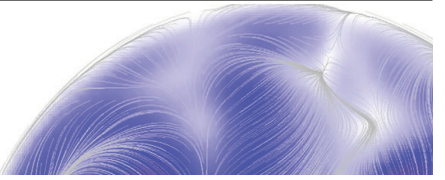
J. Fluid Mech. (2026), *vol.* 1045, P1, doi:10.1017/jfm.2026.12172

JFM PERSPECTIVES



A practical guide to machine learning in fluid mechanics: an encoder–decoder perspective

Kai Fukami[1] **and Koji Fukagata**[2]

[1]Department of Aerospace Engineering, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan

[2]Department of Mechanical Engineering, Keio University, Yokohama 223-8522, Japan

Corresponding author: Kai Fukami, kfukami1@tohoku.ac.jp

(Received 9 September 2026 UTC; revised 9 September 2026 UTC; accepted 9 September 2026 UTC)

This paper discusses how machine learning could support studying fluid mechanics. We first offer the so-called encoder–decoder perspective that can accommodate a range of problem settings fluid mechanicians often tackle with data-driven techniques. Focusing on nonlinear machine-learning analyses, representative efforts on regression, data compression, prediction and system control are introduced, following the given picture of data-driven fluid mechanics. We then provide a step-by-step guideline for studying fluid flows in a data-driven manner – (i) setting the problem up, (ii) curating the data, (iii) constructing a model, (iv) training a model, (v) assessing the machine-learned outcomes, and (vi) improving the techniques – while discussing what we fluid mechanicians should be mindful of. Importantly, each step necessitates carefully incorporating prior knowledge of flow physics to appropriately design machine-learning studies for complex fluid flows, rather than naively applying black-box techniques. We further perform a case study on data-driven compression of a turbulent wake around an aerofoil, offering example approaches for investigating three-dimensional, unsteady turbulent flows. We hope that this perspective will serve as a practical guide for designing machine-learning studies in fluid mechanics.

Key words: machine learning, low-dimensional models

© The Author(s), 2026. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Contents

1 Introduction **2**
1.1 An encoder–decoder perspective of data-driven fluid mechanics 2
1.2 Studying fluid flows with nonlinear machine learning: representative problem set-ups 4

2 A guideline for machine-learning-based analysis of fluid flows **15**
2.1 Problem design 15
2.2 Data curation 20
2.3 Model construction 25
2.4 Model training with cost function design 33
2.5 Assessments 37
2.6 Improvements 39

3 Case study: data-driven compression of turbulent wake around an aerofoil **40**

4 Prospects **48**

References **52**

1. Introduction

Fluid flows are ubiquitous. Due to their high relevance to our lives in various aspects, including transportation, defence, climate and healthcare, understanding and taming fluid flows is of importance in a range of science and engineering fields. While theoretical, experimental and numerical approaches have been considered for the long-term quest to analyse complex fluid flows in a deductive manner, data-driven fluid mechanics has emerged as a new subfield and has rapidly spread in the community to support studying fluid dynamics in an inductive manner (Brunton *et al.* 2020; Taira *et al.* 2025). Such data-driven techniques, often interchangeably referred to as linear/nonlinear machine learning, have proven effective for extracting dominant and hidden patterns residing in complex fluid flows (Brenner *et al.* 2019; Vinuesa & Brunton 2022). This perspective paper discusses how machine-learning-based studies can be designed to deepen our understanding of fluid mechanics, along with a practical guide in order to 'machine-learn' fluid flows.

1.1. *An encoder–decoder perspective of data-driven fluid mechanics*

The use of machine learning has been considered in a range of fluid mechanics problems, including sparse sensor reconstruction (super-resolution; Fukami *et al.* 2023), dimensionality reduction (data compression; Fukagata & Fukami 2025), modal analysis (Taira *et al.* 2017), optimisation/control (Garnier *et al.* 2021) and generation (Du *et al.* 2024). We note that 'machine learning' here indicates both linear and nonlinear data-driven approaches in a broad sense, analogous to a categorisation offered in Brunton *et al.* (2020). Machine-learning-based studies and their approaches in general have often been classified into supervised, semi-supervised and unsupervised, from the optimisation aspect. However, it is worth pointing out that their boundaries become blurred with the rapid growth of data-driven techniques (Taira *et al.* 2025). Such different optimisation algorithms can even be combined together, providing the state-of-the-art techniques used in a range of disciplines.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

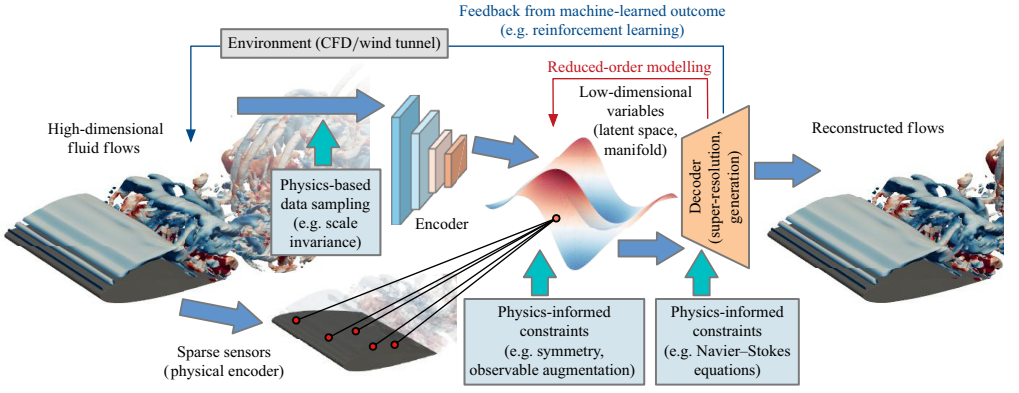


Figure 1. An encoder–decoder perspective of data-driven fluid mechanics.

In this perspective paper, let us instead consider an encoder–decoder perspective to accommodate various problem settings for machine-learning studies of fluid mechanics in a holistic manner, as presented in figure 1. While regular data-driven compression, often performed with linear modal analyses and nonlinear autoencoders, intuitively belongs to this view, the current concept may further cover diverse settings considered in the community, such as inverse problems, reduced-order dynamical modelling and other analyses taking feedback from the outcome of machine-learning models. Readers are referred to a dictionary (Table 4 in Appendix A) for the meaning of main words appearing in this perspective paper.

Cases covering the entire picture from left to right in figure 1 correspond to data compression tasks. They are often expressed with an encoder and a decoder, such that a low-dimensional compressed representation of fluid flows in an ambient space can be derived through the encoder while the reconstruction from the latent representation is performed with the decoder. A range of linear and nonlinear compressions, including proper orthogonal decomposition (POD; Lumley 1967), dynamic mode decomposition (Schmid 2010), nonlinear autoencoder (Hinton & Salakhutdinov 2006) and image/video compression (Anantharaman *et al.* 2023), can be categorised under this umbrella.

In addition, a range of inverse problems, including the reconstruction of a fluid flow state from limited information such as probe measurements, can also be accommodated in this formulation, as illustrated in figure 1. Since sparse sensors reside in the same space as the fluid flow, the process of placing the sensors can be considered a man-made encoding. While such sensors may not be the best low-dimensional representation, data-driven decoders have indeed been designed to bridge the low-order subspace of the sensors and fluid flows. Examples include Gappy POD (Bui-Thanh *et al.* 2004), a library-based linear regression (Callaham *et al.* 2019) and super-resolution reconstruction of turbulent flows (Fukami *et al.* 2019*a*; Erichson *et al.* 2020).

Although not included in figure 1, estimating closure terms in turbulence modelling from the resolved scale information, which has been extensively examined with nonlinear machine learning (Ling *et al.* 2016; Duraisamy *et al.* 2019; Lozano-Durán & Bae 2023), can be classified herein. While the input sensors in figure 1 are replaced with the resolved scale information, what the model aims to estimate, i.e. the reconstructed flows in figure 1, would be the unresolved counterpart. Some literature has grouped closure modelling studies as a class of super-resolution analysis of unsteady flows from the aspect of wavenumber recovery and deconvolution (Maulik & San 2017; Duraisamy 2021; Fukami *et al.* 2023).

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

There exist many efforts on reduced-order modelling of unsteady flows using a low-order subspace identified by some compression techniques, which has often been discussed as machine-learning-based reduced-order modelling. Focusing on whether the temporal integration is performed in a data-driven manner or in an equation-based manner, some studies have discussed them as non-intrusive or intrusive reduced-order modelling (Yu *et al.* 2019). Gaining information from either the original fluid flow state or a low-order subspace as a reward, reinforcement learning has been extensively examined for flow control and shape optimisation to improve their fluid dynamic efficacy. These studies are also covered in the current encoder–decoder perspective, as depicted in figure 1. Each component in this encoder–decoder view can be combined like LEGO blocks – in fact, it is possible to combine a low-order representation identified via nonlinear autoencoder compression with sparse sensor reconstruction for real-time aerodynamic flow estimation (Fukami *et al.* 2025*a*) and reinforcement learning for developing efficient control strategies in unsteady flows with minimal costs (Linot *et al.* 2023; Liu *et al.* 2025*b*).

What is most important and interesting in designing machine-learning studies for fluid flows is that incorporating prior knowledge of flow physics into each step and block in figure 1 not only improves the model performance but also fosters understanding of flow physics. In addition to how we design machine-learning models themselves, how we curate the data given to the model and inform domain knowledge, such as governing equations, is of importance for fluid mechanics-oriented uses of machine learning. In other words, an appropriate set-up of machine-learning studies may enable the use of models as a microscope of flow physics, rather than a sole regressor or optimiser. The guideline provided in § 2 focuses on this aspect for constructing, assessing and improving the models for fluid flow problems. While the idea of incorporating prior knowledge of flow characteristics into the problem design has been applicable for linear data-driven analyses, it becomes particularly important for nonlinear machine-learning analyses with deep networks, as they tend to be powerful but black-box models. In response, this perspective paper hereafter primarily focuses on nonlinear machine learning for fluid mechanics studies.

1.2. *Studying fluid flows with nonlinear machine learning: representative problem set-ups*

While nonlinear machine learning has begun to be actively considered for studying fluid flows since the late 2010s, there are some early stage studies for turbulent drag reduction and flow estimation from measurable information, as presented in figure 2. Lee *et al.* (1997) used a single-layer perceptron to learn the relationship between the wall shear and the control input of the opposition control (Choi *et al.* 1994) for turbulent drag reduction in a channel flow at a low Reynolds number. Because the opposition control approach requires information measured far from the wall, their idea is to estimate the velocity in a flow used for the control design from the measurable surface shear stress. Milano & Koumoutsakos (2002) performed a multi-layer perceptron (MLP)-based estimation of turbulent channel flow from the wall. Motivated by the similarity between POD and a linear neural network, they used MLPs (Rumelhart *et al.* 1986) as a nonlinear extension of POD to estimate the flow field in a low-Reynolds-number channel flow from wall shear information, showing how nonlinear functions inside the network assist in a better reconstruction than linear approaches.

Corresponding to the artificial intelligence spring, which started gradually in the late 2010s, nonlinear machine-learning approaches attracted attention again for fluid mechanics studies (Brunton *et al.* 2020). One of the reasons behind this usage is the intrinsic characteristics of fluid flows – that is, the presence of a finite-dimensional manifold

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

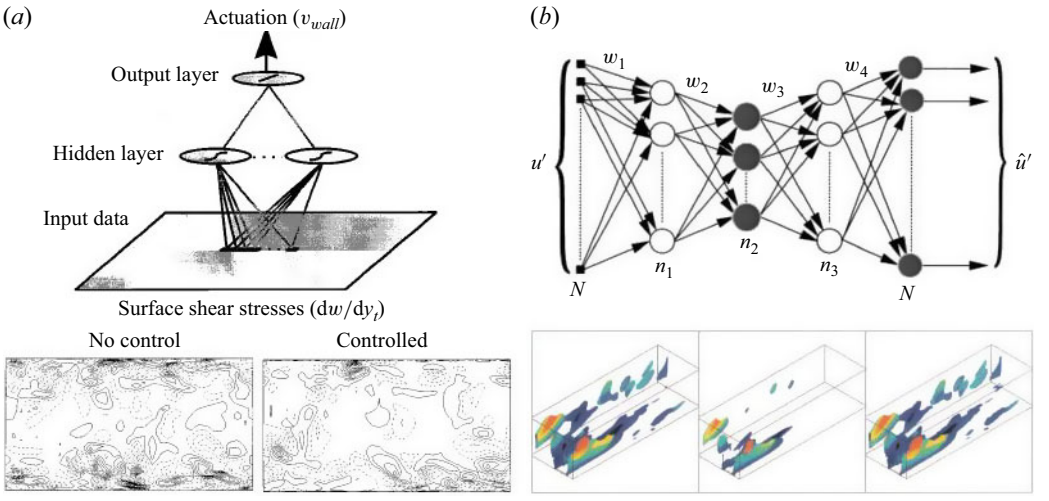


Figure 2. Seminal studies on machine-learning-based analysis for fluid flows. (*a*) A single-layer perceptron-based active control for drag reduction in wall-bounded turbulence (Lee *et al.* 1997). (*b*) A multi-layer perceptron-based compression of turbulent channel flow (Milano & Koumoutsakos 2002). Representative reconstructed flow snapshots are also depicted. Figures are reproduced from the original papers.

in state space (Hopf 1948). While the system of the Navier–Stokes equation is infinite dimensional, it is known that its long-time dynamics lie on a finite-dimensional surface that is sometimes of a much lower order than expected, suggesting the presence of spatiotemporal dominant patterns in a fluid flow (Fukami & Taira 2023). It is then natural that what modern nonlinear machine-learning techniques, well designed to extract the dominant pattern from the massive data sets, tell us about fluid flows in a data-driven manner becomes of interest. In addition, the growth of digital techniques for CPUs (central processing units) and GPUs (graphics processing units), data storage and data transfer enables handling a large collection of computationally/experimentally measured fluid flow data sets that have been accumulated over decades.

Let us introduce some representative problems of fluid mechanics that have been examined with nonlinear machine learning. Estimating integral variables, such as aerodynamic coefficients, from sparse sensors has often been considered, which can be part of either the inverse problem in figure 1, i.e. from the observable to the sensors, or the compression problem if the integral variables of interest are regarded as a low-order representation of fluid flows. An example of estimating the lift coefficient $C_L(t)$ from five sensors $\boldsymbol{s}(t)$ placed in a wake behind a NACA0012 aerofoil with a Gurney flap at a chord-based Reynolds number $Re_c = 1000$ using nonlinear machine-learning techniques is presented in figure 3 (Fukami *et al.* 2020*a*). A flow around a NACA0012 aerofoil with a Gurney flap produces various wake patterns, depending on the angle of attack α and the Gurney flap height h/c normalised by the chord length c. Shown in figure 3 is the case with $h/c = 0.1$ and $\alpha = 20°$, which exhibits time-periodic shedding of two individual vortex pairs presenting two dominant frequencies (Gopalakrishnan Meena *et al.* 2017).

They compare four different machine-learning techniques, MLP, random forest, support vector regression and extreme learning machine, from the aspect of estimation accuracy and computational cost under various training conditions. Appropriate selections of model parameters enable accurate estimation of aerodynamic coefficients even with a few sensors. While this case estimates the lift coefficient, other variables such as drag and momentum coefficients can similarly be considered. In addition, the robustness of the

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

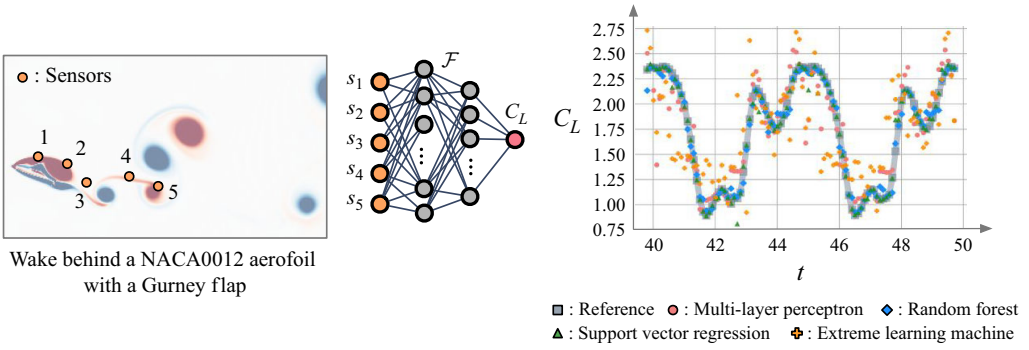


Figure 3. Nonlinear machine-learning-based estimation of the lift coefficient C_L for a flow around a NACA0012 aerofoil with a Gurney flap at $Re_c = 1000$ (Fukami *et al.* 2020*a*). Four different machine-learning models $\mathcal{F} : s(t) \to C_L(t)$ are compared. Figures are reproduced from Fukami *et al.* (2020*a*).

model performance with respect to the number of sensors and measurement noise has also been extensively examined (Fukami *et al.* 2020*a*), which will be introduced in the present guideline as a checkpoint in assessing machine-learning models.

Extracting the dominant characteristics from fluid flows in a low-order manner can be achieved with nonlinear machine learning. The resulting low-order representation often facilitates understanding of flow physics in a compact form while being leveraged for modelling, reconstructing and controlling fluid flows (Fukagata & Fukami 2025). Let us present in figure 4 an example of the nonlinear autoencoder-based analysis for extreme vortex-gust aerofoil interactions (Fukami & Taira 2023). This analysis considers a flow around a NACA0012 aerofoil at $Re_c = 100$ with a range of different angles of attack α while covering various forms of vortex-gust encounter parameterised by gust size, gust strength and its orientation against an aerofoil. As shown in figure 4(*a*), the undisturbed baseline flow at $\alpha = 20°$ is steady, while that for $\alpha \gtrsim 30°$ presents periodic wake shedding at this Reynolds number. In contrast, the flows disturbed by the gust show massive separation due to strong vortex-aerofoil interactions with a large excitation of aerodynamic responses within a short time duration.

A variant of the nonlinear autoencoder, an observable-augmented autoencoder that incorporates an observable variable of the lift coefficient C_L into the subspace identification illustrated in figure 4(*b*), is considered for the data compression of an extreme vortex-gust encounter. As discussed in § 2 later, embedding prior knowledge into the design of the model and optimisation is important to successfully extract key insights from fluid flow data. The choice of lift coefficient here is inspired by the aerodynamic prior knowledge of $\Gamma \propto C_L$, where Γ is a circulation (Fukami & Taira 2023). They found that a collection of vorticity snapshots exhibiting a variety of transient vortical patterns in a large parameter space is compressed to merely three variables, while linear techniques require approximately 300 dimensions for the same reconstruction level. While the $\xi_1 - \xi_2$ plane describes the phase and amplitude information of the aerodynamic flows, the ξ_3 axis is used to represent the effective angle of attack in this subspace. Approaches for the decision of the latent dimension will be discussed later.

Furthermore, there exists a coherent geometric structure, a submanifold, in the identified three-dimensional subspace, as depicted in figure 4(*c*). Here, the trajectories emphasised in colour correspond to the projected time trace of the undisturbed flow snapshots while those coloured in grey are encoded from the whole set of time-varying disturbed flows. For the undisturbed cases, the steady flow at $\alpha = 20°$ and the periodic shedding for $\alpha \geqslant 30°$ are expressed in a reduced-order manner as a set of a single dot and circles of different

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

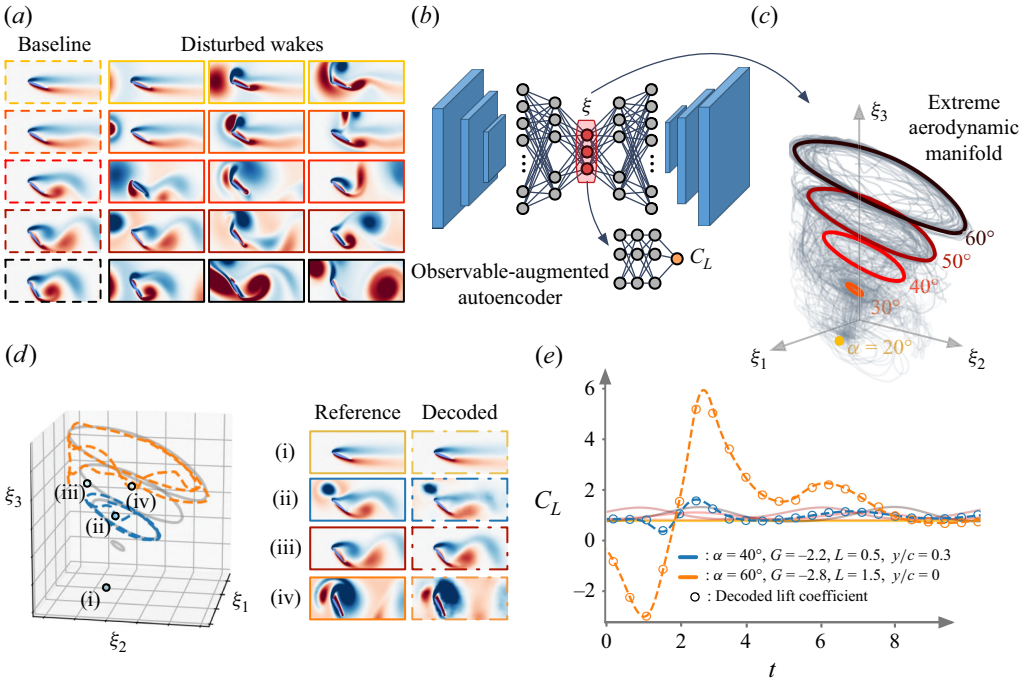


Figure 4. Nonlinear autoencoder-based analysis of extreme vortex-gust aerofoil interactions (Fukami & Taira 2023; Fukami *et al.* 2024*b*). (*a*) Data sets collected through direct numerical simulations, spanning over a huge gust-parameter space. (*b*) Observable-augmented convolutional autoencoder. (*c*) The identified low-order submanifold $\boldsymbol{\xi}$. (*d*) Representative snapshots projected to the points (i)–(iv) along with the reconstructed fields through the decoder. (*e*) The estimated lift coefficients C_L through the subnetwork. Figures are reproduced from the original papers.

sizes. The increment of radius for the circles with the angle of attack corresponds to that in the lift fluctuation from the mean state at each angle of attack.

Flow snapshots and latent trajectories of representative gusty cases are shown in figure 4(*d*). While the model is able to reconstruct a flow field, the effect of the vortex gust about the baseline dynamics is described in a low-order manner. These latent representations can also be used to estimate the lift coefficient $C_L(t)$, as presented in figure 4(*e*). As described with an encoder–decoder perspective in figure 1, an appropriate combination of nonlinear machine-learning models enables performing data compression and aerodynamic load estimation together. The physically interpretable shape of the latent manifold suggests that nonlinear machine learning is capable of extracting important vortical structures associated with lift through the observable augmentation. Their following studies have also revealed that this cone-shaped manifold can call for phase-amplitude modelling, enabling the optimal control design in the latent space to swiftly mitigate the impact of vortex gusts (Fukami *et al.* 2024*b*). This identification of a physically explainable manifold can only be achieved when the interpolation of the lift coefficient is performed in the autoencoder model (Fukami & Taira 2023).

Such nonlinear data compression can also be applied to experimental measurements. Let us introduce a nonlinear autoencoder-based analysis for transverse gust encounter data at $Re_c = 10\,000$ measured through particle image velocimetry (PIV; Smith *et al.* 2024). While the numerical example above considers leveraging prior knowledge with an observable variable, this study rather examines directly incorporating the aerodynamic

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

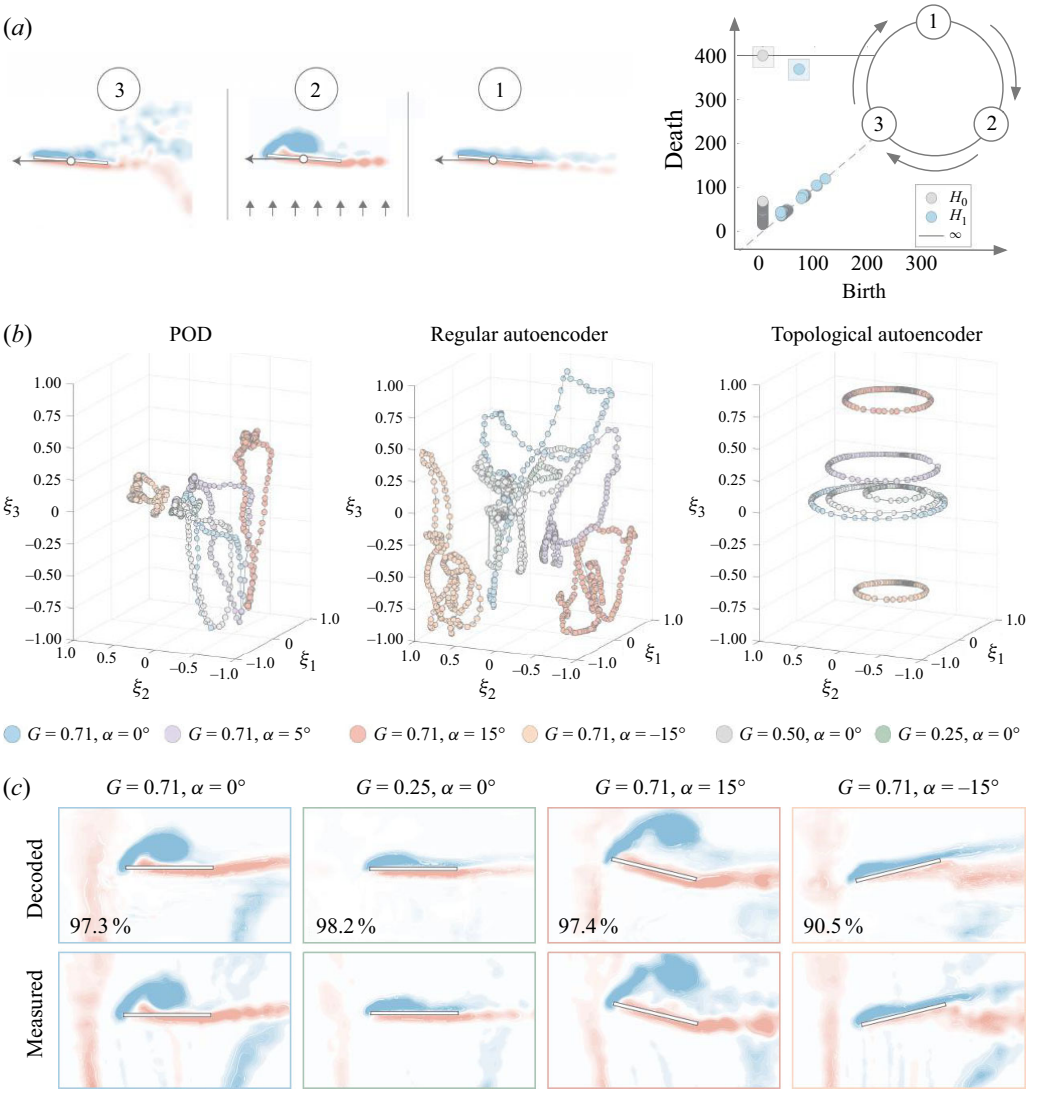


Figure 5. Nonlinear autoencoder-based analysis of transient gust encounter with experimental measurements (Smith *et al.* 2024). (*a*) A cyclic perspective on transient gust encounter. A birth–death plot is also presented. (*b*) The three-dimensional latent space identified by POD, a regular autoencoder and the topological autoencoder. (*c*) Representative reconstructed flow snapshots. The structural similarity index measure (SSIM) between the experimental measurement and the reconstructed flow field is reported inside each flow contour. Figures are reproduced from Smith *et al.* (2024).

observation into its optimisation process. They posit that a transient gust encounter can be described as a circle in state space since their transient behaviour is cyclic, as illustrated in figure 5(*a*). Indeed, the flat plate first passes through the gust ($1 \rightarrow 2$), the separating structures convect ($2 \rightarrow 3$) and then the flow field comes back to the statistically stationary state ($3 \rightarrow 1$).

This cyclicity of a transient gust encounter is mathematically extracted from the given data using persistent homology (Edelsbrunner & Morozov 2014). Although detailed formulations are omitted, the persistent homology analysis counts the number of holes in state space as that of pairs in the so-called birth–death plot, characterising a point cloud

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

based on its topology, or the connectedness of nearby points. As a result, this approach enables the description of the underlying shape from which the data points are sampled. For the case with a transient gust encounter, this technique extracts a single pair in the birth–death plot, as depicted in figure 5(*a*), suggesting that the dynamical behaviour of a transient gust encounter can indeed be described as a circle.

Based on the observation above, a topological autoencoder that constrains the persistent homology in a low-order subspace and the ambient space to be consistent has been introduced for experimental measurements of a transient gust encounter (Smith *et al.* 2024). Compared with a subspace extracted with POD and a regular nonlinear autoencoder, one with the topological autoencoder presents an organised orientation of multiple circles in figure 5(*b*), corresponding to multiple cases with different gust and wing parameters. In addition, the reconstructions are in agreement with the reference measured data, indicating that the identified space composed of cyclic trajectories is regarded as a low-order representation of the transient gust encounter. Such a flow observation-based augmentation can support extracting underlying flow physics in a low-order manner with nonlinear machine learning even for measurements with experimental noise.

The aforementioned observations with nonlinear data compression indicate that dominant patterns or characteristics of fluid flows can often be described with a few dimensional variables. In other words, it may be possible to reconstruct fluid flows from limited measurements if there exists a basis that bridges the low-order and the ambient spaces with minimal loss of information. As such, machine learning has been considered to recover fluid flows from limited information in space and time, corresponding to the decoder portion in figure 1.

These efforts on reconstructing flows with sparse data are often discussed with super-resolution analysis (Fukami *et al.* 2023). The concept of super-resolution has been traditionally examined in computer vision through various techniques such as interpolation, high-frequency transfer and neighbour embedding. While these methods are relatively simple to implement, they often struggle to accurately reconstruct high-frequency details. To overcome this limitation, nonlinear machine learning has been increasingly used for accurate image super-resolution. These models excel at mapping the complex, nonlinear relationships between input and output data, enabling high-quality reconstruction even under ill-posed conditions.

While linear data-driven techniques, including Gappy POD (Bui-Thanh *et al.* 2004), linear stochastic estimation (Nakamura *et al.* 2022) and library-based regression (Callaham *et al.* 2019), can be used for estimating fluid flows, an idea of leveraging nonlinear machine learning to extract a relationship between high-resolution data and its low-resolution counterpart has also been considered for turbulent flows. An example of neural network-based super-resolution analysis for turbulent flows is presented in figure 6 (Fukami *et al.* 2019*a*). Considering two-dimensional decaying homogeneous isotropic turbulence, Fukami *et al.* (2019*a*) revealed that it is possible to reconstruct unsteady flows on a 128^2 grid from grossly coarse data on a mere 4^2 grid. They highlight that it is important to carefully design a nonlinear super-resolution network based on prior knowledge of turbulence to successfully recover fine-scale shear and rotational structures, while non-physical artefacts appear in the reconstruction with a sole application of a deep neural network, such as a vanilla convolutional neural network (CNN), as illustrated in figure 6.

While they focused on two-dimensional decaying turbulence, a regime characterised by minimal energy transfer to small length scales, super-resolution analysis has subsequently been extended to a range of flows, including forced turbulence (Fukami *et al.* 2023). For

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

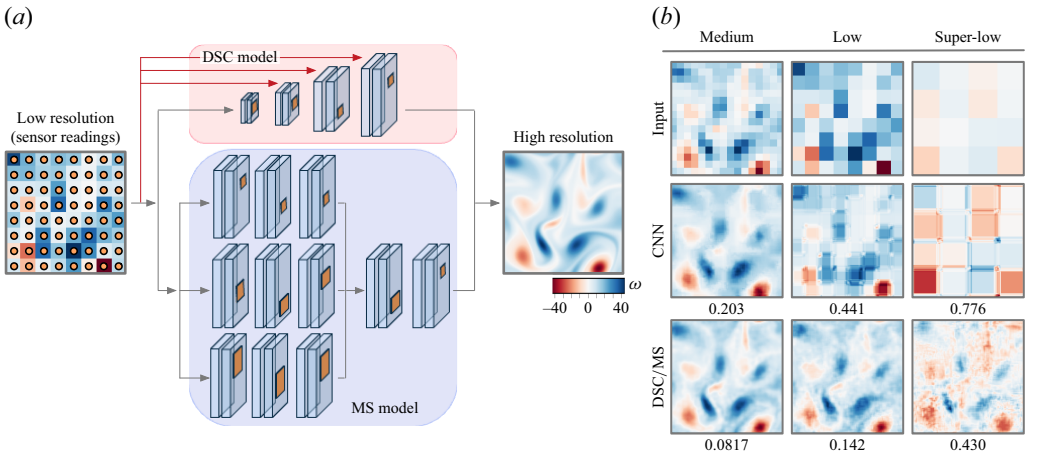


Figure 6. Super-resolution reconstruction of turbulent flows with machine learning (Fukami *et al.* 2019*a*, 2021*a*, 2023; Fukami & Taira 2024). (*a*) Sparse sensor reconstruction as a super-resolution problem, illustrated with an example machine-learning architecture of the hybrid downsampled skip-connection/multi-scale (DSC/MS) model (Fukami *et al.* 2019*a*). (*b*) Super-resolution of two-dimensional decaying turbulence (Fukami *et al.* 2019*a*). The L_2 reconstruction error norm is reported underneath each contour. Figures are reproduced from the original papers.

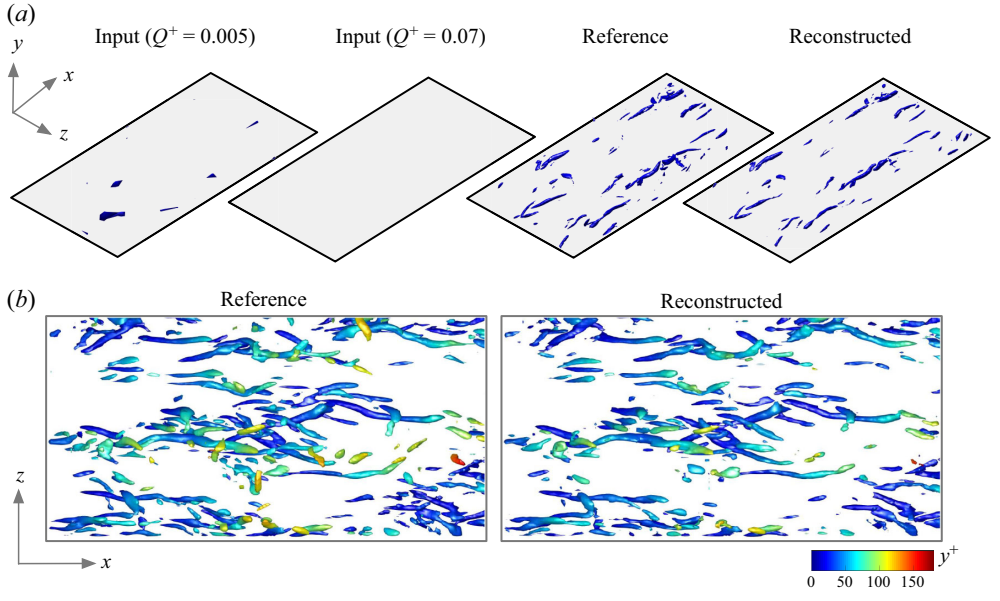


Figure 7. Super-resolution of turbulent channel flow (Fukami *et al.* 2021*a*). (*c*) The three-dimensional view of flow fields visualised with a Q isosurface. (*d*) The $x-z$ sectional view. Figures are reproduced from the original papers.

example, such an appropriately designed nonlinear network can be used to reconstruct a three-dimensional turbulent flow, as presented in figure 7 (Fukami *et al.* 2020*a*, 2021*a*). We will discuss how the incorporation of prior knowledge of turbulence into a model design can be achieved in § 2.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

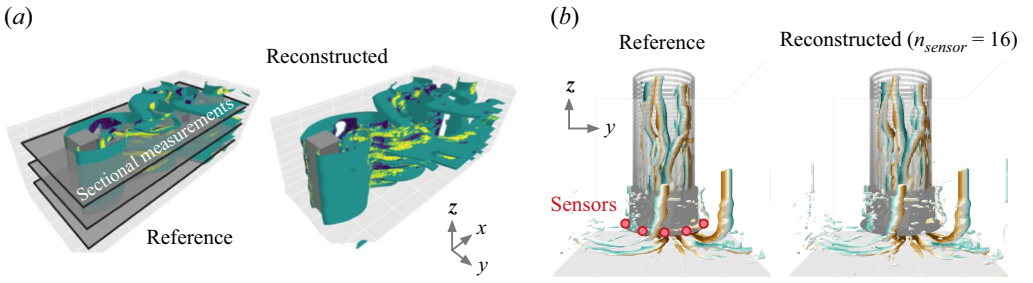


Figure 8. Examples of generalised super-resolution for unsteady flows. (*a*) Three-dimensional reconstruction of a bluff body wake from limited sectional measurements (Matsuo *et al.* 2024). (*b*) Sparse sensor reconstruction for industrial turbulence of flow in a pump sump (Fukami *et al.* 2022).

While super-resolution has been originally used for an image- or video-shaped low-resolution input of fluid flows, being such a shape is not necessary. In discussing the reconstruction of high-resolution flow data from its low-resolution counterpart, any different styles of low-resolution data in space and time could be considered. Considering experiments in which sectional measurements only are available, reconstructing the full picture of three-dimensional fluid flows from limited two-dimensional measurements has been examined, as illustrated in figure 8(*a*) (Matsuo *et al.* 2024). Equipped with generative models, it is also possible to reconstruct three-dimensional flow states across the flow variables, e.g. from the wall-normal velocity v to the streamwise velocity u, which is promising for experimental scenarios (Yousif *et al.* 2023*a*).

Sparse sensor reconstruction has been extensively studied using nonlinear machine learning, as a set of sensor readings can be viewed as a low-resolution representation of fluid flows. In this aspect, let us again mention that super-resolution reconstruction is also covered in the encoder–decoder perspective shown in figure 1. While a variety of fundamental flows, such as bluff body wakes (Marra *et al.* 2024), a flow around a wing (Zhong *et al.* 2023) and wall-bounded turbulence (Cuéllar *et al.* 2024), have been studied, complex vortical structures in industrial turbulence can be reconstructed even from sparse sensors. Shown in figure 8(*b*) is a representative snapshot of the numerically simulated field of turbulent flow in a pump sump at a high Reynolds number compared with that of machine-learning-based reconstruction (Fukami *et al.* 2022). The field of the second invariant of the velocity gradient tensor is successfully recovered from pressure sensors placed on the bell mouth. Identifying such adverse vortices that often produce noise, energy loss and vibrations would be critical to support designing energy-efficient fluid-based machines.

A number of studies have pointed out that super-resolution can be a general framework for a broad range of applications in fluid mechanics. Super-resolution analysis can be extended to denoising problems if noisy experimental measurements are considered as low-resolution fluid flows (Scherl *et al.* 2020). Large-eddy simulation can involve super-resolution reconstruction by revealing fine structures inside a low-resolution grid cell (Maulik *et al.* 2019; Duraisamy 2021). These discussions on a range of inverse problems in fluid mechanics from the super-resolution viewpoint are found in Fukami *et al.* (2023). The present paper also covers a combination of nonlinear data-driven compression and sparse sensor reconstruction as a case study offered in § 3.

In addition to reconstructing fluid flow states in an instantaneous manner, predicting the dynamics of complex fluid flows over time is of interest. Machine learning has been considered to that end, in particular with the assistance of data-driven compression. Its generic framework is essentially the same as typical reduced-order models composed of a

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

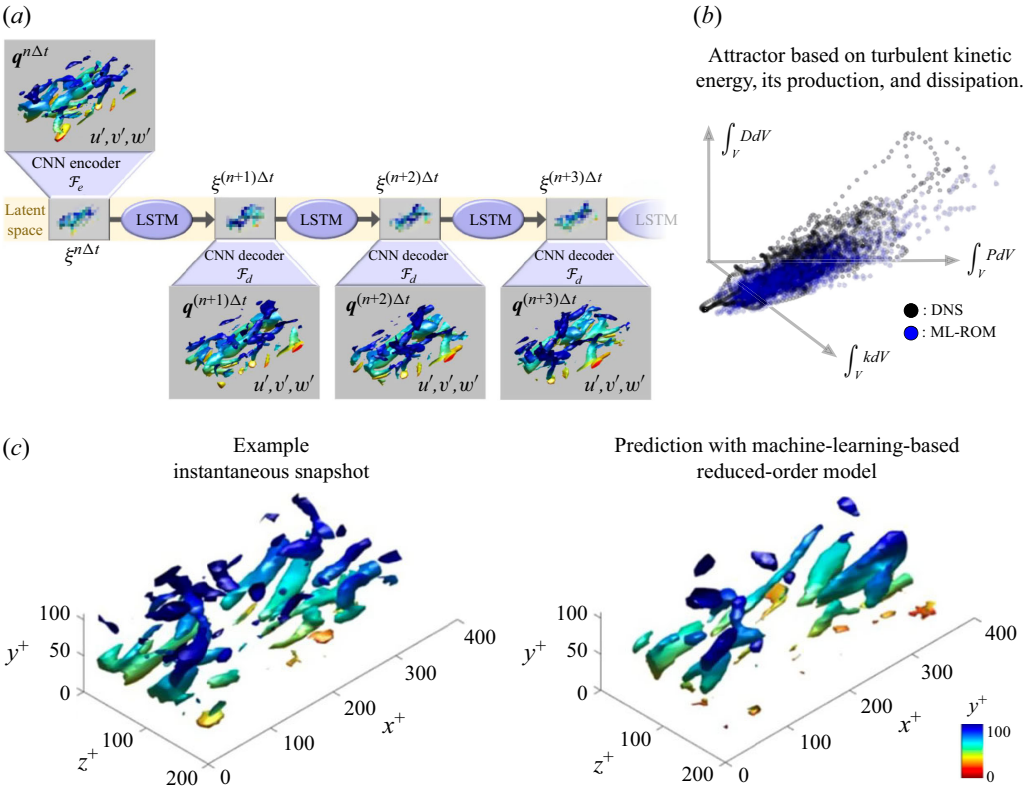


Figure 9. Machine-learning-based reduced-order modelling of a minimal turbulent channel flow (Nakamura *et al.* 2021). (*a*) A general framework of latent reduced-order modelling composed of a spatial dimensional compressor and a temporal latent integrator. Comparison of (*b*) attractor and (*c*) flow field between the reference simulation and the machine-learning-based prediction. Figures are reproduced from the original papers.

spatial dimensional compressor and a temporal integrator, as illustrated in figure 9(*a*). While POD and nonlinear autoencoders can be used as a dimensional compressor to obtain low-order representations $\boldsymbol{\xi}$ of fluid flows, their latent dynamics, $\dot{\boldsymbol{\xi}} = \mathcal{A}(\boldsymbol{\xi})$, are modelled with the integrator for an operator $\mathcal{A}$. In a linear fashion, POD with the Galerkin projection is often used to model the dynamical system of unsteady flows in a low-order manner (Holmes *et al.* 1998; Rowley & Dawson 2017). Since the temporal evolution of high-dimensional fluid flows is obtained via a decoder, this problem set, where the latent dynamics is integrated with an operator $\mathcal{A}$ in an autoregressive manner, can also be regarded as part of the present encoder–decoder perspective, as illustrated in figure 1.

With nonlinear machine-learning models, a dynamical operator $\mathcal{A}$ is often approximated with neural networks $\mathcal{F}_t$, such as MLP (Fukami *et al.* 2019*b*), long short-term memory (LSTM; Srinivasan *et al.* 2019), attention model (Gao *et al.* 2024), neural ordinary differential equations (Linot & Graham 2023) and transformer (Solera-Rico *et al.* 2024), in a discrete manner, such that $\boldsymbol{\xi}(t + n\Delta t) = \mathcal{F}_t(\boldsymbol{\xi}(t))$. While fluid flows dynamically behave as a Markov process (Norris 1998), whose next state depends only on the state attained in the previous step, some studies have also considered accounting for multiple time steps, i.e. $\boldsymbol{\xi}(t + n\Delta t) = \mathcal{F}_t(\boldsymbol{\xi}(t), \boldsymbol{\xi}(t - n\Delta t), \ldots)$, to improve the prediction stability of unsteady flows (Hasegawa *et al.* 2020; Tomasetto *et al.* 2025).

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

An example of machine-learning-based reduced-order modelling for turbulent channel flow (Nakamura *et al.* 2021) is presented in figure 9. In this case, a CNN-based autoencoder is used to reduce the data dimension, while a LSTM is used as a temporal integrator in the low-order latent space. They reported that both the spatial compressor and the temporal integrator need to be carefully designed to learn the multiple spatiotemporal length scales of wall-bounded turbulence, which will partly be discussed in § 2 in this paper.

In addition to vortical structures, long-term dynamics examined with the attractor in a three-dimensional phase space of the production P, the turbulent kinetic energy k and the dissipation D are well reproduced by nonlinear machine learning, as depicted in figures 9(b) and 9(c). A strongly intermittent event far from the basin of the attractor is not sufficiently captured, which can be improved by means of different techniques, such as a transformer as a temporal integrator (Solera-Rico *et al.* 2024). To mitigate the black-box nature in constructing a machine-learning-based temporal integrator, one can consider using grey-box approaches, such as sparse identification of nonlinear dynamics (Brunton *et al.* 2016; Fukami *et al.* 2021*d*; Fukami & Shoji 2026) and Koopman theory-based constraints (Lusch *et al.* 2018). They may result in more describable models compared with neural networks while offering insights into flow control (Ishize *et al.* 2024).

For flow control and shape optimisation, nonlinear machine learning, particularly reinforcement learning, has been considered (Rabault *et al.* 2020). The efforts on flow reconstruction and dynamical modelling introduced above are often based on supervised and unsupervised algorithms that learn the nonlinear input–output relationship and extract dominant characteristics from fluid flow data. In contrast, reinforcement learning optimally derives the time-varying action (agent) $a(t) = \pi(\boldsymbol{q}(t))$ by receiving feedback from a given state $\boldsymbol{q}(t)$. The time-varying action becomes a control input for the system control purpose. For the sake of shape optimisation, the notion of time here can be replaced with that of iteration.

Reinforcement learning-based analyses are also part of the present encoder–decoder perspective in figure 1. However, there is one key difference from other problem sets based on supervised and unsupervised learning techniques – that is, receiving feedback from the state as a reward. The state can be either a numerical solver for flow simulation or a wind tunnel, depending on the analysis of interest. To define the reward that expresses what a good or bad action is for the agent, the state then needs to be encoded into a subspace. While sparse sensors and aerodynamic coefficients are often chosen for the construction of the reward function corresponding to the subspace definition, one can consider using compressed representations of high-dimensional fluid flows obtained through POD and an autoencoder, which will be introduced later. Regardless of the definition of the state, the machine-learning model then assesses the effect of the action translated into the state, which can be regarded as a decoding process. In other words, the overall procedure resides in the encoder–decoder picture in figure 1, based on the feedback from the outcome of machine-learning models.

Reinforcement learning has been actively applied for a range of control objectives in unsteady flows, including drag reduction of wall-bounded turbulence (Guastoni *et al.* 2023; Sonoda *et al.* 2023), improving aerodynamic performance (Rabault *et al.* 2019; Paris *et al.* 2023; Garcia *et al.* 2025) and suppressing flow instabilities (Xu & Zhang 2023). They can provide better control performance and generalisable control strategies compared with what has been achievable with traditional techniques. However, taking the entire flow field as a state is computationally expensive, which becomes a unique bottleneck when applying reinforcement learning for flow control because reinforcement learning requires collecting the time-series information of states in training an agent.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

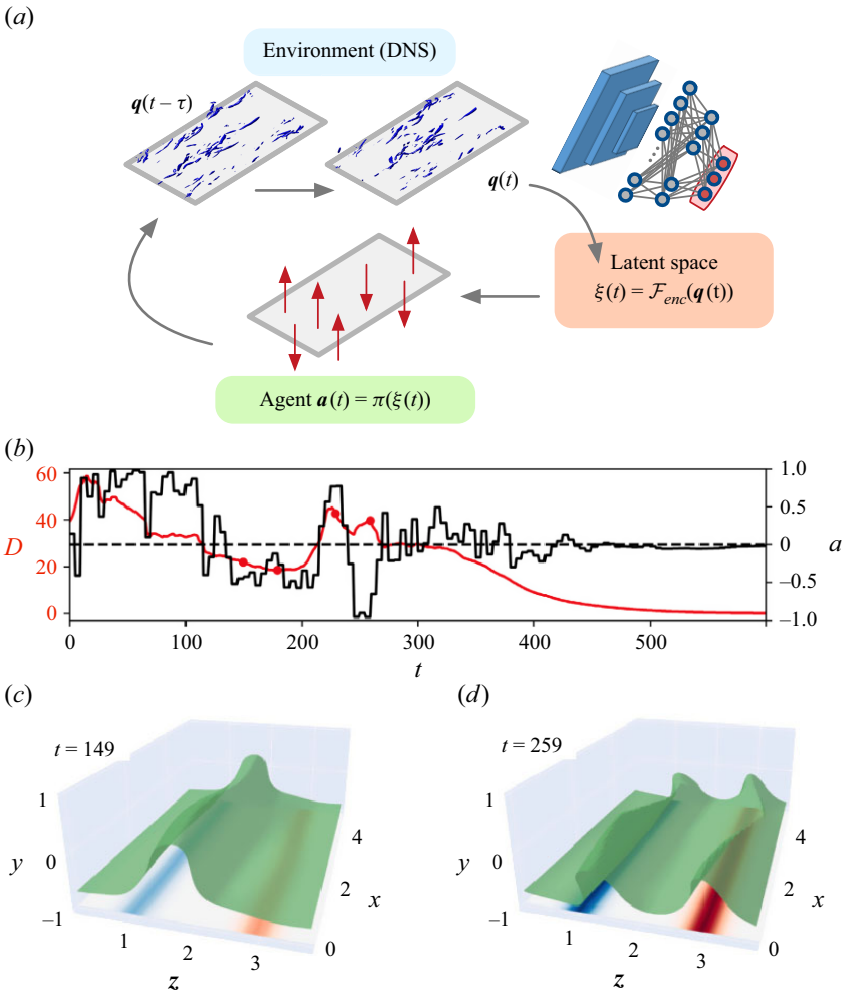


Figure 10. An example usage of machine-learned outcomes for flow control. (*a*) Reinforcement learning for control design in the latent space. (*b* – *d*) Reinforcement learning-based latent control design for plane Couette flow (Linot *et al.* 2023): (*b*) time history of drag D and action a; panels (*c*) and (*d*) show representative flow snapshots. Figures are reproduced from the original papers.

There are several remedies to address this scalability issue. One can leverage multi-agent reinforcement learning (Albrecht *et al.* 2024), which gains domain knowledge from multiple learning agents that coexist in a shared environment. This is particularly useful for unsteady flows possessing a large degree of freedom since the multi-agent approach enables the mitigation of the curse of dimensionality by exploiting invariances through training on local pseudo environments while collaborating among agents to achieve a global control objective (Bae & Koumoutsakos 2022). Its effectiveness for flow control has been demonstrated with several flow examples, including a turbulent cylinder wake (Suárez *et al.* 2025) and Rayleigh–Bénard convection (Vasanth *et al.* 2025).

Otherwise, an autoencoder-assisted reinforcement learning control of unsteady flows, which monitors latent variables $\boldsymbol{\xi}(t)$ as a state instead of the entire flow field, can be used (Linot *et al.* 2023), as illustrated in figure 10. Considering a Couette flow featuring two wall-mounted slot jets, they designed a reinforcement-learning control strategy operating

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

within a latent space obtained through a nonlinear autoencoder. The primary goal is to minimise the average turbulent drag D across both the top and bottom walls, while applying a quadratic penalty to the amplitude $a(t)$ of the actuation on the bottom wall. By training the agent to optimise $a(t)$ using a 25-dimensional low-order latent representation $\boldsymbol{\xi}(t)$, which is significantly smaller than the original flow state dimension, they enable a highly scalable analysis.

As illustrated by the red curve in figure 10(b), the turbulent drag D is successfully reduced over time. Furthermore, the agent trained in the low-order latent space uncovers a novel drag-reduction mechanism. It basically manipulates a low-speed streak into a specific location to force its breakdown, which subsequently causes two new low-speed streaks in its wake. Because these resulting streaks cannot be sustained within the flow domain, the self-sustaining turbulence cycle is disrupted, ultimately leading to laminarisation (Linot *et al.* 2023). This demonstrates that applying reinforcement learning in a latent space can yield cost-effective, nonlinear control policies that are otherwise challenging to derive through human intuition alone. Furthermore, the optimised time-varying actuation waveforms may offer valuable physical insights into mechanisms for improving their fluid dynamic performance (Godavarthi *et al.* 2023).

While these successes illustrate the transformative potential of machine learning, they simultaneously underscore the dangers of treating data-driven tools as purely black-box techniques. Such models may be prone to discovering spurious correlations, violating fundamental physical laws and failing to generalise across different flow regimes. To address this, we next offer a practical guideline for studying fluid flows in a data-driven manner, detailing what we fluid mechanicians should be mindful of throughout the research lifecycle. Crucially, we argue that successful modelling of complex fluid dynamics cannot rely on the naive application of off-the-shelf algorithms. Instead, machine-learning studies for fluid mechanics require carefully weaving prior knowledge of flow physics into every facet of the study design to ensure that machine-learned outcomes respect fundamental physical characteristics.

2. A guideline for machine-learning-based analysis of fluid flows

In this section, let us provide a practical guideline for studying fluid flows with machine learning. This paper considers six step-by-step procedures; namely, (i) setting the problem up, (ii) curating the data, (iii) constructing a model, (iv) training a model, (v) assessing the machine-learned outcomes, and (vi) improving the techniques, as graphically summarised in figure 11. It is important to note that each step necessitates a fundamental understanding of fluid mechanics, since how we proceed with each step highly depends upon what flows we consider and the physics we are interested in. Although machine learning can not work magic, it greatly supports studying fluid flows when prior knowledge of flow physics is appropriately embedded into the machine-learning process.

2.1. *Problem design*

This is the beginning and most important step for successful analysis with machine learning of fluid flows. Now, the question is what we should concretely do in designing a problem. Here, let us define that properly designing a problem corresponds to setting an appropriate mathematical optimisation problem based on given variables, data and environments. Depending on what we aim, i.e. what is set as the input and the output from the perspective picture in figure 1, how the optimisation needs to be prepared is different. This also affects the later steps of how we should curate the data, prepare a model and assess it, because there exist proper approaches and models at each step for different

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

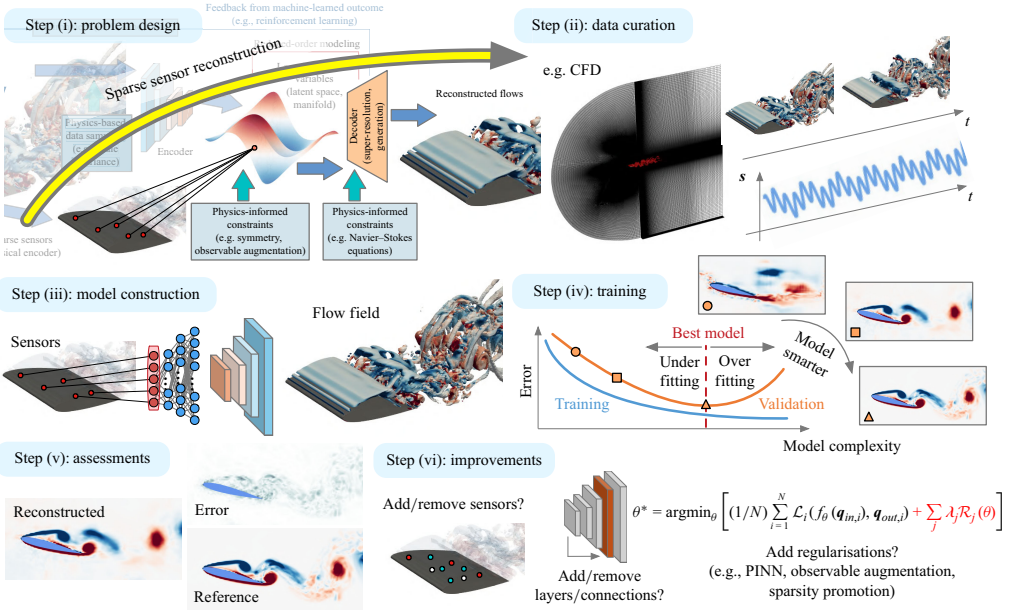


Figure 11. A flow chart of machine-learning-based studies for fluid flows.

problem set-ups and flow characteristics that we consider. Since machine learning is an optimisation, appropriately setting up the problem as a form of optimisation is key to efficiently extracting physical insights from fluid flows. Hereafter, we discuss how such a mathematical input–output relationship can be formed for different tasks in fluid mechanics to study them with the assistance of machine learning.

2.1.1. *Estimation and reconstruction*

The problem of estimating unknown or previously unmeasurable states based on limited information is ubiquitous in fluid mechanics. Denoting the input data as $\boldsymbol{q}_{in}$ and the output state as $\boldsymbol{q}_{out}$, the estimation task is framed as finding an accurate machine-learning model $\mathcal{F}$ that satisfies

$$\boldsymbol{q}_{out} = \mathcal{F}(\boldsymbol{q}_{in}; \boldsymbol{w}), \tag{2.1}$$

where $\boldsymbol{w}$ represents the trainable parameters (weights and biases) of the network.

The weight distribution $\boldsymbol{w}$ needs to be tuned to establish the relationship in (2.1). In response, the mathematical optimisation problem to find the optimal weights $\boldsymbol{w}^*$ is generally solved by minimising a predefined loss function $\mathcal{E}$ across the available training data in a supervised manner. This optimisation is formulated as

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, \mathcal{E}(\boldsymbol{w}), \tag{2.2}$$

with, for example, the L_P norm of the residual error,

$$\mathcal{E} = ||\boldsymbol{q}_{out} - \mathcal{F}(\boldsymbol{q}_{in}; \boldsymbol{w})||_P. \tag{2.3}$$

Once this optimisation is solved, the trained model $\mathcal{F}$ can be deployed to efficiently estimate or reconstruct the desired fluid states from new, unseen input data.

Typical examples for estimation in fluid mechanics are summarised in table 1. Some of them are inverse problems that aim to reconstruct the entire field from limited sensor data.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Input $\boldsymbol{q}_{in}$	Output $\boldsymbol{q}_{out}$	Example references
Sensors	Entire flow field	Erichson *et al.* (2020), Fukami *et al.* (2021*c*), Maulik *et al.* (2020), Santos *et al.* (2023)
Sensors	Mode coefficients (Field approximation)	Nair & Goza (2020), Dubois *et al.* (2022), Carter *et al.* (2021), Mousavi & Eldredge (2025)
Low-resolution field	High-resolution field	Fukami *et al.* (2019*a*), Kim *et al.* (2021), Güemes *et al.* (2022), Page (2025)
PIV particle images	High-resolution vector field	Cai *et al.* (2019), Morimoto *et al.* (2021*a*), Majewski *et al.* (2020), Masker *et al.* (2025)
Gappy field	Reconstructed field	Bui-thanh *et al.* (2004), Buzzicotti *et al.* (2021), Li *et al.* (2023), Zheng *et al.* (2024)
2-D measured fields	3-D volumetric field	Huang *et al.* (2020), Pérez *et al.* (2022), Matsuo *et al.* (2024), Xuan & Shen (2023)
Wall sensors	Off-wall flow states	Guastoni *et al.* (2021), Morimoto *et al.* (2022), Nakamura *et al.* (2022)

Table 1. Example combinations of input $\boldsymbol{q}_{in}$ and output $\boldsymbol{q}_{out}$ for data-driven state estimation in fluid mechanics.

Alternatively, instead of reconstructing the full field directly, estimation can be combined with compression techniques such as POD, dynamic mode decomposition and autoencoders. In this approach, low-dimensional coefficients are estimated from the sensors and then combined with spatial modes during post-processing to reconstruct the field, which will be covered in the present case study in § 3. From an image processing perspective, state estimation encompasses super-resolution analysis, as well as deriving high-resolution vector fields directly from experimental particle images. Furthermore, other problem settings, such as, estimating an entire three-dimensional volumetric field from limited two-dimensional cross-sections (e.g. measured by PIV), and estimating off-wall flow states at a distance using wall-mounted sensors, also belong to the group of state estimation. To address these diverse problem set-ups, various machine-learning methods have been proposed depending on the complexity and degrees of freedom of the target flow data.

2.1.2. *Temporal prediction*

Predicting the dynamical evolution of a flow field is essential for tasks such as weather forecasting and flow control. The goal is to predict the future state of a fluid system based on its past and present states. This is formulated as an autoregressive problem where the model $\mathcal{F}$ predicts the state at the next time step $\boldsymbol{q}(t + n\Delta t)$ given the state at the current time step $\boldsymbol{q}(t)$ such that

$$\boldsymbol{q}(t + n\Delta t) = \mathcal{F}(\boldsymbol{q}(t); \boldsymbol{w}), \tag{2.4}$$

where n denotes the number of time steps. The model can be recursively applied,

$$\boldsymbol{q}(t + k\Delta t) = \underbrace{\mathcal{F}(\mathcal{F}(\ldots \mathcal{F}}_{k \text{ times}}(\boldsymbol{q}(t); \boldsymbol{w}) \ldots; \boldsymbol{w}); \boldsymbol{w}), \tag{2.5}$$

for predicting the temporal evolution of fluid flows over multiple k steps.

Furthermore, the temporal prediction can be alternatively performed in a compressed latent space to reduce computational cost, rather than in the high-dimensional ambient

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

space. With the latent representation $\boldsymbol{\xi}(t)$ of the flow field, the latent dynamics model $\mathcal{F}_{latent}$ solves,

$$\boldsymbol{\xi}(t + n\Delta t) = \mathcal{F}_{latent}(\boldsymbol{\xi}(t);\, \boldsymbol{w}), \tag{2.6}$$

so that a flow field at the next step $\boldsymbol{q}(t + n\Delta t)$ is reconstructed from the predicted latent vector $\boldsymbol{\xi}(t + n\Delta t)$ through a decoder prepared with other data-driven techniques, such as POD and an autoencoder. As introduced with figure 9, machine-learning-based temporal prediction has been examined for a range of flow examples, including bluff body wake (Lee & You 2019), wall-bounded turbulence (Yousif *et al.* 2023*b*), homogeneous isotropic turbulence (Mohan *et al.* 2020), aerodynamic flows (Lui & Wolf 2019; Liu *et al.* 2025*b*) and urban models (Jaroslawski *et al.* 2025).

This type of data-driven temporal prediction model is often discussed as a class of surrogate models. By taking some parameter inputs in addition to the current state vector in (2.5) and (2.6), they can be parameterised in constructing a surrogate model. As they approximate the spatiotemporal behaviour of fluid flows at a reduced cost compared with numerical simulations and wind-tunnel experiments, they are particularly useful when considering a huge parameter space. Even if this does not need to be a complete surrogate, such approximations can guide us to figure out which cases are worth being numerically simulated or experimentally measured.

2.1.3. *Compression*

Data compression or dimensionality reduction is crucial in fluid mechanics to gain dominant flow characteristics in a compact form. The compressed representation not only facilitates understanding the flow physics but also supports downstream tasks such as reduced-complexity modelling and flow control. The general framework for compression consists of an encoding process that compresses given high-dimensional fluid data $\boldsymbol{q} \in \mathbb{R}^{n_q}$ into a latent space, and a decoding process that reconstructs the data from the compressed representation $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$.

The compressed data dimension, often referred to as the latent dimension, is set to be much smaller than the original data dimension such that $n_{\boldsymbol{\xi}} \ll n_{\boldsymbol{q}}$. This compression and the subsequent reconstruction are expressed as

$$\widehat{\boldsymbol{q}} = \mathcal{F}_d(\mathcal{F}_e(\boldsymbol{q})) = \mathcal{F}_d(\boldsymbol{\xi}), \tag{2.7}$$

where $\mathcal{F}_e$ and $\mathcal{F}_d$ are an encoder and a decoder, respectively.

The primary goal of compression approaches is to retain as much of the original information as possible. This is achieved by minimising the reconstruction error between the original data $\boldsymbol{q}$ and the reconstructed data $\widehat{\boldsymbol{q}}$. Analogous to the estimation/reconstruction problem, the encoding and decoding mappings are parameterised by weights $\boldsymbol{w}$ that are hence optimised in a manner,

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, ||\boldsymbol{q} - \mathcal{F}_d(\mathcal{F}_e(\boldsymbol{q};\, \boldsymbol{w}_e);\, \boldsymbol{w}_d)||_2, \tag{2.8}$$

where $\boldsymbol{w} = \{\boldsymbol{w}_e, \boldsymbol{w}_d\}$ represents the combined set of trainable parameters for the encoder and decoder. When the above optimisation is successful, the latent variable $\boldsymbol{\xi}$ serves as a low-dimensional latent representation of the original flow data $\boldsymbol{q}$. The specific forms of $\mathcal{F}_e$ and $\mathcal{F}_d$ depend entirely on the chosen methodology. For instance, classic modal decomposition techniques widely used in fluid mechanics, such as POD and dynamic mode decomposition, essentially function as linear encoders and decoders.

From a nonlinear machine-learning perspective, an autoencoder based on nonlinear models can be used for fluid flow data compression, often achieving better compression

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

than the linear counterpart (Fukagata 2023; Fukagata & Fukami 2025). While the encoder $\mathcal{F}_e$ and the decoder $\mathcal{F}_d$ are often structured with neural networks, a combination of various data-driven techniques for the encoder and the decoder can be considered. For example, the isomap-based encoder (Tenenbaum *et al.* 2000) and the k-nearest-neighbour-based decoder (Fix & Hodges 1989) are used for a range of controlled scenarios of fluidic pinball flows (Marra *et al.* 2024). A vision transformer-based encoder combined with a convolutional decoder is also considered to extract low-dimensional features of fluid flows (Fan *et al.* 2025). By optimising the weights of the network through the process described in (2.8), autoencoders capture a compact representation of fluid flows, which can involve information on multiple linear modes using a smaller number of nonlinear modes (Fukami *et al.* 2020*b*; Murata *et al.* 2020), thereby achieving a higher compression ratio.

2.1.4. *Generation*

While standard regression and compression approaches require pairs of input and output data for training, there exists a different optimisation principle that learns the underlying probability distribution of the data. This enables plausible data generation by sampling points on the learned probability distribution, which is particularly useful in fluid mechanics tasks such as inflow turbulence generation (driver or precursor) for numerical simulations (Kim & Lee 2020; Yousif *et al.* 2022; Liu *et al.* 2025*a*) and aerofoil shape generation from given aerodynamic constraints (Li *et al.* 2020; Wang *et al.* 2023*b*).

A prominent example of the algorithm is the generative adversarial network (GAN; Goodfellow *et al.* 2020). A GAN consists of two competing networks: a generator (G) and a discriminator (D). The generator G aims to create realistic 'fake' output data $G(\boldsymbol{n})$ from random noise $\boldsymbol{n}$ (or a latent input). In contrast, the discriminator D is trained to distinguish between 'real' data $\boldsymbol{d}$ from the actual data set and the 'fake' data produced by the generator, outputting a probability between 0 and 1.

This competition is formulated as a min–max optimisation problem,

$$\min_{G}\max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{d}\sim p_{data}(\boldsymbol{d})}[\log D(\boldsymbol{d})] + \mathbb{E}_{\boldsymbol{n}\sim p_{\boldsymbol{n}}(\boldsymbol{n})}[\log(1 - D(G(\boldsymbol{n})))], \tag{2.9}$$

where $V(D, G)$ is called the value function. The discriminator maximises this objective function to correctly identify real and fake samples, while the generator minimises it to fool the discriminator. By formulating the problem in terms of statistical likelihood rather than pixel-wise residual minimisation, generative models produce statistically plausible flow structures (Bode *et al.* 2021). Note that the GAN is treated here as a training framework rather than a specific network architecture. This is because GANs can be constructed using a variety of underlying models, including MLPs and CNNs, as detailed in § 2.3. Other generative approaches, such as the diffusion model, will also be covered later in § 2.3. Analogous to temporal prediction models, generative models can also be parameterised by taking some parameters in addition to the input vector.

2.1.5. *Action design (reinforcement learning)*

In active flow control and aerodynamic shape optimisation, the problem shifts from passive prediction to active decision making. Reinforcement learning provides a framework for this by modelling the interaction between an agent (the controller) and an environment (the flow system) (Vincent *et al.* 2018). The reinforcement-learning problem is formulated as a Markov decision process. At each time step t, the agent observes the state of the flow $\boldsymbol{s}_t$ (e.g. sensor readings), takes an action $\boldsymbol{a}_t$ (e.g. blowing/suction on the body surface) and receives a scalar reward r_t (e.g. drag reduction) from the environment. The behaviour of the agent is defined by a policy $\pi_{\boldsymbol{w}}(\boldsymbol{a}_t|\boldsymbol{s}_t)$. In practice, this policy is typically

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

approximated by a neural network where $\boldsymbol{w}$ represents the learnable weights and biases. As the state vector $\boldsymbol{s}_t$ propagates through the network, these weights govern the nonlinear transformations that map the flow information to the parameters of the action probability distribution, e.g. the mean and variance for continuous control.

The mathematical objective in action design is to find the optimised policy parameters $\boldsymbol{w}^*$ that maximise the expected cumulative reward over a trajectory of length T,

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmax}}\, \mathbb{E}_{\pi_{\boldsymbol{w}}} \left[\sum_{t=0}^{T} \gamma^t r_t \right], \tag{2.10}$$

where $\gamma \in [0, 1)$ is a discount factor that prioritises immediate rewards over future ones. By formulating flow control as a reinforcement-learning problem, we may discover complex, nonlinear control strategies that are challenging to derive using traditional linear control theories (Sonoda *et al.* 2023; Font *et al.* 2025).

2.2. *Data curation*

Once the problem setting is determined, the next step is to prepare the input and output data used for training the model. Since machine-learning-based studies in fluid dynamics often rely on extracting dominant characteristics from provided data sets, the structure and quality of these data are consequently of importance. Fluid flow data can come from various sources, including numerical simulations, experimental measurements or even empirical and theoretical formulations. In fact, the use of the input–output relationship derived through empirical formulations as an auxiliary data set can assist data-driven analysis in fluid mechanics while reducing the required amount of numerical and experimental data sets (Lee *et al.* 2022). Depending on the data source and the specific problem being addressed, the preparation of the data matrix and model architectures may need to be different. Furthermore, as discussed in subsequent sections, designing an appropriate cost function during model training should be carefully tailored to the inherent characteristics of fluid flow data and their sources. This section provides an overview of various fluid data formats, detailing how to prepare data matrices in a manner consistent with the flow characteristics, problem settings and the selected machine-learning models.

2.2.1. *Data characteristics*

A wide spectrum of formats, qualities and sizes is considered for fluid flow data. For instance, time-series data measured through surface-mounted sensors typically possess a much lower dimensionality compared with entire flow field data, often enabling the use of relatively simple machine-learning architectures such as MLPs. In contrast, the degree of freedom for flow field data acquired via computational fluid dynamics or experiments such as Schlieren photography and PIV is much larger than sensor readings.

Even under the same category of data source, the data volume and its quality become different. In PIV, the data volume and quality differ significantly depending on whether raw particle images or post-processed velocity fields are considered. For numerical simulations, the temporal axis of the data matrix depends entirely on the simulation method. Steady-state Reynolds-averaged Navier–Stokes simulations yield time-independent data, whereas direct numerical simulations, large-eddy simulations or detached eddy simulations can generate unsteady, time-resolved data sets. The spatiotemporal fidelity of fluid flow data also depends on the simulation approach.

Furthermore, the stored variables, such as velocity, pressure and vorticity, determine the data shape, which also depends on whether the flow is incompressible or compressible. The choice of computational grid, e.g. uniform, non-uniform or unstructured, and how

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

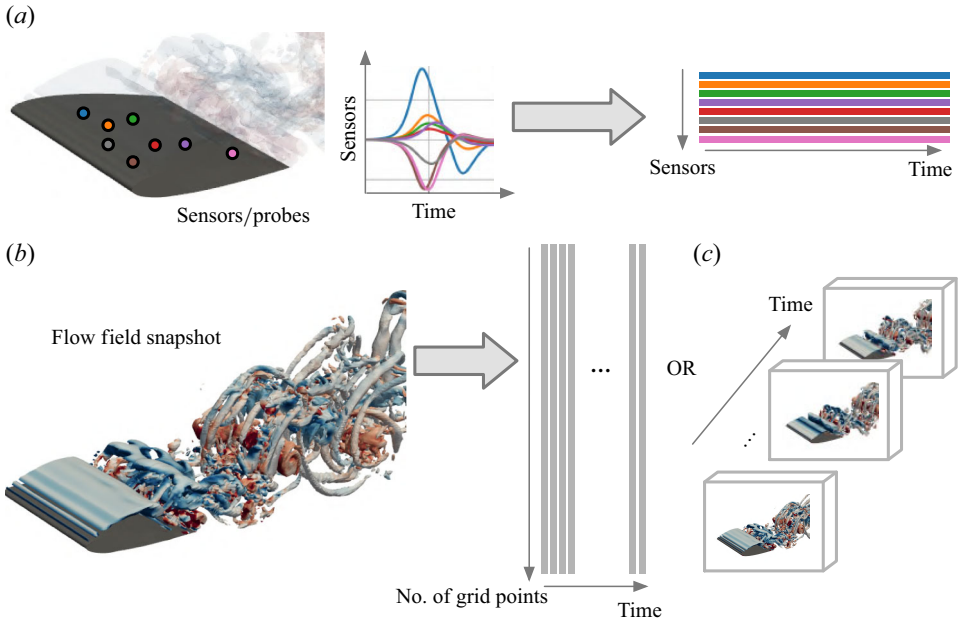


Figure 12. Data preparation for machine-learning analysis of fluid flows. (*a*) Time series of sensor readings or probe data is often handled as a matrix ($n_{sensor} \times n_t$). A series of flow snapshots can be treated with (*b*) a matrix ($n_{grid} \times n_t$), or (*c*) a higher-dimensional tensor (e.g. $n_t \times n_x \times n_y \times n_z \times n_{variable}$), depending on the machine-learning model chosen for the analysis.

flow fields on them are saved are also important. For certain architectures such as CNNs, it is sometimes advantageous to interpolate or extrapolate unstructured mesh data onto a Cartesian grid, transforming it into an image or video format to simplify subsequent processing. However, such interpolations carry the risk of obscuring the underlying flow physics. To prevent this, one can consider designing the model to directly handle unstructured data, such as point cloud networks or graph neural networks (GNNs). Which of the aforementioned variables and data structures should be considered depends highly on the problem setting and the nature of the fluid flows of interest. In other words, such prior knowledge of fluid mechanics needs to be appropriately involved in the data set preparation.

2.2.2. *Data shape*

Let us then consider what data shape we often consider for data-driven analysis in fluid mechanics. When processing time-series data from sensors or probes derived from simulations or experiments, a two-dimensional matrix is typically constructed, as illustrated in figure 12(*a*). In this formulation, the vertical axis denotes the sensor index, corresponding to the 'features' in standard machine-learning nomenclature (Goodfellow *et al.* 2016). The horizontal axis represents time, which represents the 'volume of training data.' This matrix structure serves as the standard training data format for linear regression or MLP-based analyses of problems with relatively low degrees of freedom.

An example considered in figure 3, estimating the lift coefficient $C_L(t) \in \mathbb{R}$ from the sensor readings $\boldsymbol{s}(t) \in \mathbb{R}^5$, has indeed used this matrix-based data preparation. Given the relatively low dimensionality of the input–output space, a nonlinear regression via a MLP is an appropriate choice, which will be discussed later. The input matrix $\boldsymbol{Q}_{in}$ based on the sensor time trace and the output matrix $\boldsymbol{Q}_{out}$ of the aerodynamic coefficient take the form

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

$$\boldsymbol{Q}_{in} = \begin{pmatrix} s_1(t_1) & s_1(t_2) & \cdots & s_1(t_n) \\ s_2(t_1) & s_2(t_2) & \cdots & s_2(t_n) \\ \vdots & \vdots & \ddots & \vdots \\ s_5(t_1) & s_5(t_2) & \cdots & s_5(t_n) \end{pmatrix} \in \mathbb{R}^{n\times 5}, \quad \boldsymbol{Q}_{out} = \begin{pmatrix} C_L(t_1) \\ C_L(t_2) \\ \vdots \\ C_L(t_n) \end{pmatrix} \in \mathbb{R}^{n\times 1}, \tag{2.11}$$

where n represents the number of snapshots over time. While this specific example estimates C_L, the model can be easily extended to simultaneously provide the drag coefficient C_D, which simply requires concatenating C_L and C_D to yield an output matrix $\boldsymbol{Q}_{out} \in \mathbb{R}^{n\times 2}$.

For managing two- or three-dimensional data sets of fluid flows, the same strategy as above can be considered, as illustrated in figure 12(*b*). Consider a scenario where snapshots of an unsteady flow field are collected over time. Each of the multidimensional flow field snapshots at each discrete time step is flattened into an elongated one-dimensional column vector, which is then concatenated along the temporal axis, as depicted in figure 12(*b*). The feature variables or different state variables can be populated in the vertical direction. This data matrix construction is used not only for nonlinear machine learning but also for linear modal analyses including POD (Kajishima & Taira 2016).

While the aforementioned flattened data matrices are often used, flattening can cause the loss of spatiotemporal information or connectivity of data. Topological features, such as flow structures and their dynamical motion, are critical in fluid mechanics. To preserve such topological characteristics of fluid flows in the data, an alternative approach is to maintain the data in its native multidimensional format, as illustrated in figure 12(*c*). This preparation is used for CNNs, which were originally developed for image and video processing, as introduced later.

Although the example in figure 12 illustrates the operation with vortical snapshots, one can analogously use velocity components (u, v, w), treating them similarly to RGB channels in standard video data. For a three-dimensional unsteady flow, the resulting data shape becomes a five-dimensional tensor structured as n_t (time) $\times$ n_x $\times$ n_y $\times$ n_z $\times$ $n_{variable}$ (state variables). In parametric steady-state studies, the 'time' axis can simply be replaced by a parameter axis (e.g. geometry or Reynolds number) to consider the exact same data-preparation techniques. For problems where the input and output possess significantly different degrees of freedom, the techniques shown in figure 12 can be hybridised to suit the specific model architecture.

2.2.3. *Data pre-processing*

It is possible to estimate multiple variables simultaneously with a machine-learning model. However, we should be mindful of the potential disparity in their respective orders of magnitude, which can significantly skew estimation accuracy. Since the model is trained to minimise the cost function as discussed in § 2.1, the relative magnitude of the error would be biased toward the one with a larger order of magnitude if an unweighted error metric is applied to the raw data. This can happen in fluid mechanics, as the order of velocity magnitude highly depends on the primary flow direction, e.g. the streamwise velocity and others in a turbulent channel flow. Although the magnitude itself often represents important flow characteristics, it is sometimes necessary to scale the data while being careful not to lose its physical properties.

In response, fluid flow data can be pre-processed via min–max normalisation or standardisation to balance the order and distribution of the variables. They are operated as

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

$$q_{norm} = \frac{q - q_{min}}{q_{max} - q_{min}}, \quad q_{std} = \frac{q - \mu_q}{\sigma_q}, \tag{2.12}$$

where q_{norm} and q_{std} are respectively normalised and standardised variables of q. The min–max normalisation linearly scales the data such that the minimum value q_{min} maps to 0 and the maximum q_{max} maps to 1. When processing flow variables that possess negative physical values, this range is often adjusted to $[-1, 1]$. In contrast, the standardisation centres the data around a mean μ_q of 0 and scales it to a standard deviation σ_q of 1. Once the estimation or prediction is performed with the scaled variables, the output of the machine-learning model can be scaled back for downstream assessments.

For data sets heavily contaminated by outliers, other robust transformations, such as logarithmic scaling or the Box–Cox transformation, can be considered (Sakia 1992). Determining the optimal scaling technique that appropriately balances the data magnitudes without losing its fundamental physical characteristics is entirely dependent on the underlying probability distribution. However, we caution that there exist cases where data scaling should be avoided, since the magnitude and probability distribution that the data originally possess are important characteristics of fluid flows. Rather than blindly feeding raw data into machine-learning models, performing statistical analysis is mandatory for any machine-learning pipeline.

2.2.4. *Data split*

Once the data sets are prepared with special care for the specific problem setting, the flow physics of interest and the chosen model architecture, we then start to consider the training of the model. It is critical to appropriately partition the prepared data to ensure the capability of the trained model. Failing to perform this process often leads to overfitting, resulting in an ineffective model that performs well only on the training data. While various data-splitting techniques exist, this section discusses the k-fold cross-validation while highlighting specific caveats associated with its application to fluid flow data.

Nonlinear machine-learning models are typically trained through an iterative process that minimises (or maximises) a cost function using algorithms such as backpropagation (Bishop & Nasrabadi 2006). Hence, the selection of data used to evaluate the error during this iterative training process is important. Here, let us consider a data partitioning approach that shuffles the available data set and prepares distinct subsets for specific purposes, as illustrated in figure 13(a). The subsets coloured in blue represent the training data, which are directly used to update the model parameters. However, training should not rely only on them since the model will likely overfit, adopting too closely to the training data and losing its ability to generalise to unseen scenarios, as illustrated in figure 13(b).

To prevent this, it is necessary to continuously monitor the performance of the model for unused data sets during training. This is achieved with a validation data set, highlighted in orange in figure 13(a). During the training process, the validation data are fed into the updated model at the end of each iteration (or epoch) to track the trajectory of both training and validation errors. Excessive training leads to an increase in validation error, indicating a loss of generalisation (overfitting), while insufficient training results in underfitting, as illustrated in figure 13(b). The optimised model can achieve a delicate balance between the two. To obtain such a balanced state, early stopping (Prechelt 1998), which automatically stops the learning process and saves the model parameters when the validation error ceases to improve, is often considered.

Once the model is obtained using the training and validation sets, the test data, a green box in figure 13(a), is used to evaluate the model performance on data excluded from

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

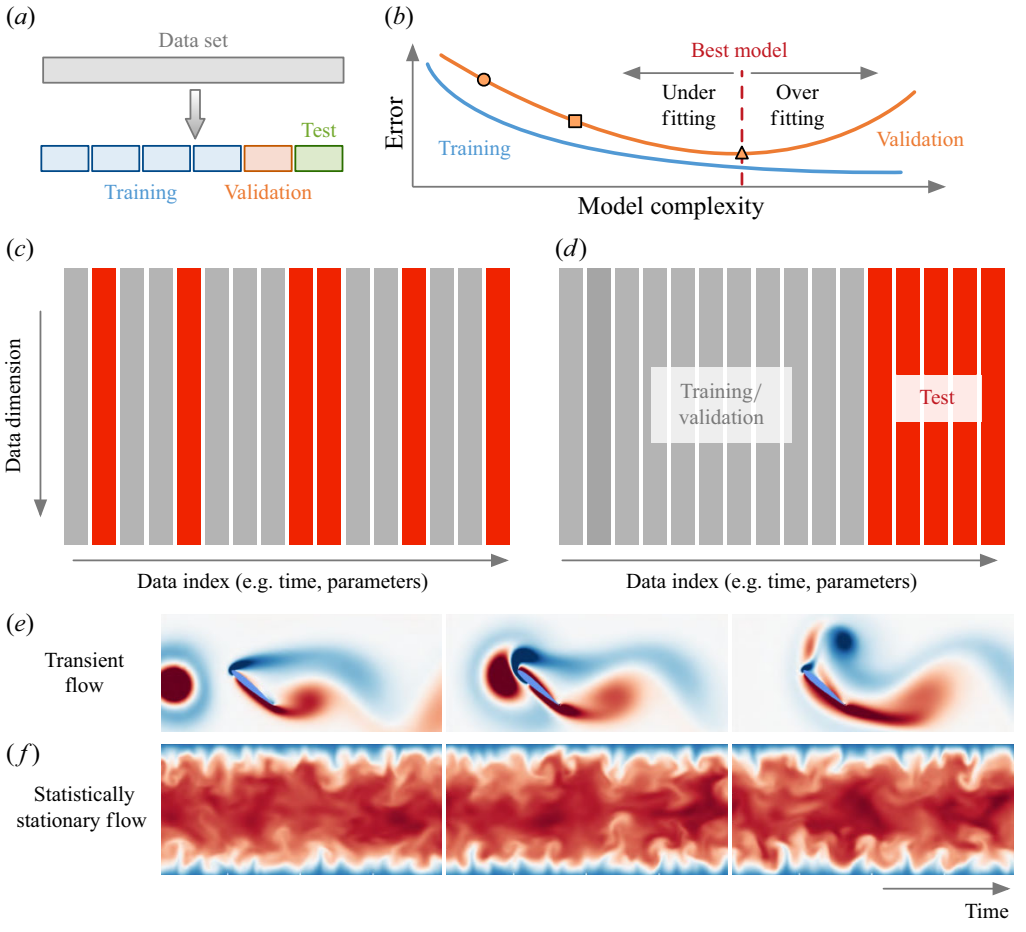


Figure 13. Data split for machine-learning analysis of fluid flows. (*a*) The prepared data sets are first divided into training, validation and test sets. (*b*) The relationship between the estimation error and the model complexity, monitored with the training and validation data. (*c–f*) Interpolation and extrapolation of machine-learning-based analysis for fluid flows. The test data preparation, such as shown in panels (*c*) and (*d*), needs to be carefully performed, depending on the flows of interest, e.g. (*e*) transient, and (*f*) statistically stationary flows.

the training process. As the model performance for the test data may be sensitive to how the data set was originally partitioned, it is common practice to perform k-fold cross-validation, where the model is trained k separate times with randomly partitioned data sets (Stone 1974). This provides the average performance across the k distinct test sets, thereby avoiding producing biased assessments for a particular test set.

When applying the data-splitting strategy to fluid flow data, however, further careful consideration based on the underlying flow physics is needed to appropriately train and assess machine-learning models. To discuss this point, a nuanced understanding of interpolation and extrapolation within the context of data-driven fluid mechanics is required. In general, a model exhibits its expected performance on data that has been 'seen' during training; this regime is termed interpolation. For 'unseen' data, the model error typically increases, which is termed extrapolation.

Defining these regimes is straightforward for low-dimensional tasks such as simple curve fitting. For example, data falling outside the numerical bounds of the training set

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

is considered extrapolation. However, identifying the clear cut between interpolation and extrapolation for high-dimensional flow field data is challenging. In fact, there are a number of studies reporting that applying a nonlinear machine-learning model to a test data set with a slightly different Reynolds number than the training data can still be regarded as interpolation, i.e. an interpolatory use of the model, when the underlying physics or topological flow structures remain similar (Fukami *et al.* 2024*a*). Nonlinear machine-learning models, particularly with some functions based on prior knowledge of flow physics, may capture scale-invariant physics embedded within the data across the spatial length scale, and their performance in these regimes can sometimes far exceed expectations based on traditional linear techniques (Taira *et al.* 2025). Therefore, carefully defining the criteria by which a model has 'seen' or 'not seen' data is important when preparing test data sets.

Let us here examine relatively intuitive cases of interpolatory and extrapolatory data sampling with two distinct unsteady flow examples. We consider two scenarios of data preparation, as illustrated in figures 13(*c*) and 13(*d*). The grey segments represent the combined training and validation data, while the red counterparts are used for the test data. In example (*c*), the training/validation and test data are randomly sampled over time. In contrast, example (*d*) prepares the test data as a temporal extrapolation, i.e. forecasting into the future.

Consider applying these data-preparation strategies to a transient system, such as a vortex-gust encounter around an aerofoil, presented in figure 13(*e*). In various problem settings, sampling data in a manner of figure 13(*c*) enables the model to comprehensively learn the temporal dynamics of the vortex-aerofoil interaction throughout the entire event over time. Thus, the red test data in figure 13(*e*) can be considered a physical interpolation. In contrast, the model trained with the data sampling strategy in chronological order, shown in figure 13(*d*), may consider the later stages of the interaction as extrapolation due to the transient nature of the flow dynamics.

It is, however, worth pointing out that such extrapolatory sampling in time can sometimes functionally manifest as physical interpolation. This can happen for data-driven analysis of statistically stationary flow, such as an unsteady turbulent channel flow, shown in figure 13(*f*). Although the instantaneous physical state chaotically evolves over time, its long-term statistics remain constant, possessing ergodicity (Walters 2000). For this type of flow, even test data sampled entirely from the future via an approach in figure 13(*d*) may still be regarded as 'seen' data from the perspective of the underlying statistical distributions. Nonlinear machine-learning analyses have indeed leveraged these characteristics for various problem settings, including turbulence modelling and flow reconstruction (Vinuesa & Brunton 2022). When preparing test data and evaluating model performance on them, we should be mindful of what results are anticipated from the aspect of data preparation for training and test phases. This becomes particularly important when the expected outcomes are not derived through machine-learning studies, so that the improvements can be considered via data-preparation perspectives based on flow physics.

2.3. *Model construction*

Once the data sets are curated and the specific problem setting is defined, the next step is to construct a model. While there exist many machine-learning architectures, e.g. see the Neural Network Zoo as highlighted by Fjodor Van Veen (https://www.asimovinstitute.org/neural-network-zoo/), an appropriate selection of the models is important based on the flow physics of interest, the Eulerian or Lagrangian frame of reference and the degrees of freedom of the input and output variables. In addition, the models can be designed

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

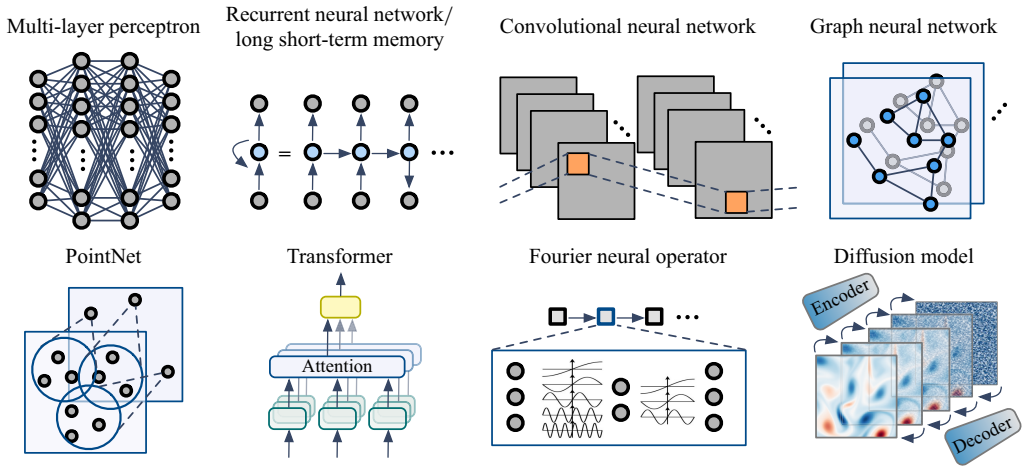


Figure 14. A variety of machine-learning models that have been considered for studying fluid mechanics.

to incorporate prior physical knowledge rather than using their vanilla forms. This often significantly reduces the learnable hypothesis space, accelerating training convergence and guaranteeing that the resulting data-driven models remain bounded by realistic fluid kinematics and dynamics.

Here, let us cover the fundamentals of machine-learning models illustrated in figure 14 often considered in studying fluid mechanics, while briefly covering examples of how physical priors are incorporated. We then discuss activation functions that are used in almost all nonlinear machine-learning models. Some tips on parameter tuning of machine learning are also offered. We also note that these fundamental techniques can be merged together depending on the flow characteristics of interest, offering the flexibility to study fluid flows in a data-driven manner. For further mathematical and algorithmic details on each architecture, please refer to the foundational papers cited in the respective subsections.

2.3.1. *Multi-layer perceptron*

The MLP (Rumelhart *et al.* 1986) is the foundational network architecture, often referred to as a feedforward neural network. The basic concept of MLP is motivated by the structure of biological neural circuits. The model maps an input state vector to an output through a sequence of fully connected hidden layers. The transition of a feature node i from layer l to layer $l+1$ is expressed as

$$h_i^{(l+1)} = \varphi\left(\sum_{j=1}^{N} W_{ij}^{(l)} h_j^{(l)} + b_i^{(l)}\right), \tag{2.13}$$

where $h_j^{(l)}$ represents the input nodes or units, $W_{ij}^{(l)}$ is the weight matrix connecting every input j to every output i, $b_i^{(l)}$ is the bias and φ is an activation function.

Multi-layer perceptrons operate on a pointwise basis while the formulation does not consider spatial topography or temporal sequencing. It is hence highly effective for mapping low-dimensional, local state variables, such as sparse sensor readings and aerodynamic coefficients (Fukami *et al.* 2020*a*). In addition, MLP-based models have been examined for closure modelling in Reynolds-averaged Navier–Stokes simulations (Duraisamy *et al.* 2019). For turbulence modelling, multiple MLP networks with a specific

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

physics-based constraint, the so-called tensor-basis neural network, are often considered (Ling *et al.* 2016; Tsai *et al.* 2026). The tensor-basis neural network guarantees the Galilean invariance of the estimated anisotropy tensor by using specialised layers with an invariant tensor basis, which becomes a standard approach in machine-learning-based studies of turbulence modelling (Wu *et al.* 2018; Zhao *et al.* 2020).

Multi-layer perceptrons also serve as the backbone for physics-informed neural networks (PINNs; Raissi *et al.* 2019), where the network takes spatial coordinates and time as input to estimate local flow variables. Physics-informed neural networks, often composed of MLPs, use automatic differentiation (Baydin *et al.* 2018) to penalise the solution by minimising the cost function based on the Navier–Stokes equations, which will be discussed later. In addition to the velocity components, they can provide a local estimate of the pressure variable without the need for pressure Poisson inversions and their associated boundary conditions.

2.3.2. *Recurrent neural network/ LSTM*

A class of neural networks designed to process sequential data by maintaining a hidden state that carries historical information forward in time is called recurrent neural networks (RNNs). The general formulation of recurrent networks is to predict the hidden vector state h_t based on the current input x_t and the previous hidden state h_{t-1},

$$h_t = \varphi(W_{hx}x_t + W_{hh}h_{t-1} + b_h), \tag{2.14}$$

where W_{hx} denotes the weights between the hidden state and the input, and W_{hh} expresses the weights between the hidden states over time. As hinted by (2.14), this formulation enables the model to account for historical information in predicting a variable.

While standard RNNs could be considered for turbulent flow forecasting, it has also been reported that they often suffer from vanishing gradients in the model learning process, as gradients across long-time sequences need to be propagated inside the model, thereby causing difficulty in stably learning the dynamics (Srinivasan *et al.* 2019). In response, LSTM (Hochreiter & Schmidhuber 1997), an RNN with a sophisticated gating mechanism based on the input, forget and output gates, can also be used to regulate long-term memory in the learning process. The use of LSTMs has been widely seen for modelling the temporal evolution of unsteady flows not only in an ambient space but also in a reduced-order space such as the time-varying coefficients of POD and nonlinear autoencoder (Wan & Sapsis 2018; Maulik *et al.* 2021).

Some recurrent networks that embed prior knowledge of flow physics into the model design have been proposed. Shown in figure 9 by Nakamura *et al.* (2021) considers a parallel LSTM in predicting the temporal evolution of minimal turbulent channel flow in a reduced-order space identified through a nonlinear autoencoder. This parallel setting includes a number of LSTM models with different numbers of units, enabling the model to learn a range of temporal length scales in turbulence. Aiming to develop robust models against noisy measurements, Ahmed *et al.* (2021) proposed an LSTM nudging that incorporates sensor priors to stabilise the prediction of the two-dimensional vorticity transport problem. They considered combining LSTM-based prediction and data assimilation approaches that have been well established in unsteady flow forecasts (Zaki 2025).

Accurate instantaneous prediction of turbulence with such recurrent models is recognised as challenging due to error accumulation in recursively using the machine-learned model, as expressed in (2.5). However, it is worth pointing out that they are powerful in learning the long-term dynamics of chaotic systems, including turbulence. Some studies have argued that nonlinear machine learning may learn the attractor of the

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

chaotic system, and the prediction can be stable unless an instantaneous prediction is not too far off from the learned attractor. As chaos itself can then automatically dump the offset of the machine-learned prediction, the resulting long-term prediction eventually resides in the vicinity of the original attractor (Linot & Graham 2020).

2.3.3. *Convolutional neural network*

While MLPs and recurrent networks are suited for problem settings with relatively low-order input and output data, the application to cases with a high degree of freedom, such as flow field reconstruction, becomes computationally prohibitive due to the fully connected nature of the model. In addition, such fully connected models generally require providing data sets in a matrix form, as introduced in figure 12(*b*), implying that the spatial arrangement of flow structures and their relative distance from boundaries may be lost before providing them to the model.

In such a scenario, a CNN (LeCun *et al.* 1998) can be chosen. The convolutional processes introduced hereafter are inspired by the connectivity pattern of the visual cortex. Convolutional neural networks have been widely used in computer vision for image and video processing, while they have also been actively considered for a variety of fluid mechanics problems (Morimoto *et al.* 2021*b*).

Convolutional neural networks modify the dense matrix multiplication of MLPs by using discrete convolutional filter operations, which enforce sparse connectivity and weight sharing across the input space. While retaining the spatial arrangement of structures in a flow snapshot, the weight sharing enables the substantial reduction of computational costs in extracting the input–output relationship. Equipped with a similar notation to the MLP, but extended to account for C input channels and K output filters, the update for the κth feature map at a spatial node i in a CNN is given by

$$h_{i,\kappa}^{(l+1)} = \varphi\left(\sum_{c=1}^{C}\sum_{j\in\mathcal{S}(i)} W_{i-j,c,\kappa}^{(l)} h_{j,c}^{(l)} + b_{\kappa}^{(l)}\right), \tag{2.15}$$

where a localised spatial neighbourhood is expressed with $\mathcal{S}(i)$. Unlike the MLP, whose every node is connected by a unique weight W_{ij}, the CNN node i only aggregates information from a localised spatial neighbourhood $\mathcal{S}(i)$ across all input channels c. Furthermore, the weight tensor $W_{i-j,c,\kappa}^{(l)}$ depends only on the relative spatial offset $i-j$, making the operation translation invariant, while a shared bias $b_{\kappa}^{(l)}$ is applied per output filter. This makes CNNs intrinsically suited for structured grids on Eulerian frames of reference (Fukami *et al.* 2021*c*).

The multi-scale nature of turbulent flows can be embedded into CNNs by employing multiple parallel filter sizes to simultaneously resolve large-scale energy-containing motions and small-scale dissipative eddies (Fukami *et al.* 2019*a*). The filter size, corresponding to the area coverage of the kernel $\mathcal{S}(i)$, can also be determined based on characteristic length scales (Zheng *et al.* 2024). Furthermore, specific convolutional kernels and layers can be hard coded and frozen to act as exact discrete differential operators, structurally enforcing divergence-free constraints (Mohan *et al.* 2023). In addition to convolutional filter operations, CNNs can involve pooling operations to adjust the data dimension inside the model, which is often used for both encoder and decoder formulations in figure 1 with various problem settings. Details on each operation from the aspect of their uses for fluid mechanics can be seen in Morimoto *et al.* (2021*b*).

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

2.3.4. *Graph neural network*

As some fluid flow data generated through computational fluid dynamics or sparsely sampled sensors are saved on a non-uniform grid, of interest is whether nonlinear machine-learning-based analysis can be considered for such cases while extracting physical insights that account for the spatial coherence of flow structures, similarly to CNNs. In response, one can consider a GNN that further generalises the concept of local receptive fields from CNNs to a non-Euclidean domain (Scarselli *et al.* 2008).

Given an arbitrary graph structure defined by nodes and edges, the core mechanism of GNNs is the so-called message passing, where a node i updates its feature state by aggregating physical information from its topological neighbourhood $\mathcal{S}(i)$. This is expressed as

$$h_i^{(l+1)} = \varphi\left(\sum_{j\in\mathcal{S}(i)} c_{ij} W^{(l)} h_j^{(l)} + b_i^{(l)}\right), \tag{2.16}$$

where the shared weight matrix $W^{(l)}$ transforms the features of neighbouring nodes and c_{ij} represents edge-specific coefficients defined by the graph geometry. This formulation is particularly useful in applying machine-learning analyses for fluid flow data on unstructured grids (Li & Farimani 2022). While standard GNNs have often been considered as they are adjustable for arbitrary mesh geometries, enforcing the equivalence of dynamic collective motions of multi-agent systems and the hierarchy of spatial length scales in fluid flows can be considered to incorporate physics-based prior knowledge into the structural design of GNNs (Linot *et al.* 2025).

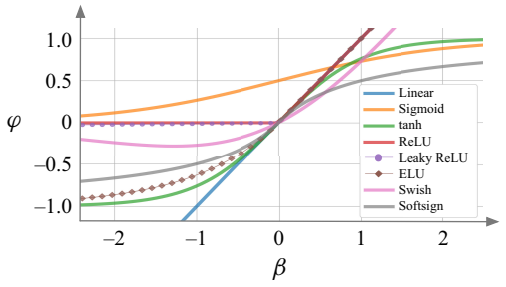
2.3.5. *Other model candidates*

In addition to the model architectures ranging from pointwise mappings to structured and unstructured spatial convolutions, there exist a variety of machine-learning models that could support studies in fluid mechanics. For instance, experimental measurements often yield sparse velocity point clouds rather than organised structured grids. To handle such data directly without relying on mesh interpolation, PointNet (Qi *et al.* 2017) is advantageous. They are specifically designed to process continuous, unordered sets of spatial points, bypassing the need for a grid topology while remaining structurally invariant to the permutation of the input data points. They have been considered for steady flow estimation (Kashefi *et al.* 2021).

Fluid flows often exhibit critical long-range, non-local dependencies. While localised filters in CNNs require a number of layers to communicate information across a wide spatiotemporal domain, transformers (Vaswani *et al.* 2017) capture such global interactions. With a self-attention mechanism, transformers dynamically weigh the importance of disparate regions across the entire flow field in parallel, effectively extracting non-local features from fluid flow data. In fluid mechanics studies, the use of transformers has been found in dynamical modelling (Solera-Rico *et al.* 2024) and turbulent flow generation (Yousif *et al.* 2023*b*).

Another challenge with standard machine-learning models is their dependence on the specific spatial grid resolution of their training data. While this highly relates to discussions in § 2.2, this challenge is caused by the fact that the majority of data-driven studies in fluid mechanics consider representing the fluid flow system as discrete finite-dimensional data. In other words, a model capable of handling data in a continuous manner may efficiently learn physics from the system without the loss of information (Wang *et al.* 2023*a*).

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press



Activation function	Equation
Linear	$\varphi(\beta) = \beta$
Sigmoid	$\varphi(\beta) = (1 + e^{-\beta})^{-1}$
tanh	$\varphi(\beta) = (e^{\beta} - e^{-\beta})/(e^{\beta} + e^{-\beta})$
ReLU	$\varphi(\beta) = \max(0, \beta)$
Leaky ReLU	$\varphi(\beta) = \beta$ (if $\beta > 0$), 0.01β (otherwise)
ELU	$\varphi(\beta) = \beta$ (if $\beta > 0$), $e^{\beta} - 1$ (otherwise)
Swish	$\varphi(\beta) = \beta/(1 + e^{-\beta})$
Softsign	$\varphi(\beta) = \beta/(1 + \lvert\beta\rvert)$

Figure 15. Activation functions.

To generalise the formulation in infinite-dimensional function spaces, Fourier neural operators, which perform nonlinear machine-learning operations in the frequency domain, can be considered (Li *et al.* 2021). Such neural operators achieve zero-shot super-resolution of turbulent flows (Li *et al.* 2022; Wang *et al.* 2024). This zero-shot capability stems from the fact that Fourier neural operators learn a continuous operator mapping between function spaces, rather than mapping between fixed-size discrete vectors. In contrast to standard convolutional models, whose learned spatial filters are intrinsically tied to the specific grid spacing of the training data, such neural operators parameterise the integral operator in the Fourier domain. By truncating the spectrum to a fixed number of low-frequency modes, the learned network weights become entirely independent of the spatial discretisation. As a result, once the continuous flow function is learned on a coarse mesh, it can be evaluated at any arbitrarily finer, unseen spatial grid points without requiring structural changes to the model or retraining.

For the generation problem discussed in § 2.1.4, it has been reported that the standard GAN algorithm can suffer from training instability while it is data hungry. Diffusion models (Ho *et al.* 2020) can instead be considered as an alternative for stochastic flow generation. By learning to synthesise data through a gradual reverse-denoising process, they enable the generation of realistic, statistically plausible flow fields from pure noise (Liu & Thuerey 2024) or low-order representations of fluid flows including sensor measurements (Shu *et al.* 2023).

It is worth noting again that these diverse machine-learning architectures are not mutually exclusive. Rather, they can be synergistically combined to analyse various flow characteristics. In fact, a combination of them, including CNN-MLP models (Zhong *et al.* 2023), CNN-LSTM models (Hasegawa *et al.* 2020), ConvLSTM models (Mohan *et al.* 2020) and graph CNNs (Ogoke *et al.* 2021), has been considered for fluid flow analysis. Such unification of machine-learning architecture may enable the model to be further versatile for a variety of problems in fluid dynamics.

2.3.6. *Activation functions*

As seen from the mathematical description for each model above, machine-learning models take activation functions to transform the input data of each node into an output signal. Nearly all modern machine-learning frameworks applied to fluid mechanics include internal operations with nonlinear activation functions, illustrated in figure 15. This is key not only to achieving better performance but also to gaining deeper physical insights compared with what has been possible with conventional linear techniques.

The fundamental role of these functions is to introduce nonlinearity into the network. In the absence of nonlinear activation, i.e. composed of a purely linear function $\varphi(\beta) = \beta$, the network entirely reduces to a sole linear transformation. In such a case, the output would

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

remain a mere linear combination of the inputs, regardless of the depth of the architecture. Given that fluid mechanics is inherently nonlinear, nonlinear activation functions assist in capturing complex spatiotemporal dynamics and turbulent multi-scale structures for machine-learning models.

Various activation functions have been proposed in the machine-learning community. Each possesses distinct mathematical properties that influence the training process and convergence. As presented in figure 15, example functions include the following.

(i) Linear: $\varphi(\beta) = \beta$, defining a simple proportional mapping
(ii) Sigmoid: $\varphi(\beta) = (1 + \mathrm{e}^{-\beta})^{-1}$, which maps inputs to a range between 0 and 1, frequently used in gating mechanisms.
(iii) tanh: $\varphi(\beta) = (\mathrm{e}^{\beta} - \mathrm{e}^{-\beta})/(\mathrm{e}^{\beta} + \mathrm{e}^{-\beta})$, a zero-centred hyperbolic tangent function bounded between -1 and 1, which often aids in stabilising the training process.
(iv) ReLU (rectified linear unit): $\varphi(\beta) = \max[0, \beta]$, often used due to its computational efficiency and mitigation of the vanishing gradient problem.
(v) Leaky ReLU: $\varphi(\beta) = \beta$ (if $\beta > 0$) and 0.01β otherwise. This variant prevents 'dying' neurones by permitting a small, non-zero gradient for negative inputs.
(vi) ELU (exponential linear unit): $\varphi(\beta) = \beta$ (if $\beta > 0$) and $\mathrm{e}^{\beta} - 1$ otherwise, which smooths the output for negative inputs to promote robust learning.
(vii) Swish: $\varphi(\beta) = \beta/(1 + \mathrm{e}^{-\beta})$, a smooth, non-monotonic function that has demonstrated superior performance in certain deep-learning applications.
(viii) Softsign: $\varphi(\beta) = \beta/(1 + |\beta|)$, an alternative to the tanh function that approaches its horizontal asymptotes more gradually.

The selection of an appropriate activation function is seldom trivial. It depends heavily on the flow physics of interest, the chosen network architecture and empirical parameter tuning to ensure smooth convergence while mitigating issues such as vanishing or exploding gradients. Because the training stability of the model also varies across different activation functions, it is essential to perform cross-validation, as discussed in § 2.2.4.

How nonlinear activation functions assist in studying fluid mechanics can be seen from various problem settings, including flow estimation and prediction. In particular, the distinction between linear and nonlinear functions inside machine-learning models is highlighted for compression problems by comparing a linear dimensional reduction of POD to its nonlinear counterparts. Although POD provides a unique solution without requiring any optimisations, let us express the POD with the reduced basis $\boldsymbol{\Phi} \in \mathbb{R}^{n_{\boldsymbol{\xi}} \times N}$ as an L_2 minimisation problem,

$$\boldsymbol{\Phi}^* = \underset{\boldsymbol{\Phi}}{\operatorname{argmin}} \, ||\boldsymbol{q}' - \boldsymbol{\Phi}^T \boldsymbol{\Phi} \boldsymbol{q}'||_2, \tag{2.17}$$

where $\boldsymbol{q}'$ denotes the fluctuating components of a variable $\boldsymbol{q}$. Equipped with (2.8), the optimisation of a nonlinear autoencoder with respect to the parameters is similarly expressed as

$$\boldsymbol{\Phi}^* = \underset{\boldsymbol{\Phi}}{\operatorname{argmin}} \, ||\boldsymbol{q}' - \varphi_d \boldsymbol{\Phi}^T \varphi_e \boldsymbol{\Phi} \boldsymbol{q}'||_2, \tag{2.18}$$

where $\boldsymbol{\Phi}$ here corresponds to the weights inside the network, and φ_e and φ_d represent the activation functions of the encoder and decoder, respectively. Based on (2.17) and (2.18), a nonlinear autoencoder becomes identical to POD when the activation function is set to be linear, such that $\varphi_e = \varphi_d = 1$ (Baldi & Hornik 1989; Fukami *et al.* 2021*b*).

In turn, machine-learning models with nonlinear activation functions could provide better compression by introducing nonlinearities in the form of (2.18). It has been widely

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

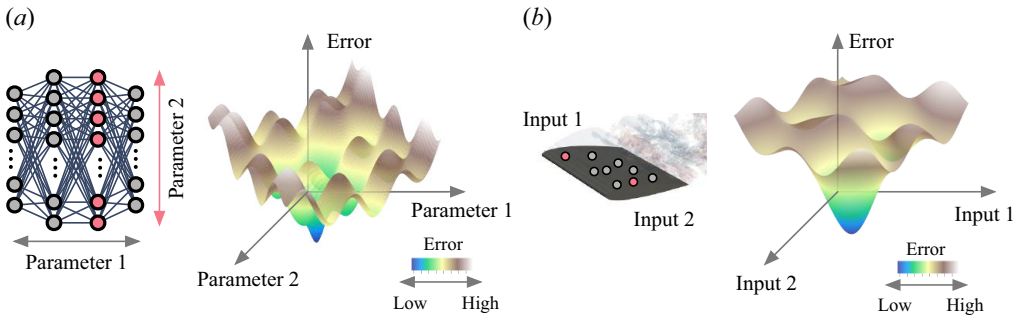


Figure 16. Loss surface with respect to (*a*) model parameters, and (*b*) input data.

reported that the use of nonlinear activation functions reduces reconstruction error at the same number of modes n_ξ compared with linear techniques for a variety of flow examples, as nonlinear autoencoders enable projecting the high-dimensional fluid flow data onto a curved, lower-dimensional manifold (Murata *et al.* 2020; Linot *et al.* 2023; Fukami & Taira 2025).

This has often been discussed with the concept of the Kolmogorov barrier or the Kolmogorov n width (Peherstorfer 2022). The Kolmogorov n width expresses the theoretical lower bound of the approximation error achievable by any optimal n-dimensional linear subspace. For advection-dominated flows, this n width decays slowly, indicating that linear methods require a large number of modes to accurately reconstruct the flow. Nonlinear autoencoders are not constrained by this barrier, capturing complex dynamics with more compact representations compared with linear techniques. The above points will also be evident through the present case study in § 3 from a comparison of data compression between POD and nonlinear autoencoders for unsteady aerodynamic flows.

2.3.7. *Parameter selection*

Once the network architecture is established with nonlinear activation functions, the subsequent challenge lies in tuning the hyperparameters of machine-learning models. In practice, finding a globally 'optimal' parameter set is nearly impossible due to several fundamental reasons. First, the loss landscape in nonlinear machine learning is generally non-convex, fraught with local minima and saddle points. Second, modern machine-learning architectures often possess a large number of tunable parameters specifically for fluid flows with a high degree of freedom in space and time, making naive parameter sweeps computationally intractable (Nakamura *et al.* 2022). Third, the sensitivity of the cost function to different parameters can significantly vary. Along with the second point, this makes finding the optimal parameter set further challenging. Fourth, the optimal model configuration also depends on the input data itself. For example, varying the spatial locations of input sparse sensors for the flow reconstruction problem may change the loss surface, as depicted in figure 16(*b*).

Because of such mathematical and computational hurdles, 'optimised' or sufficiently robust parameter sets are sought rather than the 'optimal' one. One practical remedy is to employ a step-by-step heuristic procedure, often referred to as a manual search (Bergstra & Bengio 2012; Fukami *et al.* 2020*a*). As an example, let us consider a MLP. The objective is to optimise the number of hidden layers n_{layer} and the number of nodes per layer (n_{node}), by performing an iterative procedure as follows.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

(i) Fix the number of layers n_{layer} and perform a parameter sweep to identify the number of units n^*_{unit} that minimises the loss function. To ensure the model generalises well to unseen scenarios, k-fold cross-validation should be performed during this step.
(ii) Subsequently, fix the units at the newly found n^*_{unit} and vary the number of layers to identify the n^*_{layer} that yields the minimum loss, again performing cross-validation.
(iii) The resulting parameter pair $(n_{unit,opt}, n_{layer,opt}) = (n^*_{unit}, n^*_{layer})$ is then adopted as the optimised configuration for the machine-learning model. One can also iterate over steps (i) and (ii) until convergence is achieved.

This can also be applied to other machine-learning models, e.g. CNNs, in which the number of filters and the kernel size can be optimised. An optimiser used in solving the problem based on the cost function, such as the Adam optimiser (Kingma & Ba 2014), can be an additional parameter, which may be incorporated into the above routine (Kiyani *et al.* 2025).

One can also consider automated hyperparameter optimisation approaches, such as a grid search (Bishop & Nasrabadi 2006), a random search (Bergstra & Bengio 2012) and Bayesian optimisation (Bergstra *et al.* 2015). Note, however, that these methods perform optimisation of the network parameters, which itself minimises a cost function, within an outer optimisation loop with the original cost function based on the problem setting, implying that this dual loop increases the computational cost. Based on the availability of computational resources and the requirement of model performance, an appropriate approach needs to be selected for the parameter decision of nonlinear machine-learning models.

2.4. *Model training with cost function design*

Let us discuss here how the cost function is designed for studying fluid flows with machine learning. This is a critical step, as the cost function simply determines what types of flow characteristics the model needs to prioritise in the learning process. In the standard formulation, whether for estimation, reconstruction, prediction, compression or generation, the cost function is typically defined by the data reconstruction error, such as the L_P norm of the residual $\mathcal{E}_d = ||\boldsymbol{q}_{out} - \mathcal{F}(\boldsymbol{q}_{in}; \boldsymbol{w})||_P$ expressed in (2.3).

However, data-driven models optimised solely to minimise the data reconstruction error $\mathcal{E}_d$ often suffer from overfitting, lack of interpretability or the failure to satisfy fundamental physical constraints such as conservation laws. To address these limitations, it is common practice to introduce an additional penalty (or regularisation) term, $\mathcal{E}_p$, to better constrain the optimisation landscape and the resulting solution space. The total cost function then takes the generalised form,

$$\mathcal{E} = \mathcal{E}_d + \sum_i \beta_i \mathcal{E}_{p,i}, \tag{2.19}$$

where β is a hyperparameter that scales the relative importance of the penalty term.

The design of $\mathcal{E}_p$ depends on the desired structural or physical properties of the model. For instance, one can employ an L_1 penalty on the model weights, such that

$$\mathcal{E}_p = ||\boldsymbol{w}||_1, \tag{2.20}$$

promoting sparsity within the network by effectively pruning redundant parameters to prevent overfitting (Tibshirani 1996). It is also central to sparse identification techniques, where it helps isolate the most dynamically relevant features or active terms in an underlying governing equation (Brunton *et al.* 2016).

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Alternatively, $\mathcal{E}_p$ can also be designed to embed prior knowledge of physics into the learning process. A prominent example is the physics-informed penalty, which evaluates the output of machine-learning models based on governing equations (Lagaris *et al.* 1998; Raissi *et al.* 2019). Considering the residual of the physical laws as a penalty, the model is guided toward solutions that are not only consistent with the training data but also physically valid. Equipped with careful design of these cost functions, machine-learning models may become not only robust and parsimonious but also physically sound (Brunton 2021).

The foundational application of physics-based penalties is seen in PINNs (Raissi *et al.* 2019). In the standard PINN formulation, the neural network acts as a continuous, parameterised surrogate model for the flow field itself, mapping spatiotemporal coordinates $(\boldsymbol{x}, t)$ to state variables at a particular time stamp, $\boldsymbol{q} = \mathcal{F}(\boldsymbol{x}, t; \boldsymbol{w})$. One can then perform automatic differentiation (Baydin *et al.* 2018) to obtain exact partial derivatives of the output variables with respect to space and time without relying on numerical discretisations because neural networks are often differentiable. The physical penalty $\mathcal{E}_p$ is then formulated by evaluating the residual of the governing partial differential equations, such as the Navier–Stokes equations, at a set of sampled collocation points in the domain. This physics-informed loss is expressed as

$$\mathcal{E}_p = ||\mathcal{N}(\boldsymbol{x}, t, \mathcal{F}(\boldsymbol{x}, t; \boldsymbol{w}))||_P, \tag{2.21}$$

where $\mathcal{N}$ denotes a differential operator. By minimising this residual along with any available observational data or boundary conditions $\mathcal{E}_d$, the network aims to learn a continuous representation of the flow that physically follows the governing equations.

While standard PINNs learn the relationship between coordinates and flow variables, the concept of a physics-based cost function can be extended to models that perform field-to-field mappings, such as super-resolution analysis. There are several approaches to introducing physics-based loss terms for such reconstructed fluid flows. When all state variables are available, they can directly be substituted into the governing equations, yielding an error based on the Navier–Stokes operator $\mathcal{N}$. In addition, minimising a loss function incorporating only certain terms of the Navier–Stokes equation (Lee & You 2019) can also be considered, i.e.

$$\mathcal{E}_p^j = ||\mathcal{N}_j(\boldsymbol{q}_{Ref}) - \mathcal{N}_j(\mathcal{F}(\boldsymbol{q}_{in}))||_P, \mathcal{N} = \sum_j \mathcal{N}_j, \tag{2.22}$$

where $\mathcal{N}_j$ is a term in the governing equation and $\boldsymbol{q}_{Ref}$ is a reference data, which is similar to semi-supervised learning (Zhu & Goldberg 2009). Nonlinear machine-learning models trained in a physically constrained solution space are particularly effective when the amount of training data are small (Gao *et al.* 2021; Cai *et al.* 2022).

As an example, let us present in figure 17 how a physics-based cost function could improve nonlinear machine learning with super-resolution reconstruction of two-dimensional decaying isotropic turbulence (Fukami *et al.* 2023). Here, the flow is governed by the vorticity transport equation,

$$\frac{\partial \omega}{\partial t} + \boldsymbol{u} \cdot \nabla \omega = \frac{1}{Re} \nabla^2 \omega, \tag{2.23}$$

where $\boldsymbol{u}$ is the velocity vector, ω is the vorticity and Re is the Reynolds number. The model aims to reconstruct a high-resolution vorticity field on a 512^2 grid from downsampled input data ω_{LR} on a 16^2 grid.

Since this is an instantaneous reconstruction and the time derivative $\partial\omega/\partial t$ is not available, one can consider the case with (2.22) in which some terms of the governing

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

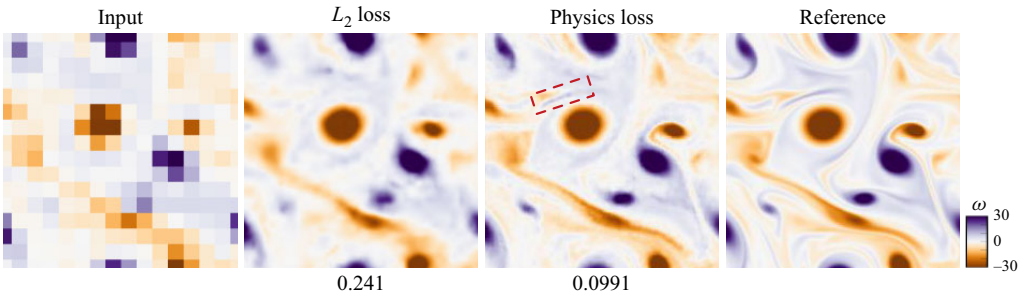


Figure 17. The impact of the cost function design on super-resolution reconstruction of turbulent flows (Fukami *et al.* 2023). The value underneath the reconstructed flow fields reports an L_2 reconstruction error norm.

equation can be obtained. Let us then augment the cost function with the advection and viscous terms from (2.23),

$$\begin{aligned}\mathcal{E} &= \mathcal{E}_d + \beta_{adv}\mathcal{E}_{adv} + \beta_{visc}\mathcal{E}_{visc},\\ \mathcal{E}_d &= ||\omega_{ref} - \mathcal{F}(\omega_{LR})||_2,\\ \mathcal{E}_{adv} &= ||\boldsymbol{u}_{ref}\cdot\boldsymbol{\nabla}\omega_{ref} - \boldsymbol{u}_{ML}\cdot\boldsymbol{\nabla}\omega_{ML}||_2,\\ \mathcal{E}_{visc} &= ||\nabla^2\omega_{ref} - \nabla^2\omega_{ML}||_2,\end{aligned} \tag{2.24}$$

where β_{adv} and β_{visc} are the weighting coefficients for the advection and viscous terms, respectively. The estimated velocity vector $\boldsymbol{u}_{ML}$ and other operators $(\cdot)_{ML}$ are obtained from the super-resolved vorticity field ω_{ML}.

The reconstructed flow fields with a regular reconstruction norm and the physics-based norm are compared in figure 17 (Fukami *et al.* 2023). The downsampled skip-connection/multi-scale (DSC/MS) model shown in figure 6 is employed as the baseline machine-learning model (Fukami *et al.* 2019*a*). While the model with a standard L_2 error norm reasonably reconstructs vortical structures, the physics-based model captures fine flow features, such as the shear layers interspersed around large structures with negative vorticity, as highlighted with a red box. The reconstruction error is correspondingly reduced from 24 % to under 10 %. We note that physics-inspired analysis can suffer from large numerical errors if the reconstructed field $\boldsymbol{q}_{HR}$ contains high-frequency noise. This approach assumes that the output fields are sufficiently smooth to evaluate spatial derivatives accurately and should therefore be used with caution.

In addition to incorporating the knowledge from governing equations into the loss function, using external physical observables can be considered, particularly for latent space identification with nonlinear machine-learning-based compression. This is achieved with an observable-augmented architecture (Fukami & Taira 2023), which is briefly discussed with an example of an extreme vortex-gust encounter in figure 4. This approach introduces an additional subnetwork that estimates a specific physical observable κ, such as sensor readings and aerodynamic loads, from the latent vector $\boldsymbol{\xi}$.

In a similar form to (2.19), the observable estimation term is added to the cost function. Training an observable-augmented CNN autoencoder is performed by minimising a joint cost function,

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, ||\boldsymbol{q} - \widehat{\boldsymbol{q}}||_2 + \beta||\kappa - \widehat{\kappa}||_2, \tag{2.25}$$

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

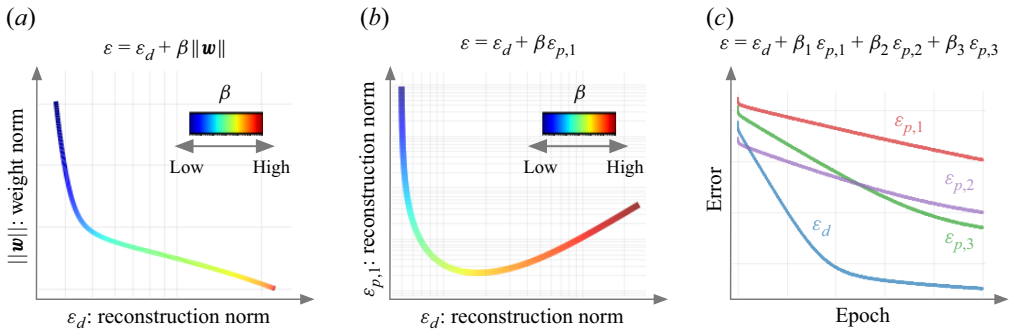


Figure 18. Selection of tuning parameters for the cost function. (*a*) Sparsity-promoting cost function: the trade-off relationship between the reconstruction term ε_d and the weight term $||\boldsymbol{w}||$ is depicted. (*b*) Observable-augmented cost function: there exists a hot spot that can balance the reconstruction terms ε_d for the primary interest and $\varepsilon_{p,1}$ for an observable. (*c*) An example trace of the multiple loss terms during the training process.

where κ is chosen based on what characteristics we would preferentially extract from a given high-dimensional flow data in a compact manner – that is, prior knowledge of flow physics. In other words, the network parameters are explicitly tuned to prioritise important flow structures associated with the given physical observables. From a training perspective, this observable-based augmentation offers a stable learning process by better constraining the solution space during fluid flow data compression (Eldredge & Mousavi 2026).

Depending on the physical characteristics of the object to be captured in a low-order space, a variety of variables can be considered for observable augmentation. For example, the lift coefficient C_L has been used to characterise important vortical structures in subspace identification following a well-known relationship between circulation Γ and C_L (Fukami & Taira 2023; Fukami *et al.* 2025*a*; Mousavi & Eldredge 2025). While the drag coefficient C_D has been incorporated for aerodynamic shape optimisation of automobiles (Tran *et al.* 2024), the friction Reynolds number has been considered to discuss the primary features of the turbulent boundary layer in a low-order subspace (Fukami & Taira 2025). Such a physics-based latent space facilitates sparse sensor-based reconstruction, shape optimisation and reinforcement learning-based control for a variety of unsteady flows (Liu *et al.* 2025*b*).

More broadly, an 'observable' may not need to be a variable. In fact, there are some endeavours seeking the use of some mathematical techniques as an additional penalty term $\mathcal{E}_p$, such as persistent homology (Smith *et al.* 2024), information theory (Fukami & Araki 2026; Koshikawa *et al.* 2026), phase-amplitude reduction (Yawata *et al.* 2024) and optimal transport (Tran *et al.* 2026), to gain physical insights from fluid flows with nonlinear data compression. This enables fluid mechanicians to enjoy learning physics from data.

Once establishing additional terms $\mathcal{E}_{p,i}$, we then need to find appropriate weighting parameters β_i. A classical approach to this hyperparameter selection is a Pareto analysis, often visualised via an L curve (Hansen & O'Leary 1993). When considering weight sparsity through an L_1 penalty as in (2.20), one can plot the reconstruction norm $\mathcal{E}_d$ against the weight norm $||\boldsymbol{w}||$ across a range of β values as depicted in figure 18, typically exhibiting an L-shaped curve. The optimal balance is chosen at the elbow of the curve, representing a parsimonious model where neither the reconstruction error nor the parameter magnitude disproportionately dominates (Hansen & O'Leary 1993).

When the additional term is coupled with the primary objective, such as in the case of an observable-augmented autoencoder, the aforementioned straightforward trade-off may not manifest, as illustrated in figure 18(*b*). Since the network simultaneously learns

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

to compress the flow field and estimate the auxiliary physical variable, excessively constraining the observable term causes the primary autoencoding process to fail. Consequently, the observable estimation cannot be accurately performed since the secondary output is a function of the latent variables that need to be a low-order representation capturing the physics. As a result, the curve rather presents a V-shaped behaviour, as shown in figure 18(*b*).

In addition to such Pareto-type strategies, adaptive weighting approaches can be considered during the optimisation process (Wang *et al.* 2021; Zhang & Deng 2025). These optimisation algorithms dynamically adjust β_i based on the gradient statistics or the relative magnitudes of the individual loss terms, ensuring stable convergence even for highly coupled physical constraints.

Beyond simply minimising the total loss, monitoring the evolution of the individual error components over the training process can serve as a diagnostic tool in studying fluid mechanics. Observing which error terms decay rapidly or stagnate provides insights into the learning dynamics, as illustrated in figure 18(*c*). In fluid mechanics, tracking these individual trajectories highlights which flow physics the network readily captures and which remain elusive during training. For example, Krishnapriyan *et al.* (2021) examined how a PINN learns the underlying physics of convection-dominated flows, accounting for the dynamic behaviour of each loss term. This transparency provides feedback for refining model architectures and optimising data-preparation strategies in fluid mechanics.

2.5. *Assessments*

Once a machine-learning model is trained, carefully assessing its performance is the next step. In fluid mechanics, standard machine-learning evaluation practices, which often rely on minimising a single-scalar loss value, may not be sufficient. Rather, evaluating machine learning with fluid flows necessitates a comprehensive understanding of both the mathematical error metrics and the underlying flow physics.

A straightforward way to evaluate a machine-learning model is to use mathematical error metrics. There exists a variety of error metrics that capture different aspects of the model error, as summarised in table 2. Pointwise error metrics, including the L_1 and L_2 norms, are often considered due to their simplicity and direct relationship to absolute mass differences or kinetic energy. However, they can respectively blur high-frequency structures or struggle with smooth gradient optimisation (Fukami *et al.* 2020*a*). Other metrics, such as the logarithmic norm, are better suited for flows with massive dynamic ranges of variables, e.g. reactive flows and compressible turbulence. In addition, the structural similarity index measure (SSIM; Wang *et al.* 2004) and cosine similarity focus on spatial coherence and phase alignment rather than raw magnitude, which can be used to assess flow fields from an image-diagnostics perspective, such as how much rotation and shear-layer structures are qualitatively captured (Morimoto *et al.* 2021*b*). Depending on the flows of interest and their statistical properties, an appropriate choice of error metrics is needed.

While these error metrics are well designed to capture a range of data characteristics, it is important to note that the over-reliance on a single-scalar error metric is sometimes dangerous. To illustrate the fragility of a scalar metric, let us present in figure 19 several flow snapshots around an aerofoil that all yield an identical L_2 error norm of 0.750 when compared with the reference field. Despite sharing the same pointwise error magnitude, their actual structural representations significantly vary. In this case, applying the SSIM provides structural fidelity ranging from low to high, which somewhat coincides with our intuition. This example may illustrate why relying purely on a global error metric is not encouraged.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Metric	Pros	Cons
L_1 norm	Robust to outliers, encourages sharper gradients and more distinct flow structures compared with L_2 norm	Treats all spatial errors equally regardless of local flow features
L_2 norm	Standard least-squares metric penalises large errors, directly relates to energy in velocity fields	Sensitive to extreme outliers, tends to yield visually blurred or smoothed flow fields
Log norm	Well suited for capturing flow quantities spanning multiple orders of magnitude (e.g. passive scalars, chemical species)	Cannot be used for zero or negative values, requires careful scaling or thresholding
SSIM	Evaluates structural, luminance and contrast similarities, correlates well with human visual perception of structures	Computationally heavier than pointwise norms, not a strict mathematical distance metric
Cosine similarity	Captures directional and phase alignment of features, useful for assessing vector field topology	Ignores magnitude differences, fields with identical shapes but different energies yield perfect similarity

Table 2. Common error metrics for assessing machine-learning analysis of fluid flows.

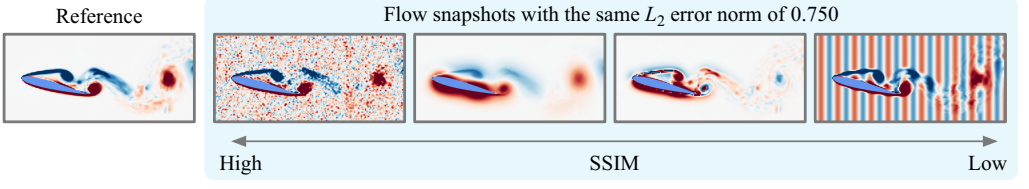


Figure 19. Fragility of a scalar error metric for assessing fluid flows. The flow fields in the blue box report the same L_2 error reconstruction norm compared with the reference field.

It is hence important to assess the flow field visually and statistically, in addition to numerically, as we have performed in computational and experimental fluid mechanics studies. High-order statistics, such as turbulent kinetic energy spectra, Reynolds stress distributions, spatial correlation functions and probability density functions, also provide a comprehensive understanding of machine-learned outcomes.

We further note that numerical accuracy is only one aspect of assessing nonlinear machine learning in fluid mechanics. For example, in manifold learning and data compression, what flow physics we gain as an outcome, i.e. the extracted latent variables and geometry, is of importance. While it is challenging to quantify how physical or physically interpretable these variables are, particularly for cases in which fluid flow data are projected into an abstract subspace, we should be mindful of the aspect of interpretability. How the 'effectiveness' of machine learning in fluid mechanics can be defined and systematically assessed remains an open question. For broader discussions on this viewpoint, we refer to Bauerheim (2026).

The convergence of the model with respect to the amount of information provided during the training needs to be carefully investigated. As the model performance depends on variations in the data capacity, such as the number of training snapshots, the number

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

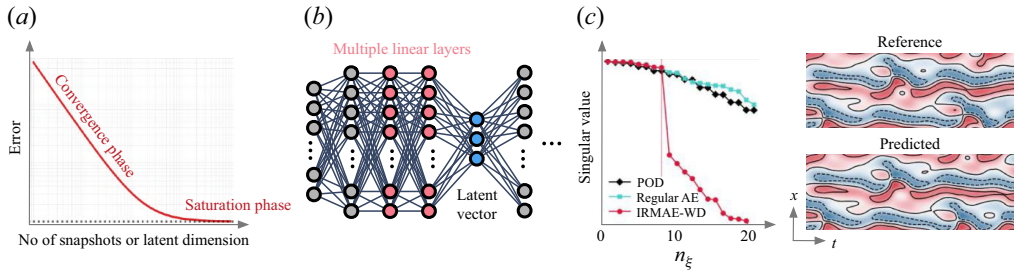


Figure 20. Convergence check for machine-learning analysis of fluid flows. (*a*) A general relationship between the model error and data amount, including the number of training snapshots and the latent dimension. (*b*) Implicit rank minimising autoencoders (IRMAE) to find the embedding dimension of given data with singular value decay (Jing *et al.* 2020). (*c*) Applications of IRMAE for the Kuramoto–Sivashinsky equation (Zeng *et al.* 2024).

of sensor measurements in flow estimation problems or the latent dimension in data-driven compression, understanding convergence properties is critical from the aspect of the reproducibility and reliability of machine-learning analyses in fluid mechanics.

As conceptually illustrated in figure 20(*a*), the relationship between the model error and the amount of given data (or latent dimension) generally exhibits two distinct regimes. During the initial convergence phase, increasing the data volume or expanding the latent dimension reduces the reconstruction or prediction error, as the network starts to capture flow characteristics from given data. The model then dives into a saturation phase, where the model no longer gains additional information from data. Identifying the onset of this saturation phase is important for building parsimonious models. Setting the latent dimension larger or excessively increasing the data size simply increases computational costs, even causing the risk of overfitting without providing additional physical insights.

Regarding the latent dimension with nonlinear compression, it is challenging to determine the optimal number *a priori* as the standard singular value decay is not directly applicable. While the convergence of reconstruction error over the latent dimension is then generally examined, one can employ implicit rank-minimising autoencoders (IRMAE; Jing *et al.* 2020). An IRMAE modifies the standard autoencoder architecture by inserting multiple linear layers between the encoder and decoder, as illustrated in figure 20(*b*). Jing *et al.* (2020) have shown that the optimisation dynamics of these linear layers introduce an implicit regularisation effect that encourages low-rank weight matrices during the training process. This algorithm actually mimics the singular value decay seen in POD, driving the variance of redundant latent variables to zero and revealing the embedding dimension of the given data.

An example of the IRMAE use for the Kuramoto–Sivashinsky equation is presented in figure 20(*c*) (Zeng *et al.* 2024). They combined IRMAE with the so-called weight decay (IRMAE-WD), enabling the nonlinear network to automatically identify the embedding dimension of chaotic systems without requiring the standard trial-and-error process. The predicted spatiotemporal evolution with the identified dimension reproduces the wave dynamics of the reference data. Such data-driven tools can be considered to facilitate machine-learning-based studies in fluid mechanics with appropriate assessments.

2.6. *Improvements*

This subsection considers how the knowledge accumulated through all the preceding steps covered above can be used to iteratively improve model performance and extract physical insights from fluid flow data. Training a machine-learning model is rarely a

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

one-shot process. The diagnostics discussed in the previous section often reveal specific shortcomings in the baseline model, offering some clues for improvements. Depending on the identified failure modes, fluid mechanicians can consider several avenues for refinement at each step in the present guide.

(i) Data augmentation: when a convergence study discussed in figure 20(*a*) indicates that the model error has not reached the saturation phase, the performance may be improved simply by adding more training snapshots.
(ii) Architectural modifications: when the model struggles to capture fine-scale turbulent structures, which can be revealed through the flow visualisation along with the error assessment, one can consider adjusting the network depth (adding or removing layers), using multiple convolutional filter sizes or introducing skip connections to better learn high-frequency information.
(iii) Cost function re-design: when the reconstructed fields lack physical fidelity despite a low L_2 error, physics-based regularisation terms can be introduced, such as physics-informed penalties, observable-augmented learning and sparsity penalties.
(iv) Sensor and input optimisation: for flow estimation tasks, poor performance may stem from insufficient or redundant input variables, which could be solved by adding, removing or relocating sensors.

Importantly, the trained machine-learning model itself can serve as a powerful tool to guide these improvements, particularly in the context of experimental design and sensor optimisation. For example, gradient-based sensitivity analyses, such as gradient-weighted class activation mapping (Grad-CAM; Selvaraju *et al.* 2017), enable the interpretation of the inner functions of a trained nonlinear network. By taking the gradient between the input and output, such techniques highlight which regions of an input mostly influence the output of machine-learning models (Jagodinski *et al.* 2023). This has been used not only to identify sensitive locations but also to reduce the number of sensors in the context of aerodynamic coefficient estimation (Chen *et al.* 2024) and flow field reconstruction (Fukami *et al.* 2025*a*). Some explainable machine-learning techniques, such as shapley additive explanations (SHAP; Lundberg & Lee 2017), can also be considered to extract the regional importance of the input flow field for data-driven estimation, which has been applied to wall-bounded turbulence (Cremades *et al.* 2024).

Note again that the incorporation of physical knowledge at each step of the present guide provides invaluable insights into how machine-learning analysis can be systematically improved in studying fluid flows. By avoiding naive applications of black-box techniques, nonlinear machine learning can then be a powerful extractor of underlying flow physics, rather than serving as a mere interpolatory function.

3. Case study: data-driven compression of turbulent wake around an aerofoil

In this section we perform a case study of the nonlinear machine-learning-based analysis for fluid flows. We consider a relatively simple problem of data-driven compression for the three-dimensional turbulent wake around an aerofoil. Our aim here is to introduce how the present guideline offered in § 2 can be employed to analyse unsteady flows in a step-by-step manner, serving as a methodological demonstration while discussing physical findings from machine-learned outcomes.

Once the problem setting is defined (step in § 2.1), let us then consider how data sets are curated (step in § 2.2). This study examines flow around a NACA0012 aerofoil at a chord-based Reynolds number of 5000 with the angle of attack $\alpha = 14°$, producing an unsteady

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

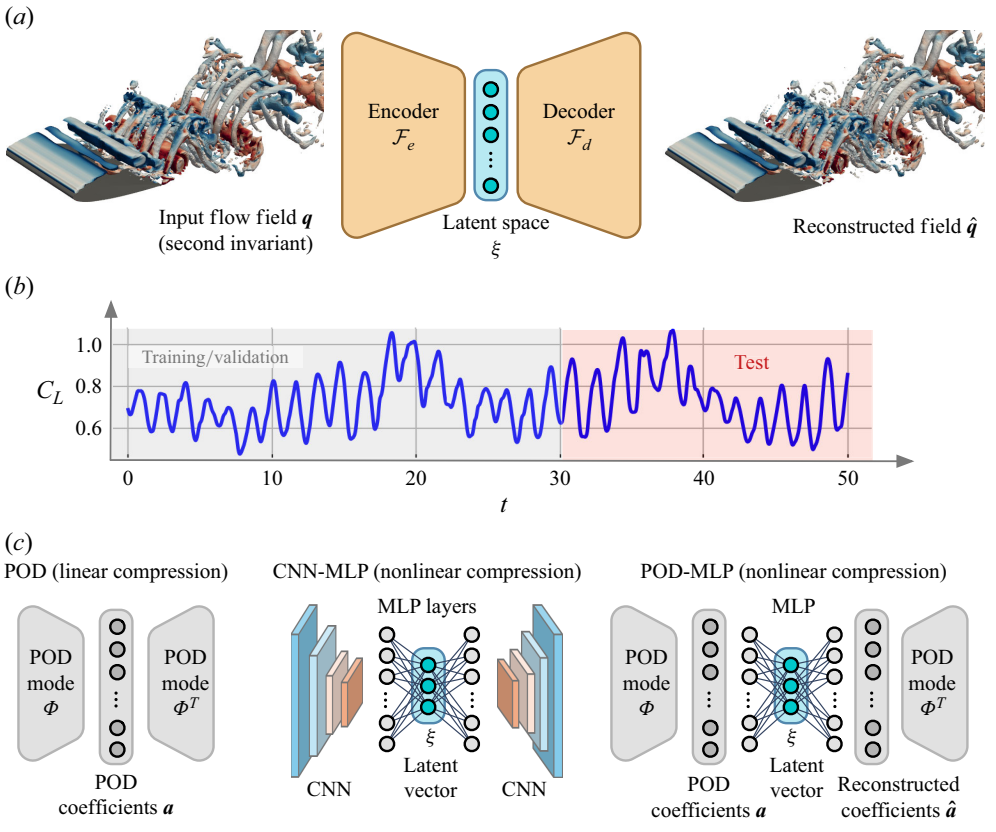


Figure 21. Autoencoder-based compression of turbulent wake around a NACA0012 aerofoil at $Re_c = 5000$ for the present case study. (*a*) A generic autoencoder formulation. An example reconstruction by the POD-MLP model with $n_{\boldsymbol{\xi}} = 3$ is shown. (*b*) Training/validation/test data split used for the example of a statistically stationary turbulent wake. (*c*) Three compression techniques considered in the case study.

turbulent separated wake as presented in figure 21(*a*) (Fukami *et al.* 2025*b*). To prepare the current data set, a large-eddy simulation is carried out using an incompressible flow solver Cliff based on the finite-volume formulation with second-order accuracy in time and space (Ham & Iaccarino 2004; Ham *et al.* 2006). The computational domain is set over $(x, y, z)/c \in [-20, 25] \times [-20, 20] \times [0, 1]$ with the leading edge of the wing positioned at the origin. In the spanwise direction, the periodic boundary condition with an extent of $1c$ is imposed so that the three-dimensional flow structures can be captured (Rolandi *et al.* 2025). The Vreman subgrid-scale model (Vreman 2004) is used for the turbulent closure. Further details on the simulation setu-p are found in Fukami *et al.* (2025*b*).

For the present analysis, we have interpolated data sets onto a Cartesian, uniform grid with spatial grid points $(n_x, n_y, n_z) = (300, 150, 50)$ considering part of the computational domain $(x, y, z)/c \in [-1.5, 4.5] \times [-1.5, 1.5] \times [0, 1]$. This is particularly motivated by the use of a convolutional model expressed later. As we not only discussed in § 2 but also performed herein, the steps of data curation and model construction can be interactive with each other. Considering the statistically steady nature of the flow field as observed from the time trace of the lift coefficient C_L in figure 21(*b*), the training/validation and test data are respectively prepared over $0 \leqslant t < 30$ and $30 \leqslant t < 50$, which follows the discussion in § 2.2.4.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Method	Strengths	Weaknesses
POD	Uniqueness orthogonal modes	Less compression compared with nonlinear approaches
CNN-MLP autoencoder	High compression accommodates spatial arrangements of vortical structures (high robustness against flow pattern variations)	High training costs non-uniqueness (latent space depends on initialisation/training)
POD-MLP autoencoder	High compression lower training costs than CNN-MLP	Non-uniqueness less robust against flow pattern variations than CNN-MLP

Table 3. Strengths and weaknesses of data-driven compression models for fluid flows covered in the current case study.

We consider three data-driven compression approaches (model construction step: § 2.3), namely, POD, a nonlinear convolutional autoencoder with MLP (CNN-MLP) and a nonlinear fully connected autoencoder with POD (POD-MLP), as illustrated in figure 21(*c*). This set-up is intended to cover a variety of model characteristics for linear and nonlinear compression techniques often used in fluid mechanics, as summarised in table 3. While POD is considered to highlight how nonlinear activations inside the autoencoder are employed, there are two nonlinear networks. The CNN-MLP models have been considered for a range of fluid flow compression problems (Fukagata & Fukami 2025). A CNN is first used to learn large-scale structures in a flow field through filter-based operations. The MLP is then inserted into the bottleneck portion of the autoencoder, where the data dimension is low and the spatial coherence of vortical structures becomes less important than that considered in the CNN counterpart. As the CNN nonlinearly learns a localised spatial neighbourhood through the filter operation while covering the entire domain, the CNN-MLP-based autoencoder often presents high adaptability for a variety of vortical patterns (Fukami & Taira 2023).

However, because the CNN-MLP model takes the entire field as the input and output, its computational cost becomes expensive. As a remedy, one can consider performing POD before employing a nonlinear autoencoder. Proper orthogonal decomposition is first used to gain mid-order representations, time-varying coefficients $\boldsymbol{a} \in \mathbb{R}^{n_a}$, and they are used as the input/output of the MLP-based autoencoder such that

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmin}} ||\boldsymbol{a} - \mathcal{F}_d(\mathcal{F}_e(\boldsymbol{a}; \boldsymbol{w}_e); \boldsymbol{w}_d)||_2, \tag{3.1}$$

similar to the operation in (2.8). The decoded POD coefficients $\widehat{\boldsymbol{a}}$ are then multiplied by the POD bases $\boldsymbol{\Phi}$ to reconstruct a flow field,

$$\widehat{\boldsymbol{q}} = \boldsymbol{\Phi}\widehat{\boldsymbol{a}} = \boldsymbol{\Phi}\mathcal{F}_d(\mathcal{F}_e(\boldsymbol{a})) = \boldsymbol{\Phi}\mathcal{F}_d(\boldsymbol{\xi}), \tag{3.2}$$

while extracting a low-order representation of flow field $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$. As the data dimension of the POD coefficient is much lower than that of the flow field $\boldsymbol{q} \in \mathbb{R}^{n_q}$ in general, i.e. $n_\xi < n_a < n_q$, the training cost for the autoencoder can be substantially reduced.

This entire process is regarded as nonlinear compression, since linear, mid-order representations of the POD coefficients are nonlinearly projected into the latent subspace. This idea has been used in a range of problems including shape optimisation (Tran *et al.* 2024) and reduced-order modelling (Linot *et al.* 2023). The robustness or extrapolation capability of the POD-MLP model against different flow configurations is expected to be likely less than that of the CNN-MLP model, as the large-scale flow structures are first

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

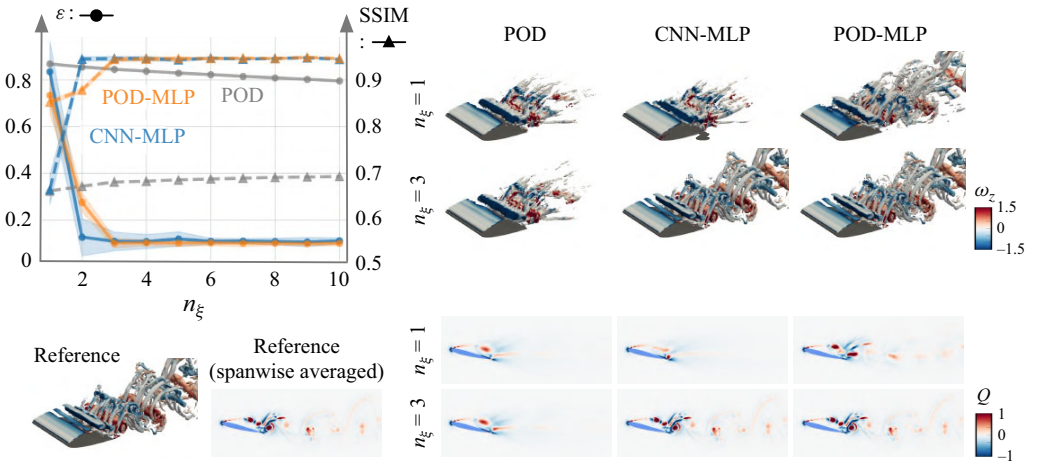


Figure 22. Autoencoder-based compression for a turbulent wake around a NACA0012 aerofoil at $Re_c = 5000$. The relationship between the latent dimension and the reconstruction errors (an L_2 error norm ε and SSIM), along with representative reconstructions using POD, a CNN-MLP autoencoder and a POD-MLP autoencoder, is presented. The three-dimensional reconstructed fields are visualised with a Q isosurface coloured by the spanwise vorticity ω_z. The spanwise-averaged reconstructed fields are also depicted.

linearly extracted. While there are pros and cons in data-driven compression techniques, they can be complemented with each other by either combining them or introducing some other techniques, such as variational autoencoders that incorporate a concept of generation into the compression problem (Solera-Rico *et al.* 2024).

Once the model is trained at step (iv) (§ 2.4), we assess its performance (step in § 2.5) and consider the possibility of model improvement (step in § 2.6). Let us examine how much the present aerodynamic flow data are compressed with data-driven techniques, as shown in figure 22. Here, we consider a three-dimensional field of the second invariant of the velocity gradient tensor Q (Hunt *et al.* 1988), as the variable of interest to highlight how complex turbulent wake structures composed of vortical cores and shear layers appear through compression.

To identify the embedding dimension in the current setting, the dependence of reconstruction performance on the latent dimension $n_{\boldsymbol{\xi}}$ is examined for all three compression techniques. These models are trained with 3000 snapshots over the time period of $0 \leqslant t < 30$ shown in figure 21. The reconstruction error is assessed with a normalised L_2 norm, $\varepsilon = ||\boldsymbol{q} - \widehat{\boldsymbol{q}}||_2/||\boldsymbol{q}||_2$, and SSIM, based on the test data sampled over $30 \leqslant t < 50$. For the CNN-MLP and POD-MLP models that require optimisation, five-fold cross-validation is performed to ensure reproducibility, and the standard deviations over five trials are shown as shaded areas in figure 22. The tanh function is used as an activation function inside the baseline nonlinear models, although the dependence of compression performance on the nonlinear activation function is examined later. Five hundred POD mode coefficients are used as the mid-dimensional representation of the present POD-MLP model.

The nonlinear models provide better compression than POD due to the use of nonlinear activation functions, as discussed in § 2.3.6. While POD presents a slow decay of the L_2 error norm over the latent dimension, the nonlinear models exhibit a sharp drop in the error at $n_{\boldsymbol{\xi}} = 2$ and plateau behaviour for $n_{\boldsymbol{\xi}} \geqslant 3$. In other words, the present turbulent wake data can be nonlinearly compressed to three variables. To achieve the same reconstruction level using POD, approximately 200 modes are required. While the two-dimensional space is

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

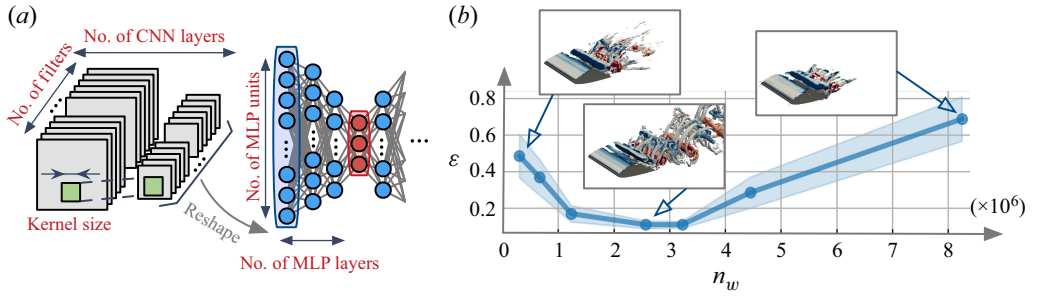


Figure 23. Impact of the model complexity on compression performance. (*a*) Adjustable parameters that contribute to the number of weights n_w inside the CNN-MLP autoencoder. (*b*) Dependence of the reconstruction error on the number of weights n_w.

needed to express the phase and amplitude information of the given flow analogous to some periodic aerodynamic flows (Murata *et al.* 2020; Fukami & Taira 2023), the third dimension may be needed to describe the multi-frequency, quasi-cyclic behaviour of the current turbulent wake.

Representative reconstructions with $n_\xi = 1$ and 3 are showcased in figure 22. With $n_\xi = 1$, POD and the CNN-MLP model poorly reconstruct the flow field, exhibiting structures near the wing only, as they have higher energy than the wake region. Note that the three-dimensional view of the POD-MLP model-based reconstruction with $n_\xi = 1$ looks plausible but is not accurate, which is evident from the spanwise-averaged field presenting somewhat blurred, elongated structures in the streamwise direction. This is because the pre-defined POD modes, which are spatially continuous, are multiplied by the decoded POD coefficients to reconstruct the entire flow field, providing a physically plausible but pseudo field.

By increasing the latent dimension to three, the nonlinear machine-learning models accurately reconstruct vortical structures across a range of spatial length scales. The reason why fine-scale information can still be retained through such a significant compression is likely due to the problem set in the present case study that considers a single, statistically steady aerodynamic flow scenario at a particular Reynolds number. Once the model aims to cover a variety of flow patterns, e.g. with different Reynolds numbers or some transient disturbances, the latent dimension for accurate reconstruction would be increased. In such a case, vortex cores are selectively extracted while fine-scale structures with shear layers are often discarded through the compression process (Fukami *et al.* 2025*b*).

Hereafter, we focus on the CNN-MLP model with $n_\xi = 3$ to demonstrate a variety of assessment approaches in studying fluid flows with nonlinear machine learning. In addition to the model convergence check with respect to the latent dimension above, we examine the dependence of the reconstruction error on the model complexity, as shown in figure 23. While the number of iterations or epochs during the training process can determine the model complexity, this case study uses the number of trainable parameters (weights) n_w inside the CNN-MLP model. As illustrated in figure 23(*a*), the CNN-MLP autoencoder can control several parameters that affect the number of weights, including the number of filters, the number of layers and the kernel size for CNN, and the number of units and layers for MLP, respectively. This study accordingly tunes them to demonstrate how the model behaviour becomes different across a wide range of the parameter amount n_w, as shown in figure 23(*b*). For detailed discussions on the impact of changing the

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

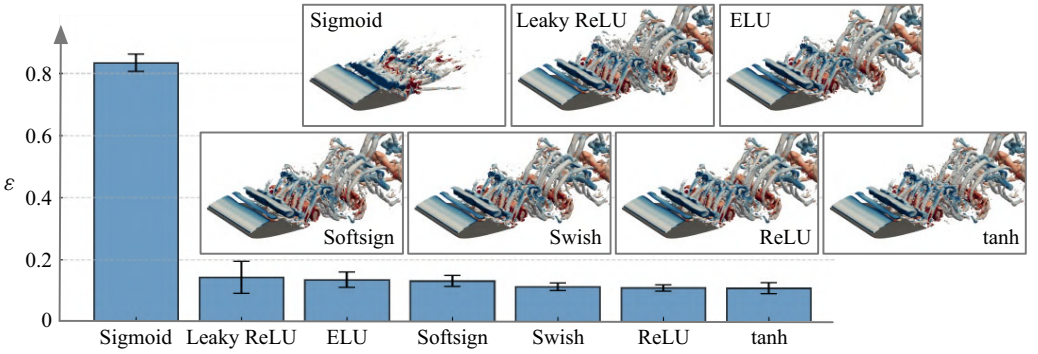


Figure 24. Dependence of the compression performance on the nonlinear activation function. The CNN-MLP autoencoder with $n_{\xi} = 3$ is considered. Error bars correspond to the standard deviation across five-fold cross-validation.

parameters of each model for CNN-MLP autoencoders, we refer the reader to Fukami *et al.* (2021*b*).

The error initially decreases as the number of weights $n_{\boldsymbol{w}}$ increases. It starts to converge with $n_{\boldsymbol{w}} \approx 2.5 \times 10^6$, revealing unsteady wake structures in the reconstructed field. However, the error then grows as the number of weights further increases. This is because training neural-network models is challenging when the number of weights is excessively large. Another observation here is that the standard deviation across the five-fold cross-validation becomes small near the bottom of the valley compared with the non-converged or overfitted states. In other words, such a converged model is needed not only for compression performance but also for the reproducibility of results in nonlinear machine learning, which non-uniquely determines its weight distributions. This exercise highlights the importance of a convergence check in performing data-driven analyses of fluid flows.

The superior compression of nonlinear autoencoders over POD stems from the use of nonlinear activation functions. These functions enable projecting high-dimensional fluid flow onto a curvilinear latent submanifold, thereby improving compression performance compared with linear techniques that map the data onto a flat manifold. While the hyperbolic tangent function (tanh) has been used for the baseline analysis, we examine the dependence of reconstruction performance on the activation functions, as shown in figure 24. In this assessment, the same activation function is uniformly applied across all hidden layers of a CNN-MLP autoencoder with $n_{\xi} = 3$, while a standard linear activation is retained for the final output layer.

The sigmoid function fails to capture the unsteady wake behaviour, resulting in a high error of $\varepsilon \approx 0.8$ across the five-fold cross-validation. In contrast, the models with other nonlinear functions yield superior and stable compression performance, reporting the L_2 error below 0.2. Correspondingly, coherent vortical structures and fine-scale information are well resolved. Although the current framework does not explicitly constrain the geometric shape of the latent submanifold, this shows that the choice of nonlinear functions determines whether the network successfully embeds multidimensional vortical flows into a latent space. The inability of the sigmoid function in this case study over the current cross-validation highlights that careful checks of both error and flow field visualisation are needed when designing network architectures for fluid flows.

Note that this activation dependence is not universal but is affected by the specific fluid flow of interest, as previously examined in Fukami *et al.* (2021*b*). The varying

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

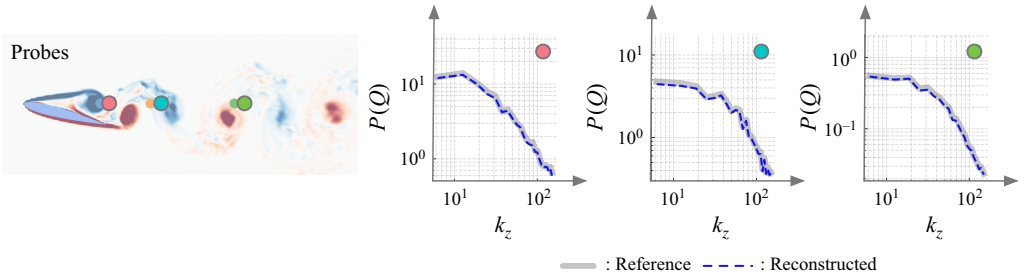


Figure 25. Spanwise-averaged spectrum of the Q criterion measured from three probes in a wake.

spatial complexity and spectral energy distributions, often characterised by the cumulative energy of singular value decomposition modes, can determine how these nonlinearities perform, which is investigated in Fukami *et al.* (2021*b*) with several flow examples including a laminar cylinder wake, sea surface temperature fields and a turbulent channel flow. A careful choice of nonlinear activation is needed not just as a machine-learning hyperparameter, but as a mapping tool considering the energy distribution of the flow of interest.

While low-dimensional representations and reconstructed flow fields are often assessed for fluid flow compression, evaluating high-order statistics is also important to see whether the trained nonlinear model respects the physical mechanisms rather than simply minimising an averaged pixel-wise loss. From this viewpoint, we examine the spanwise-averaged energy spectrum of the Q criterion, $P(Q)$, evaluated as a function of the spanwise wavenumber k_z, as shown in figure 25. These spectra are measured from three probe locations distributed throughout the wake.

The spectra obtained from the decoded flow fields are in agreement with the reference across a range of wavenumbers. The decoder accurately recovers the high-wavenumber portion of the spectrum. This suggests that the compressed latent representation is not restricted to capturing only the dominant, large-scale spatial modes. Although not examined here, one can consider investigating the temporal spectrum or correlations to observe whether the reconstructions are continuous and physical over time. Through such spectrum analysis, it is possible to examine how large spatiotemporal length scales can be covered with a designed machine-learning model.

The present results with autoencoder compression suggest that an appropriate set of low-order variables may effectively capture the underlying characteristics of the current turbulent aerofoil wake. We then posit that sparse sensors could act as such low-order variables. Here, we consider performing sparse sensor reconstruction by using the latent variables obtained from the CNN-MLP autoencoder already trained for the analysis above.

Since the pretrained decoder $\mathcal{F}_d$ recovers the flow field from a latent vector, we aim to estimate the latent vector $\boldsymbol{\xi}(t)$ directly from sparse sensor measurements $\boldsymbol{s}(t)$. To this end, we introduce an independent machine-learning model, the latent vector estimator $\mathcal{F}_s$ (Fukami *et al.* 2025*a*). The global field $\boldsymbol{q}(t)$ is reconstructed by feeding the estimated latent vector $\hat{\boldsymbol{\xi}} = \mathcal{F}_s(\boldsymbol{s}(t))$ into the pretrained decoder $\mathcal{F}_d$, as illustrated in figure 26(*a*). This procedure is expressed as

$$\boldsymbol{q}(t) \approx \hat{\boldsymbol{q}}(t) = \mathcal{F}_d(\hat{\boldsymbol{\xi}}(t)) = \mathcal{F}_d(\mathcal{F}_s(\boldsymbol{s}(t))), \tag{3.3}$$

where the weights $\boldsymbol{w}_s$ of the latent vector estimator are optimised via

$$\boldsymbol{w}_s^* = \underset{\boldsymbol{w}_s}{\operatorname{argmin}} ||\boldsymbol{\xi} - \mathcal{F}_s(\boldsymbol{s}; \boldsymbol{w}_s)||_2. \tag{3.4}$$

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

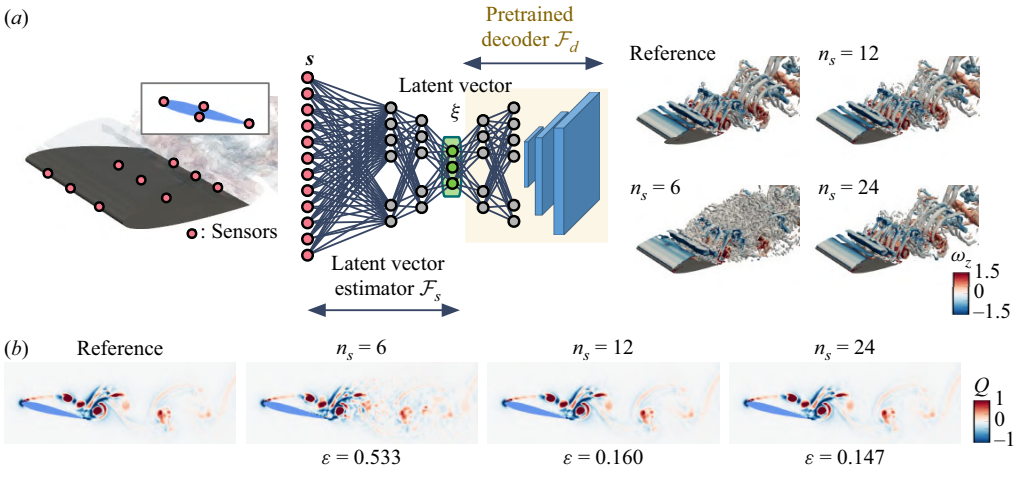


Figure 26. Sparse sensor reconstruction of a turbulent wake around a NACA0012 aerofoil via latent space. (*a*) A unification of an MLP-based latent vector estimator $\mathcal{F}_s$ and the pretrained CNN-MLP decoder $\mathcal{F}_d$. Representative reconstructed fields with $n_s = 6$, 12 and 24 are shown. (*b*) Spanwise-averaged fields corresponding to the realisations shown in (*a*). The values underneath each contour report an L_2-reconstruction error norm.

In this way, data-driven compression can be combined with sparse sensor reconstruction of unsteady flows through the latent representation, as covered in the present encoder–decoder perspective in figure 1. By mapping sparse sensors to a low-order latent vector, this approach avoids a naive learning process to relate sensor inputs directly to global field outputs (Fukami *et al.* 2023; Eldredge & Mousavi 2026). While field reconstruction is typically computationally expensive due to the dimensionality gap between input and output, this method saves costs by leveraging a pretrained decoder.

Let us perform sparse sensor reconstruction of a turbulent wake around a NACA0012 aerofoil via latent space, as presented in figure 26. Here, pressure sensors are placed on a wing at $z = 0.25$, 0.5 and 0.75. At each spanwise position, sensor readings are positioned in an equispaced manner across the aerofoil surface. With this setting, we consider three different numbers of sensors, $n_s = 6$, 12 and 24. For the estimator $\mathcal{F}_s$, we employ a MLP (Rumelhart *et al.* 1986).

The reconstruction with $n_s = 6$ shows non-physical structures in the wake region. Vortex cores and blade-type structures in the flow field are reconstructed with $n_s \geqslant 12$, reducing the error level to $\varepsilon \approx 0.150$. The current successful reconstruction suggests that the compressed representation $\boldsymbol{\xi}$ of vortical structures Q can be related to even local pressure information. It could further be argued that nonlinear machine learning bridges between these two variables that are often discussed with the Poisson equation $Q = (1/2)\nabla^2(p/\rho)$ (Davidson 2015), where ρ is the density.

We then examine the influence of the training data set size, $n_{snapshot}$, on the performance of the latent estimator $\mathcal{F}_s$ when fixing the number of sensors to be 12. To discuss how the physics-inspired cost function could help the estimation problem in fluid mechanics, let us incorporate the spatial gradients of the flow field into the cost function,

$$\mathcal{E} = ||\boldsymbol{q} - \hat{\boldsymbol{q}}||_2 + \beta||\nabla\boldsymbol{q} - \nabla\hat{\boldsymbol{q}}||_2, \tag{3.5}$$

where $\boldsymbol{q}$ and $\hat{\boldsymbol{q}} = \mathcal{F}_d(\mathcal{F}_s(\boldsymbol{s}))$ denote the reference and reconstructed fields, respectively, and β balances a regular reconstruction error and the spatial derivative penalty. This cost function encourages the model to respect local shear and vortical gradients, thereby

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

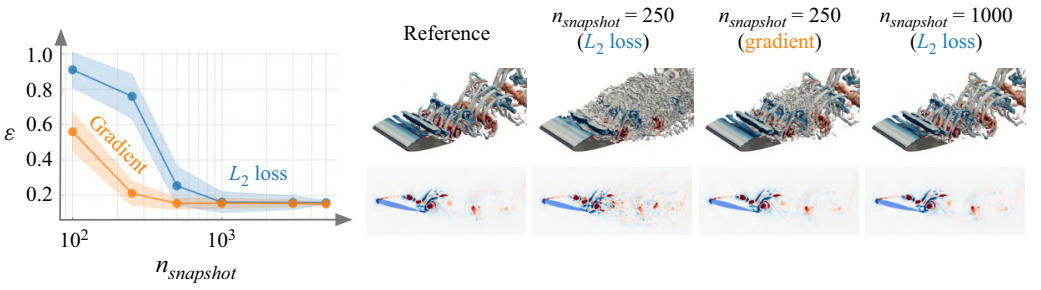


Figure 27. Dependence of the sensor reconstruction performance on the number of training snapshots $n_{snapshot}$ with $n_s = 12$. A regular L_2 cost function and gradient-based loss are considered.

avoiding the geometrically diffuse reconstruction typical of non-converged machine-learning models. Hence, this cost function design can be regarded as the incorporation of prior knowledge of flow physics, as discussed in § 2.4.

The dependence of the reconstruction performance on the number of training snapshots with both loss formulations is presented in figure 27. When trained with a limited data set of $n_{snapshot} = 250$ using the standard L_2 loss, the model fails to capture wake features, exhibiting a significantly high error. By introducing the gradient-based cost function, the model provides accurate reconstruction even with the same data limit of $n_{snapshot} = 250$, reducing the error to levels comparable to an L_2 error-based model trained with four times the amount of data. By better constraining the solution space with the assistance of a gradient-based loss, nonlinear machine learning is able to learn the relationship between sparse sensors and vortical structures even with a limited amount of data. The minimum requirement of training data for reconstruction problems can also be examined based on the covered spatial length scales in the given data sets (Fukami & Taira 2024; Wu *et al.* 2025). While the present case study primarily considers data-driven compression and reconstruction of unsteady aerodynamic flows, incorporating prior knowledge of flow physics is essential for a range of problem settings with various flow examples.

4. Prospects

This perspective paper provides a guideline for studying fluid mechanics with nonlinear machine learning. Machine learning has offered extensive opportunities to analyse fluid flows from a range of aspects, including flow reconstruction, reduced-order modelling, flow control and stochastic generation. As we have continuously mentioned throughout the paper, it is essential to embed prior knowledge of physics into the design of machine-learning studies in fluid mechanics. In other words, it is important to critically evaluate how the designed machine-learning models can be made robust, reliable and fundamentally insightful from the physical perspective. In this section, let us discuss some additional viewpoints that may be worth being mindful of in tackling fluid flow problems using machine learning, along with the present guideline.

In performing modern machine-learning analysis, the amount of data often becomes substantial. However, such machine-learning techniques may not be data hungry for fluid mechanics problems because a single flow snapshot usually includes orders of magnitude more information than a typical image frame in general machine-learning settings. This suggests that data collection can also be smarter by incorporating prior knowledge of flow physics. In fact, it is possible to accurately perform super-resolution of turbulent flows with training data composed of flow tiles sampled from merely a single snapshot (Fukami &

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Taira 2024). They have reported that the required amount of flow tiles can be further reduced by sampling the data based on local flow characteristics such as rotation and shear tensors. Embedding prior physical insights prevents practitioners from being wasteful with valuable turbulent flow data.

We need to be careful about how generalisable the developed model is. As fluid flow phenomena are determined by diverse parameters such as non-dimensional numbers and spatiotemporal characteristics, a truly generalisable model should exhibit robustness against variations in data fidelity, spatial resolution and sensor noise. While the validity of a machine-learning model is often assessed through the lens of interpolation and extrapolation, as discussed in § 2.2.4, one can extend the utility of trained models to unseen conditions by considering transfer learning. For example, fine tuning a model trained on low-Reynolds-number flows to higher Reynolds numbers drastically reduces training effort (Guastoni *et al.* 2021). In a similar vein, developing foundation models that freeze early network layers to capture broadly applicable representations while fine tuning later layers for specific downstream tasks may be promising to achieve high generalisability across a range of problem settings in fluid mechanics (Taira *et al.* 2025).

Ensuring the interpretability of nonlinear machine-learning models is also important. However, the definition of interpretability is not absolute; rather, it may vary depending on the specific problem setting. In the context of flow reconstruction, an interpretable model is one that enables the identification of which sensor locations are most influential, or explains precisely which flow structures are learned effectively and why. For data compression and reduced-order modelling, interpretability may instead hinge on the extent to which the geometric shape of the machine-learned latent space can be explained, or how intuitively those abstract variables capture the governing physics. This indicates that the interpretability of machine learning in fluid mechanics is not a static hurdle. Rather, it is an actively evolving frontier since the community would grow in understanding how nonlinear machine learning further supports studying fluid flows.

In practical scenarios where true solutions are unknown, understanding what a model does not know is just as critical as what it does. In response, performing uncertainty quantification is useful to identify how the reliability of machine-learning-based prediction can be improved. One can consider employing Bayesian networks (Morimoto *et al.* 2022) or probabilistic neural networks (Maulik *et al.* 2020) to assess the uncertainty associated with fluid flow data, systems or machine-learning models. In addition, we should be mindful that machine-learning models are often designed to capture statistical correlations rather than true physical causality. From the aspect of understanding what the model learns from fluid flow data, transitioning from correlative to causal modelling, which could be achieved with information-theoretic machine learning (Fukami & Araki 2026; Koshikawa *et al.* 2026) and explainable artificial-intelligence frameworks (Cremades *et al.* 2024), can be considered.

It is finally worth mentioning that data-driven fluid mechanics emphasises the importance of traditional subfields of theoretical, experimental and computational fluid mechanics. As discussed above, machine-learning-based studies in fluid mechanics are supported by high-fidelity data and physical knowledge gained through the three subfields. Hence, data-driven fluid mechanics should be viewed not as a replacement for traditional analysis, but as a powerful, mutually assisting mechanism. With this in mind, machine learning can serve as a microscope of flow physics rather than as a mere nonlinear regressor, supporting a range of problem settings in fluid mechanics.

Acknowledgements. We thank Kunihiko Taira for fruitful discussions on machine-learning-based analysis for fluid flows.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Funding. Kai Fukami acknowledges support from the JSPS KAKENHI Grant No. JP25K23418 and No. JP26K01129, the JST PRESTO Grant No. JPMJPR25KA, and the MEXT Coordination Funds for Promoting Aerospace Utilization Grant No. JPJ000959. Koji Fukagata acknowledges support from the JSPS KAKENHI Grant No. JP18H03758, No. JP21H05007 and No. JP26K21736.

Declaration of interests. The authors report no conflict of interest.

Appendix A. A dictionary table of data-driven fluid dynamics

Table 4 lists major technical terms used in data-driven fluid dynamics and their definitions.

Term	Definition
Activation function	A math formula (e.g. ReLU, tanh) in a neural network that lets it learn nonlinear patterns.
Adam optimiser	An algorithm that adjusts model weights during training to minimise errors efficiently.
Backpropagation	The standard method to train neural networks by working backward to correct internal weights based on errors.
Bayesian optimisation	An automated method to find the best model parameters without random guessing.
Convolutional neural network (CNN)	A network using filters, ideal for preserving the spatial layout of grid-based data.
Cost function (loss function)	A mathematical score measuring how far off the outputs are from the true data.
Cross-validation	Dividing data into multiple training and testing sets to ensure the model does not just memorise one data set.
Decoder	The part of a model that reconstructs full, high-dimensional data from compressed representations.
Diffusion model	A generative algorithm that creates realistic data by learning to reverse a noise-adding process.
Early stopping	Terminating training when performance on a validation set stops improving to prevent overfitting.
Encoder	The part of a model that compresses high-dimensional data into low-order representations.
Epoch	One complete pass of the entire training data set through the algorithm.
Fourier neural operator (FNO)	A model that learns in the frequency domain, allowing estimations on new grid resolutions without retraining.
Generative adversarial network (GAN)	Two competing networks (generator and discriminator) that train each other to produce realistic synthetic data.
Grad-CAM/SHAP	Tools that can highlight which parts of the input data the network focused on to produce an output.
Graph neural network (GNN)	A network for unstructured data that passes information between connected nodes.

Table 4. For caption see next page.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

Term	Definition
Hyperparameters	Structural settings chosen before training (e.g. layer number), unlike weights that are learned.
Latent space	A compressed space storing the important features of the data.
Long short-term memory (LSTM)	A network with gates to remember time history, used for predicting unsteady phenomena over time.
Manifold	A lower-dimensional geometric surface where complex, high-dimensional data naturally organises itself.
Multi-layer perceptron (MLP)	A standard neural network with fully connected layers that processes data point-by-point.
Overfitting	When a model memorises training data perfectly but fails completely on new, unseen data.
Physics-informed neural networks (PINNs)	Models penalised mathematically if their outputs violate physical laws such as the Navier–Stokes equations.
PointNet	A network designed to learn directly from scattered, unstructured data points without using a grid.
Recurrent neural network (RNN)	A network for time-series data that updates a hidden state at each step to carry past information forward.
Reinforcement learning	A set-up where an agent learns by trial and error in an environment to maximise a numerical reward.
Skip connections	Architectural shortcuts bypassing certain layers to retain fine details and smooth the training process.
Super-resolution	Reconstructing a high-resolution field from sparse, low-resolution sensor measurements.
Supervised learning	Training a model using data that already includes the correct, paired answers.
Surrogate model	A fast, lightweight model trained to mimic a computationally expensive simulation.
Transfer learning	Tweaking a model already trained on one task for a new task, saving computational time and providing stable training.
Transformer	A network using an attention mechanism to look at the entire data field at once, finding long-range correlations.
Underfitting	When a model is too simple or undertrained to capture the underlying patterns in the data.
Unsupervised learning	Training a model on data without answers, guiding it to find hidden structures on its own.
Vanishing gradients	A failure where update signals shrink to zero as they travel backward, stopping the network from learning.
Weights and biases	The internal numbers inside a neural network adjusted during training to form the final model.

Table 4. (cntd). A dictionary table of data-driven fluid dynamics.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

References

Ahmed, S.E., Pawar, S., San, O., Rasheed, A. & Tabib, M. 2021 A nudged hybrid analysis and modeling approach for realtime wake-vortex transport and decay prediction. *Comput. Fluids* **221**, 104895.

Albrecht, S.V., Christianos, F. & Schäfer, L. 2024 *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press.

Anantharaman, V., Feldkamp, J., Fukami, K. & Taira, K. 2023 Image and video compression of fluid flow data. *Theor. Comput. Fluid Dyn.* **37** (1), 61–82.

Bae, H.J. & Koumoutsakos, P. 2022 Scientific multi-agent reinforcement learning for wall-models of turbulent flows. *Nat. Commun.* **13** (1), 1443.

Baldi, P. & Hornik, K. 1989 Neural networks and principal component analysis: learning from examples without local minima. *Neural Networks* **2** (1), 53–58.

Bauerheim, M. 2026 Routes towards an effective AI in CFD: an epistemological and technical perspective. *J. Fluid Mech.* **1031**, P1.

Baydin, A.G., Pearlmutter, B.A., Radul, A.A. & Siskind, J.M. 2018 Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.* **18** (153), 1–43.

Bergstra, J. & Bengio, Y. 2012 Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **13** (2), 281–305.

Bergstra, J., Komer, B., Eliasmith, C., Yamins, D. & Cox, D.D. 2015 Hyperopt: a Python library for model selection and hyperparameter optimization. *Comput. Sci. Disc.* **8** (1), 014008.

Bishop, C.M. & Nasrabadi, N.M. 2006 *Pattern Recognition and Machine Learning*. Springer.

Bode, M., Gauding, M., Lian, Z., Denker, D., Davidovic, M., Kleinheinz, K., Jitsev, J. & Pitsch, H. 2021 Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows. *Proc. Combust. Inst.* **38** (2), 2617–2625.

Brenner, M.P., Eldredge, J.D. & Freund, J.B. 2019 Perspective on machine learning for advancing fluid mechanics. *Phys. Rev. Fluids* **4** (10), 100501.

Brunton, S.L. 2021 Applying machine learning to study fluid mechanics. *Acta Mechanica Sin.* **37** (12), 1718–1726.

Brunton, S.L., Noack, B.R. & Koumoutsakos, P. 2020 Machine learning for fluid mechanics. *Annu. Rev. Fluid Mech.* **52** (1), 477–508.

Brunton, S.L., Proctor, J.L. & Kutz, J.N. 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113** (15), 3932–3937.

Bui-Thanh, T., Damodaran, M. & Willcox, K. 2004 Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA J.* **42** (8), 1505–1516.

Buzzicotti, M., Bonaccorso, F., Di Leoni, P.C. & Biferale, L. 2021 Reconstruction of turbulent data with deep generative models for semantic inpainting from TURB-Rot database. *Phys. Rev. Fluids* **6** (5), 050503.

Cai, S., Mao, Z., Wang, Z., Yin, M. & Karniadakis, G.E. 2022 Physics-informed neural networks (PINNs) for fluid mechanics: a review. *Acta Mechanica Sin.* **73**, 1–12.

Cai, S., Zhou, S., Xu, C. & Gao, Q. 2019 Dense motion estimation of particle images via a convolutional neural network. *Exp. Fluids* **60**, 1–16.

Callaham, J.L., Maeda, K. & Brunton, S.L. 2019 Robust flow reconstruction from limited measurements via sparse representation. *Phys. Rev. Fluids* **4** (10), 103907.

Carter, D.W., De Voogt, F., Soares, R. & Ganapathisubramani, B. 2021 Data-driven sparse reconstruction of flow over a stalled aerofoil using experimental data. *Data-Centric Engng* **2**, e5.

Chen, D., Kaiser, F., Hu, J.-C., Rival, D.E., Fukami, K. & Taira, K. 2024 Sparse pressure-based machine learning approach for aerodynamic loads estimation during gust encounters. *AIAA J.* **62** (1), 275–290.

Choi, H., Moin, P. & Kim, J. 1994 Active turbulence control for drag reduction in wall-bounded flows. *J. Fluid Mech.* **262**, 75–110.

Cremades, A. *et al.*et al. 2024 Identifying regions of importance in wall-bounded turbulence through explainable deep learning. *Nat. Commun.* **15** (1), 3864.

Cuéllar, A., Güemes, A., Ianiro, A., Flores, Ó., Vinuesa, R. & Discetti, S. 2024 Three-dimensional generative adversarial networks for turbulent flow estimation from wall measurements. *J. Fluid Mech.* **991**, A1.

Davidson, P.A. 2015 *Turbulence: An Introduction for Scientists and Engineers*. Oxford University Press.

Du, P., Parikh, M.H., Fan, X., Liu, X.-Y. & Wang, J.-X. 2024 Conditional neural field latent diffusion model for generating spatiotemporal turbulence. *Nat. Commun.* **15** (1), 10416.

Dubois, P., Gomez, T., Planckaert, L. & Perret, L. 2022 Machine learning for fluid flow reconstruction from limited measurements. *J. Comput. Phys.* **448**, 110733.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

DURAISAMY, K. 2021 Perspectives on machine learning-augmented Reynolds-averaged and large eddy simulation models of turbulence. *Phys. Rev. Fluids* **6** (5), 050504.
DURAISAMY, K., IACCARINO, G. & XIAO, H. 2019 Turbulence modeling in the age of data. *Annu. Rev. Fluid Mech.* **51** (1), 357–377.
EDELSBRUNNER, H. & MOROZOV, D. 2014 Persistent homology: theory and practice. In *European Congress of Mathematics*. EMS Press.
ELDREDGE, J.D. & MOUSAVI, H. 2026 Practical guide to estimation and uncertainty quantification of aerodynamic flows. *AIAA J.* **64** (5), 2403–2423.
ERICHSON, N.B., MATHELIN, L., YAO, Z., BRUNTON, S.L., MAHONEY, M.W. & KUTZ, J.N. 2020 Shallow neural networks for fluid flow reconstruction with limited sensors. *Proc. R. Soc. Lond. A Math. Phys. Engng Sci.* **476**, 2238.
FAN, H., CHENG, S., DE NAZELLE, A.J. & ARCUCCI, R. 2025 ViTAE-SL: a vision transformer-based autoencoder and spatial interpolation learner for field reconstruction. *Comput. Phys. Commun.* **308**, 109464.
FIX, E. & HODGES, J.L. 1989 Discriminatory analysis: nonparametric discrimination: consistency properties. *Intl Stat. Rev.* **57**, 238–247.
FONT, B., ALCÁNTARA-ÁVILA, F., RABAULT, J., VINUESA, R. & LEHMKUHL, O. 2025 Deep reinforcement learning for active flow control in a turbulent separation bubble. *Nat. Commun.* **16** (1), 1422.
FUKAGATA, K. 2023 Reduced order modeling of fluid flows using convolutional neural networks. *J. Fluid Sci. Technol.* **18** (1), JFST0002.
FUKAGATA, K. & FUKAMI, K. 2025 Compressing fluid flows with nonlinear machine learning: mode decomposition, latent modeling, and flow control. *Fluid Dyn. Res.* **57** (4), 041401.
FUKAMI, K., AN, B., NOHMI, M., OBUCHI, M. & TAIRA, K. 2022 Machine-learning-based reconstruction of turbulent vortices from sparse pressure sensors in a pump sump. *J. Fluids Engng* **144** (12), 121501.
FUKAMI, K. & ARAKI, R. 2026 Information-theoretic machine learning for time-varying mode decomposition of separated aerodynamic flows. *AIAA J.* **64** (2), 605–613.
FUKAMI, K., FUKAGATA, K. & TAIRA, K. 2019*a* Super-resolution reconstruction of turbulent flows with machine learning. *J. Fluid Mech.* **870**, 106–120.
FUKAMI, K., FUKAGATA, K. & TAIRA, K. 2020*a* Assessment of supervised machine learning methods for fluid flows. *Theor. Comput. Fluid Dyn.* **34**, 497–519.
FUKAMI, K., FUKAGATA, K. & TAIRA, K. 2021*a* Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows. *J. Fluid Mech.* **909**, A9.
FUKAMI, K., FUKAGATA, K. & TAIRA, K. 2023 Super-resolution analysis via machine learning: a survey for fluid flows. *Theor. Comput. Fluid Dyn.* **37**, 421–444.
FUKAMI, K., GOTO, S. & TAIRA, K. 2024*a* Data-driven nonlinear turbulent flow scaling with Buckingham Pi variables. *J. Fluid Mech.* **984**, R4.
FUKAMI, K., HASEGAWA, K., NAKAMURA, T., MORIMOTO, M. & FUKAGATA, K. 2021*b* Model order reduction with neural networks: application to laminar and turbulent flows. *SN Comput. Sci.* **2** (6), 467.
FUKAMI, K., IWATANI, Y., MAEJIMA, S., ASADA, H. & KAWAI, S. 2025*a* Compact representation of transonic airfoil buffet flows with observable-augmented machine learning. *J. Fluid Mech.* **1021**, A39.
FUKAMI, K., MAULIK, R., RAMACHANDRA, N., FUKAGATA, K. & TAIRA, K. 2021*c* Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning. *Nat. Mach. Intell.* **3** (11), 945–951.
FUKAMI, K., MURATA, T., ZHANG, K. & FUKAGATA, K. 2021*d* Sparse identification of nonlinear dynamics with low-dimensionalized flow representations. *J. Fluid Mech.* **926**, A10.
FUKAMI, K., NABAE, Y., KAWAI, K. & FUKAGATA, K. 2019*b* Synthetic turbulent inflow generator using machine learning. *Phys. Rev. Fluids* **4** (6), 064603.
FUKAMI, K., NAKAMURA, T. & FUKAGATA, K. 2020*b* Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data. *Phys. Fluids* **32** (9), 095110.
FUKAMI, K., NAKAO, H. & TAIRA, K. 2024*b* Data-driven transient lift attenuation for extreme vortex gust–airfoil interactions. *J. Fluid Mech.* **992**, A17.
FUKAMI, K. & SHOJI, E. 2026 Data-driven modeling and decomposition for nanoscale liquid-film dynamics: application to superspreading nanofluid droplets. *Eur. J. Mech. B Fluids* **121**, 204596.
FUKAMI, K., SMITH, L. & TAIRA, K. 2025*b* Extreme vortex-gust airfoil interactions at Reynolds number 5000. *Phys. Rev. Fluids* **10** (8), 084703.
FUKAMI, K. & TAIRA, K. 2023 Grasping extreme aerodynamics on a low-dimensional manifold. *Nat. Commun.* **14**, 6480.
FUKAMI, K. & TAIRA, K. 2024 Single-snapshot machine learning for super-resolution of turbulence. *J. Fluid Mech.* **1001**, A32.
FUKAMI, K. & TAIRA, K. 2025 Observable-augmented manifold learning for multi-source turbulent flow data. *J. Fluid Mech.* **1010**, R4.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

GAO, H., KALTENBACH, S. & KOUMOUTSAKOS, P. 2024 Generative learning for forecasting the dynamics of high-dimensional complex systems. *Nat. Commun.* **15** (1), 8904.
GAO, H., SUN, L. & WANG, J.-X. 2021 Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels. *Phys. Fluids* **33** (7), 073603.
GARCIA, X., MIRÓ, A., SUÁREZ, P., ALCÁNTARA-ÁVILA, F., RABAULT, J., FONT, B., LEHMKUHL, O. & VINUESA, R. 2025 Deep-reinforcement-learning-based separation control in a two-dimensional airfoil. *Intl J. Heat Fluid Flow* **116**, 109913.
GARNIER, P., VIQUERAT, J., RABAULT, J., LARCHER, A., KUHNLE, A. & HACHEM, E. 2021 A review on deep reinforcement learning for fluid mechanics. *Comput. Fluids* **225**, 104973.
GODAVARTHI, V., KAWAMURA, Y. & TAIRA, K. 2023 Optimal waveform for fast synchronization of airfoil wakes. *J. Fluid Mech.* **976**, R1.
GOODFELLOW, I., BENGIO, Y. & COURVILLE, A. 2016 *Deep Learning*. MIT Press.
GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A. & BENGIO, Y. 2020 Generative adversarial networks. *Commun. ACM* **63** (11), 139–144.
GOPALAKRISHNAN MEENA, M., TAIRA, K. & ASAI, K. 2017 Airfoil-wake modification with gurney flap at low Reynolds number. *AIAA J.* **56** (4), 1348–1359.
GUASTONI, L., GÜEMES, A., IANIRO, A., DISCETTI, S., SCHLATTER, P., AZIZPOUR, H. & VINUESA, R. 2021 Convolutional-network models to predict wall-bounded turbulence from wall quantities. *J. Fluid Mech.* **928**, A27.
GUASTONI, L., RABAULT, J., SCHLATTER, P., AZIZPOUR, H. & VINUESA, R. 2023 Deep reinforcement learning for turbulent drag reduction in channel flows. *Eur. Phys. J. E.* **46** (4), 27.
GÜEMES, A., SANMIGUEL VILA, C. & DISCETTI, S. 2022 Super-resolution generative adversarial networks of randomly-seeded fields. *Nat. Mach. Intell.* **4** (12), 1165–1173.
HAM, F. & IACCARINO, G. 2004 Energy conservation in collocated discretization schemes on unstructured meshes. In *Annual Research Briefs*, pp. 3–14. Center for Turbulence Research.
HAM, F., MATTSSON, K. & IACCARINO, G. 2006 Accurate and stable finite volume operators for unstructured flow solvers. In *Annual Research Briefs*, pp. 243–261. Center for Turbulence Research.
HANSEN, P.C. & O'LEARY, D.P. 1993 The use of the L-curve in the regularization of discrete ill-posed problems. *SIAM J. Sci. Comput.* **14** (6), 1487–1503.
HASEGAWA, K., FUKAMI, K., MURATA, T. & FUKAGATA, K. 2020 Machine-learning-based reduced-order modeling for unsteady flows around bluff bodies of various shapes. *Theor. Comput. Fluid Dyn.* **34** (4), 367–383.
HINTON, G.E. & SALAKHUTDINOV, R.R. 2006 Reducing the dimensionality of data with neural networks. *Science* **313** (5786), 504–507.
HO, J., JAIN, A. & ABBEEL, P. 2020 Denoising diffusion probabilistic models. *NeurIPS* **33**, 6840–6851.
HOCHREITER, S. & SCHMIDHUBER, J. 1997 Long short-term memory. *Neural Comput.* **9** (8), 1735–1780.
HOLMES, P., LUMLEY, J. & BERKOOZ, G. 1998 *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press.
HOPF, E. 1948 A mathematical example displaying features of turbulence. *Commun. Pure Appl. Maths* **1** (4), 303–322.
HUANG, J., LIU, H., WANG, Q. & CAI, W. 2020 Limited-projection volumetric tomography for time-resolved turbulent combustion diagnostics via deep learning. *Aerosp. Sci. Technol.* **106**, 106123.
HUNT, J.C.R., WRAY, A.A. & MOIN, P. 1988 Eddies, streams, and convergence zones in turbulent flows. In *Center for Turbulence Research (CTR) Proceedings of Summer Program*, pp. 193–208.
ISHIZE, T., OMICHI, H. & FUKAGATA, K. 2024 Flow control by a hybrid use of machine learning and control theory. *Intl J. Numer. Meth. Heat Fluid Flow* **34** (8), 3253–3277.
JAGODINSKI, E., ZHU, X. & VERMA, S. 2023 Inverse identification of dynamically important regions in turbulent flows using three-dimensional convolutional neural networks. *Phys. Rev. Fluids* **8** (9), 094605.
JAROSLAWSKI, T., PATIL, A. & MCKEON, B.J. 2025 Predicting turbulence structure in street-canyon flows using deep generative modeling. *Intl J. Heat Fluid Flow* **115**, 109849.
JING, L., ZBONTAR, J. & LECUN, Y. 2020 Implicit rank-minimizing autoencoder. *NeurIPS* **33**, 14736–14746.
KAJISHIMA, T. & TAIRA, K. 2016 *Computational Fluid Dynamics: Incompressible Turbulent Flows*. Springer.
KASHEFI, A., REMPE, D. & GUIBAS, L.J. 2021 A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries. *Phys. Fluids* **33** (2), 027104.
KIM, H., KIM, J., WON, S. & LEE, C. 2021 Unsupervised deep learning for super-resolution reconstruction of turbulence. *J. Fluid Mech.* **910**, A29.
KIM, J. & LEE, C. 2020 Deep unsupervised learning of turbulence for inflow generation at various Reynolds numbers. *J. Comput. Phys.* **406**, 109216.
KINGMA, D.P. & BA, J. 2014 Adam: a method for stochastic optimization. arXiv preprint, arXiv: 1412.6980.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

KIYANI, E., SHUKLA, K., URBÁN, J.F., DARBON, J. & KARNIADAKIS, G.E. 2025 Optimizing the optimizer for physics-informed neural networks and Kolmogorov–Arnold networks. *Comput. Meth. Appl. Mech. Engng* **446**, 118308.

KOSHIKAWA, R., ARAKI, R., LIU, Q. & FUKAMI, K. 2026 Convolutional causal learning for aerodynamic flows. *J. Fluid Mech.* **1037**, A6.

KRISHNAPRIYAN, A., GHOLAMI, A., ZHE, S., KIRBY, R. & MAHONEY, M.W. 2021 Characterizing possible failure modes in physics-informed neural networks. *NeurIPS* **34**, 26548–26560.

LAGARIS, I.E., LIKAS, A. & FOTIADIS, D.I. 1998 Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans. Neural Networks* **9** (5), 987–1000.

LECUN, Y., BOTTOU, L., BENGIO, Y. & HAFFNER, P. 1998 Gradient-based learning applied to document recognition. *Proc. IEEE* **86** (11), 2278–2324.

LEE, C., KIM, J., BABCOCK, D. & GOODMAN, R. 1997 Application of neural networks to turbulence control for drag reduction. *Phys. Fluids* **9** (6), 1740–1747.

LEE, S., YANG, J., FOROOGHI, P., STROH, A. & BAGHERI, S. 2022 Predicting drag on rough surfaces by transfer learning of empirical correlations. *J. Fluid Mech.* **933**, A18.

LEE, S. & YOU, D. 2019 Data-driven prediction of unsteady flow over a circular cylinder using deep learning. *J. Fluid Mech.* **879**, 217–254.

LI, J., ZHANG, M., MARTINS, J.R.R.A. & SHU, C. 2020 Efficient aerodynamic shape optimization with deep-learning-based geometric filtering. *AIAA J.* **58** (10), 4243–4259.

LI, T., BUZZICOTTI, M., BIFERALE, L., BONACCORSO, F., CHEN, S. & WAN, M. 2023 Multi-scale reconstruction of turbulent rotating flows with proper orthogonal decomposition and generative adversarial networks. *J. Fluid Mech.* **971**, A3.

LI, Z. & FARIMANI, A.B. 2022 Graph neural network-accelerated Lagrangian fluid simulation. *Comput. Graph.* **103**, 201–211.

LI, Z., KOVACHKI, N.B. & AZIZZADENESHELI, K. 2021 Fourier neural operator for parametric partial differential equations. In *ICLR*, vol. 2, pp. 8.

LI, Z., PENG, W., YUAN, Z. & WANG, J. 2022 Fourier neural operator approach to large eddy simulation of three-dimensional turbulence. *Theor. Appl. Mech. Lett.* **12** (6), 100389.

LING, J., KURZAWSKI, A. & TEMPLETON, J. 2016 Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166.

LINOT, A.J. & GRAHAM, M.D. 2020 Deep learning to discover and predict dynamics on an inertial manifold. *Phys. Rev. E* **101** (6), 062209.

LINOT, A.J. & GRAHAM, M.D. 2023 Dynamics of a data-driven low-dimensional model of turbulent minimal Couette flow. *J. Fluid Mech.* **973**, A42.

LINOT, A.J., HANG, H., KANSO, E. & TAIRA, K. 2025 Hierarchical equivariant graph neural networks for forecasting collective motion in vortex clusters and microswimmers. *Commun. Phys.* **8**, 515.

LINOT, A.J., ZENG, K. & GRAHAM, M.D. 2023 Turbulence control in plane Couette flow using low-dimensional neural ODE-based models and deep reinforcement learning. *Intl J. Heat Fluid Flow* **101**, 109139.

LIU, Q. & THUEREY, N. 2024 Uncertainty-aware surrogate models for airfoil flow simulations with denoising diffusion probabilistic models. *AIAA J.* **62** (8), 2912–2933.

LIU, X.-Y., PARIKH, M.H., FAN, X., DU, P., WANG, Q., CHEN, Y.-F. & WANG, J.-X. 2025*a* CoNFiLD-inlet: synthetic turbulence inflow using generative latent diffusion models with neural fields. *Phys. Rev. Fluids* **10** (5), 054901.

LIU, Z., BECKERS, D. & ELDREDGE, J.D. 2025*b* Model-based reinforcement learning for control of strongly disturbed unsteady aerodynamic flows. *AIAA J.* **63** (10), 4105–4125.

LOZANO-DURÁN, A. & BAE, H.J. 2023 Machine learning building-block-flow wall model for large-eddy simulation. *J. Fluid Mech.* **963**, A35.

LUI, H.F.S. & WOLF, W.R. 2019 Construction of reduced-order models for fluid flows using deep feedforward neural networks. *J. Fluid Mech.* **872**, 963–994.

LUMLEY, J.L. 1967 The structure of inhomogeneous turbulent flows. In *Atmospheric Turbulence and Radio Wave Propagation* (ed. A.M. Yaglom & V.I. Tatarski). Nauka.

LUNDBERG, S.M. & LEE, S.-I. 2017 A unified approach to interpreting model predictions. *NeurIPS* **30**, 4768–4777.

LUSCH, B., KUTZ, J.N. & BRUNTON, S.L. 2018 Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* **9** (1), 4950.

MAJEWSKI, W., WEI, R. & KUMAR, V. 2020 Developing particle image velocimetry software based on a deep neural network. *J. Flow Vis. Image Process.* **27** (4), 359–376.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

MARRA, L., MACEDA, G.Y.C., MEILÁN-VILA, A., GUERRERO, V., RASHWAN, S., NOACK, B.R., DISCETTI, S. & IANIRO, A. 2024 Actuation manifold from snapshot data. *J. Fluid Mech.* **996**, A26.
MASKER, A.I., ZHOU, K., MOLNAR, J.P. & GRAUER, S.J. 2025 Neural optical flow for planar and stereo PIV. *Exp. Fluids* **66** (6), 129.
MATSUO, M., FUKAMI, K., NAKAMURA, T., MORIMOTO, M. & FUKAGATA, K. 2024 Reconstructing three-dimensional bluff body wake from sectional flow fields with convolutional neural networks. *SN Comput. Sci.* **5** (3), 306.
MAULIK, R., FUKAMI, K., RAMACHANDRA, N., FUKAGATA, K. & TAIRA, K. 2020 Probabilistic neural networks for fluid flow surrogate modeling and data recovery. *Phys. Rev. Fluids* **5** (10), 104401.
MAULIK, R., LUSCH, B. & BALAPRAKASH, P. 2021 Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Phys. Fluids* **33** (3), 037106.
MAULIK, R. & SAN, O. 2017 A neural network approach for the blind deconvolution of turbulent flows. *J. Fluid Mech.* **831**, 151–181.
MAULIK, R., SAN, O., RASHEED, A. & VEDULA, P. 2019 Subgrid modelling for two-dimensional turbulence using neural networks. *J. Fluid Mech.* **858**, 122–144.
MILANO, M. & KOUMOUTSAKOS, P. 2002 Neural network modeling for near wall turbulent flow. *J. Comput. Phys.* **182** (1), 1–26.
MOHAN, A.T., LUBBERS, N., CHERTKOV, M. & LIVESCU, D. 2023 Embedding hard physical constraints in neural network coarse-graining of three-dimensional turbulence. *Phys. Rev. Fluids* **8** (1), 014604.
MOHAN, A.T., TRETIAK, D., CHERTKOV, M. & LIVESCU, D. 2020 Spatio-temporal deep learning models of 3-D turbulence with physics informed diagnostics. *J. Turbul.* **21** (9–10), 484–524.
MORIMOTO, M., FUKAMI, K. & FUKAGATA, K. 2021*a* Experimental velocity data estimation for imperfect particle images using machine learning. *Phys. Fluids* **33** (8), 087121.
MORIMOTO, M., FUKAMI, K., MAULIK, R., VINUESA, R. & FUKAGATA, K. 2022 Assessments of epistemic uncertainty using Gaussian stochastic weight averaging for fluid-flow regression. *Phys. D: Nonlinear Phenom.* **440**, 133454.
MORIMOTO, M., FUKAMI, K., ZHANG, K., NAIR, A.G. & FUKAGATA, K. 2021*b* Convolutional neural networks for fluid flow analysis: toward effective metamodeling and low dimensionalization. *Theor. Comput. Fluid Dyn.* **35** (5), 633–658.
MOUSAVI, H. & ELDREDGE, J.D. 2025 Low-order flow reconstruction and uncertainty quantification in disturbed aerodynamics using sparse pressure measurements. *J. Fluid Mech.* **1013**, A41.
MURATA, T., FUKAMI, K. & FUKAGATA, K. 2020 Nonlinear mode decomposition with convolutional neural networks for fluid dynamics. *J. Fluid Mech.* **882**, A13.
NAIR, N.J. & GOZA, A. 2020 Leveraging reduced-order models for state estimation using deep learning. *J. Fluid Mech.* **897**, R1.
NAKAMURA, T., FUKAMI, K. & FUKAGATA, K. 2022 Identifying key differences between linear stochastic estimation and neural networks for fluid flow regressions. *Sci. Rep.* **12** (1), 3726.
NAKAMURA, T., FUKAMI, K., HASEGAWA, K., NABAE, Y. & FUKAGATA, K. 2021 Convolutional neural network and long short-term memory based reduced order surrogate for minimal turbulent channel flow. *Phys. Fluids* **33** (2), 025116.
NORRIS, J.R. 1998 *Markov Chains*. Cambridge University Press.
OGOKE, F., MEIDANI, K., HASHEMI, A. & FARIMANI, A.B. 2021 Graph convolutional networks applied to unstructured flow field data. *Mach. Learn.: Sci. Technol.* **2** (4), 045020.
PAGE, J. 2025 Super-resolution of turbulence with dynamics in the loss. *J. Fluid Mech.* **1002**, R3.
PARIS, R., BENEDDINE, S. & DANDOIS, J. 2023 Reinforcement-learning-based actuator selection method for active flow control. *J. Fluid Mech.* **955**, A8.
PEHERSTORFER, B. 2022 Breaking the Kolmogorov barrier with nonlinear model reduction. *Not. Am. Math. Soc.* **69** (5), 725–733.
PÉREZ, J.M., SASTRE, F., LE CLAINCHE, S., VELÁZQUEZ, A. & VEGA, J.M. 2022 Three-dimensional flow field reconstruction in the wake of a confined square cylinder using planar PIV data. *Exp. Therm. Fluid Sci.* **133**, 110523.
PRECHELT, L. 1998 Automatic early stopping using cross validation: quantifying the criteria. *Neural Networks* **11** (4), 761–767.
QI, C.R., SU, H., MO, K. & GUIBAS, L.J. 2017 PointNet: deep learning on point sets for 3-D classification and segmentation. In *Proceedings of the IEEE CVPR*, pp. 652–660.
RABAULT, J., KUCHTA, M., JENSEN, A., RÉGLADE, U. & CERARDI, N. 2019 Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *J. Fluid Mech.* **865**, 281–302.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

RABAULT, J., REN, F., ZHANG, W., TANG, H. & XU, H. 2020 Deep reinforcement learning in fluid mechanics: a promising method for both active flow control and shape optimization. *J. Hydrodyn.* **32** (2), 234–246.

RAISSI, M., PERDIKARIS, P. & KARNIADAKIS, G.E. 2019 Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707.

ROLANDI, L.V., SMITH, L., AMITAY, M., THEOFILIS, V. & TAIRA, K. 2025 Biglobal resolvent analysis of separated flow over a NACA0012 airfoil. *J. Fluid Mech.* **1021**, A53.

ROWLEY, C.W. & DAWSON, S.T.M. 2017 Model reduction for flow analysis and control. *Annu. Rev. Fluid Mech.* **49**, 387–417.

RUMELHART, D.E., HINTON, G.E. & WILLIAMS, R.J. 1986 Learning representations by back-propagation errors. *Nature* **322**, 533–536.

SAKIA, R.M. 1992 The Box–Cox transformation technique: a review. *J. R. Stat. Soc. Ser. D Stat.* **41** (2), 169–178.

SANTOS, J.E., FOX, Z.R., MOHAN, A., O'MALLEY, D., VISWANATHAN, H. & LUBBERS, N. 2023 Development of the Senseiver for efficient field reconstruction from sparse observations. *Nat. Mach. Intell.* **5** (11), 1317–1325.

SCARSELLI, F., GORI, M., TSOI, A.C., HAGENBUCHNER, M. & MONFARDINI, G. 2008 The graph neural network model. *IEEE Trans. Neural Networks Learn. Syst.* **20** (1), 61–80.

SCHERL, I., STROM, B., SHANG, J.K., WILLIAMS, O., POLAGYE, B.L. & BRUNTON, S.L. 2020 Robust principal component analysis for modal decomposition of corrupt fluid flows. *Phys. Rev. Fluids* **5** (5), 054401.

SCHMID, P.J. 2010 Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **656**, 5–28.

SELVARAJU, R.R., COGSWELL, M., DAS, A., VEDANTAM, R., PARIKH, D. & BATRA, D. 2017 Grad-CAM: visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626.

SHU, D., LI, Z. & FARIMANI, A.B. 2023 A physics-informed diffusion model for high-fidelity flow field reconstruction. *J. Comput. Phys.* **478**, 111972.

SMITH, L., FUKAMI, K., SEDKY, G., JONES, A. & TAIRA, K. 2024 A cyclic perspective on transient gust encounters through the lens of persistent homology. *J. Fluid Mech.* **980**, A18.

SOLERA-RICO, A., VILA, C.S., GÓMEZ-LÓPEZ, M., WANG, Y., ALMASHJARY, A., DAWSON, S.T.M. & VINUESA, R. 2024 β-variational autoencoders and transformers for reduced-order modelling of fluid flows. *Nat. Commun.* **15** (1), 1361.

SONODA, T., LIU, Z., ITOH, T. & HASEGAWA, Y. 2023 Reinforcement learning of control strategies for reducing skin friction drag in a fully developed turbulent channel flow. *J. Fluid Mech.* **960**, A30.

SRINIVASAN, P.A., GUASTONI, L., AZIZPOUR, H., SCHLATTER, P. & VINUESA, R. 2019 Predictions of turbulent shear flows using deep neural networks. *Phys. Rev. Fluids* **4** (5), 054603.

STONE, M. 1974 Cross-validatory choice and assessment of statistical predictions. *J. R. Stat. Soc.* **36** (2), 111–133.

SUÁREZ, P., ALCÁNTARA-ÁVILA, F., RABAULT, J., MIRÓ, A., FONT, B., LEHMKUHL, O. & VINUESA, R. 2025 Flow control of three-dimensional cylinders transitioning to turbulence via multi-agent reinforcement learning. *Commun. Engng* **4** (1), 113.

TAIRA, K., BRUNTON, S.L., DAWSON, S.T.M., ROWLEY, C.W., COLONIUS, T., MCKEON, B.J., SCHMIDT, O.T., GORDEYEV, S., THEOFILIS, V. & UKEILEY, L.S. 2017 Modal analysis of fluid flows: an overview. *AIAA J.* **55** (12), 4013–4041.

TAIRA, K., RIGAS, G. & FUKAMI, K. 2025 Machine learning in fluid dynamics: a critical assessment. *Phys. Rev. Fluids* **10** (9), 090701.

TENENBAUM, J.B., DE SILVA, V. & LANGFORD, J.C. 2000 A global geometric framework for nonlinear dimensionality reduction. *Science* **290** (5500), 2319–2323.

TIBSHIRANI, R. 1996 Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B* **58** (1), 267–288.

TOMASETTO, M., WILLIAMS, J.P., BRAGHIN, F., MANZONI, A. & KUTZ, J.N. 2025 Reduced order modeling with shallow recurrent decoder networks. *Nat. Commun.* **16** (1), 10260.

TRAN, J., FUKAMI, K., INADA, K., UMEHARA, D., ONO, Y., OGAWA, K. & TAIRA, K. 2024 Aerodynamics-guided machine learning for design optimization of electric vehicles. *Commun. Engng* **3** (1), 174.

TRAN, J., YEH, C.-A. & TAIRA, K. 2026 Using optimal transport aligned latent embeddings for separated flow analysis. *J. Fluid Mech.* **1027**, A24.

TSAI, Y.-H., WANG, Y., CHENG, C.-J., HUNG, C.-Y., CHOU, Y.-E., LI, K.-L., TSENG, C.-C., CHERN, R.-L. & CHOU, Y.-J. 2026 Data-driven turbulence closure using a tensor basis neural network without reliance on baseline models. *J. Fluid Mech.* **1033**, A15.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press

VASANTH, J., RABAULT, J., ALCÁNTARA-ÁVILA, F., MORTENSEN, M. & VINUESA, R. 2025 Multi-agent reinforcement learning for the control of three-dimensional Rayleigh–Bénard convection. *Flow Turbul. Combust.* **115** (3), 1319–1355.

VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A.N., KAISER, Ł. & POLOSUKHIN, I. 2017 Attention is all you need. *NeurIPS* **30**, 6000–6010.

VINCENT, F.-L., PETER, H., RIASHAT, I., MARC, G.B. & JOELLE, P. 2018 An introduction to deep reinforcement learning. *Foundations Trends Mach. Learn.* **11** (3–4), 219–354.

VINUESA, R. & BRUNTON, S.L. 2022 Enhancing computational fluid dynamics with machine learning. *Nat. Comput. Sci.* **2** (6), 358–366.

VREMAN, A.W. 2004 An eddy-viscosity subgrid-scale model for turbulent shear flow: algebraic theory and applications. *Phys. Fluids* **16** (10), 3670–3681.

WALTERS, P. 2000 *An Introduction to Ergodic Theory*. Springer Science & Business Media.

WAN, Z.Y. & SAPSIS, T.P. 2018 Machine learning the kinematics of spherical particles in fluid flows. *J. Fluid Mech.* **857**, R2.

WANG, H. *et al.*et al. 2023*a* Scientific discovery in the age of artificial intelligence. *Nature* **620** (7972), 47–60.

WANG, S., TENG, Y. & PERDIKARIS, P. 2021 Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM J. Sci. Comput.* **43** (5), A3055–A3081.

WANG, Y., LI, Z., YUAN, Z., PENG, W., LIU, T. & WANG, J. 2024 Prediction of turbulent channel flow using Fourier neural operator-based machine-learning strategy. *Phys. Rev. Fluids* **9** (8), 084604.

WANG, Y., SHIMADA, K. & BARATI FARIMANI, A. 2023*b* Airfoil GAN: encoding and synthesizing airfoils for aerodynamic shape optimization. *J. Comput. Des. Engng* **10** (4), 1350–1362.

WANG, Z., BOVIK, A.C., SHEIKH, H.R. & SIMONCELLI, E.P. 2004 Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13** (4), 600–612.

WU, H., CAO, Y., CHEN, Y., LAIMA, S., CHEN, W.-L., ZHOU, D. & LI, H. 2025 Generalizable super-resolution turbulence reconstruction from minimal training data. *J. Fluid Mech.* **1024**, A27.

WU, J.-L., XIAO, H. & PATERSON, E. 2018 Physics-informed machine learning approach for augmenting turbulence models: a comprehensive framework. *Phys. Rev. Fluids* **3** (7), 074602.

XU, D. & ZHANG, M. 2023 Reinforcement-learning-based control of convectively unstable flows. *J. Fluid Mech.* **954**, A37.

XUAN, A. & SHEN, L. 2023 Reconstruction of three-dimensional turbulent flow structures using surface measurements for free-surface flows based on a convolutional neural network. *J. Fluid Mech.* **959**, A34.

YAWATA, K., FUKAMI, K., TAIRA, K. & NAKAO, H. 2024 Phase autoencoder for limit-cycle oscillators. *Chaos* **34** (6), 063111.

YOUSIF, M.Z., YU, L., HOYAS, S., VINUESA, R. & LIM, H.C. 2023*a* A deep-learning approach for reconstructing 3-D turbulent flows from 2-D observation data. *Sci. Rep.* **13** (1), 2529.

YOUSIF, M.Z., YU, L. & LIM, H.C. 2022 Physics-guided deep learning for generating turbulent inflow conditions. *J. Fluid Mech.* **936**, A21.

YOUSIF, M.Z., ZHANG, M., YU, L., VINUESA, R. & LIM, H.C. 2023*b* A transformer-based synthetic-inflow generator for spatially developing turbulent boundary layers. *J. Fluid Mech.* **957**, A6.

YU, J., YAN, C. & GUO, M. 2019 Non-intrusive reduced-order modeling for fluid problems: a brief review. *Proc. Inst. Mech. Engng G* **233** (16), 5896–5912.

ZAKI, T.A. 2025 Turbulence from an observer perspective. *Annu. Rev. Fluid Mech.* **57**, 311–334.

ZENG, K., DE JESUS, C.E.P., FOX, A.J. & GRAHAM, M.D. 2024 Autoencoders for discovering manifold dimension and coordinates in data from complex dynamical systems. *Mach. Learn.: Sci. Technol.* **5** (2), 025053.

ZHANG, Y. & DENG, L.-J. 2025 Loss-driven dynamic weight and residual transformation in physics-informed neural network. *Neural Networks* **186**, 107841.

ZHAO, Y., AKOLEKAR, H.D., WEATHERITT, J., MICHELASSI, V. & SANDBERG, R.D. 2020 RANS turbulence model development using CFD-driven machine learning. *J. Comput. Phys.* **411**, 109413.

ZHENG, Q., LI, T., MA, B., FU, L. & LI, X. 2024 High-fidelity reconstruction of large-area damaged turbulent fields with a physically constrained generative adversarial network. *Phys. Rev. Fluids* **9** (2), 024608.

ZHONG, Y., FUKAMI, K., AN, B. & TAIRA, K. 2023 Sparse sensor reconstruction of vortex-impinged airfoil wake with machine learning. *Theor. Comput. Fluid Dyn.* **37** (2), 269–287.

ZHU, X. & GOLDBERG, A.B. 2009 *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers.

https://doi.org/10.1017/jfm.2026.12172 Published online by Cambridge University Press