi et al (2019), we employ a full-batch training procedure to learn function \(f\). We initiate the training using the Adam optimizer (Kingma and Ba 2017) for 1000 epochs with the learning rate of \(1 \times 10^{-3}\) and then apply the limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm (Liu and Nocedal 1989) to obtain the solution. The optimization process of the L-BFGS algorithm is stopped automatically based on the increment tolerance.
3. Physics-informed super-resolution of flow fields in time and space

We employ PINNs for super-resolution of flow-field data in time and space from only a limited set of low-resolution measurements containing noise and without having any high-resolution targets. The training data for supervised learning consists of only a few very low-resolution snapshots of the velocity components while we aim to obtain a continuous solution for velocity components and pressure. We consider three canonical test cases, i.e. Burgers’ equation, two-dimensional vortex shedding behind a circular cylinder at \( Re_D = 100 \) (Raissi et al 2019) and a turbulent minimal channel at \( Re_c = 5000 \) (Borrelli et al 2022) to show the applicability of PINNs for such super-resolution tasks in fluid mechanics. For all tests we use fully-connected networks and consider the different weighting coefficients in the loss function based on the flow case. For Burgers’ equation, we utilized a neural network comprising 8 hidden layers with 20 neurons per hidden layer. For the cylinder wake and the minimal channel, we selected the model architecture based on the findings derived from hyper-parameter tuning process detailed in A.1 and A.2, respectively. For all the models, the hyperbolic tangent is used as the activation function.

To evaluate the performance of the PINNs in the super-resolution tasks, we define the relative Euclidean norm of errors for a particular component as:

\[
\epsilon_\phi = \left\| \hat{\phi} - \phi \right\|^2 / \| \phi \|^2,
\]

where \( \hat{\phi} \) represents the PINN simulation results and \( \phi \) denotes the reference data for a particular component of the solution, which can be either one of the velocity components (\( u, v, w \)) or the pressure (\( p \)). Note that \( \| \cdot \| \) is the Euclidean norm and \( \langle \cdot \rangle \) is the average in the time. And in order to quantify the prediction accuracy, we define the correlation coefficient (\( r_\phi \)) for a particular variable \( \phi \) as:

\[
r_\phi = \frac{\sum_{i=1}^{n} (\hat{\phi}_i - \bar{\phi})(\phi_i - \bar{\phi})}{\sqrt{\sum_{i=1}^{n} (\hat{\phi}_i - \bar{\phi})^2 \sum_{i=1}^{n} (\phi_i - \bar{\phi})^2}},
\]

where \( \bar{\phi} \) denotes the mean value of the quantities. The correlation coefficient is in the range of \(-1\) to 1. Note that the higher linear correlation between prediction and reference is obtained, the closer the value to 1.

3.1. Synthetic low-resolution data with noise

In order to emulate possible measurement errors in an experiment, synthetic low-resolution data are generated from the reference velocity field by selecting a limited set of points in time and space. The low-resolution velocity field \( \hat{\mathbf{Q}}_t \) is corrupted by an independent and identically-distributed (i.i.d) Gaussian noise as:

\[
\hat{\mathbf{Q}}_t = \mathbf{Q}_t \cdot (\mathbf{1} + c \epsilon),
\]

where \( \epsilon \sim \mathcal{N}(0,1) \) represents the noise factor, \( c \in [0,1] \) indicates the level of the noise and and \( \mathbf{I} \) is the identity matrix.

3.2. Burgers’ equation

For the first test case, we consider the Burgers’ equation, which arises in various areas of applied mathematics, including fluid mechanics. We consider a one-dimensional case with Dirichlet boundary conditions and sinusoidal initial condition as:

\[
\begin{align*}
\frac{u_t + uu_x}{Re} &= 0, & x \in [-1,1], & t \in [0,0.99], \\
u(0,x) &= -\sin(\pi x), \\
u(t,-1) &= u(t,1) = 0.
\end{align*}
\]

Note that (9) is obtained from the Navier–Stokes equations for the velocity field by excluding the pressure-gradient term. We only take the data at the first time-step \( t = 0 \) and the last time-step \( t = 0.99 \) of the solution domain for supervised learning and our objective is to obtain a continuous solution in time. This can be seen as an extensive super-resolution task in time using only two snapshots of the data for training. Moreover, this is an ill-posed problem since we do not have the boundary conditions for the time-steps between \( t = 0 \) and \( t = 0.99 \). We consider \( N_s = 512 \) (256 for each time-step) and \( N_e = 2000 \) randomly-generated collocation points for the training. Results are reported in figure 2. The red vertical lines in figure 2(a) show the data used for supervised learning.

For the second test case, we consider the super-resolution in time and space for the two-dimensional vortex shedding behind a circular cylinder at \( Re_D = 100 \). This example aims to highlight the ability of PINNs in the flow-field super-resolution tasks from very low-resolution measurements both in time and space. We consider the incompressible Navier–Stokes equations as:

\[
\begin{align*}
\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} &= -\nabla p + \frac{1}{Re} \nabla^2 \mathbf{u} & \text{in } \Omega, \\
\nabla \cdot \mathbf{u} &= 0 & \text{in } \Omega,
\end{align*}
\]

where \( \mathbf{u} \) is the non-dimensional velocity vector, \( t \) is the non-dimensional time and \( p \) is the non-dimensional pressure. The reference data has a resolution of \( (N_x,N_y,N_z) = (100,50,200) \) with a non-dimensional \( \Delta t = 0.1 \). The flow field is simulated by numerically integrating equation (10) over a time interval.
Figure 2. Super-resolution in time using PINNs for the one-dimensional Burger’s equation: (a) comparison of the solution obtained from PINNs with the reference data; continuous solution of the shock-formation process (top) and the absolute error $\varepsilon = |u - \tilde{u}|$, where $u$ denotes the reference data and $\tilde{u}$ represents the PINN predictions (bottom). The red vertical lines indicate the data used for supervised learning. (b) Supervised and unsupervised losses, $L_s$ and $L_e$ respectively, during the training process. The green dashed line indicates $L_s$ and the blue line represents $L_e$.

Figure 3. A schematic view of the resolution of the training data used for the supervised learning for the two-dimensional vortex shedding behind a circular cylinder. Colors depict the streamwise velocity. The blue line shows the first temporal coefficient $a_1$ of the POD for this test case.

of almost one vortex-shedding cycle, i.e. $t \in [0, 7]$. We perform downsampling on the velocity fields with a factor of 10 in each spatial direction and only utilize three snapshots of low-resolution data at $t = 0.0, 3.5$ and 7.0 for supervised learning. Figure 3 depicts the problem set-up for this test case and the reference data used for training. The blue line shows the first temporal coefficient $a_1$ of the POD for this test case.

We utilize $N_s = 150$ points for supervised learning which is 0.14% of the total amount of data, and consider $N_e = 2000$ collocation points to compute the unsupervised loss. A hyperparameter study of the PINNs architecture is shown in A.1, where the optimal architecture is reported. The PINN model contains 4 hidden layers with 20 neurons per hidden layer with the hyperbolic-tangent activation function whereas the weights for supervised and unsupervised learning are $\alpha = 1$ and $\beta = 10$, respectively. We assess the performance of PINNs in the reconstruction of the continuous solution from the low-resolution and noisy data. Noise levels of $c = 2.5\%, 5\%$ and $10\%$ are considered, and figure 4 summarizes the results of our experiment for the noise level $c = 10\%$. The middle panel shows the noisy data. It should be noted that these two snapshots are not used for supervised learning and the noisy data are depicted only for comparison. An adequate agreement can be observed between the solution obtained from PINNs and the reference data, as discussed below quantitatively. Moreover, it is of interest to assess the performance of PINNs in the reconstruction of the high-frequency modes during the cycle of vortex shedding from only three measurements in time. To this end, we project the solution predicted by the PINN model on the POD modes computed from 200 snapshots to obtain the temporal coefficients of the POD $a_i(t)$, where $i$ indicates the mode number. Figure 4(b) shows the temporal coefficients of the POD for the 3rd, 5th and 7th modes obtained from the PINN solution in comparison with that of the reference data. The cross-correlation coefficients with respect to the reference temporal coefficients are 98.64, 96.54, and 94.37 for modes 3, 5, and 7, respectively, indicating that high-accuracy reconstruction of the temporal dynamics can be achieved even for high-frequency modes. Our results show the promising performance of the PINN model in enhancing the temporal resolution of the data and correcting the impact of experimental noise.

3.3.1. Discussion of vortex-shedding results. Table 1 shows $\varepsilon$ for the velocity components and pressure corresponding to the different noise levels. Despite the increasing noise levels leading to a rise in the relative Euclidean norm of errors $\varepsilon$, the PINN model provides solutions with high accuracy. However,
Figure 4. Results obtained based on super-resolution in time and space for the cylinder test case using PINNs from a limited set of measurements contaminated with a noise level of $\epsilon = 10\%$: (a) contours of vorticity in the $z$ direction at $t = 1.8$ and $5.3$. (b) Temporal coefficients of the POD for the 3rd, 5th and 7th modes obtained from the PINN solution in comparison with that of the reference data.

Table 1. Relative Euclidean norm of errors averaged over the whole domain and cross-correlation coefficient for the two-dimensional vortex shedding behind a circular cylinder.

| Noise level | $\epsilon_u$ | $\epsilon_v$ | $\epsilon_p$ | $r_u$ | $r_v$ | $r_p$ |
|------------|--------------|--------------|--------------|-------|-------|-------|
| clean      | 2.47%        | 5.92%        | 14.32%       | 0.997 | 0.998 | 0.979 |
| 2.5%       | 3.13%        | 6.92%        | 10.63%       | 0.995 | 0.997 | 0.990 |
| 5%         | 3.96%        | 8.66%        | 19.06%       | 0.992 | 0.996 | 0.961 |
| 10%        | 7.24%        | 13.12%       | 33.07%       | 0.973 | 0.991 | 0.890 |

Table 2. Cross-correlation coefficient for vorticity field $\omega$ obtained for each noise level in the noisy reference data and the prediction of PINNs. For each noise level, we denote the highest correlation in bold.

| Type         | Noise level | $r_\omega$ |
|--------------|-------------|-------------|
| Noisy reference | 2.5%     | 0.948     |
| PINNs        | 2.5%        | 0.985      |
| Noisy reference | 5%       | 0.831     |
| PINNs        | 5%          | 0.981      |
| Noisy reference | 10%      | 0.598     |
| PINNs        | 10%         | 0.954      |

The errors of individual quantities do not explicitly represent the quality of the obtained reconstruction of vortex shedding via the proposed method. To address this, we evaluate the denoising results on the vorticity field $\omega$ using equation (7). Table 2 reports the correlation coefficients obtained for the noisy reference and the predictions of PINNs with respect to the ground truth vorticity fields, respectively. The results indicate that PINNs exhibit promising performance in denoising and reconstructing vorticity fields with limited measurements, highlighting the potential applications of our proposed method in experimental scenarios.

Note that to evaluate the accuracy of the PINN solutions, we apply a shift to the PINN-simulation results to bring the means of the pressure for reference data and the PINN results to the same value. We observe that the error for the pressure is higher than that of the velocity components since it is not used for supervised learning. Figure 5 depicts the PINN-simulation results for the pressure at $t = 5.3$ for the clean test case. It can be observed that although the absolute error $\epsilon$ in some regions is slightly higher, the overall patterns in the flow are...
reconstructed with very good accuracy. The pressure at the peak of vorticity obtained by PINNs is $-0.237$, which meets the agreement with the ground truth of $-0.232$.

### 3.4. Minimal turbulent channel

We employ a dataset (Borrelli et al. 2022) of the minimal turbulent channel at $Re_{cl} = 5000$ generated following the parameters described in the work by Jiménez and Moin (1991). This database was obtained through DNS as discussed by Borrelli et al. (2022). The box size was set to $x_l = 0.6\pi h$, $y_l = 2h$, and $z_l = 0.18\pi h$, where $h$ denotes the half-height of the channel. The resolution is $(N_x, N_y, N_z) = (32, 129, 16)$, representing the number of grid points in the streamwise ($x$), wall-normal ($y$) and spanwise ($z$) directions, respectively. We take a time interval of $t = [0,4]$ and a domain size of $x_d = 0.6\pi h$, $y_d = h$ and $z_d = 0.01125\pi h$ for our super-resolution task. We assess the performance of PINNS in the reconstruction of a continuous solution from a limited set of measurements with different resolutions in time and space and the presence of noise. For this test case, we employ a PINN model with 10 hidden layers, each containing 100 neurons with hyperbolic tangent as the activation function and the three-dimensional incompressible Navier–Stokes equations, the results of hyper-parameter studies are reported in A.2. For time, we consider two temporal resolutions: $\Delta t = 1.0$ and $2.0$ corresponding to, respectively, taking only five and three low-resolution snapshots for supervised learning. And, we utilize two different resolutions in space, i.e. $(n_x, n_y, n_z) = (8, 8, 2)$ and $(16, 16, 2)$ where $n_x$, $n_y$, and $n_z$ represent the number of selected points in the $x$, $y$, and $z$ directions, respectively. The spacing between the selected points in the $x$ and $z$ directions is constant, but the spacing in the $y$ direction $\delta y$ is not uniform: the closer to the wall the lower $\delta y$. We represent different datasets with different resolutions as t#–s# example Figure 6 shows the instantaneous wall-normal velocity $v$ for the dataset t3–s16, which has a low resolution in time and a moderate resolution in space. As the minimal turbulent channel is focused on near-wall turbulence (Jiménez and Moin 1991), we scale the selected spatial and temporal resolutions by the viscous length $l^* = \nu/\tau_w$ (where $\nu$ is the fluid kinematic viscosity, and $\tau_w = \sqrt{\rho/\rho}$ is the friction velocity) and viscous time $t^* = \nu/\nu^*$, Table 3 summarizes the inner-scaled time interval $\Delta t/l^*$, inner-scaled spacings in the $x$ and $z$ directions ($\delta x/l^*$, $\delta z/l^*$), as well as the inner-scaled minimum and maximum spacing in the $y$ direction (min $\delta y/l^*$, max $\delta y/l^*$).

The trained PINN model with each dataset is represented as PINN–t#–s#. Figure 6 shows the instantaneous wall-normal velocity $v$ for the dataset t3–s16, which has a low resolution in time and a moderate resolution in space. As the minimal turbulent channel is focused on near-wall turbulence (Jiménez and Moin 1991), we scale the selected spatial and temporal resolutions by the viscous length $l^* = \nu/\tau_w$ (where $\nu$ is the fluid kinematic viscosity, and $\tau_w = \sqrt{\rho/\rho}$ is the friction velocity) and viscous time $t^* = \nu/\nu^*$, Table 3 summarizes the inner-scaled time interval $\Delta t/l^*$, inner-scaled spacings in the $x$ and $z$ directions ($\delta x/l^*$, $\delta z/l^*$), as well as the inner-scaled minimum and maximum spacing in the $y$ direction (min $\delta y/l^*$, max $\delta y/l^*$).

The scaling results indicate that the adopted lower resolutions drastically decrease the temporal and spatial resolutions, sampling around 1% of the total data as the ground truth for supervised learning.

Table 4 summarizes the results of our experiments for the minimal turbulent channel using the training data of different resolutions and for different noise levels. Our results show great potential from the PINN model by using the t5–s16 dataset for supervised learning, leading to $\epsilon_u = 0.53\%$, $\epsilon_v = 12.54\%$ and $\epsilon_w = 9.89\%$ whereas $r_u = 0.999$, $r_v = 0.991$ and $r_w = 0.995$. An adequate performance can also be achieved using the dataset t3–s16, which leads to $\epsilon_u = 0.81\%$, $\epsilon_v = 17.57\%$ and $\epsilon_w = 20.75\%$ whereas $r_u = 0.999$, $r_v = 0.981$ and $r_w = 0.987$. Note that the presented trend shows the evolution of the error with increasing downsampling, and although in an experiment with no ground-truth data the actual error would not be possible to quantify, it is possible to establish estimates of expected errors based on numerical trends. Our results show the good performance of PINNS for super-resolution of turbulent flows both in time and space from only a limited set of measurements and without having any high-resolution target data. This is a very relevant result in the context of data augmentation for experiments in fluid mechanics.
Further reduction of the spatial resolution leads to a considerable increase in error. For instance, the PINN model PINN–t3–s16 yields simulation results with $\epsilon_u = 2.64\%$, $\epsilon_v = 46.72\%$, and $\epsilon_w = 56.60\%$, with corresponding correlation coefficients of $r_u = 0.998$, $r_v = 0.8655$, and $r_w = 0.8563$. This may be associated with the fact that the magnitudes of $\nu$ and $\omega$ are lower than that of $u$, thus being more sensitive to the small variation when computing the error via equation (6). However, it is noteworthy that the obtained $r_v$ and $r_w$ indicate that the wall-normal and spanwise velocity components are reconstructed with adequate accuracy.

We compare the performance of the PINN models PINN–t3–s16 and PINN–t3–s8: figure 7 illustrates the evolution of the velocity components in time for a point at $(x, y) = (0.0, 0.36)$, using both of the aforementioned models. The red square shows the location of the point on the contour of the streamwise velocity $u$. For these cases, we only use the data at $t = 0.0$, 2.0, and 4.0 for supervised learning. It can be observed that although the training data for supervised learning is sparse, the PINN–t3–s16 model provides a very good reconstruction of the velocity components in time, yielding cross-correlation coefficients of $0.949$, $0.965$, and $0.947$ for the streamwise, wall-normal, and spanwise velocities, respectively. However, the results obtained from the PINN–t3–s8 model are not as accurate, with cross-correlation coefficients of $0.791$, $0.776$, and $0.584$ for $u$, $v$, and $w$, respectively.

3.4.1. Discussion of minimal-channel results. Figures 8 and 9 depict contours of the velocity components $(u, v, w)$ and the spanwise vorticity $(\omega_z)$ obtained from the PINN–t3–s16 and the PINN–t3–s8 models compared with the reference data at $t = 1.0$ and 3.0, respectively. Our results demonstrate that the PINN model PINN–t3–s16 can effectively reconstruct the continuous solution of the velocity components in all three directions, with cross-correlation coefficients yielding $r_u = 0.999$, $r_v = 0.981$, and $r_w = 0.9871$, indicating promising accuracy. Additionally, the relative Euclidean errors of the wall-normal and spanwise velocities are observed to be much higher than that of the streamwise velocity.

Moreover, it is of interest to investigate the range of scales that can be reconstructed by the various models. To this end, we computed the inner-scaled streamwise one-dimensional premultiplied power-spectral density (PSD) of the various velocity components at $y^+ = 30$ obtained by the PINN–t3–s16 and PINN–t3–s8 models, as depicted in figure 10. The reconstructed spectra obtained by PINN–t3–s16 agrees very well with the reference data for the larger wavelengths, indicating that the model can reconstruct the large-scale features of the flow. For the streamwise, wall-normal and spanwise velocity components, the spectra exhibit deviations with respect to the ground truth for the scales smaller than $\lambda^+_u = 51.0$, $39.7$ and $32.4$, respectively. Furthermore, the spectra reconstructed by PINN–t3–s8 exhibit higher deviations with respect to the ground truth with those obtained by PINN–t3–s16 model for all the velocity components, a fact that is caused by the reduced spatial resolution.

The effect of noise on the accuracy of the reconstructions is also assessed and the results are reported in table 4. We observe that very good results can be achieved, even in the presence of 5% noise, from the PINN–t5–s16 and PINN–t3–s16 models. For the PINN–t5–s16 model trained on the noisy data, the highest error is obtained for the wall-normal velocity and it is equal to $\epsilon_w = 21.78\%$, while the lowest error is again obtained for the streamwise velocity, and it is equal to $\epsilon_u = 4.61\%$. The cross-correlation coefficient of spanwise vorticity obtained by the PINN model PINN–t3–s16–c5 is 0.98, while PINN–t5–s16–c5 yields 0.94. In comparison, the noisy

### Table 4. Relative Euclidean norm of errors averaged over the whole domain and correlation coefficient for the minimal turbulent channel.

| Test          | Noise level | $\epsilon_u$ | $\epsilon_v$ | $\epsilon_w$ | $r_u$   | $r_v$   | $r_w$   |
|---------------|-------------|---------------|---------------|---------------|---------|---------|---------|
| PINN–t5–s16  | clean       | 0.53%         | 12.54%        | 9.89%         | 0.999   | 0.991   | 0.995   |
| PINN–t3–s16  | clean       | 0.81%         | 17.57%        | 20.75%        | 0.999   | 0.9807  | 0.9871  |
| PINN–t5–s8   | clean       | 1.87%         | 22.86%        | 44.15%        | 0.999   | 0.9608  | 0.9039  |
| PINN–t3–s8   | clean       | 2.64%         | 46.72%        | 56.60%        | 0.998   | 0.8655  | 0.8563  |
| PINN–t5–s16–c5 | 5.0%                | 5.0%           | 4.40%          | 36.74%        | 35.11%  | 0.9959  | 0.8792  | 0.9430  |

**Figure 7.** The evolution of the velocity components in time for a point at $(x, y) = (0.0, 0.36)$ (marked by the red square) obtained from the PINN–t3–s16 and PINN–t3–s8 models in comparison with the reference data. Contour represents the instantaneous spanwise velocity at $t = 0.0$. 

**Figure 8.** The effect of noise on the accuracy of the reconstructions is also assessed and the results are reported in table 4. We observe that very good results can be achieved, even in the presence of 5% noise, from the PINN–t5–s16 and PINN–t3–s16 models. For the PINN–t5–s16 model trained on the noisy data, the highest error is obtained for the wall-normal velocity and it is equal to $\epsilon_w = 21.78\%$, while the lowest error is again obtained for the streamwise velocity, and it is equal to $\epsilon_u = 4.61\%$. The cross-correlation coefficient of spanwise vorticity obtained by the PINN model PINN–t3–s16–c5 is 0.98, while PINN–t5–s16–c5 yields 0.94. In comparison, the noisy
reference data yields $r_{\omega_z} = 0.81$, indicating the promising performance of PINNs in removing noise from the reference data.

4. Application to experimental dataset

In this section we will discuss the application of PINNs to an experimental dataset consisting of turbulence measurements conducted by hot-wire anemometry (Bayoán Cal et al 2010). This dataset includes measurements of the mean and the Reynolds stresses in a rough turbulent boundary layer which constitutes the inflow for an array of wind turbines. We will introduce two different synthetic errors in this data, namely low resolution in the wall-normal direction and noisy measurements, and we will assess the capabilities of the PINNs approach to correct them. The interest of this particular dataset lies in the fact that it lacks information regarding the mean wall-normal velocity $V$ and pressure $P$, a fact that might complicate the predictions. The particular PINNs framework employed here is described in figure 11, where it can be observed that we employed the 2D RANS equations without viscous terms. This is in agreement with the original experimental study (Bayoán Cal et al 2010), where it is claimed that in this flow the most relevant physics takes place far from the wall.

It is worth noting that the profiles are only measured along the wall-normal direction and we use the data at $x = 3m$ (Bayoán Cal et al 2010) as the input. Since all the conservation laws need to be satisfied, some of the streamwise derivatives are computed from the available wall-normal ones. For
instance, in the continuity equation, we enforce $\frac{\partial U}{\partial x} = -\frac{\partial V}{\partial y}$ to obtain $\frac{\partial U}{\partial x}$ with adequate accuracy. The results for both synthetic errors are presented next.

4.1. Improve resolution

We begin by testing the capabilities of PINNs to enhance resolution in a one-dimensional profile by removing 6, 8, and 10 of the 13 points in the wall-normal direction of the initial profiles, respectively. To evaluate how the proposed PINN approach compares to the state-of-the-art methods, we employ spline interpolation and a neural network (NN) using the same architecture as PINNs to enhance resolution. As illustrated in figure 12, the retained 7, 5, and 3 points (marked as circles) will be used for the supervised part of the training, while the unsupervised part will involve minimizing the residual of the two-dimensional RANS equations (without viscosity) on 50 randomly-sampled points. A hyper-parameter study of the PINNs architecture is presented in A.3, where the optimal architecture is reported. Table 5 summarizes the relative errors produced by the three considered methods using different numbers of reference points for enhancing resolution. As observed in figure 12, when retaining 7 and 5 points, the newly-added points (marked as triangles) are in excellent agreement with the reference profile (in black). The averaged relative Euclidean norm of error on added data are $\epsilon$ of 2.16% and 3.76% as retaining 7 and 5 points, respectively, highlighting the potential of PINNs for super-resolution tasks. Our results demonstrate that the proposed PINNs achieve the lowest average error $\tau$ among the considered methods in all cases, highlighting the good performance of PINNs for super-resolution of turbulent measurements. It is worth noting that even when only using 3 points for training, the average error obtained by PINNs is still around 10.98%, whereas for spline interpolation, it rises to 17.84%, demonstrating the robustness of our proposed methods.

4.1.1. Discussion of super-resolution results.

Our results indicate that PINNs can effectively improve the resolution of the original velocity profiles, achieving average relative errors of 1.61%, 2.99%, and 10.98% using 7, 5, and 3, respectively. Figure 13 shows the averaged relative Euclidean norm of the errors as a function of the spacing between retained samples normalized by the boundary layer thickness ($\delta_{99}$). It can be observed that the error drastically rises with increasing spacing between retained points, while the error obtained by PINNs exhibits the lowest variation with respect to the spacing between retaining points compared with spline interpolation and NNs. It is also interesting to note that when we used 7 reference points, the vanilla NNs perform worse than spline interpolation, highlighting the advantages of incorporating physical information into neural networks.
Figure 11. Schematic representation of the PINNs approach employed here. The inputs are the coordinates, and the neural network predicts the various profiles. These profiles are used, through automatic differentiation, to construct the 2D RANS equations without viscous terms, yielding the unsupervised loss. Furthermore, some of the outputs of the network are compared with the known reference values, yielding the supervised loss. These two are combined to obtain the total loss, and the training process is continued. Note that we denote the variables without reference data by dotted lines.

Table 5. Relative Euclidean norm of errors using different number of retained reference points for enhancing the resolution. The boldface denotes the lowest averaged error among the obtained results in each case.

| Methods | reference points | $\epsilon_U$ (%) | $\epsilon_{\overline{u}^2}$ (%) | $\epsilon_{\overline{v}^2}$ (%) | $\epsilon_{\overline{u}\overline{v}}$ (%) | $\tau$ (%) |
|---------|------------------|------------------|------------------|------------------|------------------|-----------|
| PINNs   | 7                | 1.46             | 0.84             | 1.16             | 2.98             | 1.61      |
| NNs     | 7                | 3.07             | 2.72             | 1.19             | 12.46            | 4.86      |
| Spline  | 7                | 2.38             | 0.99             | 0.74             | 3.90             | 2.00      |
| PINNs   | 5                | 3.73             | 1.54             | 2.65             | 4.06             | 2.99      |
| NNs     | 5                | 5.26             | 4.38             | 2.65             | 7.65             | 4.98      |
| Spline  | 5                | 4.40             | 2.72             | 1.31             | 8.66             | 4.27      |
| PINNs   | 3                | 7.29             | 11.86            | 16.25            | 8.55             | 10.98     |
| NNs     | 3                | 10.38            | 5.74             | 39.79            | 11.10            | 16.75     |
| Spline  | 3                | 8.27             | 9.96             | 17.56            | 35.55            | 17.84     |

As depicted in figure 12, the most challenging quantity to predict is the Reynolds shear stress. This difficulty can be attributed to the inherent complexities involved in its measurement, particularly in the near-wall region. Additionally, the relative error in predicting $\overline{u}\overline{v}$ is significantly higher compared with other quantities, primarily due to its lower absolute values, rendering it highly sensitive to small variations. Note that the error incurred through spline interpolation is more than double and quadruple that of PINNs when retaining 5 and 3 points, respectively. This observation underscores the potential of PINNs-based approaches in enhancing the resolution of experimental measurements.

4.2. De-noising

In this section, we also begin with clean hot-wire measurements and then introduce Gaussian noise to simulate potential dispersion errors. We utilize the $3\sigma$ criterion to adjust the standard deviation $\sigma$ in the Gaussian distribution, controlling the added noise on the profiles in terms of noise level, as
Figure 12. Super-resolution results on (from left to right) mean flow, streamwise fluctuations, wall-normal fluctuations and Reynolds shear stress. (Black) original profile, (red) results obtained by PINNs, (yellow) results obtained by spline interpolation and (green) results obtained by NNs. Note that we denote the remaining points after reducing resolution as circle and added points as triangle markers, respectively. From the top to the bottom we show the results obtained retaining 7, 5 and 3 reference points, respectively.

Figure 13. Averaged relative Euclidean norm of the errors as a function of the averaged spacing between retained samples ($\Delta y$) normalized by the boundary layer thickness ($\delta_{99}$).

previously demonstrated in the literature (Vinuesa et al 2014). For each scaled quantity $\phi^+$ in the profile, $\sigma$ is calculated as:

$$\sigma = \frac{N[\%]}{100} \cdot \frac{\phi^+}{3},$$

(11)

where $N$ represents the noise level to be added to the profile. It is important to note that we use a friction velocity $u_\tau$ of 0.48 m s$^{-1}$ as provided in the original database (Bayoán Cal et al 2010) to scale the quantities $\phi$ as $\phi^+$, following the logarithmic law of the wall. We consider three noise level cases: 2%, 5%, and 8%. As illustrated by the green lines in figure 14, it can be observed that the synthetic profile becomes noisier when the quantity exhibits higher values. We will employ the same architecture as that discussed in section 4.1, which was selected following the hyper-parameter study detailed in A.3, where we first search for the optimal architectures and subsequently tune the respective weights of the supervised and unsupervised loss. It is worth noting that we adopt the same architecture used in the super-resolution task as it performs overall best on both tasks. Additionally, we utilize weighted least-square regression (Selesnick 2013) as a conventional de-noising method for comparison purposes. Denoting $y \in \mathbb{R}^n$ as the noisy input and $\hat{y} \in \mathbb{R}^n$ as the approximation of the clean output, the problem can be formulated as:

$$\min_{\hat{y}} \| y - \hat{y} \|_2^2 + \lambda \| D\hat{y} \|_2^2,$$

(12)

where $D\hat{y}$ represents the second-order differentiation of $\hat{y}$, and $\lambda$ is a hyper-parameter controlling the weight of $\hat{y}$ for denoising the noisy profile. This formula yields the expression
Table 6. Relative Euclidean norm of errors with respect to clean profile obtained at noise level (N) for 2%, 5% and 8%. The boldface denotes the lowest averaged error obtained in each case.

|       | N (%) | \(\epsilon_U\) (%) | \(\epsilon_u^2\) (%) | \(\epsilon_v^2\) (%) | \(\epsilon_{uv}\) (%) | \(\tau\) (%) |
|-------|-------|---------------------|----------------------|---------------------|---------------------|---------|
| Noisy profile | 2     | 1.63                | 2.19                 | 3.13                | 2.39                | 2.34    |
| PINNs   | 2     | 1.39                | 1.40                 | 2.33                | 3.56                | **2.17**|
| Least-square | 2     | 3.52                | 1.77                 | 3.85                | 4.92                | 3.52    |
| Noisy profile | 5     | 4.09                | 5.46                 | 7.83                | 5.97                | 5.84    |
| PINNs   | 5     | 3.39                | 3.59                 | 5.19                | 8.12                | **5.07**|
| Least-square | 5     | 5.08                | 3.09                 | 6.29                | 7.81                | 5.57    |
| Noisy profile | 8     | 6.55                | 8.75                 | 12.62               | 9.55                | 9.34    |
| PINNs   | 8     | 4.96                | 5.46                 | 8.16                | 9.40                | **6.99**|
| Least-square | 8     | 6.01                | 4.14                 | 8.41                | 10.10               | 7.17    |

Figure 14. De-noising results on (from left to right) mean flow, streamwise fluctuations, wall-normal fluctuations and Reynolds shear stress. (Black) original profile, (green) profile with superimposed synthetic noise, (red) profile after reducing the noise via PINNs and (blue) profile after reducing noise via least-square regression. From the top to the bottom we show results for the noise level of 2%, 5% and 8%, respectively.

Table 6 summarizes the relative Euclidean norm of error obtained with respect to the clean profiles for the proposed noise levels. PINNs yield an average error \(\epsilon\) of 2.17%, 5.07%, and 6.99% for noise levels of 2%, 5%, and 8%, respectively. Our results indicate that in all cases, the noise is reduced, leading to a better agreement with the clean original data compared to the weighted least-square regression, which yields average errors of 3.52%, 5.57%, and 7.17% for noise levels of 2%, 5%, and 8%, respectively.

4.2.1. Discussion of de-noising results. As depicted in figure 14, PINNs reduce the relative error with respect to the clean profile in all the cases. Particularly, for the \(N = 2\%\)
case, the Gaussian noise added synthetically is 2.34%, and the PINNs correction leads to an average relative error of 2.17%. As the noise level increases to 5%, the initial synthetic error of 5.84% is reduced to 5.07%. When the noise level rises to 8%, the initial synthetic error is reduced from 9.34% to 6.99%, achieving a more significant noise reduction on the profile. It’s noteworthy that the largest errors are observed close to the boundary-layer edge. This might be attributed to the smoother nature of the PINNs-based profile compared to the irregular noisy profile. Given the trend induced by the noisy points, the resulting curve leads to slightly higher errors closer to the freestream. Comparing with the results obtained by weighted least-square regression, it is evident that the profiles obtained by regression exhibit better agreement in the region closer to the freestream. However, it is important to note that the streamwise mean velocity and wall-normal fluctuation profiles denoised by weighted least-square regression demonstrate a distinct shift at the wall in all proposed cases, damaging the near-wall information. This observation implies that our PINNs method exhibits a more robust performance at the near-wall region than the linear method, which is particularly useful for turbulent boundary layer measurement.

5. Summary and conclusions

Applications of physics-informed DL have been demonstrated for data augmentation in experimental fluid mechanics, where the measurement data are generally imperfect, noisy and have low resolution. We employ PINNs for super-resolution of flow-field data both in time and space from a limited set of noisy measurements without having any high-resolution targets. Our objective is to solve the governing PDEs for an ill-posed problem, i.e. unknown initial and boundary conditions, only having a few low-resolution and noisy measurements. To this end, we use a neural network to obtain a continuous solution of the problem, providing a physically-consistent prediction at any point in the solution domain. In particular, we learn a functional mapping parameterized by the neural network from spatial (x) and temporal (t) coordinates to the flow parameters, i.e. velocity components and the pressure. We first demonstrate the applicability of PINNs to the super-resolution tasks in time using the canonical case of Burgers’ equation. We only take the data at the first and the last time-steps of the solution domain for supervised learning. This can be seen as an extensive super-resolution task in time using only two snapshots of the data for training. Our results show that the PINN model exhibits a more robust performance at the near-wall region than the linear method, which is particularly useful for turbulent boundary layer measurement.

The data that support the findings of this study are openly available at the following URL/DOI: https://github.com/KTH-FlowAI/Physics-informed-deep-learning-applications-to-experimental-fluid-mechanics.git.

Acknowledgments

R V acknowledges the financial support by the Göran Gustafsson foundation and ERC Grant No. ‘2021-CoG-101043998, DEEPCONTROL’. Part of the analysis was carried out using computational resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS).

Appendix

A.1. Fine-tuning for two-dimensional vortex shedding behind a circular cylinder

In this section we report the hyper-parameter study conducted to obtain the architecture employed across section 3.3. Note that these results correspond to the case with a noise level of 5%. We will consider variations of the number of hidden layers l, neurons per layer n, coefficients for the supervised (α) and unsupervised (β) losses in equation (5), relative errors of
Table 7. Hyper-parameter study for two-dimensional vortex shedding behind a circular cylinder fixing $\alpha = 1$ and $\beta = 10$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_v$ (%) | $\epsilon_p$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|--------|-----------------|-----------------|-----------------|-----------|------------|
| 4   | 20  | 1        | 10     | 3.96            | 8.66            | 19.06           | 414.06    | 10.56      |
| 4   | 60  | 1        | 10     | 6.29            | 11.77           | 61.76           | 595.10    | 26.61      |
| 4   | 100 | 1        | 10     | 7.25            | 14.16           | 97.69           | 1082.40   | 39.70      |
| 6   | 20  | 1        | 10     | 4.47            | 8.54            | 27.07           | 311.43    | 13.36      |
| 6   | 60  | 1        | 10     | 6.04            | 11.54           | 48.26           | 1070.28   | 21.95      |
| 6   | 100 | 1        | 10     | 7.45            | 15.57           | 55.34           | 1948.66   | 26.12      |
| 10  | 20  | 1        | 10     | 13.28           | 54.81           | 56.34           | 35.82     | 41.48      |
| 10  | 60  | 1        | 10     | 5.76            | 15.74           | 23.44           | 31.19     | 26.12      |
| 10  | 100 | 1        | 10     | 8.77            | 16.44           | 48.44           | 4261.96   | 24.55      |

Table 8. Hyper-parameter study for two-dimensional vortex shedding behind a circular fixing $l = 4$ and $n = 20$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_v$ (%) | $\epsilon_p$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|--------|-----------------|-----------------|-----------------|-----------|------------|
| 4   | 20  | 1        | 1      | 4.27            | 11.09           | 21.62           | 167.3     | 12.33      |
| 4   | 20  | 1        | 5      | 4.35            | 8.52            | 21.7            | 151.7     | 11.52      |
| 4   | 20  | 1        | 10     | 3.96            | 8.66            | 19.06           | 414.06    | 10.56      |
| 4   | 20  | 5        | 1      | 5.94            | 12.36           | 21.67           | 224.39    | 13.32      |
| 4   | 20  | 5        | 5      | 4.59            | 9.67            | 24.94           | 131.02    | 13.07      |
| 4   | 20  | 5        | 10     | 4.27            | 9.58            | 20.55           | 152.62    | 11.47      |
| 4   | 20  | 10       | 1      | 5.54            | 13.48           | 18.93           | 164.99    | 12.65      |
| 4   | 20  | 10       | 5      | 4.90            | 10.96           | 18.62           | 158.7     | 11.49      |
| 4   | 20  | 10       | 10     | 4.52            | 8.81            | 23.45           | 101.97    | 12.26      |

Table 9. Hyper-parameter study for turbulent minimal channel PINN–t5–s8 fixing $\alpha = 1$ and $\beta = 1$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_v$ (%) | $\epsilon_w$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|--------|-----------------|-----------------|-----------------|-----------|------------|
| 4   | 20  | 1        | 1      | 3.14            | 64.45           | 53.16           | 231.48    | 40.25      |
| 4   | 60  | 1        | 1      | 2.32            | 32.22           | 45.02           | 1828.71   | 26.52      |
| 4   | 100 | 1        | 1      | 2.05            | 30.41           | 45.07           | 2914.71   | 25.84      |
| 6   | 20  | 1        | 1      | 2.35            | 42.86           | 47.83           | 771.99    | 31.01      |
| 6   | 60  | 1        | 1      | 1.78            | 28.3            | 43.07           | 2690.33   | 24.38      |
| 6   | 100 | 1        | 1      | 1.65            | 26.46           | 42.29           | 5267.81   | 23.47      |
| 10  | 20  | 1        | 1      | 1.97            | 33.0            | 44.34           | 954.06    | 26.43      |
| 10  | 60  | 1        | 1      | 1.56            | 26.21           | 44.6            | 4938.71   | 24.12      |
| 10  | 100 | 1        | 1      | 1.87            | 22.86           | 44.15           | 4817.92   | 22.95      |

all the statistics $\epsilon$, average relative error $\tau$ and computational time required to train the model $t_c$. Note that $t_c$ is evaluated on a single central processing unit (CPU) from a machine with the following characteristics: AMD Ryzen 9 7950X with 16 cores, 32 threads, 4.5 Ghz and 62 GB RAM. In table 7 we show the first hyper-parameter study, where we change the values of $l$ and $n$ keeping $\alpha$ and $\beta$ constant. Based on this first study, we find that $l = 4$ and $n = 20$ yield the lowest average relative error, and proceed with the second study shown in table 8, where we keep the previous values of $l$ and $n$ and systematically change the coefficients $\alpha$ and $\beta$. The final configuration has $\alpha = 1$, $\beta = 10$, and will be employed for this flow case.

A.2. Fine-tuning for minimal turbulent channel

In this section we report the hyper-parameter study conducted to obtain the architecture employed across section 3.4. Note that these results correspond to case PINN–t5–s8. We conducted our hyper-parameter studies following the same procedure and utilizing the same machine for training as in the previous section. In table 9 we show the first hyper-parameter study, where we change the values of $l$ and $n$ keeping $\alpha$ and $\beta$ constant. Based on this first study, we find that $l = 10$ and $n = 100$ yield the lowest average relative error, and proceed with the second study shown in table 10, where we keep the previous values of $l$ and $n$ and systematically change the coefficients $\alpha$ and $\beta$. The final configuration has $\alpha = 1$, $\beta = 1$, and will be employed for the minimal turbulent channel.

A.3. Fine-tuning for experimental application

In this section we will report the hyper-parameter study conducted to obtain the architecture employed across section 4.
Table 10. Hyper-parameter study for turbulent minimal channel PINN–t5–s8 fixing $l = 10$ and $n = 100$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_v$ (%) | $\epsilon_w$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|----------|------------------|------------------|------------------|--------|---------|
| 100 | 10  | 1        | 1        | 1.87             | 22.86            | 44.15            | 4817.92| 22.95   |
| 100 | 10  | 1        | 5        | 1.56             | 27.82            | 41.76            | 5196.15| 23.71   |
| 100 | 10  | 1        | 10       | 1.68             | 28.29            | 48.7             | 5073.0 | 26.22   |
| 100 | 10  | 5        | 1        | 1.58             | 26.59            | 41.18            | 5781.06| 23.12   |
| 100 | 10  | 5        | 5        | 1.69             | 27.16            | 45.44            | 6480.06| 24.76   |
| 100 | 10  | 10       | 1        | 1.67             | 27.22            | 46.18            | 5187.21| 25.02   |
| 100 | 10  | 10       | 5        | 1.8              | 27.53            | 40.86            | 6684.72| 23.4    |
| 100 | 10  | 10       | 10       | 1.78             | 27.42            | 44.14            | 5803.82| 24.45   |

Table 11. Hyper-parameter study for application to experimental dataset fixing $\alpha = 10$ and $\beta = 1$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_v$ (%) | $\epsilon_w$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|----------|------------------|------------------|------------------|--------|---------|
| 2   | 40  | 10       | 1        | 1.57             | 0.76             | 1.23             | 3.69   | 6.17    | 1.81    |
| 2   | 60  | 10       | 1        | 1.64             | 0.97             | 2.19             | 5.45   | 6.12    | 2.56    |
| 2   | 80  | 10       | 1        | 1.49             | 0.76             | 1.94             | 3.01   | 6.31    | 1.80    |
| 2   | 100 | 10       | 1        | 2.11             | 0.78             | 0.94             | 3.56   | 8.42    | 1.85    |
| 4   | 40  | 10       | 1        | 1.46             | 0.84             | 1.16             | 2.97   | 7.41    | 1.61    |
| 4   | 60  | 10       | 1        | 1.14             | 1.68             | 1.46             | 5.16   | 9.79    | 2.36    |
| 4   | 80  | 10       | 1        | 1.49             | 0.86             | 1.81             | 3.69   | 9.40    | 1.96    |
| 4   | 100 | 10       | 1        | 1.70             | 0.85             | 2.28             | 3.28   | 11.99   | 2.03    |
| 6   | 40  | 10       | 1        | 1.19             | 0.84             | 1.24             | 4.07   | 10.38   | 1.83    |
| 6   | 60  | 10       | 1        | 1.23             | 1.13             | 1.23             | 3.43   | 13.71   | 1.76    |
| 6   | 80  | 10       | 1        | 1.50             | 0.88             | 0.93             | 3.28   | 15.49   | 1.65    |
| 6   | 100 | 10       | 1        | 1.61             | 0.93             | 0.84             | 3.38   | 18.62   | 1.69    |
| 8   | 40  | 10       | 1        | 1.53             | 1.03             | 1.27             | 3.71   | 14.47   | 1.88    |
| 8   | 60  | 10       | 1        | 1.50             | 0.96             | 1.01             | 5.35   | 18.28   | 2.21    |
| 8   | 80  | 10       | 1        | 1.80             | 0.95             | 2.34             | 3.51   | 20.84   | 2.15    |
| 8   | 100 | 10       | 1        | 1.96             | 1.43             | 0.90             | 7.67   | 23.8    | 2.99    |

Note that these results correspond to the study of super-resolution, but the same architecture is used for de-noising. We conducted our hyper-parameter studies following the same procedure and utilizing the same machine for training as in the previous section. In table 11 we show the first hyper-parameter study, where we change the values of $l$ and $n$ keeping $\alpha$ and $\beta$ constant. Based on this first study, we find that $l = 4$ and $n = 40$ yield the lowest average relative error, and proceed with the second study shown in table 12, where we keep the previous values of $l$ and $n$ and systematically change the coefficients $\alpha$ and $\beta$. The final configuration has $\alpha = 10$, $\beta = 1$, and will be employed for both super-resolution and de-noising tasks.

A.4. Discussion on hyper-parameter selection and generalizability for future applications

In this section, we discuss some guidelines for adopting the employed architecture and extending our conclusions for future applications. Since we considered several different tasks and flow cases, the employed architectures should always be fine-tuned according to the particular study to be conducted based on hyperparameter studies. One should begin by fine-tuning the architecture to achieve convergence of the loss and a lower evaluation error if possible. Subsequently, one can further fine-tune the weights of the supervised and unsupervised losses for the optimal architecture to enhance the model’s performance. Note that defining the weights of the losses before fine-tuning is possible since they usually depend on the extent to which the results rely on the input data. If there is no reference data available for comparison, which is a common scenario in many experimental applications, the convergence of the total loss $L$ might serve as a metric to evaluate the reliability of the results. Figure 15 depicts the evolution of the loss when fixing different parameters. It can be observed that the total losses converge to similar magnitudes for different models. Given that a certain degree of deviation in the averaged relative error reported in tables 11 and 12 between two individual models is acceptable, the magnitude of the loss at the convergence stage can be considered a reliable metric for performance evaluation.

Furthermore, it is worth noting that the complexity of the adopted architecture is not necessarily proportional to the scale of the data but rather depends on the degrees of freedom of the unknowns. In the present study, the employed model for two-dimensional vortex shedding has three outputs ($u$, $v$ and $p$), while in experimental applications, there are six outputs.
Table 12. Hyper-parameter study for application to experimental dataset fixing $l = 4$ and $n = 40$. Boldface denotes the architecture with the lowest average relative error.

| $l$ | $n$ | $\alpha$ | $\beta$ | $\epsilon_u$ (%) | $\epsilon_{uu}$ (%) | $\epsilon_{vv}$ (%) | $\epsilon_{uv}$ (%) | $t_c$ (s) | $\tau$ (%) |
|-----|-----|----------|---------|-----------------|-----------------|-----------------|-----------------|--------|----------|
| 4   | 40  | 1        | 1       | 1.36            | 1.00            | 1.05            | 3.78            | 9.01   | 1.95     |
| 4   | 40  | 1        | 2       | 1.13            | 1.13            | 1.01            | 3.92            | 8.50   | 1.80     |
| 4   | 40  | 1        | 5       | 1.24            | 0.84            | 0.89            | 3.61            | 9.71   | 1.64     |
| 4   | 40  | 1        | 10      | 1.60            | 0.87            | 1.24            | 4.17            | 8.43   | 1.97     |
| 4   | 40  | 1        | 20      | 1.69            | 0.97            | 1.16            | 3.58            | 8.43   | 1.85     |
| 4   | 40  | 2        | 1       | 1.53            | 1.00            | 1.51            | 3.85            | 7.32   | 1.97     |
| 4   | 40  | 2        | 2       | 1.23            | 0.85            | 1.36            | 3.59            | 8.11   | 1.76     |
| 4   | 40  | 2        | 5       | 1.59            | 0.93            | 0.91            | 3.64            | 8.54   | 1.77     |
| 4   | 40  | 2        | 10      | 1.04            | 0.79            | 0.79            | 3.62            | 10.29  | 1.56     |
| 4   | 40  | 2        | 20      | 1.43            | 0.86            | 0.91            | 4.08            | 8.61   | 1.82     |
| 4   | 40  | 5        | 1       | 1.47            | 0.90            | 1.11            | 4.14            | 7.94   | 1.91     |
| 4   | 40  | 5        | 2       | 1.51            | 0.98            | 1.28            | 3.27            | 9.11   | 1.76     |
| 4   | 40  | 5        | 5       | 1.46            | 0.77            | 0.95            | 3.46            | 7.47   | 1.66     |
| 4   | 40  | 5        | 10      | 1.38            | 1.06            | 1.46            | 3.45            | 8.28   | 1.84     |
| 4   | 40  | 5        | 20      | 1.22            | 0.83            | 1.40            | 3.21            | 9.13   | 1.67     |
| 4   | 40  | 10       | 1       | 1.46            | 0.84            | 1.16            | 2.97            | 7.41   | 1.61     |
| 4   | 40  | 10       | 2       | 1.71            | 0.91            | 0.86            | 3.41            | 7.74   | 1.72     |
| 4   | 40  | 10       | 5       | 1.66            | 0.84            | 1.17            | 3.01            | 6.92   | 1.67     |

Figure 15. The convergence of the total loss ($L$) defined by equation (5) for hyper-parameter studies in experimental application. We demonstrate the loss evolution of models fixing number of layers $l$ (upper left), number of neurons $n$ (upper right), weight of supervised-learning loss $\alpha$ (lower left) and weight of unsupervised-learning loss $\beta$ (lower right). Note that the black dashed line denotes the loss evolution obtained by the adopted model and the blue region denotes the epochs using the Adam optimizer.

($u$, $v$, $uu$, $vv$, $uv$ and $p$). This is twice the number of outputs compared to the previous one, although the two-dimensional flow case is more complex. Therefore, even though both models comprise four layers, the architecture for experimental applications adopts 40 neurons ($n$) for each layer, while 20 neurons are used for the vortex-shedding case in each layer.
References

Abadi M et al 2016 Proc. 12th USENIX Symp. on Operating Systems Design and Implementation (OSDI’16) vol 16 pp 265–83

Adamczyk A A and Rimai L 1988 2-dimensional particle tracking velocimetry (PTV): technique and image processing algorithms Exp. Fluids 6 573–80

Adrian R J 1984 Scattering particle characteristics and their effect on pulsed laser measurements of fluid flow: speckle velocimetry vs particle image velocimetry Appl. Opt. 23 1690–1

Akbari G and Montazerin N 2021 Reconstruction of particle image velocimetry data using flow-based features and validation index: a machine learning approach Meas. Sci. Technol. 33 015203

Arzani A, Wang J-X and D’Souza R M 2021 Uncovering near-wall blood flow from sparse data with physics-informed neural networks Phys. Fluids 33 071905

Baydin A G, Pearlmutter B A, Radul A A and Siskind J M 2018 Automatic differentiation in 20 years: from ADiGit to ADAM Int. J. Comput. Math. 95 1657–64

Bayoán Cal R, Lebrón J, Castillo L, Kang H S and Menéveu C 2010 Experimental study of the horizontally averaged flow structure in a model wind-turbine array boundary layer J. Renew. Sustain. Energy 2 013106

Borello G, Guastoni L, Eivazi H, Schlatter P and Vinuesa R 2222 Predicting the temporal dynamics of turbulent channels through deep learning Int. J. Heat Fluid Flow 96 109010

Brunton S L, Noack B R and Koumoutsakos P 2020 Machine learning for fluid mechanics Annu. Rev. Fluid Mech. 52 477–508

Bui-Thanh T, Damodaran M and Willcox K 2003 21st AIAA Applied Aerodynamics Conf. p 4213

Bui-Thanh T, Damodaran M and Willcox K 2004 Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition AIAA J. 42 1505–16

Cai S, Liang J, Gao Q, Xu C and Wei R 2020 Particle image velocimetry based on a deep learning motion estimator IEEE Trans. Instrum. Meas. 69 3538–54

Cai S, Mao Z, Wang Z, Yin M and Karniadakis G E 2021 Physics-informed neural networks (PINNs) for fluid mechanics: a review Acta Mech. Sin. 37 1727–38

Cai S, Wang Z, Fuest F, Jeon Y J, Gray C and Karniadakis G E 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. 915 A102

Chandramouli P, Memin E, Heitz D and Fiabane L 2019 Fast 3D flow reconstructions from 2D cross-plane observations Exp. Fluids 60 30

Durasamy K, Laccarinio G and Xiao H 2019 Turbulence modeling in the age of data Annu. Rev. Fluid Mech. 51 357–77

Eivazi H, Guastoni L, Schlatter P, Azizpour H and Vinuesa R 2021 Recurrent neural networks and Koopman-based frameworks for temporal predictions in a low-order model of turbulence Int. J. Heat Fluid Flow 90 108816

Eivazi H, Tahani M, Schlatter P and Vinuesa R 2022 Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations Phys. Fluids 34 075117

Everson R and Sirovich L 1995 Karhunen–Loève procedure for Gappy data J. Opt. Soc. Am. A 12 1657–64

Fathi M F, Perez-Rayna I, Baghaie A, Berg P, Janiga G, Arzani A and D’Souza R M 2020 Super-resolution and denoising of 4D-Flow MRI using physics-informed deep neural nets Comput. Methods Programs Biomed. 197 105729

Ferdian E, Suiiesiaputra A, Dubowitiz D J, Zhao D, Wang A, Cowan B and Young A A 2020 4DFlowNet: super-resolution 4D flow MRI using deep learning and computational fluid dynamics Front. Phys. 8 138

Fukami K, Fukagata K and Taira K 2019 Super-resolution reconstruction of turbulent flows with machine learning J. Fluid Mech. 870 106–20

Gao H, Sun L and Wang J-X 2021 Super-resolution and denoising of fluid flow using physics-infomed convolutional neural networks without high-resolution labels Phys. Fluids 33 073603

Grant I and Pan X 1995 An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images Exp. Fluids 19 159–66

Guastoni L, Güemes A, Ianario A, Discetti S, Schlatter P, Azizpour H and Vinuesa R 2021 Convolutional-network models to predict wall-bounded turbulence from wall quantities J. Fluid Mech. 928 A27

Guastoni L, Rahbati J, Schlatter P, Azizpour H and Vinuesa R 2023 Deep reinforcement learning for turbulent drag reduction in channel flows Eur. Phys. J. E 46 27

Güemes A, Discetti S, Ianario A, Sirmacek B, Azizpour H and Vinuesa R 2021 From coarse wall measurements to turbulent velocity fields through deep learning Phys. Fluids 33 075121

Güemes A, Vila C S and Discetti S 2021 14th Int. Symp. on Particle Image Velocimetry

Ham Y-G, Kim J-H and Luo J-J 2019 Deep learning for multi-year ENSO forecasts Nature 573 568–72

Hochreiter S and Schmidhuber J 1997 Long short-term memory Neural Comput. 9 1735–80

Hui T W, Tang X and Loy C Y 2018 2018 IEEE/CVF Conf. on Computer Vision and Pattern Recognition pp 8981–9

Jiang C, Vinuesa R, Chen R, Mi J, Laima S and Li H 2021 An interpretable framework of data-driven turbulence modeling using deep neural networks Phys. Fluids 33 055133

Jiménez J and Moin P 1991 The minimal flow unit in near-wall turbulence J. Fluid Mech. 225 213–40

Jiménez J 2018 Machine-aided turbulence theory J. Fluid Mech. 854 R1

Jin X, Cai S, Li H and Karmiadakis G E 2021 NSfNets (Navier-Stokes flow nets): physics-informed neural networks for the incompressible Navier-Stokes equations J. Comput. Phys. 426 109951

Karniadakis G E, Kevrekidis I G, Lu L, Perdikaris P, Wang S and Yang L 2021 Physics-informed machine learning Nat. Rev. Phys. 3 422–40

Khan I, Ullah H, Al-Salman H, Fiza M, Islam S, Rajja A, Shoaib M and Gumaei A 2021 Falkner–Skan equation with heat transfer: a new stochastic numerical approach Math. Probl. Eng. 2021 3921481

Khan I, Ullah H, Al-Salman H, Fiza M, Islam S, Shoaib M, Rajja A, Gumaei A and Bhalq F 2021 Fractional analysis of MHD boundary layer flow over a stretching sheet in porous medium: a new stochastic method J. Funct. Spaces 2021 5844741

Kim H, Kim J, Won S and Lee C 2021 Unsupervised deep learning for super-resolution reconstruction of turbulence J. Fluid Mech. 910 A29

Kingma D P and Ba J 2017 arXiv:1412.6980

Kutz J N 2017 Deep learning in fluid dynamics J. Fluid Mech. 814 1–4

Larosa F, Hoyas S, García-Martínez J, Conejero J A, Nerini F F and Vinuesa R 2023 Halting generative AI advancements may slow down progress in climate research Nat. Clim. Change 13 497–9

LeCun Y, Bengio Y and Hinton G 2015 Deep learning Nature 521 436–44
Lecun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition Proc. IEEE 86 2278–324

Ling J, Kurzawski A and Templeton J 2016 Reynolds averaged turbulence modelling using deep neural networks with embedded invariance J. Fluid Mech. 807 155–66

Liu B, Tang J, Huang H and Lu X-Y 2020 Deep learning methods for super-resolution reconstruction of turbulent flows Phys. Fluids 32 025105

Liu D C and Nocedal J 1989 On the limited memory BFGS method for large scale optimization Math. Program. 45 503–28

Méndez M A, Raiola M, Masullo A, Discetti S, Ianiro A, Theunissen R and Buchlin J-M 2017 POD-based background removal for particle image velocimetry Exp. Therm. Fluid Sci. 80 181–92

Mianroodi J R, Siboni N H and Raabe D 2021 Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials npj Comput. Mater. 7 99

Morimoto M, Fukami K and Fukagata K 2021 Experimental velocity data estimation for imperfect particle images using machine learning Phys. Fluids 33 087121

Rabault J, Kolaas J and Jensen A 2017 Performing particle image velocimetry using artificial neural networks: a proof-of-concept Meas. Sci. Technol. 28 125301

Rabault J, Kuchta M, Jensen A, Réglade U and Cerardi N 2019 Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control J. Fluid Mech. 865 281–302

Raissi M, Perdikaris P and Karniadakis G 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

Raissi M, Wang Z, Triantafyllopoulos M S and Karniadakis G E 2019 Deep learning of vortex-induced vibrations J. Fluid Mech. 861 119–37

Raissi M, Yazdani A and Karniadakis G E 2020 Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations Science 367 1026–30

Rechenberg I 1964 Annual Conf. WGLR at Berlin vol 35 p 33

Rudin C 2019 Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead Nat. Mach. Intell. 1 206–15

Rumelhart D E and McClelland J L 1987 Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations pp 318–62

Segler M H S, Preuss M and Waller M P 2018 Planning chemical syntheses with deep neural networks and symbolic AI Nature 555 604–10

Selesnick I 2013 Least squares with examples in signal processing Connexions 4 1–25

Sliwinski L and Rigas G 2023 Mean flow reconstruction of unsteady flows using physics-informed neural networks Data-Centric Eng. 4 64

Srinivasan P A, Guastoni L, Azizpour H, Schlatter P and Vinuesa R 2019 Predictions of turbulent shear flows using deep neural networks Phys. Rev. Fluids 4 054603

Stulov N and Chertkov M 2021 arXiv:2101.11950

Tang H, Rabault J, Kuhnle A, Wang Y and Wang T 2020 Robust active flow control over a range of Reynolds numbers using an artificial neural network trained through deep reinforcement learning Phys. Fluids 32 053605

Teo C, Lim K, Hong G and Yeo M 1991 A neural net approach in analysing photographs in PIV IEEE Syst. Man Cybern. 3 1535–8

Vinuesa R, Azizpour H, Leite I, Balaam M, Dignum V, Domisch S, Felländer A, Langhans S D, Tegmark M and Fusio Nerini F 2020 The role of artificial intelligence in achieving the sustainable development goals Nat. Commun. 11 233

Vinuesa R and Brunton S L 2022 Enhancing computational fluid dynamics with machine learning Nat. Comput. Sci. 2 358–66

Vinuesa R, Lehmkuhl O, Lozano-Durán A and Rabault J 2022 Flow control in wings and discovery of novel approaches via deep reinforcement learning Fluids 7 62

Vinuesa R, Schlatter P and Nagib H M 2014 Role of data uncertainties in identifying the logarithmic region of turbulent boundary layers Exp. Fluids 55 1751

Vinuesa R and Sirmacek B 2021 Interpretable deep-learning models to help achieve the sustainable development goals Nat. Mach. Intell. 3 926–926

Waldmann I P and Griffith C A 2019 Mapping Saturn using deep learning Nat. Astron. 3 620–5

Wang Z, Chen J and Hoi S C H 2021 Deep learning for image super-resolution: a survey IEEE Trans. Pattern Anal. Mach. Intell. 43 3365–87

Willcox K 2006 Unsteady flow sensing and estimation via the Gappy proper orthogonal decomposition Comput. fluids 35 208–26

Yang G and Ramanan D 2019 Advances in Neural Information Processing Systems vol 32