Eur. Phys. J. E (2023) 46:13
https://doi.org/10.1140/epje/s10189-023-00268-9

THE EUROPEAN PHYSICAL JOURNAL E

Regular Article - Flowing Matter

Assimilation of statistical data into turbulent flows using physics-informed neural networks

Sofía Angriman[1,2,a], Pablo Cobelli[1,2,b], Pablo D.Mininni[1,2,c], MartínObligado[3,d], and Patricio Clark Di Leoni[4,e]

[1] Facultad de Ciencias Exactas y Naturales, Departamento de Física, Ciudad Universitaria, Universidad de Buenos Aires, 1428 Buenos Aires, Argentina
[2] Instituto de Física del Plasma (INFIP), Ciudad Universitaria, CONICET - Universidad de Buenos Aires, Buenos Aires 1428, Argentina
[3] Université Grenoble Alpes, CNRS, Grenoble-INP, LEGI, 38000 Grenoble, France
[4] Departmento de Ingeniería, Universidad de San Andrés, Buenos Aires, Argentina

Received 9 September 2022 / Accepted 12 February 2023 / Published online 9 March 2023

Abstract When modeling turbulent flows, it is often the case that information on the forcing terms or the boundary conditions is either not available or overly complicated and expensive to implement. Instead, some flow features, such as the mean velocity profile or its statistical moments, may be accessible through experiments or observations. We present a method based on physics-informed neural networks to assimilate a given set of conditions into turbulent states. The physics-informed method helps the final state approximate a valid flow. We show examples of different statistical conditions that can be used to prepare states, motivated by experimental and atmospheric problems. Lastly, we show two ways of scaling the resolution of the prepared states. One is through the use of multiple and parallel neural networks. The other uses nudging, a synchronization-based data assimilation technique that leverages the power of specialized numerical solvers.

1 Introduction

Data Assimilation [1,2] is the discipline in charge of reconstructing dynamical systems states by merging models and observations. It has three main motivations (1) dealing with data that is, due to the collection technique used, sparse, ungridded and noisy, (2) the trajectories of chaotic systems diverge in finite times and thus need to be corrected, and (3) the models may not be capturing the full extent of the physical processes involved. Moreover, data can come in the form of direct measurements of state variables or of observations, such as precipitation or an averaged quantity of the state variables. This work will focus on the case where only average (or statistical) data is available, such as the case of asynchronous measurements of a flow, and where the flow is close to a homogeneous isotropic flow but exhibits unusual statistics and its forcing term is not properly known. In particular, our motivation comes from analyzing hot-wire measurements in the bulk of a wind tunnel being subject to an active grid forcing, although the ideas and procedures presented could be readily expanded to other usecases.

There are many families of techniques and approaches available in the data assimilation (DA) realm. The two main ones are ensemble based methods [3] and variational methods [1]. Recently, stochastic methods [4] and machine and deep learning techniques [5] have been making great strides in this area. We present a method based on physics-informed neural networks (PINNs) [6]. PINNs have been used to solve inverse problems [7,8] and to reconstruct turbulent flows out of measurements [9,10]. PINNs work by minimizing a loss function consisting of two terms, a data term involving initial conditions, boundary conditions and/or state variable measurements, and a physics term involving the residuals of the equations of motion. The method presented here takes a seed flow, e.g., coming from a homogeneous isotropic turbulence DNS, and assimilates the statistical observations using a modified PINN. The modifications are twofold, (1) we add a target term based on mean quantities to the loss function (in addition to the

T.I. : Quantitative AI in Complex Fluids and Complex Flows: Challenges and Benchmarks. Guest editors: Luca Biferale, Michele Buzzicotti, Massimo Cencini.

[a] e-mail: sangriman@df.uba.ar
[b] e-mail: cobelli@df.uba.ar
[c] e-mail: mininni@df.uba.ar
[d] e-mail: Martin.Obligado@univ-grenoble-alpes.fr
[e] e-mail: pclarkdileoni@udesa.edu.ar (corresponding author)

data and physics terms already present), and (2) the data used for the data term is updated after a certain number of epochs iteratively, facilitating the assimilation procedure. The Physics-Informed term regularizes the training by enforcing the residuals of the Navier–Stokes equation evaluated over the solution to be close to zero, and thus keeping the solution fluid-like. Stemming from the facts that PINNs can show errors when assimilating turbulent data [11] and that their parallel implementations are a matter of active research [12], we add a second step to our method where we use nudging [13,14], a synchronization-based DA protocol, to increase the resolution of the assimilated fields and increase their compliance with the equations of motion. We remark that an approach based only on (modified) PINNs is possible, using nudging provides added reassurance and simple workaround to some problems (more details below). Also, other techniques based on Generative Adversarial Networks, such as [15–17], could be used to increase the resolution of the assimilated fields, but as these are not equation-based we chose not to consider them.

As stated above, the number of DA tools is vast, and similar objectives can be achieved with other techniques. Many factors come into play when choosing a DA scheme, these range from costs, to desired accuracy, easiness of implementation and versatility. Which one to choose will depend on each application. It is not our intention to provide an exhaustive comparison between the different approaches, but to introduce a new alternative with both merits and drawbacks. Of the many works in the literature involving data assimilation of turbulent flows, two that are very close in spirit to the work presented here are [18,19]. In both cases, variational methods are used to assimilate statistical data into the evolution of a turbulence model in order to improve their accuracy. An overall view of classical DA techniques in the geosciences is presented in [2], a recent review on PINNs and its variants is published in [20], and a detailed comparison between PINNs and adjoint-based methods is performed in [11].

Finally, we would like to note that the DA setup we present here is also similar to the problem of turbulent state generation. In this problem, the goal is to generate a field that resembles a turbulent field without having to solve the Navier–Stokes equations explicitly. Techniques for generating synthetic homogeneous and isotropic turbulent states based on Fourier decompositions go back as far as Kraichnan [21] and have been extended to wall-bounded flows, where the goal is usually to generate inflow conditions for boundary layer or channel flow simulations [22–24]. The idea behind these studies is to generate turbulent states by superposing Fourier modes with random phases, but setting their amplitudes and temporal evolution so that they satisfy Kolmogorov's spectra. Another condition that is often imposed is that the resulting fields should be divergence-free. As we do not use any direct state variable measurements in our procedure, but only assimilate statistical information into a seed flow, we are generating a particular state as the aforementioned techniques do. The key difference is that our seed flow (and the addition of the nudging step too) comes from a fully resolved direct numerical simulation of a homogeneous and isotropic turbulent flow.

The paper is organized as follows. In Sect. 2, we present the preparation method and the upscale procedure. In Sect. 3, we describe the different experiments performed in this work. In Sect. 4.1, we show the results obtained. Finally, in Sect. 5 we outline our conclusions and future lines of work.

2 Methods

2.1 Modifying physics-informed neural networks

The preparation method we present is based on modified PINNs [6]. In its original form, PINNs approximate solutions of partial differential equations. They take space and time coordinates as inputs, and output the desired fields. Their loss function is comprised of a standard L^2 norm of the output and the available data, the data term, and a regularization term composed of the residuals of the partial differential equations, the so-called physics term. During training, the loss function is minimized, and the result is a regression on the data that satisfies the equations of motion. The architecture of the neural network itself is usually a standard fully connected multilayer perceptron. For our purposes, we modified the PINN in two ways. The first is by adding a target term to the loss function which enforces the target constraint that we want to impose. Taking (x, y, z) to be the three Cartesian spatial coordinates, $\boldsymbol{u}^0$ the data available on the velocity field, and $\boldsymbol{u} = (u, v, w)$ the velocity field and p the pressure outputted by the PINN, and using the incompressible Navier–Stokes equations with viscosity ν as our partial differential equations of choice, the total loss function then takes the form

$$L = L_d + \lambda_p L_p + \lambda_t L_t, \tag{1}$$

where

$$L_d = \frac{1}{N_b} \sum_{\{i\}} \left(\boldsymbol{u}_i - \boldsymbol{u}_i^0 \right)^2, \tag{2}$$

is the data term,

$$L_p = \frac{1}{N_b} \sum_{\{i\}} \left[\left(\frac{\partial \boldsymbol{u}_i}{\partial t} + \boldsymbol{u}_i \cdot \boldsymbol{\nabla} \boldsymbol{u}_i + \boldsymbol{\nabla} p_i - \nu \nabla^2 \boldsymbol{u}_i \right)^2 + \left(\boldsymbol{\nabla} \cdot \boldsymbol{u} \right)^2 \right], \tag{3}$$

is the physics term, and L_t is a problem-dependent target term. The subscript i labels the point and time in which the fields are evaluated, i.e., $\boldsymbol{u}_i = \boldsymbol{u}(x_i, y_i, z_i, t_i)$.

The whole set of points is separated into mini-batches, i.e., subsets of the dataset in which the gradient of (1) is evaluated and then used to update the hyperparameters of the network during the training procedure, of size N_b, and the actual set of points $\{i\}$ that makes up each mini-batch is picked at random out of the whole spatio-temporal domain where the problem is defined at each iteration. The hyperparameters λ_p and λ_t act as balancing terms between the different parts of the loss function. In typical neural network fashion, a PINN can then be trained using a mini-batch gradient-descent algorithm such as Adam.

The other modification we introduce is that we update the data term $\boldsymbol{u}^0$ after a certain number of iterations, and then continue with the training. The initial data used when starting the training acts as a seed, and should come from a previously-performed simulation of the partial differential equations (PDEs), which in our case are the Navier–Stokes equations. Once the first training cycle is completed, the seed data is replaced by the output of the PINN at that stage. Thus, in each data update cycle, the data in the data term are closer to satisfying the target term in the loss function. Note that a new seed is needed each time an independent realization of the fields needs to be generated. In DA parlance, the seed data would be the system state coming from a previous forecast, while the output of the whole training procedure would be the analysis.

2.2 Upscaling the prepared fields

Since the numerical study of turbulent flows requires high spatial resolution, we need to be able to increase the resolution of the PINN-generated state. This can be a challenge for neural networks, as computation of turbulent states at large Reynolds numbers usually require significant amounts of memory and computing power. One way of achieving this is by decomposing our spatio-temporal domain and running several PINNs in parallel, as per the C-PINN method [12]. While we rely on this idea to increase the total time window used (more details on this below), it still presents considerable technical challenges when trying to upscale states to very high spatial resolutions. Therefore, we present another way of upscaling the prepared fields based on the nudging technique [13,14]. Nudging works by adding a relaxation term to the r.h.s. of the equations of motion that penalizes the evolved flow when it strays away from some given reference data, $\boldsymbol{u}_{\text{ref}}$. The resulting equations take the form

$$\partial_t \boldsymbol{u} + (\boldsymbol{u}\cdot\boldsymbol{\nabla})\boldsymbol{u} = -\boldsymbol{\nabla} p + \nu\nabla^2\boldsymbol{u} - \alpha\mathcal{I}(\boldsymbol{u}-\boldsymbol{u}_{\text{ref}}), \quad (4)$$

where the last term on the r.h.s. corresponds to the nudging term, which penalizes the distance between the input reference data $\boldsymbol{u}_{\text{ref}}$ and $\boldsymbol{u}$. The amplitude of the nudging term is controlled with α, and $\mathcal{I}$ is a filter operator which acts only where data is available. In this work, $\mathcal{I}$ is a band-pass filter in Fourier space, which projects the spatial part of $\boldsymbol{u}(\boldsymbol{x},t)$ on the range of modes $[k_0,k_1]$ in which the reference field $\boldsymbol{u}_{\text{ref}}$ is known,

$$\mathcal{I}\boldsymbol{u}(\boldsymbol{x},t) = \sum_{k_0\le|\boldsymbol{k}|\le k_1} \hat{\boldsymbol{u}}(\mathbf{k},t)\exp(i\boldsymbol{k}\cdot\boldsymbol{x}). \quad (5)$$

As a rule, the reference field (in our case, the output of the PINN) is also known in a given time interval $[0,T]$, with a time cadence τ. The initial condition used to evolve Eq. (4) corresponds to setting $\boldsymbol{u}(t=0) = \boldsymbol{u}_{\text{ref}}(t=0)$. For the evolution in between successive observations of $\boldsymbol{u}_{\text{ref}}$ (i.e., for time steps shorter than τ), this field is linearly interpolated. Note that Eq. (4) is evolved in time only in the interval $[0,T]$, for which $\boldsymbol{u}_{\text{ref}}$ is available.

Under this kind of setup, nudging has been shown to be able to reconstruct the flow at the scales where information is provided, while filling in the smaller scales with dynamics that must be compatible with the nudged scales even at high Reynolds number [13,14]. Thus, the upscaled field will reproduce the prepared field in the larger scales, and have dynamically consistent turbulent smalls scales. Finally, as to solve Eq. (4) a "classical" PDE solver must be used (e.g., a pseudospectral solver), we have the added advantages of having superior error convergence in the final states. To this end, we can make use of existing and already available highly scalable parallel solvers [25].

Note that the procedure discussed here is different from methods that use large-eddy models to generate the large scales in the flow, and use neural networks to fill in the missing information on the small scales of the flow [26,27]. Instead, here the PINN is used to generate data compatible with the physics and with available large-scale or statistical information (e.g., from observational data or experiments), and a PDE solver is then used to generate physically compatible small-scale flow features. Also note that when utilizing PINNs to solve turbulence models such as the Reynolds-Averaged Navier Stokes equations [20,28], statistical data is used during training, as this is what the models themselves solve for. In our case, we are using statistical data but using the full Navier–Stokes equations, which solve for the full velocity and pressure fields.

3 Experiments

We report on three separate experiments. In all three experiments, the original seed field came from a homogeneous and isotropic forced DNS of the Navier–Stokes equation, performed with a resolution of 32^3 grid points using a pseudospectral method with periodic boundary conditions, in a computational box of size $(2\pi L_0)^3$ and in a time window of $T = 3T_0$, where L_0 and T_0 are, respectively, the characteristic length and time scales of the flow, and amounting to a Reynolds number of $\text{Re} = U_0 L_0/\nu = 80$. The low spatial resolution is associated with limitations of the PINN, but also used to highlight the upscaling procedure. In light of the moti-

vations for this work (cases in which observations or laboratory measurements are incomplete, e.g., with access to statistical information of only one field component), all preparations are performed over the x-component of the velocity. In the seed, this component has zero mean, standard deviation $\sigma_0 = 0.5U_0$, centralized third-order moment $0.175U_0$ and centralized fourth-order moment $0.66U_0$, where $U_0 = L_0/T_0$ is the flow characteristic velocity.

In Experiment 1, we use the method to impose a target mean profile on the x-component of the velocity field with the shape $u_0(y) = 0.1U_0 \sin(y)$. The target term in the loss function given by Eq. (1) then takes the form

$$L_t = \left(\left[\frac{1}{N_b} \sum_{\{i\}} u_i(y) \right] - u_0(y) \right)^2 , \tag{6}$$

where the mini-batch subsets $\{i\}$ are all at different but fixed values of y.

In Experiments 2 and 3, we impose the value of the centralized high order moments moment of u. Their target terms take the general form

$$\begin{aligned} L_t = & \left(\frac{1}{N_b} \sum_{\{i\}} u_i \right)^2 \\ & + \left(\sqrt{\frac{1}{N_b} \sum_{\{i\}} u_i^2 - \left(\frac{1}{N_b} \sum_{\{i\}} u_i \right)^2} - \sigma_0 \right)^2 \\ & + \left(\frac{1}{N_b} \sum_{\{i\}} \left[u_i - \frac{1}{N_b} \sum_{\{i\}} u_i \right]^n - m_0^n \right)^2 , \end{aligned} \tag{7}$$

where the first and second terms are added to keep the mean and standard deviation from changing values. Note that we use the n-th order moment and not its n-th root in the loss function as this results in better convergence. In Experiment 2, we use $n = 3$ and $m_0 = s_0$, while in Experiment 3 we use $n = 4$ and $m_0 = k_0$. While in principle the method works too if one tries to impose the third- and the fourth-order moments simultaneously, this can impose tension when training as the third-order moment tires to skew the distribution, while the fourth tries to symmetrize it. Therefore, due to this fact and the lack of a properly motivating case to include both moments, we decided to not include a fourth Experiment where both moments were modified.

The same neural network architecture was used in all three cases, a fully connected multilayer perceptron with sines as activation functions, in practice a SIREN [29] with 8 hidden layers of 200 neurons each. The spatio-temporal domain of volume $(2\pi L_0)^3$ and time window length $T = 3T_0$ was split into four subdomains along the temporal dimension, so four networks were trained in each experiment with the final results concatenated at the end and treated as a whole. The balancing hyperparameters were chosen to be constant and with values $\lambda_p = 10^{-4}$ and $\lambda_t = 1$, and the mini-batches had $N_b = 10{,}000$. Following standard practices [10], we added input and output normalization layers. We performed four data update cycles, and for each cycle the networks were first optimized for E_0 epochs (where an epoch equals the number of iterations required to cover the whole dataset with mini-batch samples) using a learning rate of 5×10^{-5}, followed by E_1 epochs with a learning rate of 5×10^{-6}. In Experiment 1, $E_0 = 1000$ and $E_1 = 1000$, in Experiment 2 $E_0 = 500$ and $E_1 = 1000$, and in Experiment 3 $E_0 = 1000$ and $E_1 = 2000$.

Once training was completed, the resulting PINNs were evaluated on a uniform grid of 512^3 spatial points every $\Delta t = 3.75 \times 10^{-2}\ T_0$, resulting in a total of 80 snapshots spanning a time window of $T = 3T_0$. These fields were used as the reference fields in the nudging-based upscaling procedure, as described in Sect. 2.2. The nudging was performed with the same solver and boundary conditions used to generate the seed field, but at resolution of 512^3 grid points. The band-pass filter $\mathcal{I}$ acted between $k_0 = 1/L_0$ and $k_1 = 9/L_0$, which corresponds to the range of wave numbers contained in the original 32^3 resolution seed field. The maximum wavenumber in the upscaled fields is $kL_0 = 170$. The Reynolds number of the nudged simulation is of order 1000.

4 Results

4.1 States generated by the PINN

We start by presenting results for Experiment 1. In Fig. 1a, we show the evolution of the data plus target and physics parts of the loss function, while in panel (b) we show the target mean profile compared against the mean profile of the seed and those of the output at the end of each data cycle [labeled as $P0$ to $P3$, and indicated with arrows in panel (a)]. Visualizations at $t = 1.125T_0$ and $z = \pi/L_0$ of the seed (in its original 32^3 grid point resolution) and the final PINN-prepared field (evaluated using 512^3 grid points), i.e., $P3$, are shown in Fig. 2a, b, respectively. Through the successive iterations and cycles, the preparation method is able to impose the mean profile while still resembling the original seed field and also complying with the Navier–Stokes equations (up to some error of order 5×10^{-3} as per L_p in Fig. 1a), thus retaining its fluid-like qualities.

Experiments 2 and 3 show similar results. In Fig. 3a, we show the evolution of the different parts of the loss function for Experiment 2, while in panel (b) of the same figure we show the evolution of the centralized third-order moment normalized by the target value s_0. The dashed red line indicates the target value we want to attain, while the dotted green line indicates the value of the original seed field. The prepared field

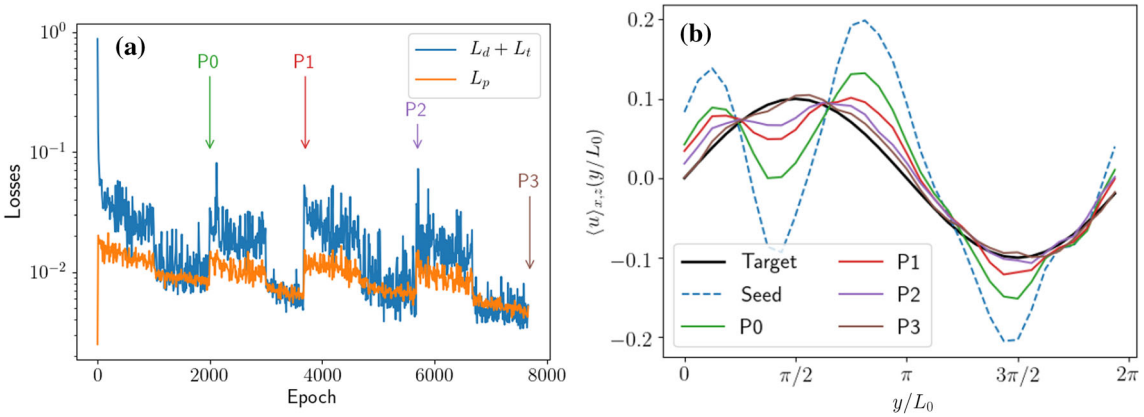

Fig. 1 Results for the training of the PINN in Experiment 1: **a** Evolution of the $L_d + L_t$ and L_p losses as a function of the training epoch. **b** Mean target profile, and mean u profile at $t = 1.125T_0$ for the seed field and for the PINN-prepared field at different instants during training ($P0, P1, P2$, and $P3$, respectively, as marked in panel **a**)

third-order moment converges to the desired value after two or three update cycles of the PINN. In Fig. 2d, we show a visualization of the prepared field, which again shows a resemblance with the seed field, although a close inspection reveals differences with the result prepared in Experiment 1 in Fig. 2b.

Finally, the results for Experiment 3 are shown in Fig. 4, where we plot the evolution of the loss function in panel (a), and the evolution of the centralized fourth-order moment normalized by k_0 in panel (b). The dashed red and dotted green lines in panel (b) indicate the target and original values of the seed, respectively. Compared to Experiment 2, the fourth-order moment converges much faster than the third-order moment. Moreover, the value obtained after convergence fluctuates less after a few data update cycles.

4.2 Upscaling

In this section, we present the results of the nudging-based upscaling procedure. As explained above, the PINN-prepared fields obtained in the previous section are used as reference fields to nudge a simulation and generate upscaled versions (i.e., a version of the fields expected to have physically valid finer details and small-scale features, while keeping the imposed conditions by the PINN). In all cases presented, the nudged simulation was able to synchronize its large-scale motion to the reference data supplied.

Figure 5a shows the energy spectra of the nudged field $\mathbf{u}$, of the reference field $\mathbf{u}_{\text{ref}}$, and of the difference between both fields, $\mathbf{u} - \mathbf{u}_{\text{ref}}$, for Experiment 1. The shaded region of the spectra indicates the range of nudged wave numbers (i.e., the range of scales in which the flow generated by the PINN has relevant information). In that region, the spectrum of the nudged simulation closely follows the spectrum of the reference field, while the spectrum of the difference of the two fields is several orders of magnitude smaller than any of the two fields, indicating that indeed the large scales of the nudged field are synchronized with the large scales of $\mathbf{u}_{\text{ref}}$. For $kL_0 \geq 10$ the nudged simulation fills in the missing scales, as energy cascades to smaller scales following the turbulent dynamics of the Navier–Stokes equations. In other words, the nudged field has a spectrum compatible with a turbulent flow, with an inertial range and a dissipative range. This is further confirmed in Fig. 2c that shows a visualization of the nudged u velocity field (i.e., the upscaled field), compared to the reference (i.e., the prepared field) in panel (b). The nudged field has finer structures while retaining the large-scale characteristics of the prepared field.

Figure 5b shows the mean profile $\langle u \rangle_{x,z}$ (i.e., averaged over x and z) as a function of y/L_0, for the nudged and reference velocity fields. The target mean profile is also shown for comparison. The nudged simulation is able to increase the scale separation of the prepared field (i.e., to create finer fluctuations) while maintaining the imposed mean profile.

Upscaling experiments 2 and 3 leads to similar results. The resulting spectra are similar to those shown in Fig. 5a and as a result are not shown for the other experiments. As in the case of Experiment 1, the nudged fields have finer structures. This fact can be appreciated in the visualizations in Fig. 2e, g, respectively, for Experiments 2 and 3, where, again, the nudged simulations have smaller and finer structures than their reference fields.

In Fig. 6a, b, we show the time evolution of the standard deviation (normalized by the target σ_0) and the centralized third-order moment (normalized by the target s_0) of the x-component of the nudged and reference velocity fields, respectively, for Experiment 2. And in Fig. 7a, b we show the same comparison for the time evolution of the (normalized) standard deviation, and the centralized fourth-order moment (normalized by the target k_0), respectively, for Experiment 3. In both cases, the nudged simulations are able to synchronize to the reference data while also being able to capture the target statistic. The standard deviation of the nudged field departs slightly from the reference as the small scales

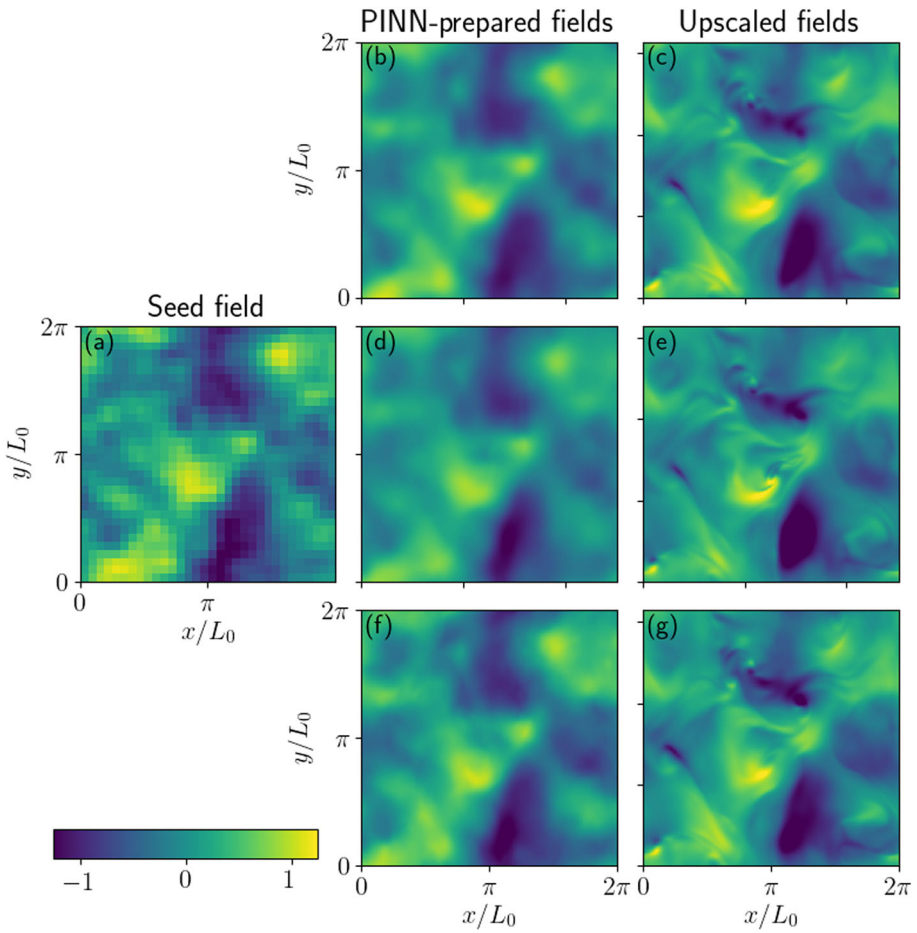

Fig. 2 Visualizations of a two-dimensional slice of **a** the seed (u velocity component), **b** the PINN output when imposing a mean velocity profile with **c** its corresponding nudging-upscaled field, **d** the PINN output with third-order moment imposed and **e** upscaled field, and **f** the PINN output with fourth-order moment imposed and **g** upscaled field. All slices correspond to time $t = 1.125T_0$ and $z = \pi/L_0$

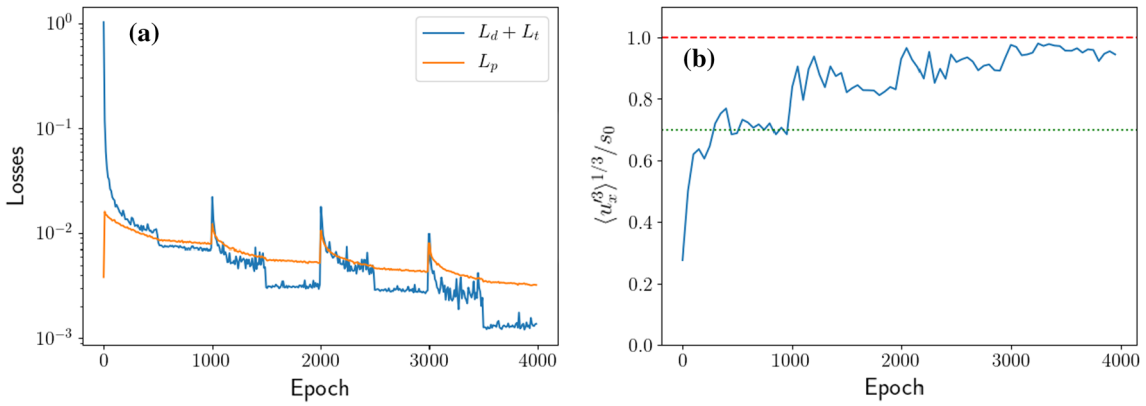

Fig. 3 PINN results for Experiment 2: **a** Evolution of the $L_d + L_t$ and L_p losses as a function of the training epoch. **b** Evolution of the centralized third-order moment normalized by the target s_0, as a function of the training epoch. The dashed red line indicates the target value, while the dotted green line indicates the value of the seed field

are filled by the turbulent energy cascade, but on all cases the generated flows remain in the vicinity of the values imposed for the statistical moments.

5 Conclusions

The physics-informed neural network-powered method we presented is a flexible and powerful tool to prepare turbulent fields given a target statistical constraint. We showed three examples, one in which the target was a mean velocity profile, and two in which the targets were the values of the third- and fourth-order moments of the

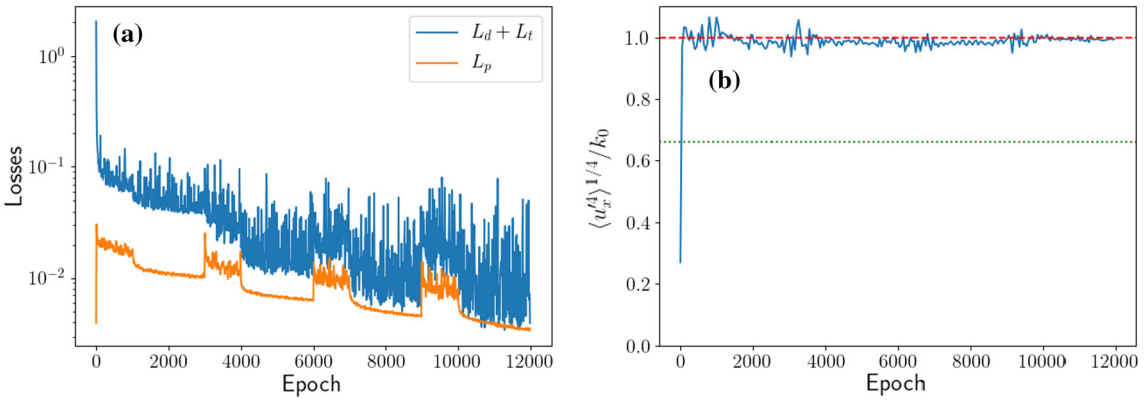

Fig. 4 PINN results for Experiment 3: **a** Evolution of the $L_d + L_t$ and L_p losses as a function of the training epoch. **b** Evolution of the centralized fourth-order moment normalized by the target k_0, as a function of the training epoch. The dashed red line indicates the target value, while the dotted green line indicates the value of the seed field

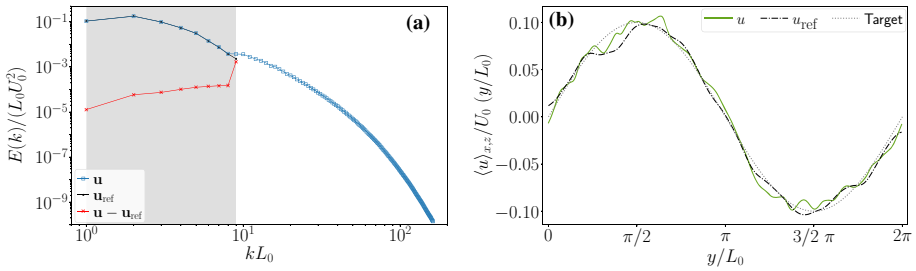

Fig. 5 Nudging of Experiment 1. **a** Instantaneous energy spectra of the nudged field $\mathbf{u}$, the reference field $\mathbf{u}_{\mathrm{ref}}$, and of the difference $\mathbf{u} - \mathbf{u}_{\mathrm{ref}}$. All of the spectra were computed at a fixed time $t/T_0 = 1.125$. The shaded area corresponds to the Fourier modes in which the nudging is imposed, the spectra for $\mathbf{u}_{\mathrm{ref}}$ and $\mathbf{u} - \mathbf{u}_{\mathrm{ref}}$ is only shown in this region. **b** Mean profile of the x-component of the nudged velocity field, as a function of y/L_0 at time $t/T_0 = 1.125$. The profile of the reference field and the target profile are shown for comparison

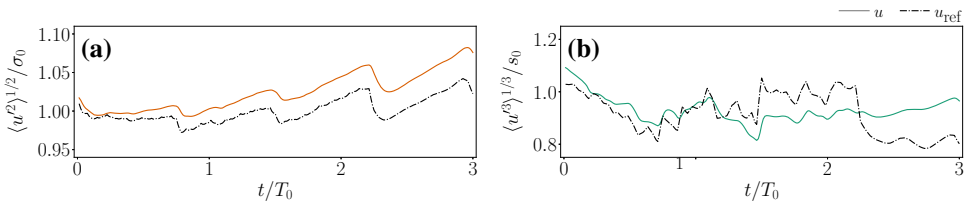

Fig. 6 Nudging of Experiment 2. **a** Time evolution of the standard deviation of u and u_{ref}, normalized by the target value σ_0. **b** Time evolution of the centralized third-order moment of u and of u_{ref}, normalized by the target value s_0

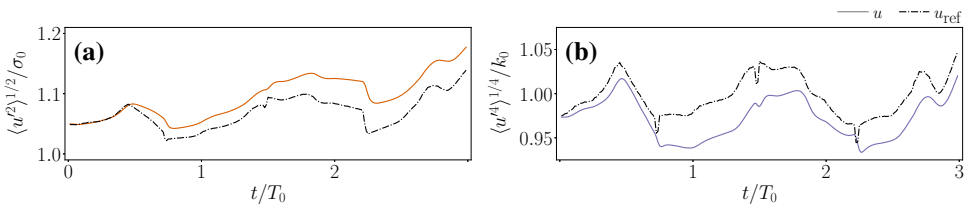

Fig. 7 Nudging of Experiment 3. **a** Time evolution of the standard deviation of u and u_{ref}, normalized by the target value σ_0. **b** Time evolution of the centralized fourth-order moment of u and of u_{ref}, normalized by the target value k_0

velocity field, respectively. In all cases, the method was able to prepare the field while retaining its fluid-like qualities. The method shows an example of the capabilities of deep learning in data assimilation problems.

Furthermore, we presented a nudging-based upscaling procedure which harnesses the power of already available and specialized numerical codes to increase the resolution of the prepared fields. The upscaling procedure is able to increase the scale separation of the prepared field (i.e., of generating physically compatible finer flow features), while maintaining the target statistics. The procedure also acts as a further reinforcement

of the physics contained in the Navier–Stokes equations. The combination of both procedures also provides an example of how neural networks can be combined with traditional numerical modeling of partial differential equations to overcome shortcomings in each separate procedure.

As the method does not use observed data directly, but knowledge gathered from it, it can serve as a way to address problems with highly heterogeneous sources, such as atmospheric flows, where data is obtained from LIDAR measurements [30], satellite observations, and many more sources. Besides applications in data assimilation, the proposed method can be useful in the classical problem of generation of initial conditions for turbulence simulations [31–33], e.g., for the study of freely decaying homogeneous and isotropic turbulence, for grid generated turbulence in wind tunnels, or for turbulent flows with prescribed inflow boundary condition [34–36].

Author contribution statement

S.A. and P.C. conceived and carried out the numerical experiments and analyzed the results. All authors worked on developing the main idea, discussed the results and contributed to the final manuscript.

Data Availability Statement The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

References

1. E. Kalnay, *Atmospheric Modeling, Data Assimilation and Predictability* (Cambridge University Press, Cambridge, 2003). (**Google-Books-ID: zx_BakP2I5gC**)
2. A. Carrassi, M. Bocquet, L. Bertino, G. Evensen, Data assimilation in the geosciences: An overview of methods, issues, and perspectives. WIREs Clim. Chang. **9**(5), 535 (2018). https://doi.org/10.1002/wcc.535. (**Accessed 22 Aug 2022**)
3. G. Evensen, *Data Assimilation: The Ensemble Kalman Filter* (Springer, Berlin, 2006). (**Google-Books-ID: VJ2oOecHhOYC**)
4. C. Cotter, D. Crisan, D. Holm, W. Pan, I. Shevchenko, Data assimilation for a quasi-geostrophic model with circulation-preserving stochastic transport noise. J. Stat. Phys. **179**, 1186–1221 (2020)
5. M. Chantry, H. Christensen, P. Dueben, T. Palmer, Opportunities and challenges for machine learning in weather and climate modelling: Hard, medium and soft AI. Philos. Trans. R. Soc. A Math. Phys. Eng. Sci. **379**(2194), 20200083 (2021). https://doi.org/10.1098/rsta.2020.0083. (**Accessed 11 March 2021**)
6. M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019). https://doi.org/10.1016/j.jcp.2018.10.045. (**Accessed 8 July 2021**)
7. M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. Science **367**(6481), 1026–1030 (2020). https://doi.org/10.1126/science.aaw4741. (**Accessed 6 April 2020**)
8. K. Shukla, P. Clark Di Leoni, J. Blackshire, D. Sparkman, G.E. Karniadakis, Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. J. Nondestr. Eval. **39**(3), 61 (2020). https://doi.org/10.1007/s10921-020-00705-1. (**Accessed 5 Aug 2020**)
9. S. Cai, Z. Wang, F. Fuest, Y.J. Jeon, C. Gray, G.E. Karniadakis, 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. (2021). https://doi.org/10.1017/jfm.2021.135
10. P. Clark Di Leoni, K. Agarwal, T. Zaki, C. Meneveau, J. Katz, Pressure pinns. In Preparation (2021)
11. Y. Du, M. Wang, T.A. Zaki, State estimation in minimal turbulent channel flow: A comparative study of 4DVar and PINN. Int. J. Heat Fluid Flow **99**, 109073 (2023). https://doi.org/10.1016/j.ijheatfluidflow.2022.109073. (**Accessed 1 Dec 2022**)
12. A. D. J. G. E. Karniadakis, Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Commun. Comput. Phys. **28**(5), 2002–2041 (2020). https://doi.org/10.4208/cicp.OA-2020-0164. (**Accessed 26 Jan 2022**)
13. P. Clark Di Leoni, A. Mazzino, L. Biferale, Inferring flow parameters and turbulent configuration with physics-informed data assimilation and spectral nudging. Phys. Rev. Fluids **3**(10), 104604 (2018). https://doi.org/10.1103/PhysRevFluids.3.104604. (**Accessed 14 Dec 2018**)
14. P. Clark Di Leoni, A. Mazzino, L. Biferale, Synchronization to big data: Nudging the Navier–Stokes equations for data assimilation of turbulent flows. Phys. Rev. X **10**(1), 011023 (2020). https://doi.org/10.1103/PhysRevX.10.011023. (**Accessed 28 Feb 2020**)
15. K. Fukami, K. Fukagata, K. Taira, Super-resolution reconstruction of turbulent flows with machine learning. J. Fluid Mech. **870**, 106–120 (2019). https://doi.org/10.1017/jfm.2019.238. (**Accessed 20 Sept 2019**)
16. M. Buzzicotti, F. Bonaccorso, P. Clark Di Leoni, L. Biferale, Reconstruction of turbulent data with deep generative models for semantic inpainting from TURB-Rot database. Phys. Rev. Fluids **6**(5), 050503 (2021). https://doi.org/10.1103/PhysRevFluids.6.050503. (**Accessed 19 May 2021**)
17. L. Yu, M.Z. Yousif, M. Zhang, S. Hoyas, R. Vinuesa, H.-C. Lim, Three-dimensional ESRGAN for super-resolution reconstruction of turbulent flows with tricubic interpolation-based transfer learning. Phys. Fluids **34**(12), 125126 (2022). https://doi.org/10.1063/5.0129203. (**Accessed 15 Dec 2022**)
18. D.P.G. Foures, N. Dovetta, D. Sipp, P.J. Schmid, A data-assimilation method for Reynolds-averaged Navier–Stokes-driven mean flow reconstruction. J. Fluid

Mech. **759**, 404–431 (2014). https://doi.org/10.1017/jfm.2014.566. (**Accessed 18 Aug 2022**)
19. V. Mons, Y. Du, T.A. Zaki, Ensemble-variational assimilation of statistical data in large-eddy simulation. Phys. Rev. Fluids **6**(10), 104607 (2021). https://doi.org/10.1103/PhysRevFluids.6.104607. (**Accessed 10 Aug 2022**)
20. S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics-informed neural networks: Where we are and what's next. J. Sci. Comput. **92**(3), 88 (2022). https://doi.org/10.1007/s10915-022-01939-z. (**Accessed 14 Sept 2022**)
21. R.H. Kraichnan, Diffusion by a random velocity field. Phys. Fluids **13**(1), 22–31 (1970). https://doi.org/10.1063/1.1692799. (**Accessed 17 Aug 2022**)
22. N.S. Dhamankar, G.A. Blaisdell, A.S. Lyrintzis, An overview of turbulent inflow boundary conditions for large eddy simulations (invited), in *22nd AIAA Computational Fluid Dynamics Conference. AIAA AVIATION Forum*. (American Institute of Aeronautics and Astronautics, 2015). https://doi.org/10.2514/6.2015-3213. (**Accessed 17 Aug 2022**)
23. X. Wu, Inflow turbulence generation methods. Annu. Rev. Fluid Mech. **49**(1), 23–49 (2017). https://doi.org/10.1146/annurev-fluid-010816-060322. (**Accessed 17 Aug 2022**)
24. M.Z. Yousif, M. Zhang, L. Yu, R. Vinuesa, H. Lim, A transformer-based synthetic-inflow generator for spatially-developing turbulent boundary layers. arXiv:2206.01618 [physics] (2022). (**Accessed 19 Dec 2022**)
25. P.D. Mininni, D. Rosenberg, R. Reddy, A. Pouquet, A hybrid MPI-OpenMP scheme for scalable parallel pseudospectral computations for fluid turbulence. Parallel Comput. **37**(6–7), 316–326 (2011). https://doi.org/10.1016/j.parco.2011.05.004. (**Accessed 7 Aug 2014**)
26. M. Gamahara, Y. Hattori, Searching for turbulence models by artificial neural network. Phys. Rev. Fluids **2**, 054604 (2017)
27. C. Xie, J. Wang, E. Weinan, Modeling subgrid-scale forces by spatial artificial neural networks in large eddy simulation of turbulence. Phys. Rev. Fluids **5**, 054606 (2020)
28. H. Eivazi, M. Tahani, P. Schlatter, R. Vinuesa, Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations. Phys. Fluids **34**(7), 075117 (2022). https://doi.org/10.1063/5.0095270. (**Accessed 6 Feb 2023**)
29. V. Sitzmann, J.N.P. Martel, A.W. Bergman, D.B. Lindell, G. Wetzstein, Implicit neural representations with periodic activation functions. arXiv:2006.09661 [cs, eess] (2020). (**Accessed 19 April 2022**)
30. S. Le Clainche, L.S. Lorente, J.M. Vega, Wind predictions upstream wind turbines from a LiDAR database. Energies **11**(3), 543 (2018). https://doi.org/10.3390/en11030543. (**Accessed 8 Sept 2022**)
31. C. Rosales, C. Meneveau, Linear forcing in numerical simulations of isotropic turbulence: Physical space implementations and convergence properties. Phys. Fluids **17**, 095106 (2005)
32. P. Lavoie, L. Djenidi, R. Antonia, Effects of initial conditions in decaying turbulence generated by passive grids. J. Fluid Mech. **585**, 395–420 (2007)
33. A. Gronskis, D. Heitz, E. Mémin, Inflow and initial conditions for direct numerical simulation based on adjoint data assimilation. J. Comput. Phys. **242**, 480–497 (2013). https://doi.org/10.1016/j.jcp.2013.01.051. (**Accessed 11 Aug 2022**)
34. L. di Mare, M. Klein, W.P. Jones, J. Janicka, Synthetic turbulence inflow conditions for large-eddy simulation. Phys. Fluids **18**(2), 025107 (2006). https://doi.org/10.1063/1.2130744
35. L. Perret, J. Delville, R. Manceau, J.-P. Bonnet, Turbulent inflow conditions for large-eddy simulation based on low-order empirical model. Phys. Fluids **20**(7), 075107 (2008). https://doi.org/10.1063/1.2957019
36. J. Kim, C. Lee, Deep unsupervised learning of turbulence for inflow generation at various Reynolds numbers. J. Comput. Phys. **406**, 109216 (2020). https://doi.org/10.1016/j.jcp.2019.109216. (**Accessed 8 Sept 2022**)