fluids

MDPI

Review

Machine Learning in Fluid Dynamics—Physics-Informed Neural Networks (PINNs) Using Sparse Data: A Review

Mouhammad El Hassan [1,*], Ali Mjalled [2], Philippe Miron [3], Martin Mönnigmann [2] and Nikolay Bukharin [4]

[1] Department of Mechanical Engineering, Prince Mohammad Bin Fahd University, Al Khobar 31952, Saudi Arabia
[2] Automatic Control and Systems Theory, Ruhr-Universität Bochum, Universitätstrasse 150, 44801 Bochum, Germany
[3] Center for Ocean-Atmospheric Prediction Studies, Florida State University, Tallahassee, FL 32306, USA
[4] The School of Manufacturing and Automation, The Southern Alberta Institute of Technology, Calgary, AB T2M 0L4, Canada; nikolay.bukharin@sait.ca
* Correspondence: melhassan@pmu.edu.sa

Abstract

Fluid mechanics often involves complex systems characterized by a large number of physical parameters, which are usually described by experimental and numerical sparse data (temporal or spatial). The difficulty of obtaining complete spatio-temporal datasets is a common issue with conventional approaches, such as computational fluid dynamics (CFDs) and various experimental methods, particularly when evaluating and modeling turbulent flows. This review paper focuses on the integration of machine learning (ML), specifically physics-informed neural networks (PINNs), as a means to address this challenge. By directly incorporating governing physical equations into neural network training, PINNs present a novel method that allows for the reconstruction of flow from sparse and noisy data. This review examines various applications in fluid mechanics where sparse data is a common problem and evaluates the effectiveness of PINNs in enhancing flow prediction accuracy. An overview of diverse PINNs methods, their applications, and outcomes is discussed, demonstrating their flexibility and effectiveness in addressing challenges related to sparse data and illustrating that the future of fluid mechanics lies in the synergy between data-driven approaches and established physical theories.

Keywords: physics informed neural networks (PINNs); sparse data; data-driven modeling; PIV; CFD; machine learning in fluids; sparse data reconstruction

Academic Editors: Wei-Tao Wu and Filippos Sofos

Received: 11 April 2025
Revised: 1 August 2025
Accepted: 18 August 2025
Published: 28 August 2025

Citation: El Hassan, M.; Mjalled, A.; Miron, P.; Mönnigmann, M.; Bukharin, N. Machine Learning in Fluid Dynamics—Physics-Informed Neural Networks (PINNs) Using Sparse Data: A Review. *Fluids* **2025**, *10*, 226. https://doi.org/10.3390/fluids10090226

1. Introduction

Over the past few years, significant progress has been made in solving partial differential equations (PDEs), which describe the physics of fluid flows in various fluid mechanics applications. Numerous numerical and analytical techniques, along with effective approximations and simplifications for each particular case, were developed. Moreover, significant progress was made in fluid mechanics experimental data post-processing methods. Despite this progress, various problems with current approaches to fluid flow prediction and analysis remain. Fluid mechanics problems often involve complex systems that are governed by a wide range of physical parameters. While some computational fluid dynamics (CFDs) and experimental techniques provide detailed flow dynamics insight, they are often costly, both in terms of computational resources and experimental setup. These methods are frequently limited by physical constraints and uncertainties, such as simplifications in

modeling assumptions, numerical errors in computational simulations, and instrumentation inaccuracies. In response to these challenges, machine learning has been increasingly applied in the field of fluid mechanics, addressing tasks related to data processing [1,2], flow control [3–6] etc. While some studies used ML for applications such as super resolution [7–9] and flow classification [10,11], the main attention has been directed to data-driven modeling of flow fields from data. Specifically, artificial neural networks have been used for this purpose due to their ability to capture nonlinear relationships. A variety of neural network architectures have been used for modeling flow dynamics, including fully connected [12–14], convolutional [15–18], recurrent [19–21], and transformer networks [22–24], to name just a few. These models often aim to predict the dynamics either in a discrete manner, i.e., the model predicts future flow states at a fixed time interval based on the current flow state, or using a continuous formulation such as neural ordinary differential equations [25].

Recently, machine learning (ML) has become a promising option that can be implemented in a wide range of fluid mechanics problems. However, the proper training of ML neural networks requires a significant amount of high-quality data, which is not always available for a given application. Despite this challenge, these neural networks can be trained from supplementary data acquired by enforcing different laws of physics. This approach, known as physics-informed learning, combines noisy data with mathematical models, executing them using neural networks. Furthermore, there is a potential to develop specialized network structures that naturally fulfill certain physical principles, enhancing accuracy and generalization capabilities and optimizing training, [26]

In general, ML falls into three main categories: supervised, unsupervised, and semi-supervised learning. Supervised learning unites methods that derive input-output relationships from labeled data; in contrast, unsupervised learning extracts patterns from unlabeled data, while semi-supervised learning blends elements of both unsupervised and supervised learning. Supervised learning is commonly used for tasks such as classification and regression [27–31]. Reinforcement Learning (RL) is typically classified among general machine learning types; however, it will only be briefly addressed in this paper. It is based on structured learning through interaction, where an algorithm is given a set of actions, goals, and parameters. By exploring various strategies within these defined rules, the algorithm learns through trial and error. While reinforcement learning (RL) generally relies on a clearly defined reward function and is predominantly used for optimal control and decision-making tasks, emerging hybrid methods are beginning to incorporate physical constraints and considerations of data sparsity into the training process. However, these approaches tend to be less robust when it comes to addressing inverse problems or partial differential equations (PDEs) with sparse datasets, particularly in comparison to physics-informed neural networks (PINNs), which are explored in this review.

Deep learning (DL), a subset of ML and Artificial Intelligence (AI) (see Figure 1), can learn from data and has made it a popular subject in computing across diverse domains such as healthcare [32], visual recognition [33,34], cybersecurity [35], text analytics [36], etc. It is also recognized as a fundamental technology of Industry 4.0. Nonetheless, developing an effective DL model presents challenges due to the dynamic nature and variability of real-world problems and data [37].

Most DL models do not consider physical constraints, resulting in a poor fit with observational data. Introducing governing physical laws and information about the domain into model training can enhance its accuracy [38]. Therefore, PINNs were introduced in [39]. These frameworks, based on deep learning, can integrate both data and physical laws represented by governing PDEs to facilitate learning. PINNs have proven effective in addressing forward and inverse problems associated with various types of PDEs.

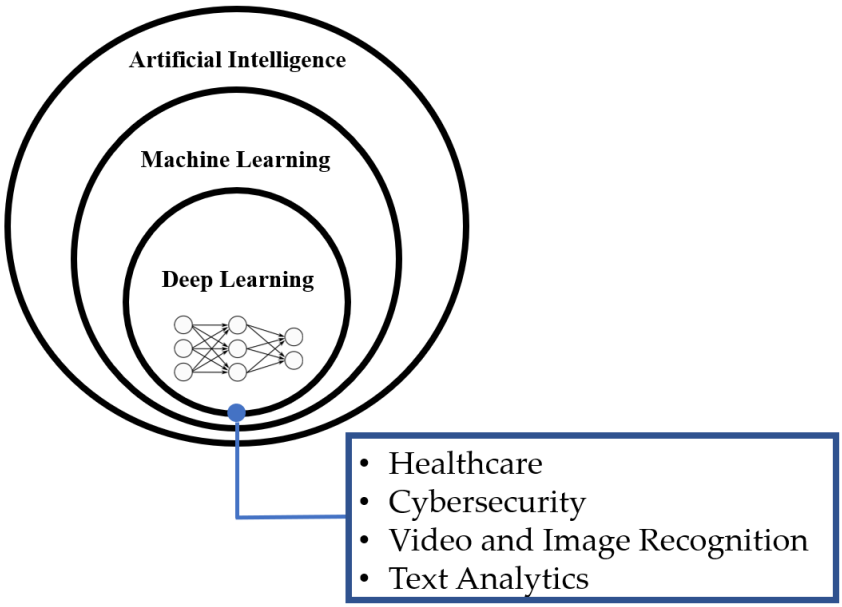

Figure 1. Deep learning applications.

The main requirement for developing an accurate data-driven model for capturing flow dynamics is access to high-dimensional spatio-temporal data. However, in many engineering applications, acquiring such data is challenging due to limitations such as experimental setup constraints, memory capacity, and sensor availability. These factors often result in sparse datasets. In the context of fluid mechanics, data sparsity refers to the condition where only a limited number of measurements are available in space and time relative to the high-dimensional nature of the flow field. Most likely, this occurs due to constraints in experimental resolution (limited sensors, optical access, probe locations, etc.) or computational resources (storage and post-processing limitations in high accuracy simulations). A dataset is considered sparse when the quantity, density, or coverage of the data points is insufficient to resolve the full flow structure or flow dynamics directly and requires the use of reconstruction or interpolation techniques.

Unlike general definitions where sparsity implies many missing or zero-valued entries, in fluid mechanics, sparsity more specifically reflects the incompleteness of physical observations relative to the complexity of the governing phenomena. Sparse data problems arise in various fluid mechanics applications [40–42], including turbulent flow analysis [38,43,44], multiphase flow modeling [45], and environmental fluid dynamics [46,47], where obtaining comprehensive datasets across the entire flow domain or time frame can be difficult and impractical.

Effective handling of sparse data in fluid mechanics is crucial for improving the accuracy of flow predictions, optimizing designs, and enhancing the understanding of physical phenomena. Traditional methods based on interpolating or extrapolating sparse data can introduce uncertainties or inaccuracies, especially in highly nonlinear systems. To mitigate these challenges, modern approaches such as data assimilation [48–51], machine learning techniques [52], and reduced-order modeling [53–56] have been adopted. These techniques leverage existing sparse datasets to reconstruct full-field flow information.

This review aims to explore various methods for handling sparse data in fluid mechanics, focusing on how recent advances in machine learning can overcome these challenges. Key applications where sparse data is particularly problematic are discussed, and a review of the current state-of-the-art solutions based on physics-informed neural network (PINN) models is provided.

2. Physics-Informed Neural Network Methods Tailored for Navier–Stokes Equations

PINNs are commonly used to simulate fluid flows governed by Navier–Stokes (NS) equations [57,58]. The goal is to parameterize the solution using a neural network $\mathcal{N}$. For a given spatial location (x, y) defined in a 2D domain and time t, the neural network predicts the velocity components $\widetilde{u}$ and $\widetilde{v}$ in x and y directions, respectively, as well as the pressure $\widetilde{p}$, i.e.,

$$\widetilde{u},\ \widetilde{v},\ \widetilde{p} = \mathcal{N}(\theta, x, y, t), \tag{1}$$

where θ represents the parameters of the network. Training a PINN involves minimizing a loss function that is regularized by incorporating the physical laws as soft constraints. Depending on the specific characteristics of the underlying fluid problem, availability of data, and computational constraints, various PINN approaches can be utilized to model fluid flows effectively. Below, we provide an overview of the most commonly used PINN variants for solving Navier–Stokes equations (see Table 1 for a summary).

Table 1. PINN methods tailored for Navier–Stokes.

Method	Description	Usage	Limitation
Standard PINNs	Minimize the residuals of the continuity and momentum equations	Well-posed boundary and initial value problem	Struggle with high Reynolds numbers or complex geometries
Data-driven PINNs	Leverage labelled data to train the model in addition to physical constraints	Problems with available simulation or experimental data	Requires careful balance between physical loss and data-driven loss
Extended PINNs	The domain is decomposed into subregions	Large-scale problems	Requires handling on interfaces
Variational PINNs	The training considers an integral-based loss rather than a pointwise evaluation	Problems with discontinuity	Complex implementation
Stochastic PINNs	Incorporate probabilistic elements to model uncertainties	Fluid flows with uncertainty parameters	Suffer from instability during training
PINNs with turbulence	The turbulence equations are provided as additional constraints to the optimization problem	Highly turbulent flows	Rely on turbulence models, which are usually derived empirically

2.1. Standard PINNs

Standard PINNs are suitable for solving forward and inverse problems for the steady state or transient NS equations in well-defined geometries with known initial and boundary conditions, [39,59]. These models are particularly effective in canonical domains such as laminar flow over flat plates, flow in straight arteries, and benchmark lid-driven cavities. The loss function of the network $\mathcal{L}_{\text{PINN}}$ includes terms to satisfy the continuity and momentum equations, as well as boundary and initial conditions. For 2D incompressible NS equations, the loss function reads

$$\mathcal{L}_{\text{PINN}} = \mathcal{L}_{\text{Con}} + \mathcal{L}_{\text{Mom}-x} + \mathcal{L}_{\text{Mom}-y} + \mathcal{L}_{\text{BC}} + \mathcal{L}_{\text{IC}}, \tag{2}$$

where:

$$\mathcal{L}_{\text{Con}} = \frac{1}{N_f}\sum_{i=1}^{N_f} \epsilon_{\text{con}_i}^2 = \frac{1}{N_f}\sum_{i=1}^{N_f}\left(\frac{\partial \widetilde{u}}{\partial x_i} + \frac{\partial \widetilde{v}}{\partial y_i}\right)^2, \tag{3}$$

$$\mathcal{L}_{\text{Mom}-x} = \frac{1}{N_f}\sum_{i=1}^{N_f}\epsilon_{x_i}^2 = \frac{1}{N_f}\sum_{i=1}^{N_f}\left(\frac{\partial\widetilde{u}}{\partial t_i} + \widetilde{u}\frac{\partial\widetilde{u}}{\partial x_i} + \widetilde{v}\frac{\partial\widetilde{u}}{\partial y_i} + \frac{\partial\widetilde{p}}{\partial x_i} - \nu\left(\frac{\partial^2\widetilde{u}}{\partial x_i^2} + \frac{\partial^2\widetilde{u}}{\partial y_i^2}\right)\right)^2, \quad (4)$$

$$\mathcal{L}_{\text{Mom}-y} = \frac{1}{N_f}\sum_{i=1}^{N_f}\epsilon_{y_i}^2 = \frac{1}{N_f}\sum_{i=1}^{N_f}\left(\frac{\partial\widetilde{v}}{\partial t_i} + \widetilde{u}\frac{\partial\widetilde{v}}{\partial x_i} + \widetilde{v}\frac{\partial\widetilde{v}}{\partial y_i} + \frac{\partial\widetilde{p}}{\partial y_i} - \nu\left(\frac{\partial^2\widetilde{v}}{\partial x_i^2} + \frac{\partial^2\widetilde{v}}{\partial y_i^2}\right)\right)^2, \quad (5)$$

with ρ and ν denoting the density and kinematic viscosity, respectively. The terms ϵ_{con_i}, and $\epsilon_{(x/y)_i}$ are the residuals for the continuity and momentum equations, respectively, evaluated at (x_i, y_i, t_i). Similarly, $\mathcal{L}_{\text{BC}}$ and $\mathcal{L}_{\text{IC}}$ are residual terms introduced to enforce boundary and initial conditions, respectively. During the training, N_f training points, usually denoted as collocation points, are used to evaluate the residuals and to backpropagate the errors to update the network parameters. Altogether, this loss formulation ensures the learned velocity and pressure fields simultaneously satisfy the governing PDEs and the initial-boundary conditions, integrating domain knowledge directly into the optimization process.

Although standard PINNs can handle well-posed boundary-value problems, they usually struggle when considering complex geometries and high Reynolds numbers. In such cases, the loss landscape becomes stiff [60], and the optimizer struggles to converge to a physically meaningful solution [61].

2.2. Data-Driven PINNs

Data-driven PINNs extend the standard framework by integrating partial simulation or experimental data, especially useful when observations are spatially or temporally sparse, as in biomedical flow imaging, environmental sensing, or wind tunnel measurements [62]. This is usually the case when simulation or experimental flow data are spatially or/and temporally sparse. Specifically, the neural network learns to interpolate between the available data points while ensuring that the predictions satisfy the Navier–Stokes (NS) equations. The loss function in this case incorporates an additional term penalizing deviation from the observed measurements, which reads

$$\mathcal{L}_{\text{data}-\text{driven}} = \mathcal{L}_{\text{PINN}} + \alpha\mathcal{L}_{\text{data}} = \mathcal{L}_{\text{PINN}} + \alpha\frac{1}{N_d}\sum_{i=1}^{N_d}(\widetilde{u}_i - u_i)^2 + (\widetilde{v}_i - v_i)^2 + (\widetilde{p}_i - p_i), \quad (6)$$

where α is a weighting factor for the multi-objective optimization problem, N_d is the number of observation points, and $u, v,$ and p are the observed velocity and pressure measurements. The data-driven PINN framework is outlined in Figure 2.

The total loss $\mathcal{L}_{\text{data}-\text{driven}}$ requires careful calibration between the physical loss term $\mathcal{L}_{\text{PINN}}$ and the data loss term $\mathcal{L}_{\text{data}}$. When α is set too low, the neural network overfits the physical equations and ignores the measurement data. In contrast, setting α too high reproduces the measurement data at the expense of violating the underlying governing equations. Accordingly, various methods have been proposed to mitigate this trade-off, e.g., gradient normalization [63], and self-adaptive learning [64].

2.3. Extended PINNs for Domain Decomposition

Extended PINNs (XPINNs) consist of decomposing the domain into multiple regions and solving the N-S equations in each subdomain, allowing for parallel computing [65]. Within each subdomain, a separate neural network is trained to satisfy the governing physical laws, such as the Navier–Stokes equations.

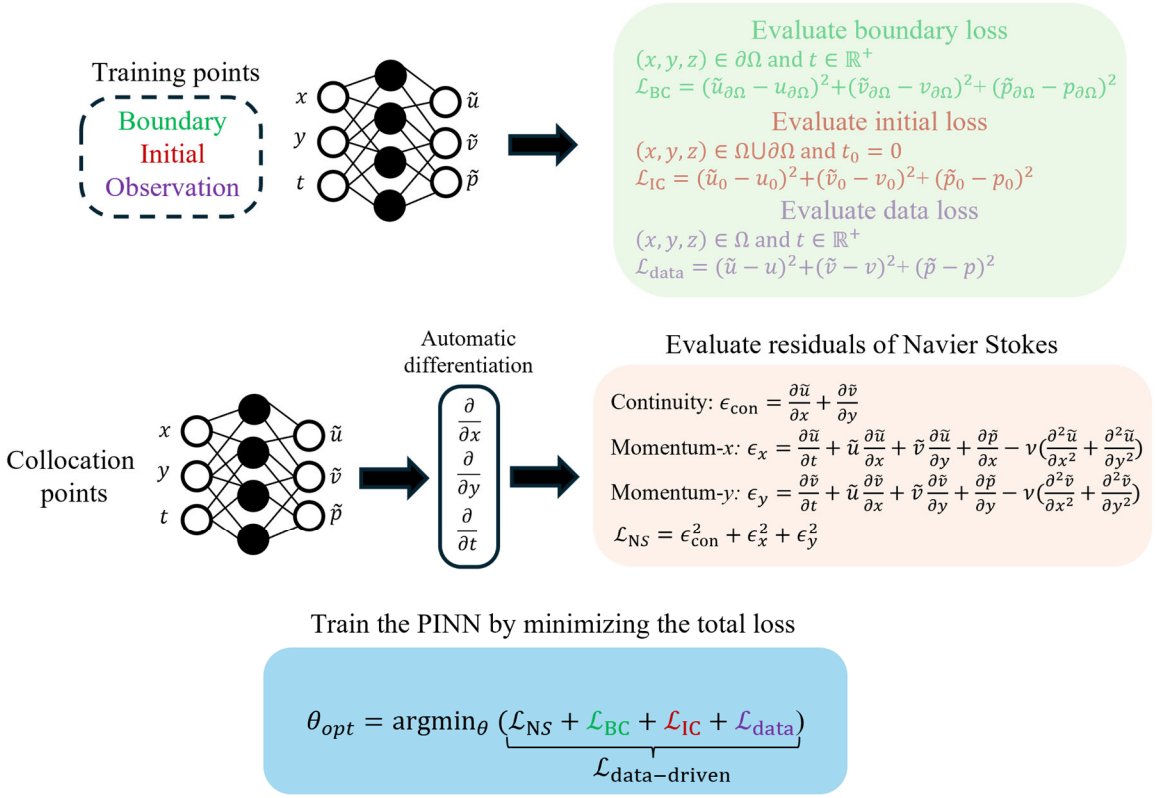

Figure 2. PINN framework for the 2D incompressible Navier–Stokes equations with observations. θ denotes the trainable parameters of the NN.

This domain decomposition approach brings several advantages:

- Parallelization: each subdomain can be trained independently, enabling efficient parallel computation.
- Scalability: XPINNs are particularly well-suited for large or complex geometries (such as vascular networks, combustor chambers, or porous geological formations), where a monolithic network would struggle with convergence or memory limitations.
- Flexibility: local variations in physical properties (e.g., heterogeneous media) can be modeled more effectively using specialized networks in different regions.

The loss function of XPINNs reads

$$\mathcal{L}_{\text{XPINN}} = \sum_k \mathcal{L}_{\text{PINN,k}} + \mathcal{L}_{\text{inter,k}}, \tag{7}$$

where $\mathcal{L}_{\text{PINN,k}}$ is the PINN loss (data, boundary, and PDE residuals) evaluated in each region k and $\mathcal{L}_{\text{inter,k}}$ denotes the interface loss, which enforces continuity of the solution and its derivatives across adjacent subdomain boundaries.

The residuals in this context refer to how well the neural network predictions satisfy the PDE and interface conditions. Specifically, for each subdomain, the residual of the PDE is minimized at a set of collocation points. At interfaces, additional residuals are defined to ensure the physical compatibility between subdomains—this typically includes continuity in both the solution and fluxes (e.g., pressure, velocity gradients).

The XPINN framework has been successfully applied in various applications, including flow simulations through highly heterogeneous media [66] and modeling of supersonic flows [67].

2.4. Variational PINNs

Variational PINNs (VPINNs) use a weak formulation of the NS equations, allowing better stability in irregular or highly curved domains, such as aneurysms, coronary arteries, or bioinspired microfluidic channels [68,69]. The training process consists of minimizing an integral-based loss, denoted as $\mathcal{L}_{\text{VPINN}}$, rather than pointwise evaluation, i.e.,

$$\mathcal{L}_{\text{VPINN}} = \sum_{i=1}^{N_b} \left(\int_{\Omega} \phi_i \cdot \left(\partial_t u + (u \cdot \nabla) u + \frac{1}{\rho} \nabla p - \nu \Delta u \right) d\Omega \right)^2, \tag{8}$$

where $\{\phi\}_{i=1}^{N_b}$ is a basis of test functions spanning the test space, and Ω denotes the fluid domain. In [68], a Petrov–Galerkin variational PINN was introduced by representing the trial functions—i.e., the unknowns being approximated using neural networks—while the test functions were taken as Legendre polynomials on each element. In this formulation, Legendre polynomials were chosen because they (i) are orthogonal on the reference element, (ii) offer good approximation properties for smooth functions, and (iii) permit efficient methods for computing integrals and derivatives.

2.5. Stochastic PINNs

Stochastic PINNs (SPINNs) address uncertainty in the fluid flow, such as random initial or boundary conditions [70,71]. Applications include weather prediction, turbulent combustion with input uncertainty, and patient-specific flows with uncertain measurements. The neural network treats the PDE constraints probabilistically, which is essential for modeling fluid flows with uncertain parameters or noisy observations. Within the SPINN framework, the individual terms of the loss function are enforced as expected values, with the option of incorporating an additional regularization term to minimize the variance of the predicted solution.

Let $\mathcal{L}_{\text{PINN}}(\xi)$ denote the standard PINN loss evaluated on a stochastic input sample ξ, e.g., drawn from uncertain boundary data, and similarly let $\mathcal{L}_{\text{data}}(\xi)$ be the corresponding data loss. The corresponding SPINN loss reads:

$$\mathcal{L}_{\text{SPINN}} = \mathbb{E}_{\xi}[\mathcal{L}_{\text{PINN}}(\xi)] + \alpha \mathbb{E}_{\xi}[\mathcal{L}_{\text{data}}(\xi)] + \lambda \sum_{\varphi \in \{u,v,p\}} \mathbb{V}_{\xi}[\varphi], \tag{9}$$

where $\mathbb{E}_{\xi}[\cdot]$ denotes the expectation over uncertain input samples, λ is a weighting factor controlling the regularization rate, $\mathbb{V}\xi[\cdot]$ denotes the pointwise variance of predictions, and α is a weighting factor that controls the relative importance of the data loss.

2.6. PINNs with Turbulence

In this family of methods, turbulence models are incorporated within the PINN framework (e.g., $\kappa - \epsilon$ [72,73] or Spalart–Allmaras [43] model). This extension helps model the effects of turbulence rather than directly solving them. The corresponding loss function is written as

$$\mathcal{L}_{\text{PINN,tr}} = \mathcal{L}_{\text{PINN}} + \mathcal{L}_{\text{data}} + \mathcal{L}_{\text{tur}}, \tag{10}$$

where $\mathcal{L}_{\text{tur}}$ denotes the residual of the turbulence model. These methods are used in aerodynamic simulations, jet mixing studies, and flow separation predictions.

2.7. B-PINNs: Bayesian Physics-Informed Neural Networks

In [70], the authors propose a Bayesian physics-informed neural network (B-PINN) framework to solve forward and inverse partial differential equation (PDE) problems under uncertainty. Unlike standard PINNs, B-PINNs incorporate Bayesian neural networks (BNNs) to enable uncertainty quantification and improve robustness against noisy data.

1. Problem setup

   Given a general PDE: $N_x(u;\lambda) = f(x),\ x \in D$ $(d-\text{dimensional physical domain})$, and boundary condition operator acting on the domain boundary $\Gamma:\ B_x(u;\lambda) = b(x),\ x \in \Gamma$ with noisy data:

$$\overline{u}^{(i)} = u\left(x_u^{(i)}\right) + \varepsilon_u^{(i)}, \tag{11}$$

$$\overline{f}^{(i)} = f\left(x_f^{(i)}\right) + \varepsilon_f^{(i)}, \tag{12}$$

$$\overline{b}^{(i)} = b\left(x_b^{(i)}\right) + \varepsilon_b^{(i)}, \tag{13}$$

   with $\varepsilon \sim N\left(0, \sigma^2\right)$. This noise reflects aleatoric uncertainty.

2. Bayesian Surrogate and Posterior Inference

   The surrogate solution $\widetilde{u}(x;\theta)$, parameterized by a BNN, satisfies:

$$\widetilde{f}(x;\theta) = N_x(\widetilde{u}(x;\theta)),\ \widetilde{b}(x;\theta) = B_x(\widetilde{u}(x;\theta)). \tag{14}$$

   The likelihood function over the noisy data is:

$$P(D|\theta) = P(D_u|\theta)P\left(D_f\middle|\theta\right)P(D_b|\theta), \tag{15}$$

   for example:

$$P(D_u|\theta) = \prod_{i=1}^{N_u} \frac{1}{\sqrt{2\pi\sigma_u^{(i)^2}}} \exp\left(-\frac{\left(\widetilde{u}(x_u^i;\theta) - \overline{u}^{(i)}\right)^2}{2\pi\sigma_u^{(i)^2}}\right). \tag{16}$$

   The posterior is obtained from Bayes' theorem:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \simeq P(D|\theta)P(\theta), \tag{17}$$

   which is approximated via:

   - Hamiltonian Monte Carlo (HMC)—accurate but computationally expensive
   - Variational Inference (VI)—efficient but less reliable due to mean-field assumptions
   - Dropout—used as a non-Bayesian baseline (performed poorly in uncertainty estimation)

3. Inverse Problem Extension

   For inverse problems, where PDE parameters λ are also unknown:

$$P(\theta, \lambda|D) \simeq P(D|\theta,\lambda)P(\theta,\lambda) = P(D|\theta,\lambda)P(\theta)P(\lambda). \tag{18}$$

   This enables joint inference of both the solution and the unknown physical parameters.

4. Main Results

   - B-PINN-HMC yields superior predictive accuracy and reliable uncertainty estimates.
   - B-PINN-VI underestimates uncertainty due to Gaussian factorization.
   - Dropout-based PINNs fail to capture meaningful uncertainty.
   - Replacing BNNs with a truncated Karhunen–Loève (KL) expansion yields similar accuracy at lower computational cost for low-dimensional problems.
   - Deep Normalizing Flows (DNFs) are evaluated as alternative posterior estimators.

In general, it was demonstrated that B-PINNs offer a robust, probabilistically grounded alternative to deterministic PINNs, particularly in the presence of noise and limited data.

The Bayesian formulation enables principled uncertainty quantification but requires careful consideration of prior choices and scalability for high-dimensional applications.

A similar approach was used in [74].

3. Physics-Informed Neural Networks in Experimental and Numerical Fluid Dynamics

Examples of PINNs in Laminar and Turbulent Flows

The physics-informed neural network (PINN) was applied in [44] to address certain limitations in simulating incompressible turbulent and laminar flows. Navier–Stokes flow nets (NSFnets) were introduced, which are designed based on the Navier–Stokes equations using two mathematical formulations: velocity-velocity (VV) and velocity-pressure (VP).

Databases from direct numerical simulation (DNS) and analytical solutions were used to provide boundary conditions and reference baselines for the NSFnets. The networks take spatiotemporal coordinates as input and generate instantaneous velocity as output. In the VP-NSFnet, the output includes vorticity fields, while the VV-NSFnet outputs both pressure fields and instantaneous velocity.

This approach eliminates the need for labeled data beyond the boundary conditions, properties of the fluid, and initial conditions. Residuals of the governing equations (VV or VP), together with boundary and initial conditions, are incorporated into the loss function of the NSFnets.

It is interesting to note that the VP-NSF-net does not receive direct pressure data; instead, this information is treated as a hidden state and is derived through the incompressibility constraint without any additional computational costs. It should be noted that this approach might not be directly applicable for compressible flows where pressure plays a crucial role in the governing equations. For compressible flows, pressure cannot be obtained from velocity and incompressibility constraints alone, since the flow's density and thermal energy also vary and influence the pressure field. Therefore, in models such as NSFnet for compressible flows, pressure must typically be treated as an explicit variable rather than a hidden state, and its evolution must be calculated directly based on the compressible Navier–Stokes equations, including the equation of energy and other thermodynamic relationships.

For laminar flow, it was demonstrated by [44] that both the VV and VP VV formulations achieve similar accuracy. In contrast to traditional numerical methods, NSFnets adopt neural network (NN) properties, leading to a total error composition comprising approximation, optimization, and generalization errors. In this study, an attempt was made to empirically quantify these errors by changing NN architectures, iterative solvers, and sampling ("residual") points. For the turbulent channel flow, NSFnets showed the ability to sustain turbulence at the skin-friction-velocity Reynolds number $Re_\tau \sim 1000$ [75]. However, due to intensive training requirements, the focus was made on enforcing velocity boundary conditions based on DNS data subdomain boundaries and a subset of the channel domain.

In [76], PINN was applied to study a flow with a low Reynolds number around a cylinder without simulation or experimental data input. Freestream conditions were used for PINN training and to define the boundaries of the computation domain.

The physics-informed neural network (PINN) consisted of layers (fully connected), where the biases and weights were adjusted according to specified boundary conditions and the governing physical equations. Every node in one layer of a fully connected network is connected to every other node in the next layer, which contains a number of nodes according to the features fed into the network and is known as the input layer. For problems related to fluid flow, these characteristics typically include time and spatial coordinates. The final layer, or output layer, provides the expected values for the input

features, with the number of nodes matching the dimension of the prediction. The layers, which exist between the output and input layers, are known as hidden layers and contain a variable number of nodes.

This PINN effectively captures the overall flow field trends, with velocity fields generated by the PINN comparable to the fields obtained from computational fluid dynamics (CFDs). The Navier–Stokes equations solution was achieved numerically with simpleFoam, which utilizes the SIMPLE algorithm to iteratively update the pressure and velocity fields. Second-order upwind schemes were used for spatial discretization.

However, the PINN encountered challenges with pressure fields due to the absence of an explicit pressure term in the loss function. It was shown that an increased number of layers had the most significant impact on result accuracy, followed by expanding the point cloud size. The smallest performance improvement was observed when enhancing the nodes in each hidden layer. Furthermore, the PINN approach was more memory-efficient than CFD but did not necessarily outperform it in computational time. In simpler cases, where the CFD mesh contained a low number of cells, PINN took more time to compute results as compared to CFD. However, in scenarios involving high cell counts in CFD meshes, PINN demonstrated greater time efficiency.

Cases with different amounts of data demonstrated that the solution's resolution was significantly affected by the point cloud density. A low-density point cloud can fail to resolve near-wall flow conditions properly, while an overly dense point cloud can overcomplicate the neural network's training. To isolate the effect of point cloud density, each hidden layer was set to 50 nodes and kept constant for each of the 20 hidden layers of the network's architecture. PINN accuracy was tested with point clouds of 2000, 5000, 10,000, and 15,000 points.

It was shown that with only 2000 points, PINN failed to match CFD results, especially in capturing the leading-edge velocity profiles and pressure field magnitudes. This indicates that a 2000-point cloud is too scattered to resolve the pressure and velocity fields accurately for this scenario. When the point cloud density was increased to 5000 points, PINN began to show improved accuracy in both the field of pressure and velocity. The velocity profile became closer to CFD results; PINN still predicted a smaller flow acceleration region due to capturing a lower magnitude of velocity.

For the 10,000-point cloud, no significant improvements in the velocity components were observed compared to the 5000-point case. Although u-velocity components captured the wake profile, the magnitude of velocity near the leading edge was reduced in the high flow region. The v-velocity profile followed similar trends, with diminished high-velocity regions. However, the pressure profile from PINN showed increased symmetry, though the pressure magnitudes remained lower than those produced by CFD.

4. Physics-Informed Neural Networks Using Sparse Data

4.1. Introducing PINN Using Sparse Data

PINNs' approach was used in many applications [77–82], but in this review paper, the focus will be on applications of flow reconstruction using data from noisy and sparse data. Researchers are facing challenges in reconstructing a flow field from noisy and limited measurements, which are encountered across various engineering applications. To compensate for the sparsity and incompleteness of the data, supplementary information is required, which can be sourced from physics-based models or offline flow databases. With access to an extensive database of flow fields, correlation features, and coherent structures of fluid flow, it is possible to extract and utilize these features to reconstruct high-resolution flow fields using sparse data. Flow feature extraction techniques such as Proper Orthogonal Decomposition (POD) [83,84] or Dynamic Mode Decomposition (DMD) [85] are usually

employed in this context. Recent progress in the development of deep learning algorithms for image super-resolution has opened up new possibilities for flow reconstruction from limited measurements. NNs were applied to either learn to directly construct end-to-end mapping between high-resolution flow fields and sparse measurements or to learn the POD coefficients.

However, these deep learning models heavily rely on sufficient training data (offline), which may not be available in many applications. To address the problem related to sparsity of data, a physics-constrained deep learning approach has been proposed for flow reconstruction [86].

4.2. PINNs Using Navier–Stokes Equations and Sparse Set of Video Frames

In [87], a novel method for reconstructing fluid flows utilizing the Navier–Stokes equations (governing physics) was introduced with an end-to-end optimization process. This approach bypasses the need for explicit inputs such as boundary conditions and geometry, using only a sparse set of video frames. This strategy establishes a smooth spatio-temporal representation of scenes, employing neural networks to model density and velocity solutions for fluid flows. A novel reconstruction of fluid interactions with static obstacles was achieved without requiring additional human labeling or geometry by segregating dynamic and static elements. This approach adheres to physical principles and benefits from image supervision.

To enhance optimization from sparse input views, a layer-by-layer progressive growth strategy was implemented, gradually increasing the network capacity of the resulting neural representation. Utilizing this progressively growing model alongside a newly introduced regularization term, density–color ambiguity in radiance fields was effectively addressed without encountering overfitting issues.

During training, a sliding window approach was used to progressively involve early-layer neurons, similar to shallow model training, and then to cover the whole architecture by slowly moving the window towards the deeper layers. For every hidden layer $m \in [0, N-1]$, the following weighting parameter governed the sliding window: $w_{lm} = \text{clamp}(1 + m_a - m, 0, 1) \cdot \text{clamp}\,(1 + m - m_a, 0, 1)$, where $m_a = 1 + (N-2)/S$ represents the current last hidden layer. Here, s refers to the present training iteration, and S represents the iteration step at the completion of progressive growth. The layer $m = 0$ (with $w_{l0} = 0$) is a permanent hidden layer. For $m = 1$, the layer fades out, whereas intermediate layers fade in and out, and the last layer ($m = N - 1$) only fades in. A reasonable density profile was learned by the growing model in the second row after applying progressive growing. However, at the 5000-iteration step, "ghost density" artifacts appeared. To address and penalize this ghost density, a regularization technique in 2D image space was proposed.

$$\mathcal{L}_{\text{ghost}}\big(C(r_{ijt}), B_{ijt}, a(r_{ijt})\big) = \text{sigmoid}\Big[-\big(C(r_{ijt}) - B_{ijt}\big)^2\Big] \cdot A(r_{ijt}), \tag{19}$$

Background B_{ijt} is a known input, and $\mathcal{L}_{\text{ghost}}$ penalizes the opacity $t = k = 1KT(hk)\alpha(hk)$ when the pixel color $C(r_{ijt})$ is close to the background color in the 2D image space. After applying $\mathcal{L}_{\text{ghost}}$ on the progressively growing model, it effectively refined the density profile during training. This approach, "ghost density" removal, proved effective across many scenes.

4.3. PINNs in Biomedical Applications Without Boundary Conditions

In [88], PINN was applied to enhance the quantification of the wall shear stress (WSS) in diseased arterial flow, particularly in scenarios where some boundary conditions, such as outlet and inlet boundaries, were not known. The WSS and blood flow near arterial walls play crucial roles in regulating major cardiovascular diseases. However, accurately

quantifying them poses challenges due to issues such as uncertainty, low resolution, and noise in experimental and computational measurements of WSS for a specific patient. The PINN offers a flexible deep learning approach by combining mathematical equations of blood flow with experimental data. By incorporating the governing equations, particularly the Navier–Stokes equations, the substantial data requirements of the PINN can be dramatically reduced. In this study, an example using idealized aneurysm and stenosis models combined partial knowledge of flow physics with partial measurements, yielding a precise prediction of near-wall flow. The mathematical problem in [88] was presented with steady, incompressible, Newtonian Navier–Stokes equations and the continuity equation.

In cases where we have accurate knowledge of the parameters within the Navier–Stokes equations (represented by μ and ρ) and the boundary conditions within the computational domain, traditional computational fluid dynamics (CFDs) methods can effectively solve the equation mentioned above. However, in applications related to patient-specific cardiovascular fluid mechanics, there is often partial knowledge or uncertainty associated with these parameters and boundary conditions. In this scenario, the issue is that the problem lacks a clear definition, leading to a lack of a unique solution. However, it has been suggested that by using sparse measurements within the domain, a solution might be achievable. Suppose sparse set points were used to evaluate the velocity vector field. The main aim of PINN is to find the optimal solution $u(x)$ that not only conforms to these measured data points but also verifies the Navier–Stokes equations under unknown parameters or partially specified boundary conditions. In [88], fully connected neural networks (NNs) were specifically used; artificial neuron layers were stacked to form a composite function. An affine transformation was applied to every layer by the previous layer, introducing nonlinearity through an element-wise activation function. The biases and weights are determined by using Adam optimization, which is a variant of stochastic gradient descent optimization [89]. For all layers and cases, the Swish activation function was used.

With a proper definition of the boundary conditions (BC), a mathematically well-posed problem can be constructed by the two objectives mentioned above (BC and Navier–Stokes), which can be solved using a physics-informed neural network. However, in some cases of cardiovascular modeling of a specific patient, the boundary conditions are not fully known. Therefore, in [88], a boundary condition loss L_{BC} was only applied on certain parts of the boundary. While with conventional numerical techniques, solving partial differential equations (PDEs) is not usually feasible without well-defined BCs, PINNs can still converge to a solution by minimizing the residual of the field equation and handling fully unknown or partially known BCs. Despite this, the resulting solution may not be the true one, as many or infinite numbers of possibilities are present to find the solution of PDE when BCs are not fully defined. To address this challenge, sparse data from the measurements was introduced to define a new data loss term ($\mathcal{L}_{\mathrm{data}}$):

$$\mathcal{L}_{\mathrm{data}} = \|z - f\|_{\mathrm{G}}, \tag{20}$$

where f is the function of velocity (it was assumed that the velocity field was measured in sparse data points $u(x_i) = f(x_i),\ x_i \in \Gamma$; with $\Gamma = \{x_i\},\ i = 1, \ldots, N$ representing discrete measurement locations where the velocity was obtained by the function f and z is the velocity measurement in sparse locations G. The total loss function was defined as

$$\mathcal{L}_{\mathrm{tot}}(W_i, b_i) = \mathcal{L}_{\mathrm{phys}} + \lambda_b \mathcal{L}_{\mathrm{BC}} + \lambda_d \mathcal{L}_{\mathrm{data}}, \tag{21}$$

$$W_i^*, b_i^* = \mathcal{L}_{\mathrm{tot}}(W_i, b_i), \tag{22}$$

For each pressure network and velocity component b_i^* and W_i^*, the optimal weights and biases were obtained such that the total loss function $\mathcal{L}_{\text{tot}}$ was minimized.

The λ_d and λ_b are known as hyperparameters and were used to account for the contribution of the measurement data and boundary conditions (BCs), respectively. The framework presented in [88] was tested across various scenarios, including 2D blood flow simulation in both aneurysmal and stenotic conditions, 1D advection-diffusion transport, and blood flow simulations in 3D in an idealized aneurysm, and achieved highly accurate predictions of the wall shear stress (WSS). Specifically, two unknown parameters were required to solve the 1D problem analytically, but the boundary conditions were not specified. The coefficient was obtained using a regression problem that uses the sparse measurement data (three points in this case). In this example, a sharp boundary layer is applied to make the regression problem ill-conditioned, requiring at least one point within the boundary layer to address this issue. In the two-dimensional problems, the inlet boundary condition was not accurately identified by PINN; however, it correctly reconstructed the flow field where the sensors were placed and downstream of them. This suggests that the PINN framework in [88] should be viewed as an inverse modeling problem by which the inlet boundary condition is to be predicted. A common issue with many inverse problems is that for the same flow rate, different inlet velocity profiles can yield similar downstream solutions, and consequently, there may be no unique solution. Therefore, to accurately identify the inlet boundary conditions, sensors need to be placed near the inlet of the domain. It should be noted that the results in [88] demonstrated that accurately determining the WSS in the region of interest does not necessarily require knowing the inlet velocity profile. This framework illustrates the potential of integrating partial knowledge of cardiovascular flow physics, such as the Navier–Stokes equations, with sparse data gathered from localized regions without specific boundary conditions. This combination enables the extraction of clinically relevant hemodynamic parameters such as WSS.

4.4. PINNs Using Reynolds-Averaged Navier–Stokes Equation Without Closure Turbulence Model

In [90], PINNs incorporating RANS equations were used for two specific scenarios: the periodic hill problem ($5600 \leq Re_b \leq 37{,}000$; Re_b is the Reynolds number based on bulk velocity) and an adverse-pressure-gradient (APG) boundary layer ($1000 < Re_\theta < 3000$, where θ is the momentum thickness and Re_θ—Reynolds number based on momentum thickness). In order to close the RANS equations, this approach does not rely on a turbulence model; instead, the neural network learns to infer the Reynolds stress fields during training as a means to obtain closure. Regarding the potential of PINNs to provide reliable estimations with sparse training data, this study indicates their capability to be trained using data typically collected in experiments. The goal is to determine the quantity and distribution of data required to model a time-averaged turbulent flow accurately with PINNs. For the APG boundary layer, the goal is to determine the amount and distribution of data needed to model a time-averaged turbulent flow accurately with PINNs. It was shown that PINNs can more effectively model wall pressure and WSS and show their performance when trained without crucial flow features such as a separation bubble. Additionally, it was demonstrated that the training cost of the network remains consistent across different Reynolds numbers. The PINN architecture overview is shown in Figure 2. The loss function in [90] was defined as

$$\mathcal{L} = \mathcal{L}_b + \lambda_r \mathcal{L}_r, \tag{23}$$

$$\mathcal{L}_b = \frac{1}{N_b} \sum_{j=1}^{N_d} \sum_{i=1}^{N_b} \left[T_{ij} - \hat{U}_{ij} \right]^2, \tag{24}$$

$$\mathcal{L}_r = \frac{1}{N_r} \sum_{j=1}^{N_e} \sum_{i=1}^{N_r} \epsilon_{ij}^2, \tag{25}$$

where $\mathcal{L}_b$ represents the mean squared error between training data T_{ij} and network prediction, it ensures that the network remains grounded in available empirical or simulated data, making it particularly valuable when measurements are sparse, noisy, or unevenly distributed. $\mathcal{L}_r$ is the residual loss. The weight λ_r was used to scale residual loss, N_d represents the count of network outputs, N_b is considered the number of training points provided to the network, N_e represents the residual equations to be solved, N_r is referred to as the number of residual points in the domain. The equation residual is denoted by ϵ_{ij}. Physical reasoning can be applied to impose Dirichlet conditions at specific points along the wall by introducing an additional term in the loss function in experimental campaigns. This was completed by using a set of spatial coordinates corresponding to the domain boundaries.

It was shown in [90] that PINNs could make accurate predictions for APG boundary layer problems with minimal data points. Accurate prediction of C_f (friction coefficient) and C_p (pressure coefficient) was achieved with just thirteen training points within δ_o (inside the BL) and six data points δ_{in} (velocity and Reynolds-stress), along with scattered pressure data on additional boundaries, including the wall and downstream locations.

However, the estimation of C_f can slightly degrade with less data availability. To test the network's performance without information on specific flow features such as separation, it was trained on a smaller dataset, excluding data from the separated flow region across different Re_b values. The network successfully predicted the recirculation zone extent but had issues with the estimation of the velocity magnitude within the separated region.

It was also shown that there is no significant increase in the computational cost of PINNs with increasing Reynolds numbers, making them suitable for modeling high Reynolds-number flows.

4.5. PINNs Using RANS with Reynolds Stresses; Solenoidal Forcing; Prediction Validation with Sparse Experimental Data

Sliwinski and Rigas in [91] focused on utilizing PINNs for reconstructing time-averaged quantities of an unsteady flow based on sparse velocity data. The schematic of physics-informed neural networks, which was used in [91], is shown in Figure 3. In the green box (Figure 4), network inputs are depicted, which can be either data point coordinates or collocation points. The flow variable modeled by the function and implemented through a neural network is represented by the blue box. The specific derivatives related to flow variables are denoted by the red box, computed through automatic differentiation. The residual operator is represented by the purple box, which takes the flow variable along with its derivatives to assess the physical residuals.

The RANS equations with the Reynolds stresses are incorporated in the top version of the network. On the other hand, RANS with solenoidal forcing is used by the bottom version of the network. When the network input consists of data points, the output layer is activated, and the residuals of the governing equations are not computed.

The data-assimilation framework aims to find a model function $g(x) = [g_1(x), g_2(x), \ldots, g_R(x)]^T$, represented as a neural network, by minimizing the objective function $\mathcal{L}$. This model maps the domain coordinates x and y to flow variables R.

$$\mathcal{L}(g) = \mathcal{L}_D(g) + \mathcal{L}_B(g) + \mathcal{L}_P(g), \tag{26}$$

where $\mathcal{L}_D$ represents data loss, the boundary condition loss is $\mathcal{L}_B$, and the physical loss by enforcing RANS equations is $\mathcal{L}_P$. The data loss is the mean squared error, computed at all points where flow variables are known.

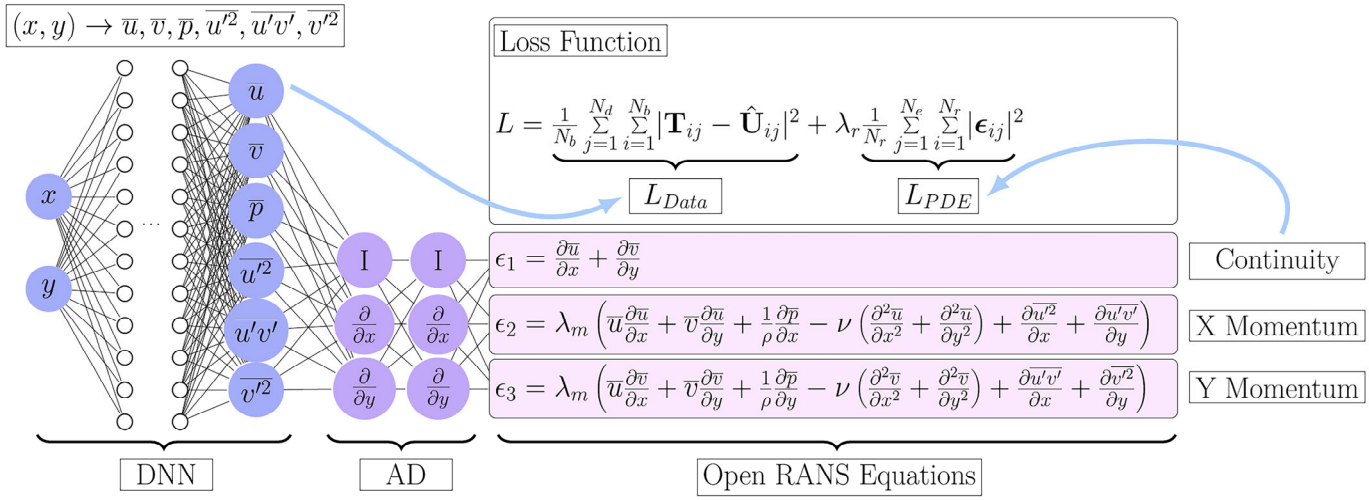

Figure 3. A PINN with embedded RANS equations [90], Network Quantities: $\mathcal{L}$—Loss, T_{ij}—Training data, $\hat{U}_{ij}$—Network predictions, ϵ_{ij}—Residual, N_b—No. data points, N_r—No. residual points, Flow Parameters: x, y—Residual coordinates, $\bar{u}$, $\bar{v}$—Velocity, $\bar{p}$—Pressure, $\overline{u'^2}$, $\overline{u'v'}$, $\overline{v'^2}$—Reynolds stresses, ρ—Density, ν-Viscosity. (Figure adopted under a Creative Commons license). The same figure was used in [92].

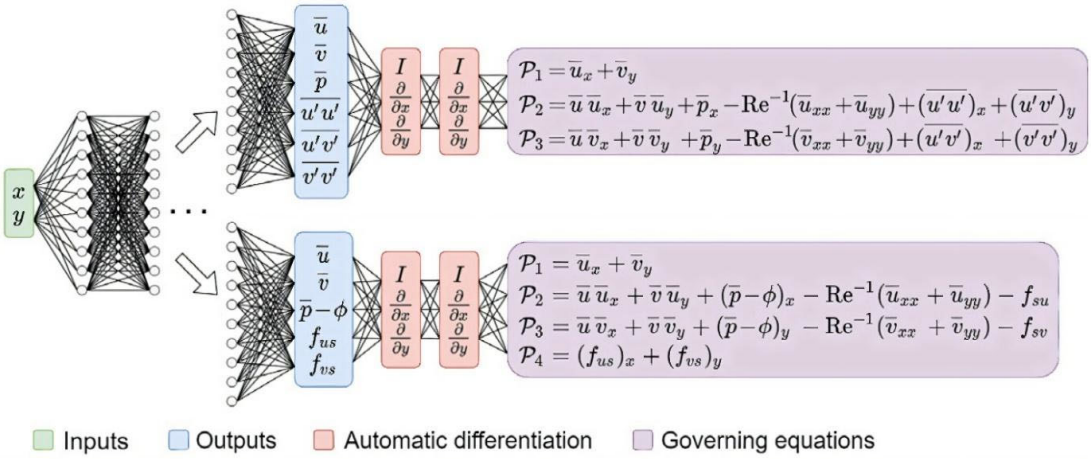

Figure 4. A diagram illustrating two different approaches of PINNs for flow reconstruction [91]. Figure adopted under a Creative Commons license.

To find some unknown closure quantities, such as the curl of unsteady RANS forcing, and to interpolate fields from sparse data [91] utilized the time-averaged velocity data. The capabilities of this technique were expanded to assimilate Reynolds stresses, while PINNs effectively incorporated the data to complete both velocity and stress fields. This process also provided insights into the pressure field.

The presented framework was validated in the context of unsteady cylinder flow at Re = 150, showing satisfactory predictions in the cylinder wake, while discrepancies were observed near the cylinder surface. An investigation into the impact of noise in the velocity data on PINN predictions revealed that it reduced the accuracy of the predicted forcing. However, PINN effectively filtered the noise from the input velocity data.

Additionally, PINNs demonstrated the ability to interpolate flows that depended on limited experimental velocity data, showcasing their potential for reconstructing flows from sparse measured data. The scope of PINN was expanded to explicitly incorporate Reynolds stresses (rather than implicitly through the forcing term), enabling the utilization of pressure measurement and second-order velocity statistics. Leveraging sparse measurements of mean Reynolds stresses and mean velocities, the method successfully

interpolated velocity and Re-stress fields for the cylinder flow, although some discrepancies remained. Incorporating pressure measurements on the cylinder surface notably enhanced interpolation accuracy. The interpolated velocity fields also enabled accurate estimation of the skin friction distribution over the cylinder surface.

4.6. PINNs Using Experimental Data with Different Spatial Sparsities

In [93], an approach to reconstruct flow fields using PINN, which incorporates known data alongside physical principles to handle imperfect data, was introduced. The circular cylinder wake flow was used as a test scenario. The study explores two types of training CFD sets: one comprises velocity data with varying levels of sparsity, while the other includes data of velocity that is missing from different regions. Training convergence was improved using a learning rate schedule, specifically demonstrating the effectiveness of the cosine-annealing algorithm.

To replicate the conditions of a real experiment, data for continuous time were given to the physics-informed neural network at specific fixed points. Specifically, 1%, 5%, 20%, and 100% of the actual velocity data were used for fitting and prediction. It is important to note that the training set included only velocity information, as the nonlinearity in the flow is driven by velocity transport.

To quantitatively compare the deviation between the original and the predicted flow field across the whole time domain, the commonly used relative norm R_{L2} was introduced:

$$R_{L2} = \frac{||\hat{U} - U||}{||U||}, \tag{27}$$

where $||\hat{U} - U||$ represents the L2 norm of the deviation between the predicted quantity (e.g., u, v, p) at a particular time, and $||U||$ is the original quantity L2 norm at that specific time. R_{L2} metric provides a reliable measure of prediction accuracy at each time step. It was found that the average R_{L2} between the predicted and actual or original flow fields at a particular time, utilizing 100% of the velocity data for training was 3.5×10^{-2} for the pressure, 1.2×10^{-2} for the spanwise velocity, and 3.9×10^{-3} for the streamwise velocity. These indicate the successful wake flow fitting by the PINN.

It was also observed that the PINN had low dependency on the amount of data. With a suitable NN architecture and efficient training strategy, precise flow predictions were achieved using only 1% of the original data. Thus, PINNs can accurately reconstruct flow fields even with relatively sparse data.

After successfully fitting the momentum equation, the partial derivative of pressure was trained as a constraint, making the PINN-computed pressure gradient consistent with the original data. However, the pressure field differed by a constant value at each snapshot since pressure information was not directly included in the training set—only its spatial gradient was used.

The agreement between the predicted and original flow fields showed that by regularly selecting a few measurement points and using data from these points to build a training set, the PINN could reconstruct the flow field with high accuracy due to its adherence to physical principles. This makes the approach a promising data assimilation method for experimental fluid dynamics. It could potentially be used with particle image velocimetry (PIV) for velocity field reconstruction, allowing PINNs to predict flow in unmeasured regions and effectively expand the coverage of PIV measurements.

4.7. PINNs Compared to RANS and URANS

In [43], precise sparse mean velocity points along with the RANS equations with data assimilation techniques were integrated. The main goal was to bridge the gap between

PINNS (data assimilation method) and the methods using traditional spatial discretization (variational methods).

A comparative analysis of both approaches was conducted using a turbulent flow case. By applying sparse data to constrain PINNs and solving the underdetermined RANS equations without full boundary conditions, superior accuracy in reconstructing the mean flow was demonstrated compared to a RANS solver employing the Spalart–Allmaras (SA) turbulence model. The following SA-augmented RANS equations were considered:

i. Continuity equation (mass conservation):

$$\frac{\partial U_i}{\partial x_i} = 0. \tag{28}$$

By penalizing deviations from zero divergence in the velocity field U_i, the model ensures that the volume of fluid is conserved throughout the domain.

ii. Momentum conservation:

$$U_j \frac{\partial U_i}{\partial x_j} + \frac{1}{\rho}\frac{\partial (P-\Phi)}{\partial x_i} - \frac{\partial \left[2(\nu+\nu_t)S_{ij}\right]}{\partial x_j} - f_{s,i} = 0. \tag{29}$$

iii. Forcing source term continuity:

$$\frac{\partial f_{s,i}}{\partial x_i} = 0. \tag{30}$$

This condition maintains solenoidal consistency for the applied forcing field $f_{s,i}$, ensuring that any synthetic or modeled body forces introduced into the system do not artificially create or destroy mass or momentum.

iv. Transport equation for turbulent viscosity:

$$U_j \frac{\partial \widetilde{\nu}}{\partial x_j} - S_p - S_{\text{diff}} - S_C - S_d = 0, \tag{31}$$

where P represents the mean pressure field, S_{ij} denotes the mean strain rate tensor, and the Reynolds forcing vector f_i was decomposed using Helmholtz decomposition into a divergence-free solenoidal, a potential part (Φ), a scalar, and a vector component ($f_{s,i}$); ν_t is an actual eddy viscosity and $\widetilde{\nu}$ is a pseudo eddy-viscosity variable. S_p represents the production, the diffusion is S_{diff}, the cross-diffusion is S_C, and S_d is the destruction term.

In PINN, the solution $\left(\hat{U}_i, \hat{P}-\hat{\Phi}, \hat{f}_{s,i}, \hat{\widetilde{\nu}}\right)(x;\theta)$ was determined by the biases and weights, θ, which characterized the neural network. The optimal set of biases and weights, $\hat{\theta}$ was obtained by minimizing a loss function $\mathcal{L}$,

$$\hat{\theta} = \mathcal{L}(U_i, P-\Phi, f_{s,i}, \widetilde{\nu}\,)(x;\theta), \tag{32}$$

$\mathcal{L}$ represents the loss function used to enforce both the boundary conditions and the governing equations. These components were sufficient for optimizing the forward problem. For the inverse problem (data assimilation), however, a measurement error term—namely J—must be included, as follows:

$$\begin{aligned}\mathcal{L} = \ & \tfrac{\lambda^D}{N_m} J + \tfrac{\lambda^P}{N_C}\sum_{k=1}^{N_C} \text{RANS}[(U_i, P-\Phi, f_{s,i}, \widetilde{\nu}\,)(x_k;\theta)]^2 + \tfrac{\lambda^P}{N_C}\sum_{k=1}^{N_C}\left(U_j\tfrac{\partial \widetilde{\nu}}{\partial x_j} - S_p - S_{diff} - S_C - S_d\right)(x_k;\theta)^2 \\ & + \tfrac{\lambda^B}{N_b}\sum_{k=1}^{N_C} \text{BC}[(U_i, P-\Phi, f_{s,i}, \widetilde{\nu}\,)(x_k;\theta)]^2 + \tfrac{\lambda^R}{2}\tfrac{1}{N_C}\sum_{k=1}^{N_C}\sum_{i=1}^{3} f_{s,i}(x_k;\theta)^2.\end{aligned} \tag{33}$$

To simplify the equation, we can break it down into distinct terms

$$\mathcal{L} = \underbrace{\frac{\lambda^D}{N_m} J}_{\text{Data term}} + \underbrace{\frac{\lambda^P}{N_C} \sum_{k=1}^{N_C} \mathrm{R}_k^2}_{\text{RANS Residual}} + \underbrace{\frac{\lambda^P}{N_C} \sum_{k=1}^{N_C} T_k^2}_{\text{Transport Residual}} + \underbrace{\frac{\lambda^B}{N_b} \sum_{k=1}^{N_C} B_k^2}_{\text{Boundary conditions}} + \underbrace{\frac{\lambda^R}{2N_C} \sum_{k=1}^{N_C} \sum_{i=1}^{3} f_{s,i}(x_k;\theta)^2}_{\text{Regularization}}, \quad (34)$$

where R_k, T_k, B_k denote the residuals of the Reynolds-averaged Navier–Stokes equations, transport, and boundary conditions at point x_k. The PINN architecture is illustrated in Figure 5.

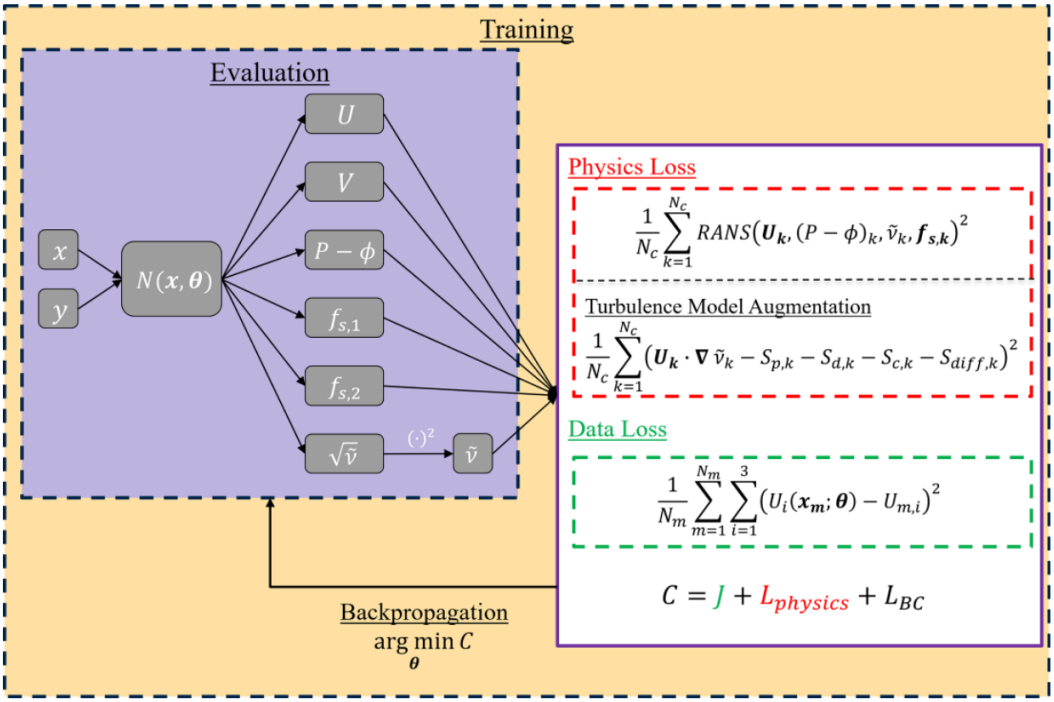

Figure 5. PINN's structure, which was used in [43]. Figure adopted under a Creative Commons license.

The governing equations were determined at N_C collocation points, while the boundary conditions were imposed at points N_b along the boundary. BC represents the error in the enforced boundary conditions, and *RANS* represents the residuals of the mean-flow equations. To compute the governing equations, the *RANS* equations gradients are precisely calculated by using automatic differentiation, which uses the chain rule to find the derivatives of all operations in the mapping $N(x;\theta)$. The physics term, the data term, and the boundary condition term are adjusted by the factors λ^P, λ^D, and λ^B, respectively. The purpose of applying additional regularization was to maintain the uniqueness of the distribution between the corrected and modeled forcing during optimization.

This hybrid approach significantly reduces mean velocity reconstruction errors by up to 73% when using coarse measurements. The incorporation of Spalart–Allmaras (SA) physics constraints prove particularly beneficial in improving reconstructions of flow in areas with pressure gradients, separation zones, and high-velocity.

The PINN-DA-SA methodology was compared to a variational data assimilation approach using the same physics constraints and sparse velocity measurements. It was shown that the PINN-DA-SA method yielded lower reconstruction errors across various data resolutions. This advantage is attributed to the avoidance of discretization errors—commonly observed in variational methodologies—which are effectively mitigated by PINNs.

The methodology was validated using high-fidelity measurements obtained directly from DNS simulations of turbulent periodic hill flow at Re = 5600.

A similar approach was used by [94]: 3D variational data assimilation of time-averaged sparse data into an unsteady Reynolds-averaged Navier–Stokes (URANS) simulation, using a stationary divergence-free forcing term within the URANS equations, was conducted. The presented approach enhances efficiency and speed by using coarse URANS simulations and the stationary discrete adjoint method for solving the time-averaged URANS equations. The data assimilation algorithms were produced utilizing OpenFOAM for URANS and adjoint problem solutions, alongside Python for gradient-based optimization.

The results in [93] have shown that assimilating sparse time-average velocity measurements enhanced both vortex dynamics prediction (shedding frequency) and facilitated correct reconstruction of the mean flow.

The effectiveness of this method was verified by applying it to turbulent flows around cylinders of different geometries for Reynolds numbers from 3000 to 22,000. The crucial role of including data points near the cylinder to improve the accuracy of vortex shedding frequency predictions was demonstrated, while additional downstream data points were essential for reconstructing the mean velocity field in the wake region.

A discrepancy term and a regularization term formed the cost function. In [94], total variation (TV) regularization was selected to decrease the ambiguity in the inverse problem by imposing a smoothness constraint on the parameter field. It was shown that in the optimization process, the regularization weight was a critical parameter.

Data-driven PINNs have rapidly emerged as a powerful approach for solving fluid dynamics problems, particularly when dealing with sparse data. In many examples, [76] and [68] found that PINNs models outperformed their CFD counterparts while requiring a fraction of the computation cost.

Similar to turbulence models, there is now a wide variety of PINN formulations available. The selection of an appropriate PINN method depends on several key factors: flow regime, boundary conditions, data availability, and computational constraints (see Table 2 for a non-exhaustive list of recent applications).

For the laminar regime [44] (NSFnets) and [76] show that PINNs can achieve accuracy comparable to traditional CFD while requiring significantly less computational memory.

Various promising approaches were also identified in the literature for situations involving turbulent flows. Eivazi et al. [38] highlight the excellent accuracy of PINN models in the turbulent case, even without providing a turbulent model.

Wang et al. [78] TF-Net leveraged multi-level spectral decomposition to predict the vortex dynamics at different scales, outperforming baseline models such as U-Net and ResNet.

Patel et al. [43] proposed a framework combining data measurements, RANS equations, and turbulence modeling, which offers greater accuracy for turbulent flows than traditional PINN methods.

Table 2. PINN examples for fluid flows.

Papers	PINN Methodology	Applications	Main Results
[38]	PINN neural network to solve the stationary two-dimensional RANS equations without any specific turbulent model. The inputs of the PINN are the spatial coordinates, and the outputs are the normal velocity, streamwise velocity, and Reynolds-stress components.	Falkner-Skan boundary layer, Adverse-pressure-gradient boundary layer, zero-pressure gradient boundary layer, turbulent flow over a periodic hill, and turbulent flow over a NACA4412 airfoil.	The PINN shows excellent ability in predicting laminar boundary layer flows (Falkner-Skan) and very good accuracy for the turbulent case.
[44]	Navier–Stokes flow nets (NSFnets) were introduced for simulating incompressible laminar and turbulent flows. Given the temporal and spatial coordinates as inputs, VP-NSFnet predicted the instantaneous velocity and pressure fields, while VV-NFSnet predicts instantaneous velocity and vorticity fields. The initial and boundary conditions are used as supervised loss, and the residuals of the Navier–Stokes equations are used for the unsupervised physics-informed loss.	2D steady Kovasznay flow, 2D unsteady cylinder wake, 3D unsteady Beltrami flow, and turbulent channel flow.	VP-NSFnet and VV-NSFnet show comparable accuracy for laminar flows.For turbulent flows, VP-NSFnet sustained turbulence at Re $\approx$ 1000 for a long period of time.
[76]	PINN for low Reynolds number flows. The network is trained without labeled data, i.e., unsupervised learning is used to minimize the residuals of Navier–Stokes equations and to satisfy initial and boundary conditions.	2D flow over a cylinder.	In terms of computational demands, the memory usage of PINN is 5–10 times smaller than the memory usage of CFD. For a small number of CFD cells, PINN takes longer to converge than the CFD solver. The number of layers is the most sensitive hyperparameter affecting the accuracy of the predictions.
[78]	Turbulent flow net (TF-Net) was introduced, which is a hybrid deep learning framework based on multi-level spectral decomposition for predicting the dynamics of different vortical structures The loss function is regularized by the continuity equation.	2D turbulent flow of the Rayleigh–Benard convection flow.	TF-net outperforms baseline models, such as U-net [95] and ResNet [34] in terms of prediction accuracy. Regularizing the loss function by penalizing the deviation from zero velocity divergence results in a more accurate model compared to the purely data-driven TF-net.

Table 2. *Cont.*

Papers	PINN Methodology	Applications	Main Results
[68]	PINN to solve forward and inverse problems for high-speed flows. The PINN predicts the density, velocity, and pressure fields for the Euler equations. The network is optimized to minimize a composite loss, integrating deviation between predicted states and ground truth values, and the residuals of the physical laws.	Forward problem: 1D Euler equation with moving contact discontinuity, 2D oblique shock wave problem Inverse problem: Sod's shock tube problem [96] and Lax's shock tube problem [97].	PINNs can approximate solutions for forward problems but are not as accurate as numerical methods for simulating high-speed flows. For inverse problems, PINNs show superior performance compared to classical methods. The clustering of collocation points around discontinuities improved the performance of the PINN.
[81]	PINN combined with the phase-field method for modeling two-phase incompressible flow. The Cahn-Hilliard equation and Navier–Stokes are encoded in the loss function of the neural network. A time-marching strategy, which consists of dividing the sampling domain into different parts, was introduced to help the network convergence.	Reversed single vortex case, bubble-rising problem in two-phase flow at a large density ratio.	The PINNs were able to capture the interface dynamics and velocity fields. The time-marching strategy employed for training the PINNs is required for obtaining accurate results.
[82]	PINN to predict 3D velocity and pressure fields using 3D temperature snapshots obtained via tomographic background-oriented Schlieren (Tomo-BOS) imaging. The loss function of the PINN contains data mismatch term as well as the residuals of the Navier–Stokes and heat transfer equations.	Buoyancy-driven flow over an espresso cup.	PINN results were validated using PIV experimental data It was shown that PINNs can handle sparse and noisy experimental data, yielding accurate and continuous descriptions of flow fields.
[86]	Physics-constrained Bayesian neural network (PC-BNN) for reconstruction of flow fields from noisy and sparse data. The violation of Navier–Stokes equations is penalized during the training of the neural network.	Vascular flow in stenosis and aneurysm geometries.	The PC-BNN model showed higher accuracy in reconstructing flow fields compared to purely data-driven methods, where the physics is not integrated.

Table 2. *Cont.*

Papers	PINN Methodology	Applications	Main Results
[87]	PINN combined with Neural Radiance Fields to handle complex fluid interactions from sparse Multi-view RGB videos. The loss function of the neural network is constrained with the residuals of Navier–Stokes equations and trained in an end-to-end optimization to learn the spatio-temporal fields without providing geometry information or boundary conditions as input.	Smoke flow involving fluid and static obstacles.	Accurate flow reconstruction for density and velocity fields reduces the need for detailed lighting or boundary conditions by using image sequences and PINN.
[88]	PINN incorporates Navier–Stokes equation and is trained using very sparse boundary data to estimate cardiovascular flow characteristics such as wall shear stress.	2D and 3D flow fields in diseased arteries (stenosed and aneurysmal) without full knowledge of boundary conditions.	PINNs can accurately predict wall shear stress from sparse velocity measurements, even in the case of unknown boundary conditions. The proposed method achieved high accuracy compared to CFD models for flow fields in stenosed and aneurysmal arteries.
[90]	PINN based on the RANS equations without using a turbulent closure model. The loss function contains a data mismatch term as well as residuals of the Navier–Stokes equations.	Adverse-pressure-gradient boundary layer and turbulent periodic hill flow.	Accurate predictions with limited training data, especially for wall shear stress and pressure predictions. Near-wall velocity gradients were poorly captured due to insufficient training data. PINN was able to predict reattachment points, even at high Reynolds numbers, for the periodic hill case.
[91]	A PINN methodology has been proposed to reconstruct time-averaged quantities of unsteady fluid flows using sparse velocity data. The training integrates sparse flow measurements with the governing RANS equations.	2D flow around a circular cylinder.	Accurate prediction of mean flow and force distribution, with small discrepancies near the cylinder. Errors were reduced by introducing inlet boundary conditions and pressure measurements. PINNs reduce noise in input velocity data and show good performance in interpolating missing data.

Table 2. *Cont.*

Papers	PINN Methodology	Applications	Main Results
[93]	PINN is employed to reconstruct high-dimensional flow fields with sparse or incomplete data. The residuals of the Navier–Stokes equations are integrated in the loss function of the network, alongside sparse velocity measurements.	2D flow around a circular cylinder.	PINNs can reconstruct velocity and pressure fields even if the data sparsity reaches 1%, where the average L2-norm error was shown to be as low as 0.0087 (compared to 0.0039 when sparsity reaches 100%). The cosine annealing learning rate scheduler reduced the number of training epochs required for convergence.
[43]	The PINN framework combined with the Spalart–Allmaras (SA) turbulence model to improve the accuracy of reconstructing mean flow fields from sparse high-fidelity measurements. The loss function of the network combines sparse data measurements, RANS equations, and the SA turbulence model.	Turbulent periodic hill flow.	Incorporating the SA turbulence model in the PINN framework improved the results by up to 73% compared to the baseline PINN model. The accuracy of PINN-RANS-SA degrades when using coarser data resolutions.

In [98], a reactive flow physics-informed neural network (RF-PINN) framework aimed at reconstruction of steady-state mean flow fields from limited particle image velocimetry (PIV) data was introduced without predicting the chemical species themselves. As illustrated in Figure 6, the method utilizes sparse data within the computational domain to accurately recover mean flow features. To reduce reliance on detailed species transport equations, an alternative strategy based on the progress variable (θ) equation was used, which included the reaction rate term. This approach streamlines the training process by minimizing the number of governing equations that act as physical constraints.

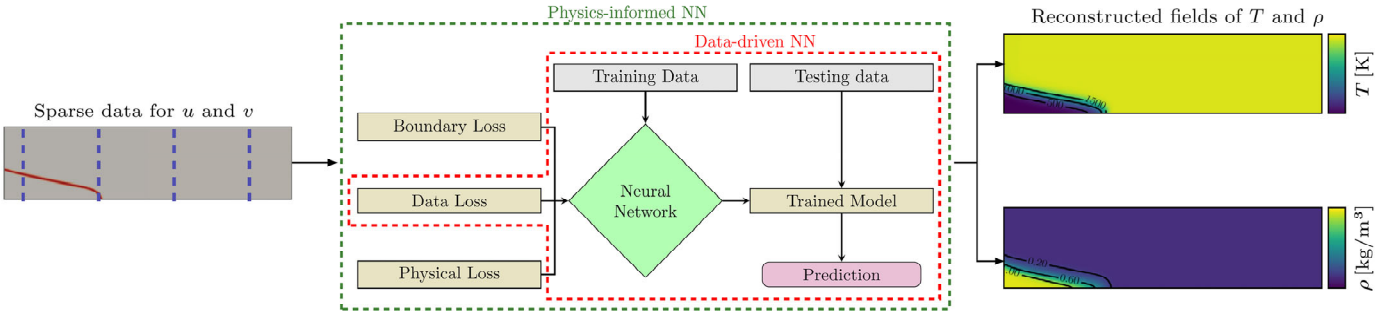

Figure 6. Reconstruction via RF-PINNs with sparse data. One way to use RF-PINNs is to reconstruct the T and ρ fields based on PIV sparse data [98]. Figure adopted under a Creative Commons license.

In cases where the boundary conditions are unknown or limited, models such as the one presented in [88] were able to predict wall shear in cardiovascular flow with higher accuracy than traditional CFD methods. Similarly, approach in [90] achieves a reliable wall shear stress approximation with limited training data. For even more challenging situations, Chu et al. [87] developed a methodology for complex fluid-structure interactions that eliminates the need for detailed boundary conditions and geometric information.

While most models perform well with large amounts of data, some specialize in handling sparse or noisy data. The model presented in [93] maintains a high level of accuracy with as little as 1% of available data across the domain. Furthermore, in [86], a physics-constrained Bayesian neural network, specifically addresses sparse as well as noisy datasets. Methodology in [91] extended these capabilities to time-averaged quantities, successfully reconstructing mean flows and forces despite the presence of noisy inlet data.

Recent studies have also focused on applying machine learning techniques to supersonic flow regimes [99], where traditional modeling approaches face significant challenges due to shock interactions, flow separation, and sparse measurements.

Deng et al. [100] introduced an intelligent reconstruction framework for combustion flow fields that integrates heuristic-guided learning (HGL) to address the challenge of sparse data environments. Rather than relying on conventional end-to-end learning approaches, the proposed HGL-MSFNN model adopts a two-stage learning strategy: in the first stage, it learns a broad range of unknown prior distributions; in the second stage, these priors guide the posterior reconstruction of the flow field, thereby enhancing both generality and generalization capacity.

The model was validated using supersonic wind tunnel experiments at free-stream Mach numbers of 2.5 and 3.0, where both flame and schlieren flow fields were captured. Notably, the system reduced pressure measurement points from 32 to just 14, demonstrating the model's effectiveness under extreme data sparsity. Comparative results indicate that the HGL-MSFNN significantly outperforms baseline models such as Sin-Transformer in reconstructing spatiotemporal dynamical fields with high accuracy and robustness.

Despite its promising performance, the model still exhibits limitations in capturing fine-scale turbulence structures, particularly those associated with multi-scale and chaotic

vortex dynamics. The authors suggest that this limitation stems from a linear imbalance in the available training features and a lack of high-quality turbulence descriptors. To address this, they propose that data augmentation strategies could be employed to improve sample diversity and enhance the model's ability to resolve smaller-scale flow features.

5. Conclusions

Physics-informed neural networks (PINNs) have significantly advanced fluid dynamics in recent years by integrating physical laws (e.g., Navier–Stokes equations) with data-driven learning. Key successes include:

- Mesh-free solving: PINNs bypass complex meshing required in traditional CFD, enabling simulations on irregular geometries and high-dimensional parametric spaces.
- Inverse problem capability: they uniquely derive unknown parameters (such as boundary conditions and turbulence model coefficients) from sparse/noisy data, which is not possible with conventional solvers.
- Data assimilation: PINNs effectively deal with sparse experimental data (e.g., particle image velocimetry, video frames) to reconstruct flow fields, even with <1% spatial coverage.
- Hybrid approaches: techniques such as re-initialization escape local minima in stiff flows, while gradient-free PINNs and U-Net++ architectures capture vortex shedding in bluff-body flows.

Despite significant progress in the application of PINNs in different fields of fluid mechanics, this approach still has some important limitations:

- High-frequency/transient dynamics: PINNs struggle with vortex shedding, shocks, or turbulence due to spectral bias (neural networks favor low-frequency features) and inadequate periodic inductive bias.
- Optimization complexity: loss landscapes for multi-scale PDEs (e.g., RANS with Reynolds stresses) induce training instability, slow convergence, and local minima.
- Theoretical gaps: rigorous error bounds and convergence guarantees for PINNs remain underdeveloped, especially for turbulent flows.
- Computational costs: training deep PINNs for 3D flows demands extensive resources, offsetting gains in online prediction speed.

Ultimately, the future of fluid mechanics depends on how well data-driven approaches can be combined with traditional physical theories. As researchers continue to improve these methods and explore new applications, PINNs have the potential to deepen our understanding and improve how we model fluid systems, both in theory and in practice. Integrating machine learning into fluid dynamics not only unlocks numerous opportunities for obtaining more accurate results but also enables engineers to develop faster and more practical solutions for real-world problems.

Author Contributions: Conceptualization, M.E.H.; methodology, M.E.H., N.B. and A.M.; writing—original draft preparation, M.E.H., A.M., P.M., M.M. and N.B.; writing—review and editing, M.E.H., A.M., P.M., M.M. and N.B.; supervision, M.E.H. and M.M.; project administration, M.E.H. and N.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No original data generated.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Brunton, S.L. Applying machine learning to study fluid mechanics. *Acta Mech. Sin.* **2021**, *37*, 1718–1726. [CrossRef]
2. Stroeva, P.; Frolova, E.; Nichik, M.; Dulin, V.; Markovich, D. Flow Dynamics of an Axisymmetric Impinging Jet under Two-Frequency External Forcing. A Study by Time-Resolved PIV and DMD. *Int. J. Heat Fluid Flow* **2023**, *103*, 109196. [CrossRef]
3. Xie, Y.; Franz, A.; Chu, M.; Thuerey, N.; Franz, E. TempoGAN: A Temporally Coherent, Volumetric GAN for Super-Resolution Fluid Flow. *ACM Trans. Graph.* **2018**, *37*, 1–15. [CrossRef]
4. Ren, F.; Hu, H.-B.; Tang, H. Active flow control using machine learning: A brief review. *J. Hydrodyn.* **2020**, *32*, 247–253. [CrossRef]
5. Li, Y.; Chang, J.; Kong, C.; Bao, W. Recent progress of machine learning in flow modeling and active flow control. *Chin. J. Aeronaut.* **2022**, *35*, 14–44. [CrossRef]
6. Renn, P.I.; Gharib, M. Machine learning for flow-informed aerodynamic control in turbulent wind conditions. *Commun. Eng.* **2022**, *1*, 45. [CrossRef]
7. Fukami, K.; Fukagata, K.; Taira, K. Super-Resolution Reconstruction of Turbulent Flows with Machine Learning. *J. Fluid Mech.* **2019**, *870*, 106–120. [CrossRef]
8. Bright, I.; Lin, G.; Kutz, J.N. Compressive Sensing Based Machine Learning Strategy for Characterizing the Flow around a Cylinder with Limited Pressure Measurements. *Phys. Fluids* **2013**, *25*, 127102. [CrossRef]
9. Su, H.; Li, Y.; Xu, Y.; Fu, X.; Liu, S. A review of deep-learning-based super-resolution: From methods to applications. *Pattern Recognit.* **2025**, *157*, 110935. [CrossRef]
10. Colvert, B.; Alsalman, M.; Kanso, E. Classifying Vortex Wakes Using Neural Networks. *Bioinspiration Biomim.* **2018**, *13*, 025003. [CrossRef] [PubMed]
11. Roxas, R., II; Evangelista, M.A.; Sombillo, J.A.; Nnabuife, S.G.; Pilario, K.E. Machine Learning Based Flow Regime Identification using Ultrasonic Doppler Data and Feature Relevance Determination. *Digit. Chem. Eng.* **2022**, *3*, 100024. [CrossRef]
12. Lui, H.F.S.; Wolf, W.R. Construction of Reduced-Order Models for Fluid Flows Using Deep Feedforward Neural Networks. *J. Fluid Mech.* **2019**, *872*, 963–994. [CrossRef]
13. Zhu, L.; Sun, X.; Liu, Y.; Zhang, W. One neural network approach for the surrogate turbulence model in transonic flows. *Acta Mech. Sin.* **2022**, *38*, 321187. [CrossRef]
14. Cuomo, S.; Di Cola, V.S.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. *J. Sci. Comput.* **2022**, *92*, 88. [CrossRef]
15. Wu, P.; Sun, J.; Chang, X.; Zhang, W.; Arcucci, R.; Guo, Y.; Pain, C.C. Data-Driven Reduced Order Model with Temporal Convolutional Neural Network. *Comput. Methods Appl. Mech. Eng.* **2020**, *360*, 112766. [CrossRef]
16. Guastoni, L.; Güemes, A.; Ianiro, A.; Discetti, S.; Schlatter, P.; Azizpour, H.; Vinuesa, R. Convolutional-network models to predict wall-bounded turbulence from wall quantities. *J. Fluid Mech.* **2021**, *928*, A27. [CrossRef]
17. Mjalled, A.; El Hassan, M.; Assoum, H.H.; Mönnigmann, M. Data-Driven Modeling Using Convolutional Neural Network for Experimental Velocity Fields of an Impinging Jet. In Proceedings of the 35th European Modeling and Simulation Symposium, Athens, Greece, 18–20 September 2023. [CrossRef]
18. Rana, P.; Weigand, T.M.; Pilkiewicz, K.R.; Mayo, M.L. A scalable convolutional neural network approach to fluid flow prediction in complex environments. *Sci. Rep.* **2024**, *14*, 23080. [CrossRef] [PubMed]
19. Maulik, R.; Lusch, B.; Balaprakash, P. Reduced-Order Modeling of Advection-Dominated Systems with Recurrent Neural Networks and Convolutional Autoencoders. *Phys. Fluids* **2021**, *33*, 037106. [CrossRef]
20. Hasegawa, K.; Fukami, K.; Murata, T.; Fukagata, K. CNN-LSTM Based Reduced Order Modeling of Two-Dimensional Unsteady Flows around a Circular Cylinder at Different Reynolds Numbers. *Fluid Dyn. Res.* **2020**, *52*, 065501. [CrossRef]
21. Xie, H.-R.; Hua, Y.; Li, Y.-B.; Aubry, N.; Wu, W.-T.; He, Y.; Peng, J.-Z. Estimation of sequential transient flow around cylinders using recurrent neural network coupled graph convolutional network. *Ocean. Eng.* **2024**, *293*, 116684. [CrossRef]
22. Fu, R.; Xiao, D.; Navon, I.; Fang, F.; Yang, L.; Wang, C.; Cheng, S. A Non-linear Non-intrusive Reduced Order Model of Fluid Flow by Auto-encoder and Self-attention Deep Learning Methods. *Int. J. Numer. Methods Eng.* **2023**, *124*, 3087–3111. [CrossRef]
23. Zhang, B. Airfoil-Based Convolutional Autoencoder and Long Short-Term Memory Neural Network for Predicting Coherent Structures Evolution around an Airfoil. *Comput. Fluids* **2023**, *258*, 105883. [CrossRef]
24. Jiang, J.; Li, G.; Jiang, Y.; Zhang, L.; Deng, X. TransCFD: A transformer-based decoder for flow field prediction. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106340. [CrossRef]
25. Mjalled, A.; El Hassan, M.; Boldocky, J.; Gulan, M.; Mönnigmann, M. Improved Neural Ordinary Differential Equation-Based Reduced Model for Impinging Jet Using Wall Shear Stress. *Phys. Fluids* 2024, *in press*. [CrossRef]
26. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-Informed Machine Learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. [CrossRef]
27. Bishop, C.M.; Nasrabadi, N.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006; Volume 4.
28. Sharma, P.; Chung, W.T.; Akoush, B.; Ihme, M. A Review of Physics-Informed Machine Learning in Fluid Mechanics. *Energies* **2023**, *16*, 2343. [CrossRef]

29. Li, X.; Sun, W.; Qin, C.; Yan, Y.; Zhang, L.; Tu, J. Evaluation of supervised machine learning regression models for CFD-based surrogate modelling in indoor airflow field reconstruction. *Build. Environ.* **2025**, *267*, 112173. [CrossRef]
30. Jagtap, A.D.; Shin, Y.; Kawaguchi, K.; Karniadakis, G.E. Deep Kronecker neural networks: A general framework for neural networks with adaptive activation functions. *Neurocomputing* **2022**, *468*, 165–180. [CrossRef]
31. Shukla, K.; Jagtap, A.D.; Karniadakis, G.E. Parallel physics-informed neural networks via domain decomposition. *J. Comput. Phys.* **2021**, *447*, 110683. [CrossRef]
32. Miotto, R.; Wang, F.; Wang, S.; Jiang, X.; Dudley, J.T. Deep learning for healthcare: Review, opportunities and challenges. *Brief. Bioinform.* **2018**, *19*, 1236–1246. [CrossRef]
33. Chai, J.; Zeng, H.; Li, A.; Ngai, E.W. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Mach. Learn. Appl.* **2021**, *6*, 100134. [CrossRef]
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; IEEE: New York, NY, USA, 2016; pp. 770–778. [CrossRef]
35. Dixit, P.; Silakari, S. Deep Learning Algorithms for Cybersecurity Applications: A Technological and Status Review. *Comput. Sci. Rev.* **2021**, *39*, 100317. [CrossRef]
36. Asudani, D.S.; Nagwani, N.K.; Singh, P. Impact of word embedding models on text analytics in deep learning environment: A review. *Artif. Intell. Rev.* **2023**, *56*, 10345–10425. [CrossRef] [PubMed]
37. Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Comput. Sci.* **2021**, *2*, 420. [CrossRef]
38. Eivazi, H.; Tahani, M.; Schlatter, P.; Vinuesa, R. Physics-Informed Neural Networks for Solving Reynolds-Averaged Navier–Stokes Equations. *Phys. Fluids* **2022**, *34*, 075117. [CrossRef]
39. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
40. Jia, L.C.; Zhu, Y.D.; Jia, Y.X.; Yuan, H.J.; Lee, C.B. Image Pre-Processing Method for near-Wall PIV Measurements over Moving Curved Interfaces. *Meas. Sci. Technol.* **2017**, *28*, 035201. [CrossRef]
41. Kähler, C.J.; Scharnowski, S.; Cierpka, C. On the Uncertainty of Digital PIV and PTV near Walls. *Exp. Fluids* **2012**, *52*, 1641–1656. [CrossRef]
42. Schlueter-Kuck, K.L.; Dabiri, J.O. Coherent Structure Colouring: Identification of Coherent Structures from Sparse Data Using Graph Theory. *J. Fluid Mech.* **2017**, *811*, 468–486. [CrossRef]
43. Patel, Y.; Mons, V.; Marquet, O.; Rigas, G. Turbulence Model Augmented Physics-Informed Neural Networks for Mean-Flow Reconstruction. *Phys. Rev. Fluids* **2024**, *9*, 034605. [CrossRef]
44. Jin, X.; Cai, S.; Li, H.; Karniadakis, G.E. NSFnets (Navier-Stokes Flow Nets): Physics-Informed Neural Networks for the Incompressible Navier-Stokes Equations. *J. Comput. Phys.* **2021**, *426*, 109951. [CrossRef]
45. Zhang, J.; Braga-Neto, U.; Gildin, E. Physics-Informed Neural Networks for Multi-Phase Flow in Porous Media Considering Dual Shocks and Interphase Solubility. *Energy Fuels* **2024**, *38*, 17781–17795. [CrossRef]
46. Wei, C.; Ooka, R. Indoor airflow field reconstruction using physics-informed neural network. *Build. Environ.* **2023**, *242*, 110563. [CrossRef]
47. Ohara, Y.; Moteki, D.; Muramatsu, S.; Hayasaka, K.; Yasuda, H. Physics-informed neural networks for inversion of river flow and geometry with shallow water model. *Phys. Fluids* **2024**, *36*, 106633. [CrossRef]
48. Cotter, S.L.; Dashti, M.; Robinson, J.C.; Stuart, A.M. Bayesian Inverse Problems for Functions and Applications to Fluid Mechanics. *Inverse Probl.* **2009**, *25*, 115008. [CrossRef]
49. Gao, X.; Wang, Y.; Overton, N.; Zupanski, M.; Tu, X. Data-assimilated computational fluid dynamics modeling of convection-diffusion-reaction problems. *J. Comput. Sci.* **2017**, *21*, 38–59. [CrossRef]
50. Di Leoni, P.C.; Mazzino, A.; Biferale, L. Synchronization to Big Data: Nudging the Navier-Stokes Equations for Data Assimilation of Turbulent Flows. *Phys. Rev. X* **2020**, *10*, 011023. [CrossRef]
51. Angriman, S.; Cobelli, P.; Mininni, P.D.; Obligado, M.; Di Leoni, P.C. Assimilation of statistical data into turbulent flows using physics-informed neural networks. *Eur. Phys. J. E* **2023**, *46*, 13. [CrossRef] [PubMed]
52. Vinuesa, R.; Brunton, S.L. Enhancing Computational Fluid Dynamics with Machine Learning. *Nat. Comput. Sci.* **2022**, *2*, 358–366. [CrossRef]
53. Kaiser, E.; Noack, B.R.; Cordier, L.; Spohn, A.; Segond, M.; Abel, M.; Daviller, G.; Östh, J.; Krajnović, S.; Niven, R.K. Cluster-Based Reduced-Order Modelling of a Mixing Layer. *J. Fluid Mech.* **2014**, *754*, 365–414. [CrossRef]
54. Buoso, S.; Manzoni, A.; Alkadhi, H.; Kurtcuoglu, V. Stabilized Reduced-Order Models for Unsteady Incompressible Flows in Three-Dimensional Parametrized Domains. *Comput. Fluids* **2022**, *246*, 105604. [CrossRef]
55. Amsallem, D.; Zahr, M.J.; Farhat, C. Nonlinear Model Order Reduction Based on Local Reduced-order Bases. *Int. J. Numer. Methods Eng.* **2012**, *92*, 891–916. [CrossRef]

56. Kadeethum, T.; Ballarin, F.; O'mAlley, D.; Choi, Y.; Bouklas, N.; Yoon, H. Reduced Order Modeling for Flow and Transport Problems with Barlow Twins Self-Supervised Learning. *Sci. Rep.* **2022**, *12*, 20654. [CrossRef] [PubMed]
57. Khademi, A.; Dufour, S. A novel discretized physics-informed neural network model applied to the Navier–Stokes equations. *Phys. Scr.* **2024**, *99*, 076016. [CrossRef]
58. Khademi, A.; Dufour, S. Physics-informed neural networks with trainable sinusoidal activation functions for approximating the solutions of the Navier-Stokes equations. *Comput. Phys. Commun.* **2025**, *314*, 109672. [CrossRef]
59. Sun, L.; Gao, H.; Pan, S.; Wang, J.-X. Surrogate Modeling for Fluid Flows Based on Physics-Constrained Deep Learning without Simulation Data. *Comput. Methods Appl. Mech. Eng.* **2020**, *361*, 112732. [CrossRef]
60. Sharma, P.; Evans, L.; Tindall, M.; Nithiarasu, P. Stiff-PDEs and Physics-Informed Neural Networks. *Arch. Comput. Methods Eng.* **2023**, *30*, 2929–2958. [CrossRef]
61. Rathore, P.; Lei, W.; Frangella, Z.; Lu, L.; Udell, M. Challenges in Training PINNs: A Loss Landscape Perspective. *arXiv* **2024**, arXiv:2402.01868. [CrossRef]
62. Raissi, M.; Yazdani, A.; Karniadakis, G.E. Hidden Fluid Mechanics: Learning Velocity and Pressure Fields from Flow Visualizations. *Science* **2020**, *367*, 1026–1030. [CrossRef]
63. McClenny, L.D.; Braga-Neto, U.M. Self-adaptive physics-informed neural networks. *J. Comput. Phys.* **2023**, *474*, 111722. [CrossRef]
64. Chen, Z.; Badrinarayanan, V.; Lee, C.-Y.; Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *PMLR, Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018*; ML Research Press: Cambridge, MA, USA, 2018; pp. 794–803. [CrossRef]
65. Jagtap, A.D.; Karniadakis, G.E. Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations. *Commun. Comput. Phys.* **2020**, *28*, 2002–2041. [CrossRef]
66. Alhubail, A.; He, X.; AlSinan, M.; Kwak, H.; Hoteit, H. Extended physics-informed neural networks for solving fluid flow problems in highly heterogeneous media. In Proceedings of the International Petroleum Technology Conference, Dhahran, Saudi Arabia, 21–23 February 2022; p. D031S073R001.
67. Jagtap, A.D.; Mao, Z.; Adams, N.; Karniadakis, G.E. Physics-informed neural networks for inverse problems in supersonic flows. *J. Comput. Phys.* **2022**, *466*, 111402. [CrossRef]
68. Kharazmi, E.; Zhang, Z.; Karniadakis, G.E. Variational Physics-Informed Neural Networks for Solving Partial Differential Equations. *arXiv* **2019**, arXiv:1912.00873. [CrossRef]
69. Kharazmi, E.; Zhang, Z.; Karniadakis, G.E. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Comput. Methods Appl. Mech. Eng.* **2021**, *374*, 113547. [CrossRef]
70. Yang, L.; Meng, X.; Karniadakis, G.E. B-PINNs: Bayesian Physics-Informed Neural Networks for Forward and Inverse PDE Problems with Noisy Data. *J. Comput. Phys.* **2021**, *425*, 109913. [CrossRef]
71. O'LEary, J.; Paulson, J.A.; Mesbah, A. Stochastic physics-informed neural ordinary differential equations. *J. Comput. Phys.* **2022**, *468*, 111466. [CrossRef]
72. Ghosh, S.; Chakraborty, A.; Brikis, G.O.; Dey, B. Using Parametric PINNs for Predicting Internal and External Turbulent Flows. *arXiv* **2024**, arXiv:2410.18917. [CrossRef]
73. Luo, S.; Vellakal, M.; Koric, S.; Kindratenko, V.; Cui, J. Parameter identification of rans turbulence model using physics-embedded neural network. In *International Conference on High Performance Computing*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 137–149.
74. Dabrowski, J.J.; Pagendam, D.E.; Hilton, J.; Sanderson, C.; MacKinlay, D.; Huston, C.; Bolt, A.; Kuhnert, P. Bayesian Physics Informed Neural Networks for data assimilation and spatio-temporal modelling of wildfires. *Spat. Stat.* **2023**, *55*, 100746. [CrossRef]
75. Wenzel, C.; Selent, B.; Kloker, M.; Rist, U. DNS of Compressible Turbulent Boundary Layers and Assessment of Data/Scaling-Law Quality. *J. Fluid Mech.* **2018**, *842*, 428–468. [CrossRef]
76. Ang, E.H.W.; Wang, G.; Ng, B.F. Physics-Informed Neural Networks for Low Reynolds Number Flows over Cylinder. *Energies* **2023**, *16*, 4558. [CrossRef]
77. Zhang, E.; Dao, M.; Karniadakis, G.E.; Suresh, S. Analyses of Internal Structures and Defects in Materials Using Physics-Informed Neural Networks. *Sci. Adv.* **2022**, *8*, eabk0644. [CrossRef]
78. Wang, R.; Kashinath, K.; Mustafa, M.; Albert, A.; Yu, R. Towards Physics-Informed Deep Learning for Turbulent Flow Prediction. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 6–10 July 2020; ACM: New York, NY, USA, 2020; pp. 1457–1466. [CrossRef]
79. Mao, Z.; Jagtap, A.D.; Karniadakis, G.E. Physics-Informed Neural Networks for High-Speed Flows. *Comput. Methods Appl. Mech. Eng.* **2020**, *360*, 112789. [CrossRef]
80. Cai, S.; Mao, Z.; Wang, Z.; Yin, M.; Karniadakis, G.E. Physics-Informed Neural Networks (PINNs) for Fluid Mechanics: A Review. *Acta Mech. Sin.* **2021**, *37*, 1727–1738. [CrossRef]

81. Qiu, R.; Huang, R.; Xiao, Y.; Wang, J.; Zhang, Z.; Yue, J.; Zeng, Z.; Wang, Y. Physics-Informed Neural Networks for Phase-Field Method in Two-Phase Flow. *Phys. Fluids* **2022**, *34*, 052109. [CrossRef]
82. Cai, S.; Wang, Z.; Fuest, F.; Jeon, Y.J.; Gray, C.; Karniadakis, G.E. Flow over an Espresso Cup: Inferring 3-D Velocity and Pressure Fields from Tomographic Background Oriented Schlieren via Physics-Informed Neural Networks. *J. Fluid Mech.* **2021**, *915*, A102. [CrossRef]
83. Berkooz, G.; Holmes, P.; Lumley, J.L. The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows. *Annu. Rev. Fluid Mech.* **1993**, *25*, 539–575. [CrossRef]
84. Ye, M.; Xu, M.; Liu, H.; Yao, W. A Method of Flow Field Reconstruction via Proper Orthogonal Decomposition. In Proceedings of the 2010 2nd International Conference on Information Engineering and Computer Science, Ternopil, Ukraine, 25–26 December 2010; IEEE: New York, NY, USA, 2010; pp. 1–4. [CrossRef]
85. Schmid, P.J. Dynamic Mode Decomposition of Numerical and Experimental Data. *J. Fluid Mech.* **2010**, *656*, 5–28. [CrossRef]
86. Sun, L.; Wang, J.-X. Physics-Constrained Bayesian Neural Network for Fluid Flow Reconstruction with Sparse and Noisy Data. *Theor. Appl. Mech. Lett.* **2020**, *10*, 161–169. [CrossRef]
87. Chu, M.; Liu, L.; Zheng, Q.; Franz, A.; Seidel, H.-P.; Theobalt, C.; Zayer, R.; Franz, E. Physics Informed Neural Fields for Smoke Reconstruction with Sparse Data. *ACM Trans. Graph.* **2022**, *41*, 1–14. [CrossRef]
88. Arzani, A.; Wang, J.-X.; D'SOuza, R.M. Uncovering Near-Wall Blood Flow from Sparse Data with Physics-Informed Neural Networks. *Phys. Fluids* **2021**, *33*, 071905. [CrossRef]
89. Ilya, L.; Frank, H. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv* **2016**, arXiv:1608.03983.
90. Hanrahan, S.; Kozul, M.; Sandberg, R. Studying Turbulent Flows with Physics-Informed Neural Networks and Sparse Data. *Int. J. Heat Fluid Flow* **2023**, *104*, 109232. [CrossRef]
91. Sliwinski, L.; Rigas, G. Mean Flow Reconstruction of Unsteady Flows Using Physics-Informed Neural Networks. *Data Centric Eng.* **2023**, *4*, e4. [CrossRef]
92. He, C.; Li, S.; Liu, Y. Data assimilation: New impetus in experimental fluid dynamics. *Exp. Fluids* **2025**, *66*, 94. [CrossRef]
93. Xu, S.; Sun, Z.; Huang, R.; Guo, D.; Yang, G.; Ju, S. A Practical Approach to Flow Field Reconstruction with Sparse or Incomplete Data through Physics Informed Neural Network. *Acta Mech. Sin.* **2023**, *39*, 322302. [CrossRef]
94. Plogmann, J.; Brenner, O.; Jenny, P. Variational Assimilation of Sparse Time-Averaged Data for Efficient Adjoint-Based Optimization of Unsteady RANS Simulations. *Comput. Methods Appl. Mech. Eng.* **2024**, *427*, 117052. [CrossRef]
95. Olaf, R.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Munich, Germany, 5–9 October 2015; pp. 234–241. [CrossRef]
96. Sod, G.A. A Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. *J. Comput. Phys.* **1978**, *27*, 1–31. [CrossRef]
97. Toro, E.F. *Riemann Solvers and Numerical Methods for Fluid Dynamics*; Springer: Berlin/Heidelberg, Germany, 2009. [CrossRef]
98. Yadav, V.; Casel, M.; Ghani, A. RF-PINNs: Reactive flow physics-informed neural networks for field reconstruction of laminar and turbulent flames using sparse data. *J. Comput. Phys.* **2025**, *524*, 113698. [CrossRef]
99. Deng, X.; Tian, Y.; Chen, E.; Yang, M.; Zhang, H.; Le, J. Research on intelligent prediction method of supersonic flow field in scramjet based on deep learning: A review. *Expert Syst. Appl.* **2025**, *279*, 127500. [CrossRef]
100. Deng, X.; Tian, Y.; Yang, M.; Chen, E.; Deng, J.; Zhang, H. Intelligent reconstruction of supersonic combustion flow fields using sparse sensor data and heuristic-guided learning. *Adv. Eng. Inform.* **2025**, *66*, 103486. [CrossRef]