Data-Centric Engineering (2023), 4: e4
doi:10.1017/dce.2022.37

RESEARCH ARTICLE

Mean flow reconstruction of unsteady flows using physics-informed neural networks

Lukasz Sliwinski* and Georgios Rigas

Department of Aeronautics, Imperial College London, London SW7 2AZ, United Kingdom
*Corresponding author. E-mail: ls2716@ic.ac.uk

Received: 16 January 2022; **Revised:** 01 September 2022; **Accepted:** 24 October 2022

Keywords: Data assimilation; flow interpolation; physics-informed neural networks

Abstract

Data assimilation of flow measurements is an essential tool for extracting information in fluid dynamics problems. Recent works have shown that the physics-informed neural networks (PINNs) enable the reconstruction of unsteady fluid flows, governed by the Navier–Stokes equations, if the network is given enough flow measurements that are appropriately distributed in time and space. In many practical applications, however, experimental measurements involve only time-averaged quantities or their higher order statistics which are governed by the under-determined Reynolds-averaged Navier–Stokes (RANS) equations. In this study, we perform PINN-based reconstruction of time-averaged quantities of an unsteady flow from sparse velocity data. The applied technique leverages the time-averaged velocity data to infer unknown closure quantities (curl of unsteady RANS forcing), as well as to interpolate the fields from sparse measurements. Furthermore, the method's capabilities are extended further to the assimilation of Reynolds stresses where PINNs successfully interpolate the data to complete the velocity as well as the stresses fields and gain insight into the pressure field of the investigated flow.

Impact Statement

In fluid dynamics, efficient data assimilation can significantly reduce the costs of running wind tunnel experiments and performing numerical simulations. In this study, we have used physics-informed neural networks (PINNs) to leverage synthetic mean velocity measurements in order to reconstruct the time-averaged statistics of the unsteady flow past a circular cylinder. Due to the closure problem, such flows are inherently difficult and expensive to simulate. The network was able to infer missing flow information, interpolate the flow from sparse measurements, as well as denoise artificially corrupted velocity data.

1. Introduction

Data assimilation methods in fluid dynamics leverage available measurement data, in combination with a partially or fully known model, to extract more information about the flows from which the data originated (Foures et al., 2014; Symon et al., 2017; Fukami et al., 2019; Franceschini et al., 2020; Mons and Marquet, 2021). The growth of machine learning techniques over the last decade has brought numerous new

This research article was awarded Open Materials badge for transparent practices. See the Data Availability Statement for details.

methods for analysis and assimilation of flow data (Brunton et al., 2020). Fukami et al. (2019) used convolutional neural networks to super-resolve unsteady laminar and turbulent flows from low resolution images. In other studies (Ling et al., 2016; Beck et al., 2019), the authors employed neural networks to learn closures for turbulence models based on unsteady flow data.

Most recently, PINNs (Raissi et al., 2019; Cai et al., 2021), have brought forth a new paradigm in solving fluid dynamics problems with machine learning. By including the governing partial differential equation (PDE) residuals within the training loss, it is possible to incorporate domain knowledge, that is, the physical constraints of a system, directly into the neural network architecture. Jin et al. (2021) used PINNs as a forward solver for both steady and unsteady flow problems. Constrained by the governing Navier–Stokes equations and well-defined boundary conditions (BCs) describing the flow dynamics, the network was able to infer the flow solution in the computational domain, thus fully substituting for conventional finite-difference or finite-volume numerical methods.

PINNs have had a particularly strong impact on data assimilation problems since the output field reconstructions are explicitly constrained to be physically realizable solutions. Raissi et al. (2019) used PINNs for data-driven PDE discovery, identifying the equation variables and constants based on data generated by an unknown governing system. Raissi et al. (2020) showed that PINNs can leverage space- and time-resolved dye concentration measurements to accurately infer velocity and pressure fields with no prior information about those flow fields. In the study by Fathi et al. (2020), the authors used PINNs to super-resolve and denoise sparse four-dimensional (in space and time) magnetic resonance imaging blood flow measurements.

Both Raissi et al. (2020) and Fathi et al. (2020) performed unsteady flow data assimilation using instantaneous data, that is, each velocity/dye concentration measurement corresponded to a specific time t at which it was taken. However, in many practical applications, we are interested in time-averaged quantities, which are consequently a function of the mean flow field. For example, using Particle Image Velocimetry (PIV), we can obtain time-averaged velocity fields or higher order velocity statistics by averaging randomly sampled or under-resolved in time flow snapshots. The main difficulty associated with the assimilation of mean flow data is that the underlying physical system is governed by the Reynolds-averaged Navier–Stokes (RANS) equations, an under-determined set of equations. While in comparison to standard neural networks, PINNs are more robust to a small amount of input data, they are still subject to mathematical limitations of the governing setup. Thus, care has to be taken to ensure that enough system information and data are available to the networks. Eivazi et al. (2021) have applied PINNs to RANS modeling of turbulent two-dimensional boundary layers. To make the data assimilation problem determinate, in addition to mean velocity measurements on the domain boundaries, the networks are given the second-order velocity statistic—the in-plane component of the Reynolds' stress tensor.

Foures et al. (2014) tackled the problem of the data assimilation of synthetic PIV measurements of an unsteady, averaged 2D cylinder flow using adjoint variable optimization. To overcome the closure problem, the authors used a variant of RANS equations by replacing the unknown Reynolds stress tensor with an unknown forcing vector. Using the synthetic, sparse data, base-flow RANS solution and BCs, the authors were able to recover the solenoidal (divergence free) part of the averaged unsteady forcing and also super-resolve the flow in scenarios where the velocity data was sparse. Symon et al. (2017) applied the adjoint variable optimization approach to assimilate experimental data for a 2D flow past an idealized airfoil with $\mathrm{Re} = 13{,}500$, showing that the method is also applicable to realistic applications at high Reynolds numbers.

The PINN method and the adjoint-based data assimilation method presented in Foures et al. (2014), while applied to the same problem, are fundamentally different. In Foures et al. (2014), the authors used the variational formulation to solve a RANS constrained optimization problem for assimilating partially known velocity information. The RANS equations form a hard constraint to the optimization problem, such that the dynamic equations are satisfied in the entire domain. In the case of PINNs, the RANS equations are weakly enforced through the loss function. Another difference between the PINN approach and the adjoint-based method presented in Foures et al. (2014) is the numerical aspect of implementation. Adjoint variable optimization requires a numerical discretization of the continuous equations on a computational mesh and the optimization steps are performed in a deterministic manner using gradient descent. In contrast, PINN method is a neural network optimization with randomly distributed collocation points (mesh-free) and stochastic gradient descent (SGD) for updating the network weights. While the absence of mesh simplifies the optimization, the PINN approach requires more computational resources.

In this study, we apply PINNs to the reconstruction of averaged quantities of unsteady flows using sparse velocity data. Thanks to the versatility of neural networks, the proposed method will be able to leverage the RANS formulation from Foures et al. (2014) and extend to more variations of the problem. The article is organized as follows: Section 2 presents the methodology of the PINN flow reconstruction including the governing equations. Section 3 describes the implementation of PINNs, as well as the simulations that were performed to generate data. Section 4 presents the results of the study. Specifically, Section 4.1 covers the problem of inferring the average unsteady RANS forcing and Section 4.2 describes the application of PINN optimization to the interpolation of flows using sparse data. Section 5 summarizes the findings and suggests directions for future work.

2. Methodology

Typical experimental flow measurements of unsteady flows involve time-averaged and spatially sparse (coarse-grained and/or partially observed) quantities. Without loss of generality, the method will be demonstrated for a two-dimensional incompressible flow. In this case, the data points contain time-averaged velocities $\overline{\mathbf{u}}(\mathbf{x}) = [\overline{u}(\mathbf{x}), \ \overline{v}(\mathbf{x})]^T$ (first-order statistics) with $\mathbf{x} = [x, y]^T$, and optionally time-averaged Reynolds stresses (second-order statistics)

$$\overline{\mathbf{u}'\mathbf{u}'}(\mathbf{x}) = \begin{bmatrix} \overline{u'u'}(\mathbf{x}) & \overline{u'v'}(\mathbf{x}) \\ \overline{v'u'}(\mathbf{x}) & \overline{v'v'}(\mathbf{x}) \end{bmatrix}. \tag{1}$$

These constitute typical first- and second-order statistics that can be directly obtained from PIV or hotwire anemometry methods in flow experiments. In the above, the unsteady flow field has been decomposed using a standard Reynolds decomposition as

$$\mathbf{u}(\mathbf{x}, t) = \overline{\mathbf{u}(\mathbf{x})} + \mathbf{u}'(\mathbf{x}, t) \tag{2}$$

where $\overline{()}$ denotes time-averaged quantities and $()'$ fluctuating ones, and a similar decomposition holding for the pressure field $p(\mathbf{x}, t)$.

The mean-flow dynamics of an unsteady flow are governed by the RANS equations

$$\begin{aligned} 0 &= \nabla \cdot \overline{\mathbf{u}}, \\ 0 &= \overline{\mathbf{u}} \cdot \nabla \overline{\mathbf{u}} + \nabla \overline{p} - \mathrm{Re}^{-1} \nabla^2 (\overline{\mathbf{u}}) + \nabla \cdot \overline{\mathbf{u}'\mathbf{u}'}, \end{aligned} \tag{3}$$

with appropriate BCs dictated by the geometry and inflow and far-field conditions. The above set of equations is under-determined (three equations and six unknowns in 2D). The last term in the momentum equation, $\nabla \cdot \overline{\mathbf{u}'\mathbf{u}'} \equiv \mathbf{f}$, is the divergence of the symmetric Reynolds stress tensor and its modeling constitutes the closure problem for solving for the mean flow quantities in unsteady laminar and turbulent regimes.

2.1. Problem setup

The data-assimilation framework seeks a model function $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), .., g_R(\mathbf{x})]^T$ which maps the domain coordinates x and y to values of R flow variables $g_1, g_2, .., g_R$ (e.g., u, v velocities). The model function is expressed as a neural network and can be obtained by minimization of the objective function $\mathcal{L}$:

$$\mathcal{L}(\mathbf{g}) = \mathcal{L}_D(\mathbf{g}) + \mathcal{L}_B(\mathbf{g}) + \mathcal{L}_P(\mathbf{g}), \tag{4}$$

where $\mathcal{L}_D$ is the data loss, $\mathcal{L}_B$ is the BCs loss, and $\mathcal{L}_P$ is the physics loss by enforcing the governing equations, in this case the RANS equations. The data loss is defined as mean squared error computed over all points at which values (data) of any of the flow variables are given. Denoting $N_{D,i}$ as the number of data points for the i-th flow variable and denoting a j-th data point for the i-th flow variable as $\mathbf{x}_{i,j}^D = \left[x_{i,j}^D, y_{i,j}^D\right]^T$ with the corresponding value as $g_{i,j}^D$, we can write as

$$\mathcal{L}_D(\mathbf{g}) = \sum_{i=1}^{R}\left[\frac{1}{N_{D,i}}\sum_{j=1}^{N_{D,i}}\left[g_i\left(\mathbf{x}_{i,j}^D\right) - g_{i,j}^D\right]^2\right], \tag{5}$$

where the outer summation is over the flow variables and the inner summation is over the data points. In the later sections, g_i is replaced with symbols of the flow variables. Superscript D indicates values at data points. Otherwise, flow variables symbols indicate quantities computed within the neural network. The physics loss is defined using the residual operator $\mathcal{P}$—a function which takes the model function $\mathbf{g}$ and returns a function which evaluates residuals of the governing PDEs at an input point. The operator not only depends on the governing equations (RANS), but also on the choice of the flow variables expressed by the model function $\mathbf{g}$ and therefore it will be defined in more detail in the following sections. For the physics loss computation, the residuals are evaluated at “collocation” points which are not data points. Denoting the number of collocation points as N_P and an j-th collocation point as $\mathbf{x}_j^P = \left[x_j^P, y_j^P\right]^T$, the physics loss can be written in compact form as

$$\mathcal{L}_P(\mathbf{g}) = \frac{1}{N_P}\sum_{j=1}^{N_P}\|\mathcal{P}(\mathbf{g})|_{\mathbf{x}_j^P}\|_2^2, \tag{6}$$

with the summation performed over the collocation (evaluation) points. Notation $\cdot|_{\mathbf{x}_j^P}$ expresses evaluation of the residual at point $\mathbf{x}_j^P$ and $\|\cdot\|_2$ denotes l_2 norm. The boundary loss is a mixture of data loss and physical type loss, depending if the boundary is specified using data or a BC (e.g., no slip condition on walls).

The above definitions highlight the difference between data points and collocation points. Data points are given by the experiment and they correspond to measurements. Their purpose is to enforce the agreement between the predicted fields and the data points—at these points, during training, the algorithm will compare the PINN predicted values with the data values and use the corresponding loss to constrain the model function $\mathbf{g}(\mathbf{x})$ to the data. The collocation points are set during the flow reconstruction procedure and they can be set irregardless of the data points. The aim of the collocation points is to enforce physical constraints—at these points, during training, the algorithm will evaluate the residuals and use the corresponding loss to constrain the model function $\mathbf{g}(\mathbf{x})$ to be a valid solution to the governing equations.

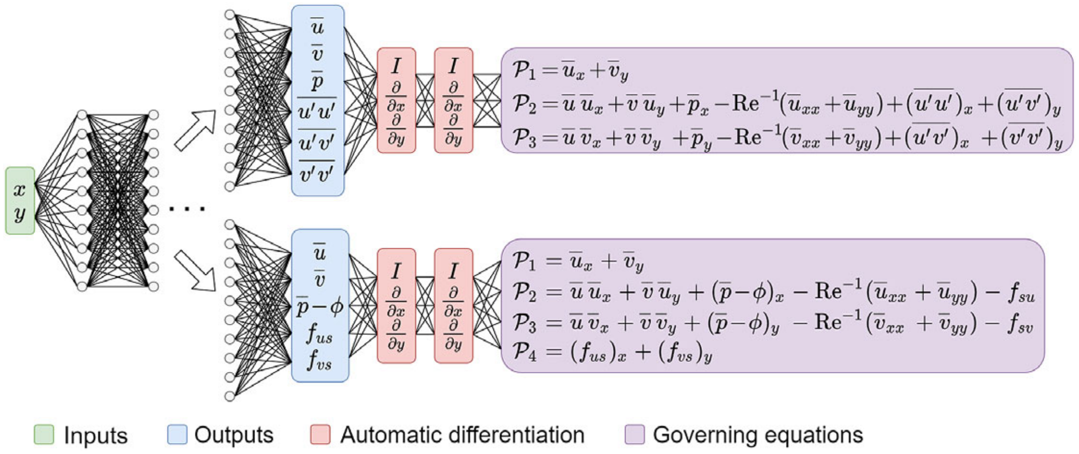
2.1.1. RANS-assimilation with explicit Reynolds stresses

To solve the RANS system using the PINN, the RANS equations are transformed into a system of residuals, defined over a set of domain points. As mentioned in the previous section, we denote the residuals with an operator $\mathcal{P}$ and we can write as

$$\begin{aligned}
\mathcal{P}_1(\mathbf{g}) &= \bar{u}_x + \bar{v}_y, \\
\mathcal{P}_2(\mathbf{g}) &= \bar{u}\bar{u}_x + \bar{v}\bar{u}_y + \bar{p}_x - \mathrm{Re}^{-1}\left(\bar{u}_{xx} + \bar{u}_{yy}\right) + \left(\overline{u'u'}\right)_x + \left(\overline{u'v'}\right)_y, \\
\mathcal{P}_3(\mathbf{g}) &= \bar{u}\bar{v}_x + \bar{v}\bar{v}_y + \bar{p}_y - \mathrm{Re}^{-1}\left(\bar{v}_{xx} + \bar{v}_{yy}\right) + \left(\overline{u'v'}\right)_x + \left(\overline{v'v'}\right)_y,
\end{aligned} \tag{7}$$

where $()_x$ and $()_y$ denote partial derivatives with respect to x and y directions. The neural-network parametrized vector $\mathbf{g}$, which is also the output of the network is

$$\mathbf{g} = \left[\bar{u}, \bar{v}, \bar{p}, \overline{u'u'}, \overline{u'v'}, \overline{v'v'}\right]^T. \tag{8}$$



x
y
$\overline{u}$
$\overline{v}$
$\overline{p}$
$\overline{u'u'}$
$\overline{u'v'}$
$\overline{v'v'}$
I
$\frac{\partial}{\partial x}$
$\frac{\partial}{\partial y}$
$\mathcal{P}_1 = \overline{u}_x + \overline{v}_y$
$\mathcal{P}_2 = \overline{u}\,\overline{u}_x + \overline{v}\,\overline{u}_y + \overline{p}_x - \mathrm{Re}^{-1}(\overline{u}_{xx} + \overline{u}_{yy}) + (\overline{u'u'})_x + (\overline{u'v'})_y$
$\mathcal{P}_3 = \overline{u}\,\overline{v}_x + \overline{v}\,\overline{v}_y + \overline{p}_y - \mathrm{Re}^{-1}(\overline{v}_{xx} + \overline{v}_{yy}) + (\overline{u'v'})_x + (\overline{v'v'})_y$
$\overline{u}$
$\overline{v}$
$\overline{p} - \phi$
f_{us}
f_{vs}
$\mathcal{P}_1 = \overline{u}_x + \overline{v}_y$
$\mathcal{P}_2 = \overline{u}\,\overline{u}_x + \overline{v}\,\overline{u}_y + (\overline{p} - \phi)_x - \mathrm{Re}^{-1}(\overline{u}_{xx} + \overline{u}_{yy}) - f_{su}$
$\mathcal{P}_3 = \overline{u}\,\overline{v}_x + \overline{v}\,\overline{v}_y + (\overline{p} - \phi)_y - \mathrm{Re}^{-1}(\overline{v}_{xx} + \overline{v}_{yy}) - f_{sv}$
$\mathcal{P}_4 = (f_{us})_x + (f_{vs})_y$
🟩 Inputs
🟦 Outputs
🟥 Automatic differentiation
🟪 Governing equations

Figure 1. *Schematics of PINNs for flow reconstruction based on two different approaches. Green box: network inputs (collocation or data point coordinates). Blue box: flow variables modeled by the function* $\mathbf{g}(\mathbf{x})$*, expressed by a neural network. Red box: spatial derivatives of the flow variables obtained using automatic differentiation. Purple box: residuals operator which takes the flow variables and their derivatives to evaluate the physical residuals. The top version of the network utilizes the RANS equations with Reynolds stresses. The bottom version uses RANS with solenoidal forcing* $\mathbf{f}_\mathbf{s}$*. When data points are used as network inputs, the network computation stops at the output layer and the residuals of the governing equations are not computed.*

In the setup above, the output vector $\mathbf{g}$ has a dimension of 6 but there are only three equations. Thus, the system is under-determined. If the true mean velocity field is fully known on a fine/resolved grid, the continuity equation is automatically satisfied, leaving the system with four unknown fields for two equations. The system is still under-determined. Therefore, in the flow reconstruction framework, where velocity data is provided at sparse data points only, the PINN is not guaranteed to converge to a unique solution.

If, in addition to the mean velocity field, the Reynolds stresses are fully known on a fine/resolved grid, the system is determinate and can be solved for the unknown pressure p (with an arbitrary offset). In the flow reconstruction framework, where mean velocities and Reynolds stresses are provided at sparse data points, the system is not fully determined. However, because the idealized system with fully known velocity and Reynolds' stresses fields is determinate, we hypothesize that PINN may be able to converge to a solution that is an accurate representation of the true flow. The assimilation with first- and second-order statistics is utilized in Section 4.2.2. A similar formulation was applied in Eivazi et al. (2021) using time-averaged first- and second-order velocity data at the boundaries of the domain. The top path in Figure 1 presents the PINN formulation corresponding to this approach.

2.1.2. RANS-assimilation with forcing

To alleviate the problem of having an under-determined system, a different form of the RANS equation will also be investigated, as proposed by Foures et al. (2014), where the unknown stress tensor is replaced by an unknown forcing vector in the optimization. This is done by setting $\nabla \cdot \overline{\mathbf{u}'\mathbf{u}'} \equiv \mathbf{f}$ in the governing RANS equations. The forcing $\mathbf{f}$ is decomposed into potential and solenoidal components:

$$\mathbf{f} = \nabla\phi + \mathbf{f}_\mathbf{s}, \tag{9}$$

where

$$\nabla \cdot \mathbf{f}_\mathbf{s} = 0. \tag{10}$$

Then, denoting $\mathbf{f_s} = (f_{us}, f_{vs})^T$, we obtain a new residuals operator $\mathcal{P}$:

$$\begin{aligned}
\mathcal{P}_1(\mathbf{g}) &= \overline{u}_x + \overline{v}_y, \\
\mathcal{P}_2(\mathbf{g}) &= \overline{u}\overline{u}_x + \overline{v}\overline{u}_y + (\overline{p} - \phi)_x - \mathrm{Re}^{-1}\left(\overline{u}_{xx} + \overline{u}_{yy}\right) - f_{us}, \\
\mathcal{P}_3(\mathbf{g}) &= \overline{u}\overline{v}_x + \overline{v}\overline{v}_y + (\overline{p} - \phi)_y - \mathrm{Re}^{-1}\left(\overline{v}_{xx} + \overline{v}_{yy}\right) - f_{vs}, \\
\mathcal{P}_4(\mathbf{g}) &= (f_{us})_x + (f_{vs})_y.
\end{aligned} \tag{11}$$

Now, defining PINN outputs $\mathbf{g}$ to be

$$\mathbf{g} = [\overline{u}, \overline{v}, \overline{p} - \phi, f_{us}, f_{vs}]^T, \tag{12}$$

we get four equations for five unknowns.

This is still an under-determined system. If the mean velocity fields are fully known on a fine grid, then, excluding the automatically verified mass conservation equation, we obtain a system of three equations with three unknowns, to which a solution can be obtained. In the flow reconstruction framework, for which the mean velocities are provided at sparse data points only, the system is not fully determined. Nonetheless, because the idealized system with fully known velocity fields is determinate, we hypothesize that PINN may be able to converge to a solution that is an accurate representation of the true flow. The bottom path in Figure 1 presents the PINN formulation corresponding to this approach with appropriately set network outputs in the blue box and the physical residuals in the purple box.

The reorganization of the Reynolds stresses into the forcing term and the subsequent decomposition into potential and solenoidal components together with fully known velocity fields makes the idealized system determinate. It was possible to add one equation and decrease the number of unknowns by 1. This does not mean that some new information was introduced to the problem—rather, it was possible to lump the unknown potential variables $\overline{p}$ and ϕ together. The problem of recovering pressure $\overline{p}$ from the compound $(\overline{p} - \phi)$ field is under-determined and this remains true for the solution to the data assimilation problem.

3. PINNs Implementation and Problem Setup

The methodology for the assimilation of mean-flow data using PINNs will be demonstrated for the two-dimensional unsteady flow around a circular cylinder at diameter-based Reynolds number equal to 150.

3.1. Mean flow database

Mean flow data were obtained from direct numerical simulations using Nektar++ (Cantwell et al., 2015), a spectral/hp element framework. The simulation domain for the 2D cylinder flow was a rectangle spanning from −5 to 15 in the stream-wise direction and from −5 to 5 in the vertical direction. The cylinder with diameter 1 was placed so that its center coincided with the origin.

The unsteady Navier–Stokes equations were solved using a velocity correction scheme with convective advection, Galerkin spatial projection and IMEX second order time integration. The following BCs were imposed. A uniform free stream at the inlet, no-slip conditions at the cylinder surface and outflow conditions as described in Dong et al. (2014), at the outflow boundary. At the top and bottom boundaries, Dirichlet condition was used for the v-velocity component and Neumann condition for the stream-wise velocity component u. For more information about the scheme and BCs, readers are referred to Section 11.1 of Nektar++ user guide (available at www.nektar.info) and Dong et al. (2014). Additionally, the Nektar++ simulation files with the simulation setup are available online on the Github repository (see the Section "Data Availability Statement").

The simulation was run for 100 dimensionless time units and, after the initial transient, statistics were computed based on three full vortex shedding cycles. The maximum CFL number was 1.2. Plots in Figure 2 show the first (mean velocity and pressure fields) and second order (Reynolds stresses) time-averaged quantities.

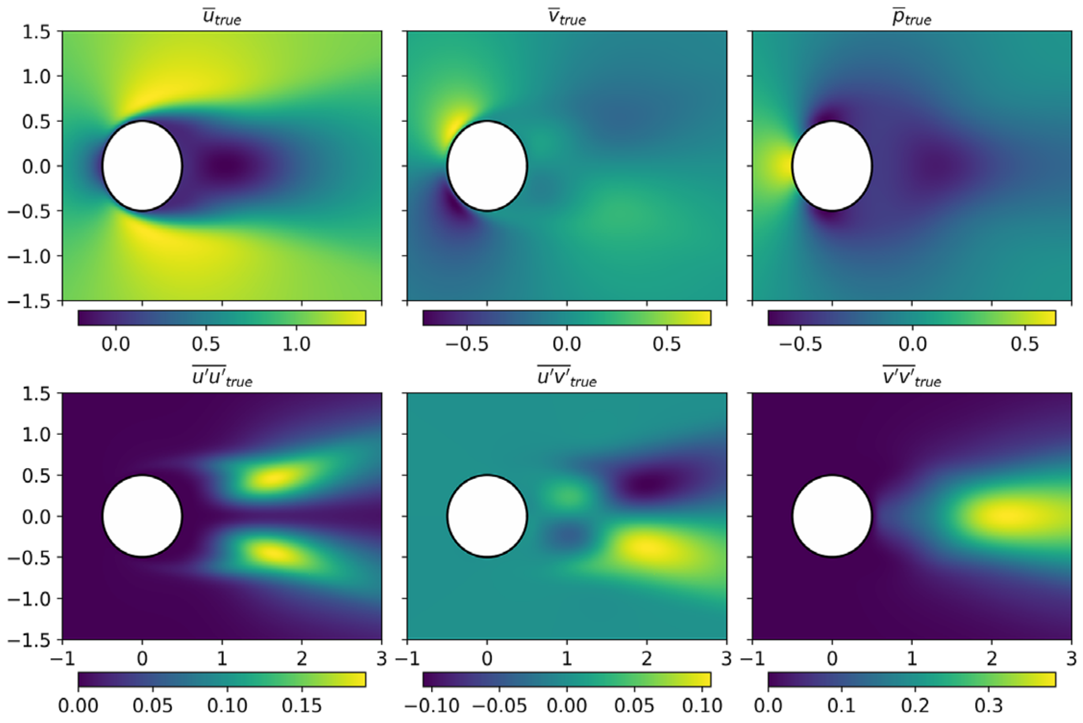


Figure 2. *Direct numerical simulation (ground truth) of the 2D cylinder flow at* $Re = 150$ *using Nektar++. Top: time-averaged stream-wise, cross-stream-wise velocities and pressure. Bottom: time-averaged Reynolds stresses.*

3.2. PINN architecture

The PINNs were implemented using DeepXDE Python library (Lu et al., 2021) which provides a level of abstraction for constructing the PINNs, setting up the numerical domain with appropriate BCs and training of the neural network.

All of the results presented in this work were obtained using the same PINN architecture. It was a multi-layer perceptron with two input variables, x and y, followed by seven fully connected layers with 100 units each terminating to an output layer. The number of units in the output layer was set to be equal to the number of field functions modeled by the network, thus dependent on the formulation used. For all hidden layers, a tanh activation function was applied. Additionally, the weights were initialized with the Glorot uniform algorithm. This setup was found to perform well for all PINN optimization runs. Furthermore, this architecture is consistent with PINNs presented in Raissi et al. (2020) as well as with the networks from the article about the Navier–Stokes flow nets (Jin et al., 2021).

3.3. PINN training

The PINN optimization requires data points, as well as collocations points for both the BCs and the PDE loss. The data points are the input of the PINN optimization and therefore their spatial distribution is not the hyperparameter of the flow reconstruction. In contrast, the number and distribution of the collocation points for the BCs and for the physics loss can be adjusted to aid optimization. The main requirement is the adequacy of points so as to enforce that the output model functions are smooth and that they obey the governing equations in the entire domain. Consequently, the number and distribution of collocation points depend on the flow structure, the domain size and the Reynolds number. In this study, the number of collocation points was determined based on multiple runs and inspection of the results—the number of

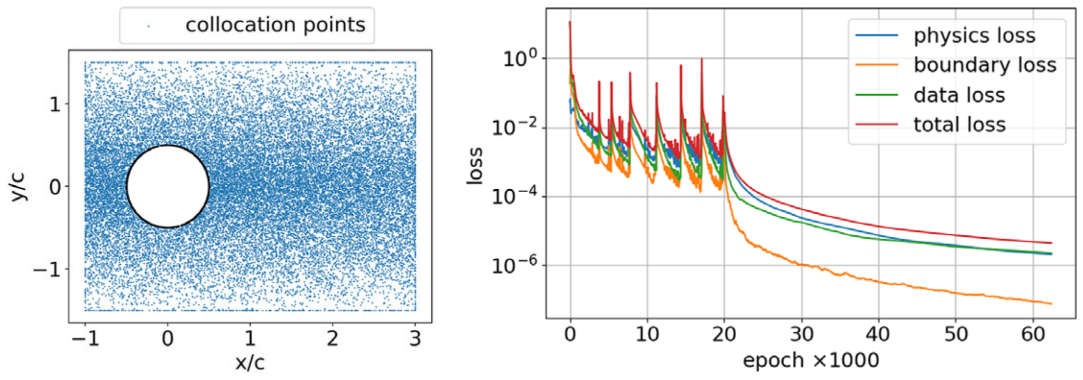


Figure 3. *Left: scatter plot of collocation points around the circular cylinder for minimizing the PDE physics loss. Right: representative loss during the PINN training. One epoch denotes one transit of the training data (data points and collocation points) through the training algorithm. The training history and the collocation points shown correspond to one of the optimization runs for a PINN formulation described in Section 4.1. The qualitative description is representative of all optimizations run in this study.*

collocation points was increased until the point where adding collocation points did not change the output of the PINN optimization. For the BCs, the number of points was chosen so that at each of the boundaries the distance between neighboring points was around 0.01 which translated to a relatively fine resolution. For example, at the cylinder surface, there was around 750 points for enforcing the no-slip wall condition. For the physics loss, the points were clustered near the cylinder surface and the centerline of the flow, as these are the regions with the highest variation of the field functions. Furthermore, the numerical experiments indicated that approximately 25,000 collocation points in the optimization domain (described in the next section) suffice to enforce physical solutions. Therefore, in all PINN optimizations presented in this article, the number of collocation points was approximately 25,000. Figure 3 provides a visual indication of the number of the collocation points and their distribution.

The PINN setup used in this study also employed dynamic loss weighting. This was implemented in order to combat gradient problems which impact PINN optimizations (Wang et al., 2020). The loss weighting scheme is described in Appendix A of the Supplementary Material.

The minimization of the cost function was split into two segments. First, the network was trained for 20,000 epochs with SGD using Adam optimizer. Afterward, the PINNs were optimized using L-BFGS-B scheme until one of the stopping criteria was reached. In all the cases run, the optimizations terminated due to the relative improvement falling below the threshold value which was set to be equal to the machine precision ($1e-7$). Figure 3 shows a training loss history plotted for one of the cases. It provides a qualitative representation of all the training runs performed. Up until 20,000 epochs, when SGD is used, the loss decreases but the curve is ragged and there are multiple loss increases. Afterward, L-BFGS-B algorithm slowly reduces the total loss until the stopping criterion is reached and the network weights are considered to have converged. The PINN optimization was performed using high-performance GPUs ($2 \times$ NVIDIA RTX6000) and each computation took about 4 hrs.

4. Results

4.1. PINNs with RANS forcing as outputs

The performance of PINNs is evaluated by inferring the potential-free part of the unsteady RANS forcing using mean velocity data distributed on a square grid (PIV data format) around the cylinder. This is the same problem that Foures et al. (2014) tackled with adjoint variable optimization (see Section 4.1 in Foures et al., 2014).

The numerical domain for the optimization was a cut out of the simulation domain—a rectangle spanning $(-1, 3)$ in x and $(-1.5,1.5)$ in y directions. The cylinder with diameter 1 was centered at the origin. The resolution of the data grid was 0.02×0.02 which means that the data points were spaced by 0.02 in both x and y directions. Furthermore, the data grid spanned the entire numerical domain and was positioned so that the coordinates of the bottom left corner and upper right corner were $(x, y) = (-1, -1.5)$ and $(x, y) = (3,1.5)$, respectively. Thus, in total there were 201 points in x direction and 151 points in y direction.

The network outputs were set to be

$$\overline{u}, \overline{v}, \overline{p} - \phi, f_{us}, f_{vs},$$

following the PINN output description in Section 2.1.2. The formulation of PINN to output the above flow variables and the corresponding form of the physics residuals will be referred to as the forcing formulation. The data loss $\mathcal{L}_D$, following the definition (5), was:

$$\mathcal{L}_D = \frac{1}{N_D} \sum_{j=1}^{N_D} \left[\left[\overline{u}\left(\mathbf{x}_j^D\right) - \overline{u}_j^D \right]^2 + \left[\overline{v}\left(\mathbf{x}_j^D\right) - \overline{v}_j^D \right]^2 \right]. \tag{13}$$

The physics loss $\mathcal{L}_\mathcal{P}$, following definition (6), was:

$$\mathcal{L}_\mathcal{P} = \frac{1}{N_P} \sum_{j=1}^{N_P} \left[\begin{array}{l} [(\overline{u}_x + \overline{v}_y)|_{\mathbf{x}_j^P}]^2 \\ + [(\overline{u}\overline{u}_x + \overline{v}\overline{u}_y + (\overline{p} - \phi)_x - \mathrm{Re}^{-1}(\overline{u}_{xx} + \overline{u}_{yy}) - f_{us})|_{\mathbf{x}_j^P}]^2 \\ + \left[\left(\overline{u}\overline{v}_x + \overline{v}\overline{v}_y + (\overline{p} - \phi)_y - \mathrm{Re}^{-1}(\overline{v}_{xx} + \overline{v}_{yy}) - f_{vs} \right)|_{\mathbf{x}_j^P} \right]^2 \\ + \left[\left((f_{us})_x + (f_{vs})_y \right)|_{\mathbf{x}_j^P} \right]^2 \end{array} \right]. \tag{14}$$

In the above formula, each row denotes squared residual evaluated at the collocation point $\mathbf{x}_j^P$.

Initially, the only BCs given to the network were the wall BCs at the cylinder surface:

$$\overline{u}_{wall} = \overline{v}_{wall} = f_{us,wall} = f_{vs,wall} = 0 \tag{15}$$

and thus the BCs loss was defined as

$$\mathcal{L}_B = \frac{1}{N_{B,w}} \sum_{j=1}^{N_{B,w}} \left[\overline{u}\left(\mathbf{x}_j^{B,w}\right)^2 + \overline{v}\left(\mathbf{x}_j^{B,w}\right)^2 + f_{us}\left(\mathbf{x}_j^{B,w}\right)^2 + f_{vs}\left(\mathbf{x}_j^{B,w}\right)^2 \right], \tag{16}$$

where $N_{B,w}$ denotes the number of collocation points at the cylinder surface and $\mathbf{x}_j^{B,w}$ denotes j-th wall collocation point.

Using only these BCs and mean flow velocity data, the network was able to regress to the true curl of forcing, as shown in Figure 4 (first row). Comparing it with the exact field from the DNS, it can be deduced that the structure and the magnitude of the field are well inferred in the bulk of the domain. However, there are high magnitudes of $\nabla \times \mathbf{f}$ near the cylinder surface with the maximum value of about 3.2—this can be clearly seen from the error plot. The $\nabla \times \mathbf{f}$ contour from the PINN can be qualitatively compared against the analogous result of the adjoint variable optimization (see Figure 4g,h in Foures et al., 2014). The $\nabla \times \mathbf{f}$ field presented in Foures et al. (2014) is in good agreement with the true field but also shows discrepancies near the cylinder surface.

It is worth noting that in the region of a highest discrepancy—the front of the cylinder, the exact curl of forcing as well as forcing itself is trivially zero. Thus, it should not be a problem for the network to accurately represent both $\mathbf{f}_s$ and $\nabla \times \mathbf{f}$ in this region. However, the velocity field features high spatial gradients and the PINN may not be able to accurately represent those high variations. Consequently, errors in the prediction of the velocity may cause discrepancies in the predicted curl of forcing. One could presume that this could be the issue of insufficient network capacity (the ability of the network to represent complex functions, which is influenced by the number and size of the network's layers).

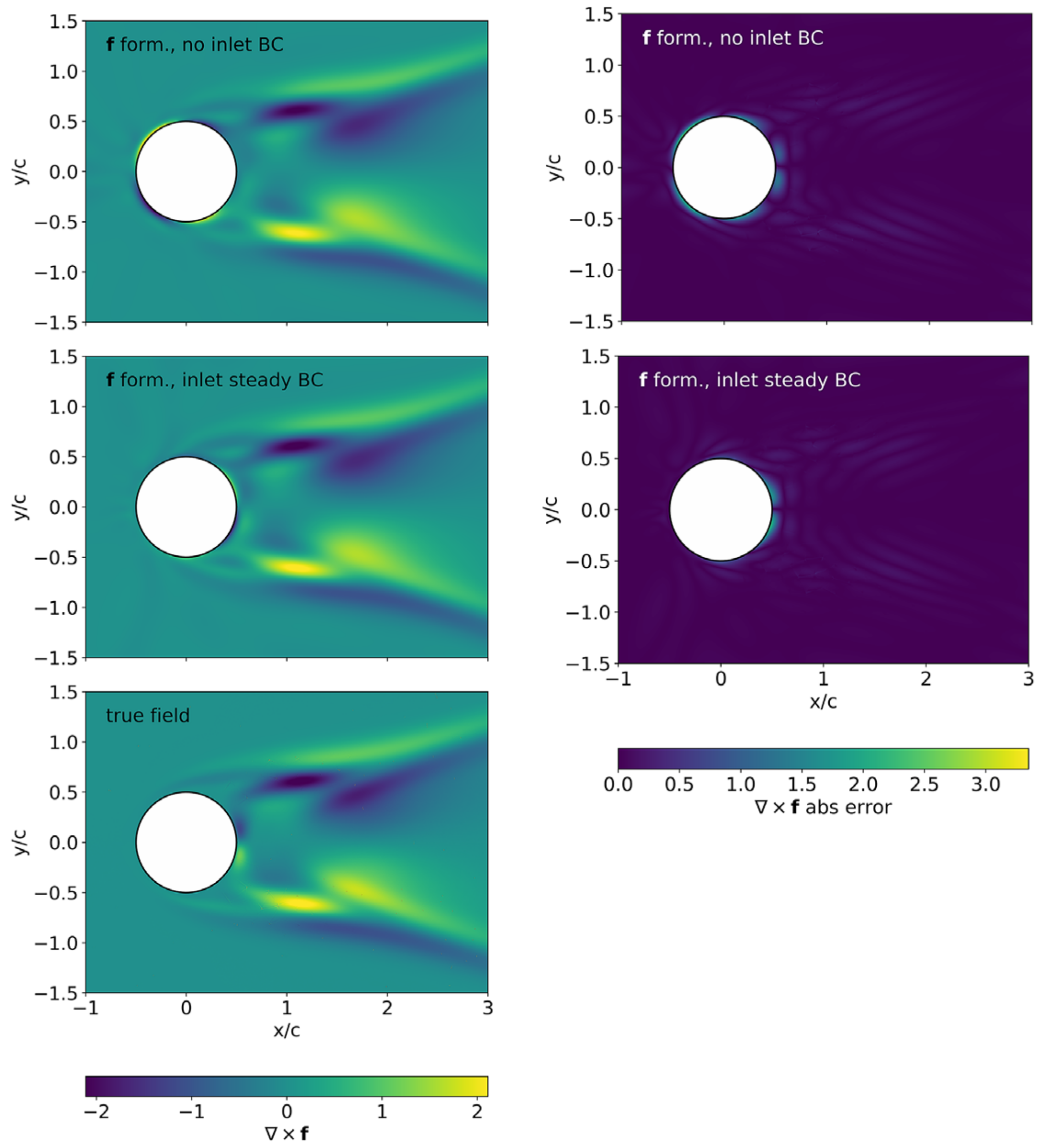


Figure 4. $\nabla \times \mathbf{f}$ *predictions for the 2D cylinder flow obtained with PINNs. Left: regressed fields obtained with forcing formulation without inlet boundary conditions (first row), forcing formulation with inlet steady boundary conditions (second row), as well as the true field (third row). Right: absolute error between regressed and true field. For the left plot in the first row (* $\nabla \times \mathbf{f}$ *with forcing formulation, no inlet BC), the color range was clipped to the true field range to aid comparison.*

Consequently, if the number of network parameters was increased, the predicted $\nabla \times \mathbf{f}$ field might be more accurate. This approach was tested but failed to provide any improvements. Furthermore, since similar errors appeared in the results of adjoint variable optimization, the issue does not appear to be specific to PINNs.

Table 1. *Error measures E_2 and E_∞ for $\nabla \times \mathbf{f}$ prediction using forcing formulation without inlet boundary conditions or with steady inlet boundary condition.*

Case	Region	Field	E_2	E_∞	H_1	H_1 true	$\Delta\%H_1$
Forcing form., no inlet BC	Full	$\nabla \times \mathbf{f}$	27.7%	77.7%	–	–	–
Forcing form., inlet steady BC	Full	$\nabla \times \mathbf{f}$	19.7%	63.4%	–	–	–
Forcing form., no inlet BC	Wake	$\nabla \times \mathbf{f}$	7.03%	8.87%	–	–	–
Forcing form., inlet steady BC	Wake	$\nabla \times \mathbf{f}$	6.15%	7.87%	–	–	–
Forcing form., no inlet BC	Full	$\overline{u}$	0.03%	7.91%	1.7405	1.7409	−0.02%
Forcing form., no inlet BC	Full	$\overline{v}$	0.20%	3.43%	0.9356	0.9363	−0.07%
Forcing form., inlet steady BC	Full	$\overline{u}$	0.03%	7.90%	1.7409	1.7509	−0.002%
Forcing form., inlet steady BC	Full	$\overline{v}$	0.18%	3.43%	0.9363	0.9363	0.009%

Note. In addition, the errors are presented for both the "full" and "wake" regions of the optimization. The lower part of the table shows the error measures E_2 and E_∞ for the predicted velocity fields, as well their H_1 semi-norms, together with H_1 semi-norm of the true fields and the corresponding percentage difference.

In order to quantitatively evaluate the PINN accuracy, we introduce the error measures E_2 and E_∞:

$$E_2 = \frac{||h - h_{true}||_2}{||h_{true}||_2} \cdot 100\% \quad \text{and} \quad E_\infty = \frac{||h_h - h_{true}||_\infty}{||h_{true}||_\infty} \cdot 100\%, \tag{17}$$

where $||\cdot||_2$ and $||\cdot||_\infty$ denote the l_2 and l_∞ norms respectively, and h denotes the analyzed field, in this case $\nabla \times \mathbf{f}$, and subscript "true" denotes DNS output (ground truth). For this and for all subsequent error calculations, unless stated otherwise, the predicted fields were evaluated on a rectangular grid spanning the whole numerical domain and with spacing between points $\Delta x = \Delta y = 0.002$. As shown in Table 1, the E_2 error for the reconstruction of $\nabla \times \mathbf{f}$ is 27.7%. The plots indicate that this high value might be caused by the region of high discrepancy at the cylinder surface. This is confirmed by recomputing the error only for the wake region of the flow—rectangular area restricted to (0.75,3) in x and $(-1.5,1.5)$ in y directions. These values are denoted in Table 1 as "wake," in contrast to "full" region which includes the entire optimization domain. The magnitudes of E_2 and E_∞ for the wake region are considerably smaller, 7 and 9%, respectively, confirming that most of the error is confined to the region of high velocity variation close to the cylinder surface.

In addition to the error measures above, it is worth analyzing the smoothness of the reconstructed velocity fields by introducing the H_1 semi-norm:

$$||h||_{H_1} = ||\nabla h||_2, \tag{18}$$

where h denotes the analyzed field function.

Figure 5 presents the predictions of the velocity fields with the associated absolute error plots. Table 1 shows the E_2, E_∞ and H_1 values for the reconstructed velocity field. In addition, the table shows the H_1 semi-norm for the true field (DNS solution) and the relative change of the semi-norm between the true field and the prediction. In terms of E_2 and E_∞ errors, the reconstructed fields are in very good agreement with the true fields, which was expected as the density of velocity data was high. Furthermore, the reconstructed fields have almost the same value of H_1 semi-norm as the true fields which means that both fields have almost identical smoothness.

In order to avoid high magnitude values of the forcing near the cylinder surface, more information about the flow can be supplied to the network. At the inlet of the domain, the flow is steady which implies that Reynolds stresses, as well as their derivatives, are zero and consequently that both potential and solenoidal forcings are zero. Thus, new BCs can be added:

$$f_{us,inlet} = f_{vs,inlet} = (\nabla \times \mathbf{f}_s)_{inlet} = 0. \tag{19}$$

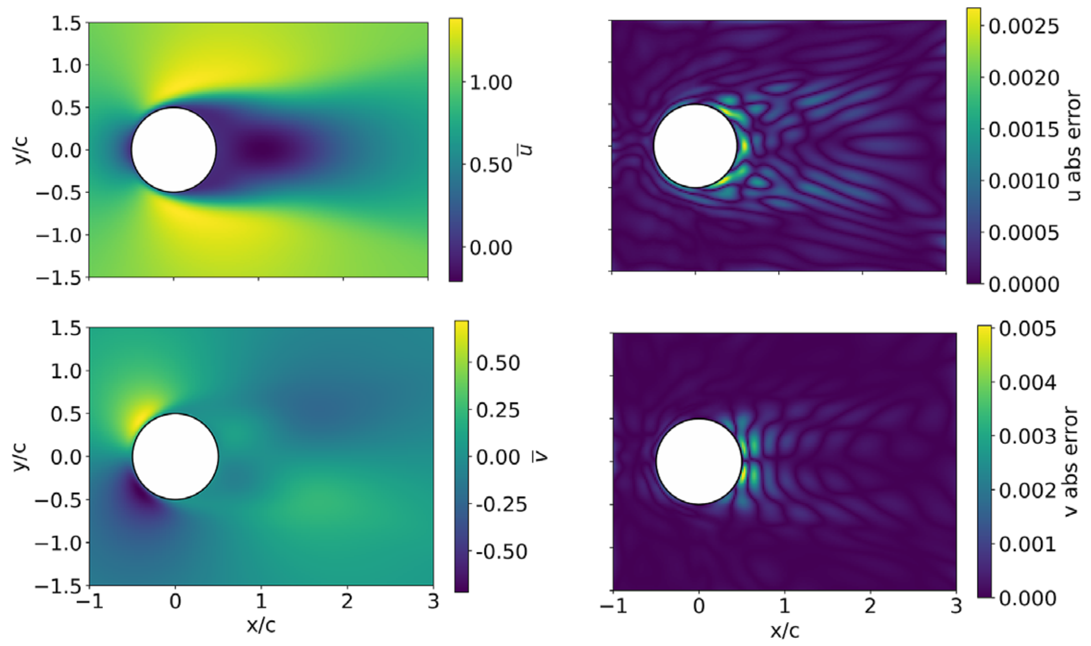


Figure 5. *Reconstructed mean velocity fields and corresponding error plots obtained using forcing formulation PINN without imposing steady inlet boundary condition. The resolution of the input data grid with velocity measurements was* 0.02×0.02.

And thus an additional term was added to the BCs loss. Denoting $N_{B,i}$ as the number of inlet collocation points and the corresponding points as $\mathbf{x}_j^{B,i}$:

$$\begin{aligned}\mathcal{L}_B = &\frac{1}{N_{B,w}}\sum_{j=1}^{N_{B,w}}\left[\left[\overline{u}\left(\mathbf{x}_j^{B,w}\right)\right]^2+\left[\overline{v}\left(\mathbf{x}_j^{B,w}\right)\right]^2+\left[f_{us}\left(\mathbf{x}_j^{B,w}\right)\right]^2+\left[f_{vs}\left(\mathbf{x}_j^{B,w}\right)\right]^2\right]\\ &+\frac{1}{N_{B,i}}\sum_{j=1}^{N_{B,i}}\left[\left[f_{us}\left(\mathbf{x}_j^{B,i}\right)\right]^2+\left[f_{vs}\left(\mathbf{x}_j^{B,i}\right)\right]^2+\left[\left(\nabla\times\left(f_{us},\,f_{vs}\right)^T\right)\big|_{\mathbf{x}_j^{B,i}}\right]^2\right].\end{aligned} \tag{20}$$

Gradients for the curl were obtained within PINN, using automatic differentiation—the same method as for the physics residuals.

The regression results with the above BCs are shown in Figure 4 (second row). The inferred forcing curl field, $\nabla \times \mathbf{f}$, is similar to the one without inlet BCs, but the magnitude near the front cylinder surface is diminished. This conclusion is reinforced by a comparison of the error plots. The error value comparison in Table 1, with the new results marked as "forcing form., inlet steady BC," also confirms the improvement of accuracy. In the "full" region, the E_2 decreases from 27.7 to 19.7%, whereas E_∞ decreases from 77.7 to 63.4%. The improvement also extends to the "wake" region, where E_2 decreases from 7.03 to 6.15% and E_∞ decreases from 8.87 to 7.87%. Again, the relative difference of errors between the "full" and the "wake" region indicates that most of the error is caused by the high discrepancy region near the cylinder surface. The addition of the inlet BC condition also provides accuracy improvements for the reconstruction of the velocity fields—this is evident from E_2 and E_∞ values presented in the lower portion of Table 1.

For the specific cylinder flow, the assumption of steady inlet BCs is valid and thus the second formulation is more advantageous. This demonstrates how the PINN approach can be augmented with extra information to enhance the reconstruction quality.

Nevertheless, for the no inlet BCs case, while there is a local error in $\nabla \times \mathbf{f}$ near the cylinder surface, the prediction in the wake is still satisfactory. Due to its generality (imposing outer BCs is not always valid), the forcing formulation without inlet BC is chosen for further analysis.

In conclusion, PINN networks can successfully leverage the available mean flow velocity data to find the correct representation of the curl of the RANS forcing for the unsteady 2D cylinder flow. In the wake, the regressed field is in good agreement with the true field. Near the cylinder walls, PINN's prediction shows discrepancies with $\nabla \times \mathbf{f}$ reaching relatively high magnitudes. Adding inlet boundary information can alleviate this discrepancy and improve the prediction in the entire domain. The results obtained with the adjoint variable optimization in Foures et al. (2014) also show considerable discrepancies at the cylinder surface, indicating that the error is not specific to the PINN approach.

4.1.1. Predicting forces on the cylinder

The total force distribution exerted on the cylinder is due to pressure forces and frictional shear stresses on the surface. Using the forcing formulation, the network infers $p - \phi$, which is the difference between the pressure and the potential component of the forcing fields. Without additional information or assumptions, it is not possible to separate the two fields (see Foures et al., 2014) and thus obtain an accurate estimate of the pressure distribution. However, the skin friction can be extracted since it depends only on the inferred velocity field. Figure 6 presents the friction coefficient over the top surface of the cylinder as a function of the angle θ with 0° at the front of the cylinder and 180° at the rear of the cylinder.

The results are compared with the results from the Nektar++ simulations as well as the experiments from Dimopoulos and Hanratty (1968). The friction coefficient is defined as

$$C_f = \frac{\mu \frac{\partial u_t^*}{\partial n^*}}{\frac{1}{2}\rho U_\infty^2} = \frac{2}{Re}\frac{\partial u_t}{\partial n}, \tag{21}$$

where $\frac{\partial u_t}{\partial n}$ is the derivative of the non-dimensional tangential velocity normal to the surface (* corresponds to dimensional values). The prediction of the friction coefficient is in excellent agreement with the Nektar++ result (ground truth). The good level of accuracy for the C_f distribution was expected as the density of the velocity data points is relatively high.

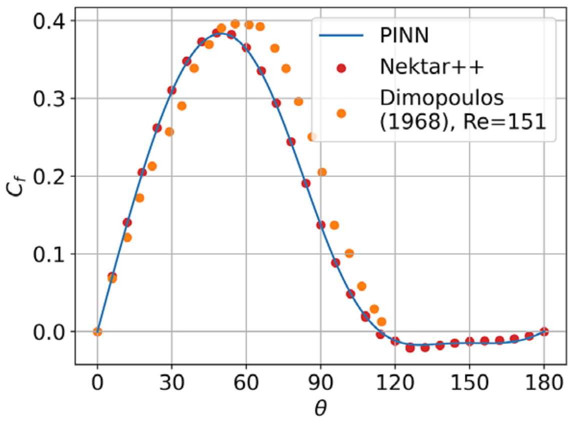


Figure 6. *Friction coefficient over the cylinder surface obtained using output from the PINN optimization performed using forcing formulation without inlet boundary condition (line). Additionally, the plot presents DNS results obtained with Nektar++ (red circles) and experimental measurements (orange circles).*

Table 2. *Error measures E_2 and E_∞ for $\nabla \times \mathbf{f}$ field obtained using PINN optimization on velocity data with added noise.*

Case	Region	Field	Noise level	E_2	E_∞
Forcing form., no inlet BC	Full	$\nabla \times \mathbf{f}$	0.02	64.6%	209%
Forcing form., no inlet BC	Full	$\nabla \times \mathbf{f}$	0.05	60.8%	158%
Forcing form., no inlet BC	Full	$\nabla \times \mathbf{f}$	0.1	45.7%	56.9%
Forcing form., no inlet BC	Wake	$\nabla \times \mathbf{f}$	0.02	20.3%	25.2%
Forcing form., no inlet BC	Wake	$\nabla \times \mathbf{f}$	0.05	22.1%	22.9%
Forcing form., no inlet BC	Wake	$\nabla \times \mathbf{f}$	0.1	24.9%	25.7%

4.1.2. Influence of noise

Here, we investigate how the noise in the velocity data affects the accuracy of the prediction of $\nabla \times \mathbf{f}$ and of the velocity fields. The analysis was performed on the same numerical domain and data grid as in Section 4.1. Zero-mean Gaussian noise was added at all data points, to both velocity components. The PINN optimization, utilizing forcing formulation without inlet BC, was run for three levels of noise with variance 0.02, 0.05, and 0.1. The resolution of the input data was unchanged—0.02×0.02.

Table 2 presents the errors of the predicted $\nabla \times \mathbf{f}$ field for the different noise levels. The division between the "full" and "wake" region (same as in the previous section) shows that in the "wake," the level of errors is again significantly lower and in this region the error is an increasing function of the noise level. Interestingly, this is not the case for the "full" region and thus the region closer to the cylinder surface. There, increasing the amount of noise decreases the errors of the predicted $\nabla \times \mathbf{f}$.

Table 3 presents the errors for the reconstruction of the velocity fields. For these calculations, the errors were evaluated on the same grid as in the previous section (with point spacing $\Delta x = 0.002$ and $\Delta y = 0.002$). Column "E_2 before denoising" presents the E_2 error measure between ground truth (DNS) velocity fields with and without noise. For example,

$$\text{“}u\,\text{field } E_2 \text{ before denosing”} = \frac{\|(u_{true} + \text{noise}) - u_{true}\|_2}{\|u_{true}\|_2} \cdot 100\% = \frac{\|\text{noise}\|_2}{\|u_{true}\|_2} \cdot 100\%, \tag{22}$$

where "noise" denotes a noise field. Thus, the "E_2 before denoising" value directly corresponds to the level of noise and indicates the amount of noise present in the data before PINN optimization was performed. Column "E_2 after denoising" presents the E_2 error measure between ground truth and the reconstructed field obtained using PINN applied to the noisy input data. In all cases, E_2 error of PINN prediction is an order of magnitude lower than the E_2 value before denoising which implies that the application of PINN has effectively decreased the amount of noise present in the velocity data.

4.2. Flow interpolation

Another application of PINNs is the interpolation of fluid flows. If the density of the velocity data is low, instead of trying to find the unknown forcing, PINNs can be applied to complete the partial information about the velocity fields. Given a set of data points in the domain $\left\{\left(\mathbf{x}^{\mathbf{i}}_{\mathbf{1}}, h^i_1\right), \left(\mathbf{x}^{\mathbf{i}}_{\mathbf{2}}, h^i_2\right), \dots, \left(\mathbf{x}^{\mathbf{i}}_{\mathbf{N_{in}}}, h^i_{N_{in}}\right)\right\}$, PINNs can assimilate that data to find an accurate representation of the true field by providing a synthetic set of field points located in-between the original set of points $\left\{\left(\mathbf{x}^{\mathbf{o}}_{\mathbf{1}}, h^o_1\right), \left(\mathbf{x}^{\mathbf{o}}_{\mathbf{2}}, h^o_2\right), \dots, \left(\mathbf{x}^{\mathbf{o}}_{\mathbf{N_{out}}}, h^o_{N_{out}}\right)\right\}$ which can be significantly larger than the input data set ($N_{in} << N_{out}$). In the case of PINNs, instead of a set of output points, the result of the interpolation comes in the form of a converged neural network capable of evaluating the velocity at any point within the optimization domain. The interpolation of velocity fields on sparse data from 2D cylinder flow was also performed using adjoint variable optimization in the article by Foures et al. (2014) (Section 4.2.1 in their article) and again their study will provide a means of performance assessment.

Table 3. *E_2 error measure of the velocity fields before and after application of PINN to the velocity data with added noise.*

Case	Region	Field	Noise level	E_2 before denoising	E_2 after denoising
Forcing form., no inlet BC	Full	$\overline{u}$	0.02	2.11%	0.16%
Forcing form., no inlet BC	Full	$\overline{v}$	0.02	11.3%	0.66%
Forcing form., no inlet BC	Full	$\overline{u}$	0.05	5.27%	0.30%
Forcing form., no inlet BC	Full	$\overline{v}$	0.05	28.1%	1.34%
Forcing form., no inlet BC	Full	$\overline{u}$	0.1	10.5%	0.55%
Forcing form., no inlet BC	Full	$\overline{v}$	0.1	56.3%	2.42%

4.2.1. Flow interpolation with first-order velocity statistics

Here, a coarse distribution of first-order velocity statistics ($\overline{u}$ and $\overline{v}$) is known and used as data points. The PINN configuration was identical to the one presented in Section 2.1.2, using the forcing formulation. The data was distributed on a coarse rectangular grid, with spacing equal to 0.5 in both x and y directions (resolution 0.5×0.5). This is the same data resolution that was used for interpolation in Foures et al. (2014). The bottom left and the upper right corners of the data grid were located at $(x, y) = (-1, -1.25)$ and $(x, y) = (3, 1.25)$, respectively.

The results of the interpolation, along with the distribution of the training points are presented in Figure 7. Table 4 presents the quantitative measures E_2 and E_∞, as well as the H_1 semi-norm for the reconstructed velocity fields. As in Section 4.1, the accuracy of interpolated fields was evaluated on a rectangular grid spanning the entire numerical domain and with the spacing between points $\Delta x = \Delta y = 0.002$. The interpolation obtained with PINN is satisfactory—for both velocity components

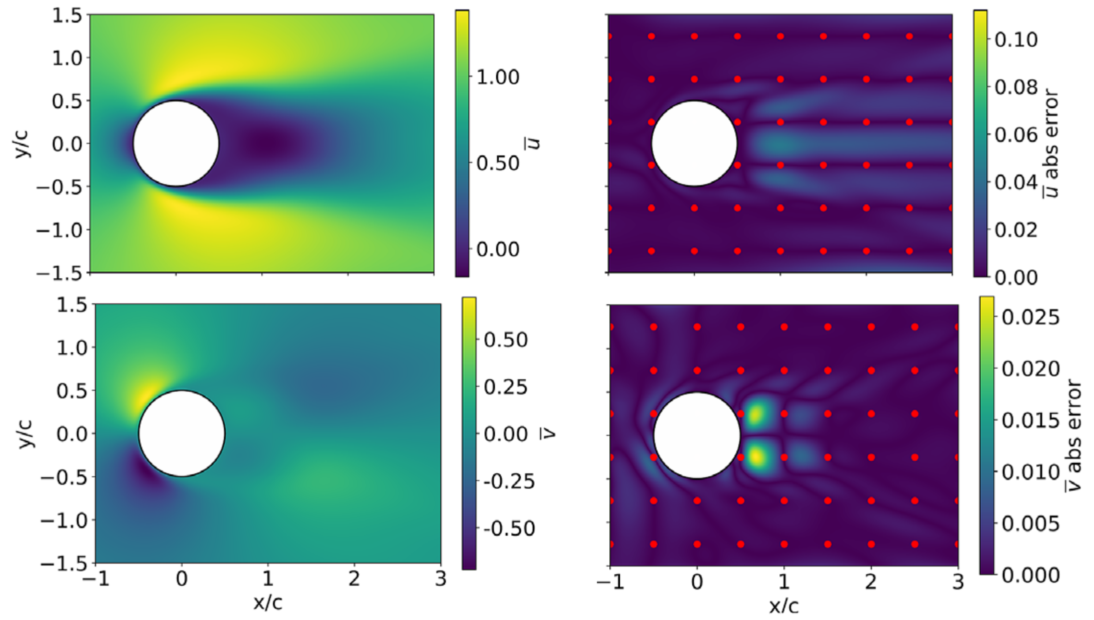


Figure 7. *Flow interpolation using PINN forcing formulation without inlet BC on a coarse data grid of resolution* 0.5×0.5. *At each data point, first-order velocity statistics were provided ($\overline{u}$ and $\overline{v}$). The plots show the predicted velocity fields (left column) and the corresponding absolute error fields (right column) for the stream-wise (top) and the cross-stream-wise (bottom) velocities. The red dots indicate the locations of the data points.*

Table 4. *Error measures and H_1 semi-norm comparison for the interpolated velocity fields shown in Figure 7 and for the predicted $\nabla \times \mathbf{f}$ field.*

Region	Field	E_2	E_∞	H_1	H_1 true	$\Delta\%\ H_1$
Full	$\overline{u}$	1.13%	8.10%	1.7362	1.7401	−0.27%
Full	$\overline{v}$	1.58%	3.75%	0.9306	0.9363	−0.61%
Full	$\nabla \times \mathbf{f}$	56.3%	161%	–	–	–
Wake	$\nabla \times \mathbf{f}$	25.7%	31.2%	–	–	–

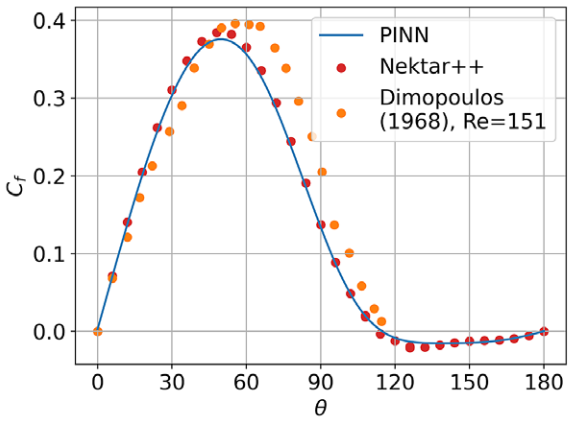


Figure 8. *Prediction of the friction coefficient over the cylinder surface for the interpolated velocity fields shown in Figure 7.*

E_∞ is well below 10% and E_2 is below 1.6%. In terms of H_1 semi-norm, the reconstructed velocity fields are in good agreement with the true fields. This data resolution used here is the same as in Foures et al. (2014) and the PINNs approach exhibits similar errors as the adjoint optimization (see Figure 6 in Section 4.2.1 of Foures et al., 2014).

The accuracy of the interpolation is further evaluated by calculating the friction coefficient over the cylinder surface, which is shown in Figure 8. The PINN predicted friction distribution is still in very good agreement with the Nektar++ results with minor deviations in the regions where the friction is maximum or minimum. Most importantly, the level of accuracy is retained despite the optimization being performed on a significantly coarser data grid.

4.2.2. Flow interpolation with first- and second-order velocity statistics

Here, sparse second-order velocity statistics, that is, Reynolds stresses $\overline{u'u'}$, $\overline{u'v'}$, and $\overline{v'v'}$ are provided along with first-order statistics $\overline{u}$ and $\overline{v}$. The PINN, constrained by the explicit RANS equations described in Section 2.1.1, was employed to interpolate the mean velocity fields and the mean Reynolds stresses fields, as well as to extract the unknown pressure field. This formulation will be referred to as explicit Reynolds stress formulation. The network outputs were set to

$$\overline{u}, \overline{v}, \overline{p}, \overline{u'u'}, \overline{u'v'}, \overline{v'v'}.$$

Consequently, the physics loss $\mathcal{L}_P$, following definition (6), became

$$\mathcal{L}_P = \frac{1}{N_P}\sum_{j=1}^{N_P}\left[\begin{array}{l} \left[\left(\overline{u}_x+\overline{v}_y\right)|_{\mathbf{x}_j^P}\right]^2 \\ +\left[\left(\overline{u}\overline{u}_x+\overline{v}\overline{u}_y+\overline{p}_x-\mathrm{Re}^{-1}\left(\overline{u}_{xx}+\overline{u}_{yy}\right)+\left(\overline{u'u'}\right)_x+\left(\overline{u'v'}\right)_y\right)|_{\mathbf{x}_j^P}\right]^2 \\ +\left[\left(\overline{u}\overline{v}_x+\overline{v}\overline{v}_y+\overline{p}_y-\mathrm{Re}^{-1}\left(\overline{v}_{xx}+\overline{v}_{yy}\right)+\left(\overline{u'v'}\right)_x++\left(\overline{v'v'}\right)_y\right)|_{\mathbf{x}_j^P}\right]^2 \end{array}\right]. \tag{23}$$

Additionally, the BCs at the cylinder surface were expanded to include wall constraints on Reynolds' stresses:

$$\overline{u}_{wall}=\overline{v}_{wall}=\overline{u'u'}_{wall}=\overline{u'v'}_{wall}=\overline{v'v'}_{wall}=0 \tag{24}$$

and the boundary loss $\mathcal{L}_B$ was updated accordingly to include no-slip condition on the Reynolds stress terms:

$$\mathcal{L}_B = \frac{1}{N_{B,w}}\sum_{j=1}^{N_{B,w}}\left[\begin{array}{l} \overline{u}\left(\mathbf{x}_j^{B,w}\right)^2+\overline{v}\left(\mathbf{x}_j^{B,w}\right)^2+\overline{u'u'}\left(\mathbf{x}_j^{B,w}\right)^2 \\ +\overline{u'v'}\left(\mathbf{x}_j^{B,w}\right)^2+\overline{v'v'}\left(\mathbf{x}_j^{B,w}\right)^2 \end{array}\right]. \tag{25}$$

Lastly, the addition of new data requires modification of the data loss $\mathcal{L}_D$. For this case, where Reynolds stresses are explicitly given, the pressure gradient is uniquely defined. In contrast to the forcing formulation, there is no potential field ϕ absorbed in the pressure gradient. To make the pressure prediction unique and remove any arbitrary constant pressure offset, two pressure data points located at the top and bottom of the cylinder were added to the PINN training by augmenting the data loss with the pressure data error term. Thus, the new data loss was

.

$$\begin{aligned} \mathcal{L}_D = \frac{1}{N_D}\sum_{j=1}^{N_D} & \left[\begin{array}{l} \left[\overline{u}\left(\mathbf{x}_j^D\right)-\overline{u}_j^D\right]^2+\left[\overline{v}\left(\mathbf{x}_j^D\right)-\overline{v}_j^D\right]^2+\left[\overline{u'u'}\left(\mathbf{x}_j^D\right)-\overline{u'u'}_j^D\right]^2 \\ +\left[\overline{u'v'}\left(\mathbf{x}_j^D\right)-\overline{u'v'}_j^D\right]^2+\left[\overline{v'v'}\left(\mathbf{x}_j^D\right)-\overline{v'v'}_j^D\right]^2 \end{array}\right] \\ & +\left[\overline{p}\left(\mathbf{x}_{bottom}^D\right)-\overline{p}_{bottom}^D\right]^2+\left[\overline{p}\left(\mathbf{x}_{top}^D\right)-\overline{p}_{top}^D\right]^2 \end{aligned} \tag{26}$$

The distribution of data points was the same as for interpolation using first-order velocity statistics in Section 4.2.1. The inferred fields are shown in Figure 9. The upper section of Table 5 presents the error measures, as well as H_1 semi-norm comparison for the predicted fields.

The PINN is able to infer the time-averaged velocity fields $\overline{u}$ and $\overline{v}$, however, with higher errors than in the previous case when only first-order statistics were provided and the forcing formulation was used. This is despite having more information about the flow—in the previous case, the network did not have access to the values of the Reynolds stresses at the data points. This drop in accuracy is caused by the fact that in the regions between the data points, the RANS equations with Reynolds stresses have more unknowns and fewer constraints than in the case of forcing formulation. If RANS forcing is used instead of stresses, the system has five unknowns for four equations. Using Reynolds stresses in the RANS equations, the system has six unknown flow variables for three equations and thus it is easier for PINN to converge to an incorrect solution in regions where no data is provided.

Inspection of the values in Table 5 indicates the levels of E_2 and E_∞ errors are quite high for all Reynolds stresses, as well as for the pressure field. Inspecting the plots of these fields and the corresponding error plots, it can be seen that in all cases there is a significant discrepancy in front of the cylinder, whereas the prediction in the wake looks relatively better. As the region in front of the cylinder is free from unsteadiness, the Reynolds stresses are zero and PINN should have no problem representing that field in this region. Thus, it can be concurred that the error is caused by low data density. In order to improve the accuracy of the reconstruction, more data points have to be supplied.

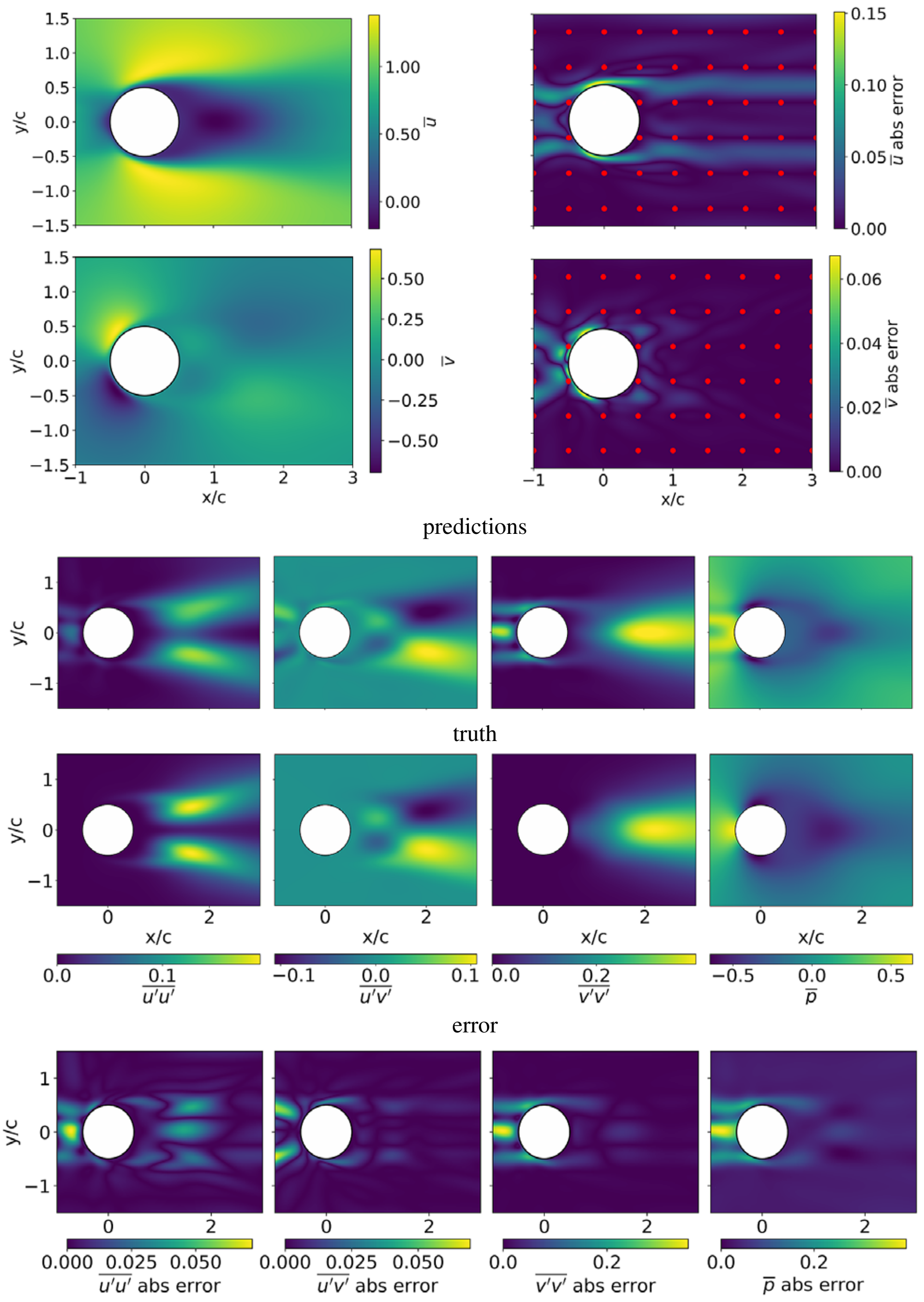


Figure 9. *Flow interpolation results with second-order velocity statistics of resolution* 0.5×0.5*. Plots in the first and second rows show predicted velocity fields (left column) and the corresponding error (right column) for the stream-wise and the cross-stream-wise velocities, respectively. Beneath, in four columns, plots of regressed (top), true (middle), and error (bottom) fields are shown for Reynolds stresses:* $\overline{u'u'}$, $\overline{u'v'}$, $\overline{v'v'}$ *and pressure* $\overline{p}$ *(respectively, going from left to right).*

Table 5. *Error measures and H_1 semi-norm comparison for the interpolated fields obtained using the Reynolds stresses formulation on the first- and second-order velocity data without or with pressure data over the cylinder surface.*

Case	Field	E_2	E_∞	H_1	H_1 for true field	$\Delta\%\ H_1$
Re stresses form.	$\overline{u}$	2.16%	10.9%	1.7351	1.7409	−0.33%
Re stresses form.	$\overline{v}$	3.48%	9.36%	0.8902	0.9363	−4.92%
Re stresses form.	$\overline{u'u'}$	19.1%	37.4%	0.1600	0.1775	−9.86%
Re stresses form.	$\overline{u'v'}$	22.5%	65.7%	0.1457	0.1188	22.6%
Re stresses form.	$\overline{v'v'}$	36.8%	91.2%	0.5603	0.2140	136%
Re stresses form.	$\overline{p}$	26.6%	61.5%	0.6239	0.4952	26.0%
Re stresses form. with pressure	$\overline{u}$	1.01%	8.02%	1.7482	1.7409	0.42%
Re stresses form. with pressure	$\overline{v}$	2.23%	6.58%	0.9241	0.9363	−1.30%
Re stresses form. with pressure	$\overline{u'u'}$	18.5%	31.6%	0.1550	0.1775	−12.7%
Re stresses form. with pressure	$\overline{u'v'}$	14.5%	36.5%	0.1434	0.1188	20.7%
Re stresses form. with pressure	$\overline{v'v'}$	17.2%	52.6%	0.3201	0.2140	49.6%
Re stresses form. with pressure	$\overline{p}$	8.31%	26.9%	0.5429	0.4952	9.63%

4.2.3. Flow interpolation with first- and second-order velocity and pressure statistics

For the formulation with Reynolds stresses as outputs, the PINN infers pressure which does not include the potential part of the forcing ϕ. Thus, in the event of having pressure tappings on the cylinder surface, those time-averaged measurements can be supplied to the network to aid the interpolation. This is implemented by adding the error from the pressure data to the data loss in (26):

$$\mathcal{L}_D = \frac{1}{N_D}\sum_{j=1}^{N_D}\left[\begin{array}{l}\left[\overline{u}\left(\mathbf{x}_j^D\right)-\overline{u}_j^D\right]^2+\left[\overline{v}\left(\mathbf{x}_j^D\right)-\overline{v}_j^D\right]^2+\left[\overline{u'u'}\left(\mathbf{x}_j^D\right)-\overline{u'u'}_j^D\right]^2\\ +\left[\overline{u'v'}\left(\mathbf{x}_j^D\right)-\overline{u'v'}_j^D\right]^2+\left[\overline{v'v'}\left(\mathbf{x}_j^D\right)-\overline{v'v'}_j^D\right]^2\end{array}\right] \\ +\frac{1}{N_{D,p}}\sum_{j=1}^{N_{D,p}}\left[\overline{p}\left(\mathbf{x}_j^{D,p}\right)-\overline{p}_j^{D,p}\right]^2, \tag{27}$$

where $N_{D,p}$ denotes the number of pressure data points and $\overline{p}_j^{D,p}$ denotes the value of pressure at j-th data point located at $\mathbf{x}_j^{D,p}$. To simulate having dense pressure measurements over the cylinder surface, 100 pressure data points were specified. Results from an optimization with added pressure data can be seen in Figure 10. The bottom part of Table 5 presents the error measures for the new interpolation. The addition of pressure data improves the reconstruction accuracy, especially for Reynolds stresses $\overline{u'v'}, \overline{v'v'}$ and pressure field $\overline{p}$. Nonetheless, the errors are still considerable and present for the Reynolds stresses in the region in front of the cylinder. Thus, it can be again argued that the data density should be increased in order to obtain satisfactory interpolation accuracy.

4.2.4. Resolution versus velocity reconstruction error

Finally, we investigate the dependence of the reconstruction errors on the data grid resolution. Specifically, PINN regressions were performed for the cases with first-order velocity data only and with both first- and second-order velocity data (Section 4.2.1 with forcing formulation and Section 4.2.2 with Reynolds stresses formulation, respectively) for square grid resolutions of cell length 0.02, 0.05, 0.1, 0.2, 0.5, 0.5, 0.6, 0.7, and 1.0. Figure 12 in Appendix B of the Supplementary Material shows the data grids for the above data grid resolutions, overlaid on $\overline{u}_{true}$ velocity field.

Figure 11 shows the $\overline{u}$ velocity error dependence on the resolution ($\Delta x \times \Delta x$) for the two formulations, with forcing and with explicit Reynolds stresses as outputs. In the same graph, we have plotted the errors

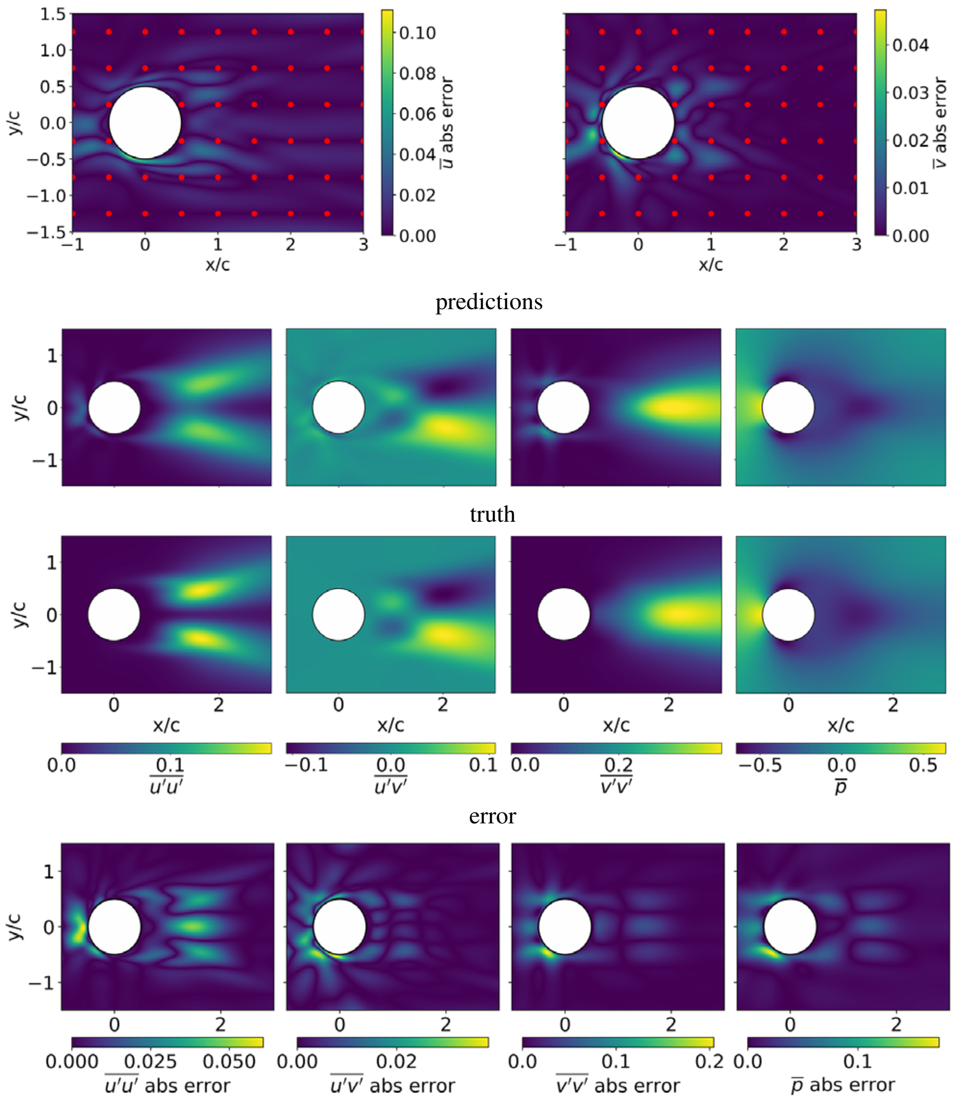


Figure 10. *Flow interpolation results with second-order velocity statistics of resolution* 0.5×0.5 *and pressure data points over the cylinder surface. The top row shows absolute error fields for the velocity components with red points indicating the locations of data points. Beneath, in four columns, plots of regressed (top), true (middle), and error (bottom) fields are shown for Reynolds stresses:* $\overline{u'u'}$, $\overline{u'v'}$, $\overline{v'v'}$ *and pressure* $\overline{p}$ *(from left to right).*

obtained using cubic bivariate spline interpolation. For all resolutions and for both formulations of PINNs, the interpolation with PINNs outperforms the spline interpolation. The relative improvement is the greatest between the smallest analyzed resolution 0.02×0.02 and 0.6×0.6. In that range, the errors obtained with PINNs are an order of magnitude smaller than in the case of spline interpolation. It can be reasoned that in the case of the finest resolution (fine data grid), the spatial variation of the velocity fields is small and thus spline interpolation is very effective. As the resolution increases, the variation of the velocity in between data points increases and the spline interpolation is less successful. At the same time, enforcement of physics constraints allows PINN to produce a better interpolation, as the reconstructed field has to satisfy the governing RANS equations. Finally, as the resolution decreases even more (coarse

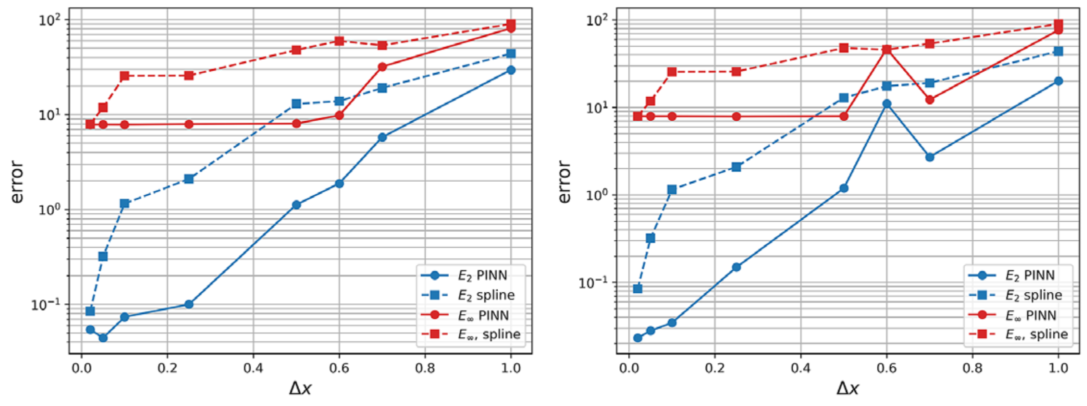


Figure 11. *E_2 and E_∞ errors of the interpolated $\overline{u}$ velocity field against data grid resolution. Forcing formulation using first order data (left) and explicit Reynolds stress formulation using first and second order data (right). Solid lines indicate PINN errors, whereas dashed lines indicate errors obtained by performing spline interpolation on the data points.*

grid), the relative gap between spline and PINN interpolation decreases. The PINN system is under-determined and thus, if the spacing of data points is large, the PINN might converge to a solution that satisfies the constraints (under-determined RANS) but different from the ground truth.

The analysis of the effect of resolution on the reconstruction errors shows that the data resolution for successful interpolation of the velocity fields can be relatively low. However, after reaching a critical resolution (here equal to 0.6×0.6) the system becomes under-determined and reconstruction becomes inaccurate.

5. Conclusions and Future Work

In this article, we have introduced a framework for mean flow reconstruction from sparse flow measurements using PINNs. The method was tested and validated for the unsteady cylinder flow at $\mathrm{Re} = 150$.

Using the forcing formulation of the RANS governing equations, as in Foures et al. (2014), the method deduced the unknown solenoidal part of the RANS forcing. While the prediction in the cylinder wake was satisfactory, the results showed discrepancies near the cylinder surface. An analysis of the influence of the noise in the velocity data on the PINN prediction showed that addition of noise deteriorates the accuracy of the predicted forcing. However, PINN considerably reduced the amount of noise present in the input velocity data. Furthermore, PINNs were able to interpolate the flow based on a limited number of velocity measurements, showing their capability to reconstruct flows from sparse experimental data. In both cases, the accuracy of the predictions was similar to the results obtained using adjoint variable optimization in Foures et al. (2014), despite the differences of the two approaches.

The PINN was extended to explicitly take into account the Reynolds stresses (instead of implicitly through the forcing term) which opens a possibility of leveraging data with second-order velocity statistics as well as pressure measurements. Using sparse measurements of mean velocities and mean Reynolds stresses, the method was able to interpolate both the velocity and the stresses fields for the cylinder flow, albeit with some discrepancies. Furthermore, addition of pressure measurements over the surface of the cylinder significantly improved the interpolation accuracy. Using the interpolated velocity fields it was possible to accurately predict the distribution of skin friction over the cylinder surface.

Finally, the analysis of the influence of data resolution on the interpolation accuracy revealed that PINN interpolation performs better than spline interpolation and the relative improvement is the greatest when data spacing is moderate—bigger than small flow variations but few times smaller than size of the cylinder.

A significant advantage of the PINNs to the reconstruction of the mean flow is the ease of numerical implementation. However, the PINN method requires more computational resources compared to adjoint-based gradient optimization methods. That said, the PINNs are a developing area of research. For example, there exists successful attempts to increase the computational efficiency of PINN optimization. Jagtap and Karniadakis (2020) improved the speed of regression using numerical domain splitting which was further enhanced through parallelization by Shukla et al. (2021). Moreover, Linka et al. (2022) introduced Bayesian PINNs that explicitly consider measurement uncertainty which might provide further performance improvements for noisy input data.

Future steps will focus on applying the above method to 3D flows at higher Reynolds numbers. Especially valuable would be a validation on experimental data with PIV measurements. Furthermore, future applications should employ recent technical advances in PINN architectures and PINN optimization.

Author Contributions. L.S. performed the calculations and wrote the manuscript with continuous input from G.R. Both authors approved the final submitted draft.

Competing Interests. The authors declare no competing interests exist.

Data Availability Statement. All code and files necessary to perform PINN optimizations, as well as to generate mean flow data (Nektar++ simulation files) can be found in Github repository available at https://github.com/RigasLab/PINNS-Mean-Flow-Reconstruction.

Ethics Statement. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Funding Statement. This work received no specific grant from any funding agency, commercial or not-for-profit sectors.

Supplementary Materials. To view supplementary material for this article, please visit http://doi.org/10.1017/dce.2022.37.

References

Beck A, **Flad D and Munz C-D** (2019) Deep neural networks for data-driven LES closure models. *Journal of Computational Physics 398*, 108910.

Brunton SL, **Noack BR and Koumoutsakos P** (2020) Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics 52*(1), 477–508.

Cai S, **Mao Z**, **Wang Z**, **Yin M and Karniadakis GE** (2021) Physics-informed neural networks (PINNs) for fluid mechanics: A review. Preprint, *arXiv*:2105.09506.

Cantwell C, **Moxey D**, **Comerford A**, **Bolis A**, **Rocco G**, **Mengaldo G**, **De Grazia D**, **Yakovlev S**, **Lombard J-E**, **Ekelschot D**, **Jordi B**, **Xu H**, **Mohamied Y**, **Eskilsson C**, **Nelson B**, **Vos P**, **Biotto C**, **Kirby R and Sherwin S** (2015) Nektar++: An open-source spectral/hp element framework. *Computer Physics Communications 192*, 205–219.

Dimopoulos HG and Hanratty TJ (1968) Velocity gradients at the wall for flow around a cylinder for Reynolds numbers between 60 and 360. *Journal of Fluid Mechanics 33*(2), 303–319.

Dong S, **Karniadakis G and Chryssostomidis C** (2014) A robust and accurate outflow boundary condition for incompressible flow simulations on severely-truncated unbounded domains. *Journal of Computational Physics 261*, 83–105.

Eivazi H, **Tahani M**, **Schlatter P and Vinuesa, R** (2022) Physics-informed neural networks for solving Reynolds-averaged Navier–okes equations. *Physics of Fluids 34*(7), 075117.

Fathi MF, **Perez-Raya I**, **Baghaie A**, **Berg P**, **Janiga G**, **Arzani A and D'Souza RM** (2020) Super-resolution and denoising of 4D-Flow MRI using physics-informed deep neural nets. *Computer Methods and Programs in Biomedicine 197*, 105729.

Foures DPG, **Dovetta N**, **Sipp D and Schmid PJ** (2014) A data-assimilation method for Reynolds-averaged Navier–Stokes-driven mean flow reconstruction. *Journal of Fluid Mechanics 759*, 404–431.

Franceschini L, **Sipp D and Marquet O** (2020) Mean-flow data assimilation based on minimal correction of turbulence models: Application to turbulent high Reynolds number backward-facing step. *Physical Review Fluids 5*, 094603.

Fukami K, **Fukagata K and Taira K** (2019) Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics 870*, 106–120.

Jagtap AD and Karniadakis GE (2020) Extended Physics-Informed Neural Networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics 28*(5), 2002–2041.

Jin X, **Cai S**, **Li H and Karniadakis GE** (2021) NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics 426*, 109951.
Ling J, **Kurzawski A and Templeton J** (2016) Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics 807*, 155–166.
Linka K, **Schafer A**, **Meng X**, **Zou Z**, **Karniadakis GE and Kuhl E** (2022) Bayesian physics-informed neural networks for real-world nonlinear dynamical systems. *Computer Methods in Applied Mechanics and Engineering 402*, 115346.
Lu L, **Meng X**, **Mao Z and Karniadakis GE** (2021) DeepXDE: A deep learning library for solving differential equations. *SIAM Review 63*(1), 208–228.
Mons V and Marquet O (2021) Linear and nonlinear sensor placement strategies for mean-flow reconstruction via data assimilation. *Journal of Fluid Mechanics 923*, A1.
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. *Journal of Computational Physics 378*, 686–707.
Raissi M, **Yazdani A and Karniadakis GE** (2020) Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science 367*(6481), 1026–1030.
Shukla K, **Jagtap AD and Karniadakis GE** (2021) Parallel physics-informed neural networks via domain decomposition. *Journal of Computational Physics 447*, 110683.
Symon S, **Dovetta N**, **McKeon B**, **Sipp D and Schmid P** (2017) Data assimilation of mean velocity from 2D PIV measurements of flow over an idealized airfoil. *Experiments in Fluids 58*, 61.
Wang S, **Teng Y and Perdikaris P** (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing 43*(5), A3055–A3081.

Cite this article: Sliwinski L and Rigas G (2023). Mean flow reconstruction of unsteady flows using physics-informed neural networks. *Data-Centric Engineering*, 4, e4. doi:10.1017/dce.2022.37