Author/s:
Hanrahan, SK;Kozul, M;Sandberg, RD

Title:
Data Assimilation of Transitional and Separated Turbomachinery Flows With Physics-Informed Neural Networks

Date:
2025-11-01

Citation:
Hanrahan, S. K., Kozul, M. & Sandberg, R. D. (2025). Data Assimilation of Transitional and Separated Turbomachinery Flows With Physics-Informed Neural Networks. Journal of Turbomachinery, 147 (11), https://doi.org/10.1115/1.4068396.

Persistent Link:
https://hdl.handle.net/11343/361893

ASME Journal of Turbomachinery
Online journal at:
https://asmedigitalcollection.asme.org/turbomachinery

Data Assimilation of Transitional and Separated Turbomachinery Flows With Physics-Informed Neural Networks

Sean K. Hanrahan[1]
Department of Mechanical Engineering,
University of Melbourne,
Parkville, VIC 3010, Australia
e-mail: sean.hanrahan@unimelb.edu.au

Melissa Kozul
Department of Mechanical Engineering,
University of Melbourne,
Parkville, VIC 3010, Australia
e-mail: kozulm@unimelb.edu.au

Richard D. Sandberg
Department of Mechanical Engineering,
University of Melbourne,
Parkville, VIC 3010, Australia
e-mail: richard.sandberg@unimelb.edu.au

Despite the demonstrated utility of Reynolds-averaged Navier–Stokes (RANS) calculations for many industrially relevant problems, the method yields unsatisfactory representations of many flows of engineering interest, such as nonequilibrium turbulence, massive flow separation, coherent unsteadiness, and secondary flow features. Due to the Reynolds-averaging process, a turbulence model is required to close the RANS equations, and the simple physical arguments and approximations used in many turbulence models can cause erroneous results when applied to the flows that feature strong pressure gradients, sudden changes in mean-strain-rate, surface curvature, and turbulence anisotropy. Physics-informed neural networks (PINNs) offer a way to model aerodynamic problems without explicitly requiring turbulence closure. The network can use sparse training data and unclosed RANS equations to reconstruct the flow without a turbulence model. In this work, PINNs are applied to two problems of relevance in the turbomachinery community. First, we consider a variable area channel known as the periodic hills, which features a shear layer, a separation bubble, as well as favorable and adverse pressure gradients. Second, a PINN is applied to the T106C low-pressure turbine blade with two different levels of inlet turbulence intensity, featuring the additional challenges of transition and laminar separation. We demonstrate that PINNs are capable of modeling wall-bounded quantities such as C_f and C_p in such complex flows, capturing sensitive features such as the change in separation length when the turbulent inlet conditions are altered. This article undertakes a considered and pragmatic assessment of the state of PINNs when applied to complex high Reynolds number flows, highlighting where the method is comparable to the quality of high-fidelity simulations, and conversely where the method degrades with a lack of training data around regions of interest. [DOI: 10.1115/1.4068396]

Keywords: turbulence, transition, physics-informed neural networks, low-pressure turbine, closure modeling, machine learning

1 Introduction

Conducting analyses of turbomachinery flows using the Reynolds-averaged Navier–Stokes (RANS) equations is a cost-effective and practical method for aeroengine design. While the method is an affordable way to undertake parametric design studies, simulating a flow with RANS has its own challenges. For 2D flows, the Reynolds-averaging process leaves three unknown Reynolds-stress terms that require a model to close the RANS equations. The Reynolds-stress terms can be modeled with several different techniques, but the industry has historically preferred eddy-viscosity turbulence models due to low computational cost and simplicity. Many eddy-viscosity turbulence models can fail to capture important features such as the separation bubble length [1], as well as the correct wake loss profiles downstream of turbine blades [2]. Both quantities are crucial for understanding the performance of the blade and the interaction with the prior and subsequent stages of the engine. This poor performance is partly due to the inherent assumptions used in the turbulence model, such as the Boussinesq approximation [3] or the tuning of parameters in the transport equations to represent the production and decay of turbulent kinetic energy and dissipation rate. The Boussinesq approximation models Reynolds-stress tensor components as the sum of isotropic and anisotropic contributions [2] and is defined as

$$\overline{u_i' u_j'} = \frac{2}{3} k \delta_{ij} - 2\nu_T S_{ij} \tag{1}$$

where $\overline{u_i' u_j'}$ is a component of the Reynolds-stress tensor τ_{ij}, ν_T is the turbulent viscosity, $\overline{u}_i$ is the Reynolds-averaged velocity components, and δ_{ij} is the Kronecker delta function. The mean-strain-

[1]Corresponding author.

Contributed by the International Gas Turbine Institute (IGTI) of ASME for publication in the Journal of Turbomachinery. Manuscript received March 28, 2024; final manuscript received March 31, 2025; published online June 2, 2025. Tech. Editor: David G. Bogard.

rate tensor is defined as $S_{ij} = (1/2)((\partial \overline{u}_i/\partial x_j) + (\partial \overline{u_j}/\partial x_i))$, and the turbulent kinetic energy of the fluid is defined as $k = (1/2)(\overline{u_i' u_i'})$. The description assumes that the principle axis of the Reynolds-stress tensor is coincident with the mean strain-rate tensor for all locations in the turbulent flow [3], which is known to not hold for many flows of engineering interest. The true normalized anisotropy tensor is defined as $b_{ij} = \overline{u_i' u_j'}/2k - \delta_{ij}/3$ [3], and because of the strong nonlinearity in the Navier–Stokes equations, there are no complete mathematical descriptions that can be used for a posteriori predictions of b_{ij}. Corrections to the description of the anisotropy typically improve separation and secondary flows in RANS calculations [4,5].

In the past decade, machine learning has widely been adopted in the turbulence modeling community as a method to make meaningful improvements in certain modeling challenges [6]. Because of the unsatisfactory description of anisotropy in Eq. (1), machine learning has led to the development of improved and generalizable anisotropy tensors [4,7]. The anisotropy tensor can be added to different eddy-viscosity turbulence models, modifying the relationship in Eq. (1) to a more sophisticated explicit algebraic Reynolds-stress model. This methodology has demonstrated improvements in secondary flows in ducts [5,7], flow separation in the periodic hills [4], as well as passage and wake effects in low-pressure turbine (LPT) blades [2,8]. While the method has demonstrated notable improvements in turbulence modeling performance, it has been observed that anisotropy corrections are unable to improve all aspects of the flow around the LPT at engine-relevant operating conditions [2,8]. Rather the anisotropy expression can be trained to either represent the wake mixing downstream of the blade (the physics of which is a strain-dominated phenomenon) or to represent the near-wake region to capture the separation bubble behind the blade (which has a higher rotation rate than the wake region of the flow) [2].

Motivated by this observation, we seek to explore recent developments in physics-informed neural networks (PINNs) and demonstrate how this methodology can study time-averaged complex flow phenomena in turbomachinery-relevant applications. PINNs are a meshless partial differential equation (PDE) solver that uses a deep neural network (DNN) to act as a solution function for the problem being studied, and this DNN allows for derivatives of network predictions to be calculated with a process known as automatic differentiation. DNNs are used in this work for their unique ability to assimilate observational data to solve inverse problems, and should not be considered generalizable to any a posteriori problem. The present network architecture is a regularized DNN, as the data loss term is combined with a PDE regularization term which weakly enforces the governing equations [9]. The data loss is evaluated at locations where training data are available, and the residual (PDE) loss is evaluated at discrete locations in the domain known as residual points.

In the field of fluid dynamics, PINNs have been applied to a wide range of incompressible flows in the temporal [9–15] and time-averaged [16–18] forms of the Navier–Stokes equations. Inviscid compressible problems have also been studied by Refs. [12,19–21], with the networks typically demonstrating good shock-capturing abilities. The present work builds on the work of Eivazi et al. [16], who demonstrated that the network could infer the closure of the embedded RANS equations without a turbulence model when the network is trained on sparse data. The appeal of this method is that the flow is solved in a model-free manner (unlike a RANS calculation) as the method avoids all turbulence modeling assumptions and approximations. The methodology of Ref. [16] has been applied to a range of turbulent flows such as favorable and adverse-pressure-gradient incompressible turbulent boundary layers [16,17,22], the backward-facing step benchmark [18], and the periodic hills [16,17]. The work of Pioch et al. [18] demonstrated that using the network to infer the stress field had better performance than the k–ω turbulence model proposed by Wilcox [23] when applied to the backward-facing step flow. Furthermore, Hanrahan et al. [17] applied the methodology to sparse experimental data of the periodic hills at high Reynolds numbers, noting that the costs of the method do not necessarily scale with Reynolds numbers.

In this work, we take advantage of the inherent properties of PINNs and apply the network to two turbomachinery-related problems. First, we investigate periodic hills at $\mathrm{Re}_b = 10,595$, a problem that has relevance to variable geometry ducts and channel flows in engine designs. We place emphasis on the predicted friction coefficient (C_f) and pressure coefficient (C_p), as these distributions are valuable to engine designers. We then apply PINNs for the first time to compressible viscous fluid flows and study multiple test cases relating to the flow around a T106C turbine blade. No turbulence models are used to calculate the Reynolds-stress fields, rather the Reynolds-stresses are inferred by the network during training. For the compressible cases, the turbulent heat-flux terms are also inferred by the network to close the energy equation. The advantage of this inference is that the predicted Reynolds-stress fields have been solved in a model-free manner and have not been modeled with the Boussinesq approximation or transport equations. PINNs are still in the early stages of technological readiness and are typically applied to small problems and benchmarks. We seek to document the performance of PINNs when applied to turbomachinery-related problems and to understand whether this method could be used to assimilate sparse experimental data for specific problems where RANS calculations are known to produce erroneous predictions. This work seeks to demonstrate the current state of the art of PINNs when applied to problems of industrial relevance.

2 Method

2.1 The Physics-Informed Neural Network. The PINN architecture utilizes a DNN to act as a universal functional approximator to represent quantities relevant to the incompressible or compressible flow regimes in domain Ω (Fig. 1). Let $\mathcal{N}^D : R^{\dim_{\text{in}}} \rightarrow R^{\dim_{\text{out}}}$ represents a neural network of D layers which consists of an input layer, $D-1$ hidden layer, and a linear output layer. $\mathbf{x} \in R^{\dim_{\text{in}}}$ are the input coordinates of the network. An intermediate layer of the network is described as

$$\mathcal{N}^d(\mathbf{x}) = \sigma\left(\mathbf{W}^d \mathcal{N}^{d-1}(\mathbf{x}) + \boldsymbol{\phi}^d\right) \in R^{\mathcal{N}^d} \tag{2}$$

where the layer-wise weight matrix is represented by $\mathbf{W}^d$, and the layer-wise bias vector is represented by $\boldsymbol{\phi}^d$. $\mathcal{N}^d$ is the layer-wise output vector, and σ is the nonlinear activation function for each neuron in the network. For the incompressible case, $\sigma = \tanh(\cdot)$ is used as the activation function with a fixed gradient, but for the compressible cases, this was modified to be a layer-wise locally adaptive activation function (L-LAAF) to improve the learning efficiency of the network [24]. For this modification, a trainable layer-wise gradient scaling term a^d is included in each layer of the DNN architecture so that

$$\mathcal{N}^d(\mathbf{x}) = \sigma\left(n \times a^d\left(\mathbf{W}^d \mathcal{N}^{d-1} + \boldsymbol{\phi}^d\right)\right) \in R^{\mathcal{N}^d} \tag{3}$$

For the L-LAAF architecture, the activation function $\sigma = \tanh(\cdot)$ is still used, and the constant $n = 5$ is used to adjust the sensitivity of the function, which was informed by prior work on the topic [25].

The key objective of using a PINN is to represent a surrogate model $\hat{\mathbf{U}}$ of a hidden solution $\mathbf{U}_h$, so that

$$\hat{\mathbf{U}} - \mathbf{U}_h \approx 0 \in \Omega \quad \text{and} \quad \hat{\mathbf{U}} \approx g \in \partial\Omega \tag{4}$$

are satisfied. The output layer $\mathcal{N}^D$ is the prediction of state $\hat{\mathbf{U}}$, and g represents a prescribed condition on the domain boundary $\partial\Omega$. Automatic differentiation is used to determine the symbolic derivatives of the network [26], with respect to a set of residual points (a

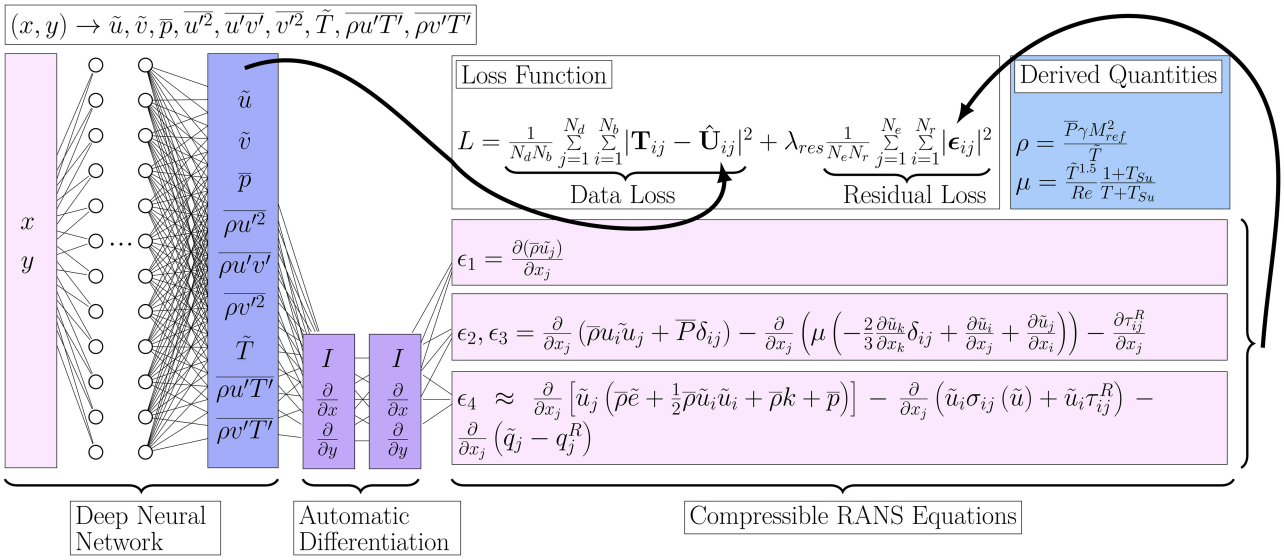

Fig. 1 The PINN architecture with Favre-averaged Navier–Stokes equations and a simple loss function

set of spatial coordinates $\mathbf{x}$ within Ω). The network predictions and their associated derivatives are substituted into the relevant governing equations, and the residual loss L_{res} is calculated for each residual point. L_{res} contributes to the loss function L, which is used to optimize the network during training, and further details regarding the formulation of the loss function are discussed in Secs. 2.2 and 2.3.

2.2 Incompressible Flow Equations. The network outputs are $\overline{u}$, $\overline{v}$, $\overline{p}$, $\overline{u'^2}$, $\overline{u'v'}$, $\overline{v'^2}$, and the governing equations are continuity, and the unclosed 2D x and y momentum equations, which are

$$\varepsilon_1 = \frac{\partial \overline{u}}{\partial x} + \frac{\partial \overline{v}}{\partial y} \tag{5a}$$

$$\varepsilon_2 = \overline{u}\frac{\partial \overline{u}}{\partial x} + \overline{v}\frac{\partial \overline{u}}{\partial y} + \frac{1}{\rho}\frac{\partial \overline{p}}{\partial x} - \nu\left(\frac{\partial^2 \overline{u}}{\partial x^2} + \frac{\partial^2 \overline{u}}{\partial y^2}\right) + \frac{\partial \overline{u'^2}}{\partial x} + \frac{\partial \overline{u'v'}}{\partial y} \tag{5b}$$

$$\varepsilon_3 = \overline{u}\frac{\partial \overline{v}}{\partial x} + \overline{v}\frac{\partial \overline{v}}{\partial y} + \frac{1}{\rho}\frac{\partial \overline{p}}{\partial y} - \nu\left(\frac{\partial^2 \overline{v}}{\partial x^2} + \frac{\partial^2 \overline{v}}{\partial y^2}\right) + \frac{\partial \overline{u'v'}}{\partial x} + \frac{\partial \overline{v'^2}}{\partial y} \tag{5c}$$

The network is trained on loss function L which is minimized by the optimization algorithm during training, and this is defined as

$$L = \lambda_{\text{data}} L_{\text{data}} + \lambda_{bc} L_{bc} + \lambda_{pbc} L_{pbc} + \lambda_{\text{res}} L_{\text{res}} \tag{6}$$

The loss function consists of terms that relate to relevant components of the optimization problem, such as the data loss L_{data}, the residual loss L_{res}, the Dirichlet boundary conditions loss L_{bc}, and the periodic boundary condition (PBC) loss L_{pbc}. All loss components are scaled by an appropriate weighting λ, and this is used to increase or decrease the representation of the term in the loss function. The training data loss term L_{data} is defined as

$$L_{\text{data}} = \frac{1}{N_d N_b} \sum_{j=1}^{N_d} \sum_{i=1}^{N_b} |\mathbf{T}_{ij} - \hat{\mathbf{U}}_{ij}|^2 \tag{7}$$

where $\mathbf{T}_{ij}$ is the supervised training data matrix, N_b is the number of training data points, and N_d is the number of predicted quantities of the network. The loss of the Dirichlet boundary conditions L_{bc} is also calculated in the same manner as Eq. (7). The periodic boundary conditions are solved with the difference between the network predictions at the inlet ($\mathbf{x}_{\text{in}}$) and the outlet ($\mathbf{x}_{\text{out}}$) as

$$L_{pbc} = \frac{1}{N_d N_{pbc}} \sum_{j=1}^{N_d} \sum_{i=1}^{N_{pbc}} |\hat{\mathbf{U}}_{ij}(\mathbf{x}_{\text{in}}) - \hat{\mathbf{U}}_{ij}(\mathbf{x}_{\text{out}})|^2 \tag{8}$$

where N_{pbc} is the number of residual points along the PBC interface. Finally, the residual loss term L_{res} is defined as

$$L_{\text{res}} = \frac{1}{N_e N_r} \sum_{j=1}^{N_e} \sum_{i=1}^{N_r} |\boldsymbol{\epsilon}_{ij}|^2 \tag{9}$$

where N_r is the number of residual points in the domain Ω, and N_e is the number of governing equations in the problem.

The training data are normalized by the absolute maximum value of each quantity in $\mathbf{T}_{ij}$, and during runtime, Eq. (7) was solved with normalized $\mathbf{T}_{ij}$ and $\hat{\mathbf{U}}_{ij}$. The residual loss in Eq. (9) is solved in quantities with the magnitude of the original dataset before normalization.

2.3 Compressible Flow Equations. The network outputs are $\tilde{u}$, $\tilde{v}$, $\overline{p}$, $-\overline{\rho u'^2}$, $-\overline{\rho u'v'}$, $-\overline{\rho v'^2}$, $\log_e(\tilde{T})$, $\overline{\rho u'T'}$, $\overline{\rho v'T'}$, and the PINN is regularized with the Favre-averaged Navier–Stokes equations, which are written as

$$\epsilon_1 = \frac{\partial \overline{\rho}\tilde{u}_j}{\partial x_j} \tag{10a}$$

$$\epsilon_2, \epsilon_3 = \frac{\partial}{\partial x_j}\left(\overline{\rho}\tilde{u}_i\tilde{u}_j + \overline{p}\delta_{ij}\right) - \frac{\partial}{\partial x_j}\left(\sigma_{ij}(\tilde{u})\right) + \frac{\partial}{\partial x_j}\left(\tau_{ij}^R\right) \tag{10b}$$

$$\epsilon_4 = \frac{\partial}{\partial x_j}\left[\tilde{u}_j\left(\overline{\rho}\tilde{e} + \frac{1}{2}\overline{\rho}\tilde{u}_i\tilde{u}_i + \overline{\rho}k + \overline{p}\right)\right] - \frac{\partial}{\partial x_j}\left(\tilde{u}_i\sigma_{ij}(\tilde{u}) + \tilde{u}_i\tau_{ij}^R\right) - \frac{\partial}{\partial x_j}\left(\tilde{q}_j - q_j^R\right) - \frac{\partial}{\partial x_j}(\mathcal{F}_j) - \frac{\partial}{\partial x_j}(\mathcal{V}) \tag{10c}$$

The Favre-averaged viscous stress tensor and the Reynolds-stress tensor are described, respectively, as

$$\sigma_{ij}(\tilde{u}) = \mu\left(-\frac{2}{3}\frac{\partial \tilde{u}_k}{\partial x_k}\delta_{ij} + \frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) \tag{11}$$

$$\tau_{ij}^R = -\overline{\rho}\left(\widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j\right) = \overline{-\rho u_i' u_j'} \tag{12}$$

The turbulent heat-flux tensor q_j^R, $\mathcal{V}$ (effects of viscosity on the velocity fluctuations), and $\mathcal{F}$ (the total flux of the turbulent kinetic energy) are defined as

$$q_j^R = \overline{\rho} C_p \left(\widetilde{T u_j} - \tilde{T}\tilde{u}_j\right) = \overline{\rho u_i' T'} \tag{13}$$

$$\mathcal{V} = \overline{u_i''}\sigma_{ij}(\tilde{u}) + \tilde{u}_i \sigma_{ij}\left(\overline{u''}\right) \tag{14}$$

$$\mathcal{F}_j = -\overline{\frac{1}{2}\rho u_j'' u_i'' u_i''} - \overline{p'' u_i''}\delta_{ij} + \overline{u_i'' \sigma_{ij}(u'')} \tag{15}$$

$\mathcal{V}$ and $\mathcal{F}$ have been neglected from the energy equation as these effects are considered negligible for the flow around the LPT. The equation of state and Sutherland's law have been used to calculate $\overline{\rho}$ and the kinematic viscosity μ in the fluid, and these are described as

$$\overline{\rho} = \frac{\overline{P}\gamma M_{\text{ref}}^2}{\tilde{T}} \tag{16}$$

$$\mu = \frac{\tilde{T}^{1.5}}{\text{Re}_{\text{Ref}}} \frac{1 + T_{\text{Su}}}{\tilde{T} + T_{\text{Su}}} \tag{17}$$

where γ is the specific heat ratio, M_{ref} is the reference Mach number, and T_{Su} is Sutherland's temperature. The energy equation (10*c*) is closed with the turbulent heat-flux terms $\overline{\rho u' T'}$ and $\overline{\rho v' T'}$ which are predicted by the network in the same way as the Reynolds-stresses, avoiding modeling assumptions. The loss function L for the compressible cases is defined as

$$L = L_{\text{data}} + \lambda_{\text{res}} L_{\text{res}} + \lambda_{sr} L_{sr} \tag{18}$$

where a slope recovery term L_{sr} is used to increase the gradients of the L-LAAF when possible during training [24]. This is achieved by forcing the network to increase a^d via minimizing

$$L_{sr} = \frac{1}{1/(D-1)\sum_{i=1}^{D-1} e^{a^i}} \tag{19}$$

in the loss function. A periodic boundary condition in the pitchwise direction was not implemented for the LPT case, rather the pitchwise boundary was enforced in the loss function with training data as defined in Eq. (18).

The compressible data are normalized with min–max scaling, and the normalized network predictions were only used for calculating Eq. (6), meaning that the governing equations are solved with quantities with original magnitudes. In addition to this, the $\tilde{T}$ field is forced to be positive to ensure stability during training. This was achieved by applying an exponential function to the network prediction (i.e., $e^{\log_e(\hat{\tilde{T}})} = \tilde{T}$) when the predictions need to be in the original magnitudes. This method has been utilized in other works to enforce positive values of $\overline{P}$ and $\overline{\rho}$ [19,20]. For simplicity in Fig. 1, this feature is just described as $\tilde{T}$.

3 Periodic Hills at $\text{Re}_b = 10, 595$

3.1 Numerical Setup. The periodic hills are a common benchmark in the turbulence modeling community, as they capture many of the features that are challenging to simulate with RANS and historic large eddy simulation (LES) attempts [4,27]. The benchmark is an incompressible channel flow with a symmetric and periodic converging–diverging bottom wall, a feature that is often described as a "periodic hill." There is a flat extent between the two hills as shown in Fig. 2. The 2D computational domain is normalized by the hill height h, with the stream-wise length $L_x = 9h$ and the wall-normal length $L_y = 3.035h$. The hills constrict the cross-sectional area of the flow by 33%. This restriction in the cross section of the flow causes a favorable-pressure-gradient to form on the windward side of the hill and an adverse-pressure-gradient on the leeward side of the hill. The hill is a smooth curved surface, with flow separation occurring on the leeward side of the hill. Reattachment occurs on the flat plate between the hills, and inter-hill distance allows the flow to recover away from the acceleration due to the convergent region of the channel [27].

The residual points are extracted from the LES mesh of Ref. [27], which is stretched away from the bottom wall with an expansion ratio of 1.05. The top wall of the channel has a coarse resolution, as this region was not of interest for the original LES campaign. The mesh has 129 residual points in the wall-normal direction and 197 points in the stream-wise direction. The profiles of velocity and Reynolds-stress from Ref. [27] have been used as "sensor data" for this problem, which are located at $x/h = [0.05, 0.5, 1, 2, 3, 4, 5, 6, 7, 8]$. For this case, the network will predict the pressure field from a hidden state, as no pressure data are given in these profiles. Dirichlet boundary conditions are enforced at 3600 discrete points located at the top and bottom walls of the channel. The PBC is enforced for all prediction quantities in the network including pressure and is applied at 2000 equidistant discrete points along the boundary at $x/h = 0$ and $x/h = 9$. The network is trained with the Adam optimizer [28] for 1000 epochs with a learning rate of 0.001—where an epoch is one training cycle of the network. The limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) optimizer [29] is used for the remaining 50, 000 epochs of training. Adam is a first-order gradient descent method with variable learning rate step sizes to accelerate learning [30], and L-BFGS is a second-order quasi-Newton method that uses an approximation of the Hessian of the loss function [30]. Adam is initially used in training, as the method has robust training performance and can avoid being caught in local loss function minima that are typically found in early training. L-BFGS is known to get caught in these minima, and this optimization method is used as a second step of training for its superior convergence performance. All calculations in the network are made at double network precision.

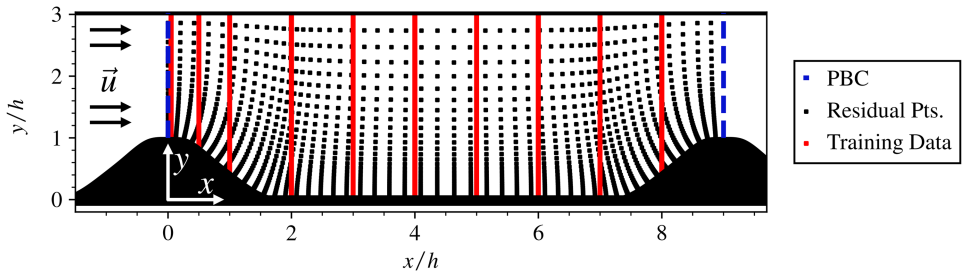

Fig. 2 The numerical setup of the periodic hills, with the residual points of the domain decreased by a factor of 16 for visualization purposes

Table 1 A summary of the numerical setup and run time for the compressible and incompressible cases

Name	Net.	Res. Pts.	Act. Func.	Runtime (h)[a]
PH	80	25, 413	Fixed	1.70
Wake	60	29, 450	L-LAAF	8.19
BL	60	64, 000	L-LAAF	22.43
TI = 0.0%	60	134, 921	L-LAAF	31.05
TI = 0.9%	60	134, 921	L-LAAF	31.01

Note: [a]The reported runtime for all compressible cases included additional scripts for monitoring the convergence of the PINN. This added approximately 1 h to the reported runtime of a full domain case.

3.2 Results and Discussion. The network has several parameters that can affect the prediction capability of the network, and this can be better understood with a network hyperparameter study. For this study, λ_{res} was varied between 0.1 and 10 with the other weightings fixed at unity ($\lambda_{bc} = \lambda_{pbc} = \lambda_{\text{data}} = 1.0$). The network width w was varied between 20 and 100 neurons, and eight layers were used in all runs. $\lambda_{\text{res}} = 0.1$ and $w = 80$ yield excellent quality predictions, and these quantities are used for the remainder of this study. To acknowledge and discuss the robustness of PINNs with considerations to the overfitting of network hyperparameters, a detailed discussion has been included in Appendix A. Training for this network architecture was conducted on a single 80 GB Nvidia A100 graphics processing unit, with each instance of the network taking 1.70 h to complete training. All PINNs runtimes are shown in Table 1, with the periodic hills parameters denoted as *PH*.

It can be a challenge to assess sensitive network predictions such as C_f, and this is because the training of a DNN is a stochastic process where each run of the network will have minor differences in the predictions. We use ensemble averaging to overcome this by taking the mean and standard deviation of ten network models to provide a simple statistical analysis of network convergence in regions with sensitive quantities.

The first observation from the hyperparameter study is that decreasing λ_{res} decreases the mean prediction error of all quantities. The relaxation of the governing equations in the early stages of training allows the network to prioritize fitting the training data. This in turn allows the residual loss to become a more prominent feature of the loss function in the later stages of training. The second observation from the hyperparameter study is that larger network widths are associated with smaller standard deviations in the ensemble-averaged quantities. A wider network of the same depth is more expressive than a narrow network. Generally speaking, a more expressive network requires fewer iterations to fit the solution space as there are more degrees-of-freedom. This means that for a given amount of training, the wide networks will not need to prioritize (and compromise) predicted quantities when moving toward an optimal solution.

The network predictions of the velocities, and Reynolds-stresses are of excellent quality as shown in the mean error plots of Fig. 3, and only regions near the wall and the boundary of the separation bubble have errors greater than 5%. These errors may be spurious due to very small quantities in some of the regions in the separation bubble and along the wall. The absolute error

$$\varepsilon = 100 \times \frac{|\mathbf{U}_h - \hat{\mathbf{U}}|}{|\mathbf{U}_h|} \tag{20}$$

is calculated at every point in the domain. As the pressure field is only accurate up to a constant when trained as a hidden state (i.e., no training data provided) [10], the pressure field is scaled by

$$\Delta\overline{p} = \overline{p} - \overline{p}_{\text{wall, inlet}} \tag{21}$$

before calculating the absolute error. The $\overline{p}$ error field is sensitive λ_{res}, and the magnitude of the $\varepsilon > 5\%$ region on the leeward side of the hill tended to decrease with smaller λ_{res}. The profiles of the pressure have the same shape as the reference quantities throughout the domain, and the focus of this work is on modeling the quantities of the wall.

The ensemble average and the standard deviation of the network predictions are shown in Fig. 4. All velocity and Reynolds-stress profiles $0.05 < x/h < 8$ are of excellent quality, and this clearly demonstrates the capabilities of PINNs as the network predictions are comparable to the velocity and Reynolds-stress profiles of a highly resolved LES study. It is encouraging to see the high-quality predictions of the Reynolds-stress fields when the network is solving the RANS equations without turbulence closure. There are several differences between the network predictions and reference data in Figs. 3 and 4. The artifact on the predicted $\overline{u}$ profile at $2.8 < y/h < 3$ is due to a cubic interpolation of the network predictions in a region with sparse information. Close to the top wall of the channel, oscillations can be observed in the $\overline{v}$ error field of Fig. 3, and this is due to small oscillations found in the reference dataset. The network solutions filter out the oscillations as shown in Fig. 4, and this is an indication of a well-fitted network.

The predictions of $\overline{u'^2}$, $\overline{u'v'}$, and $\overline{v'^2}$ become erroneous close to the wall in the region of $8 \leq x/h \leq 9$, and the velocity profiles in the boundary layer are over-predicted (Fig. 4). The PINN ensemble velocity predictions have a low standard deviation in the wall-normal direction, yet the C_f curve in Fig. 5 indicates a region of higher standard deviation in the region. In our work, we have observed that the standard deviations of network predictions tend to increase when the network is not provided with enough data in a region where the physics of the flow is rapidly changing. In this region in the periodic hills problem, the boundary layer thins due to the acceleration of the flow [27], and after $x/h > 7$, a redistribution occurs from the stream-wise component of the Reynolds-stress

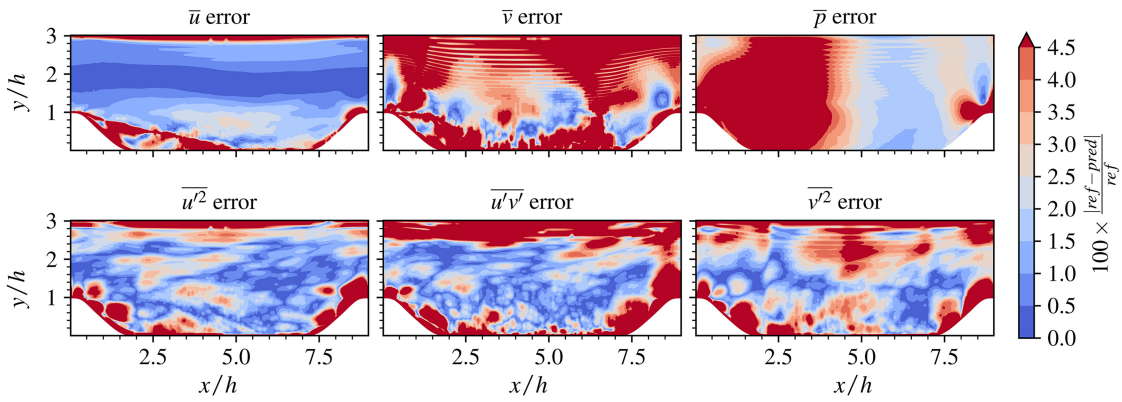

Fig. 3 The mean ensemble error fields of the periodic hills. The error is limited to 5% for visualization purposes.

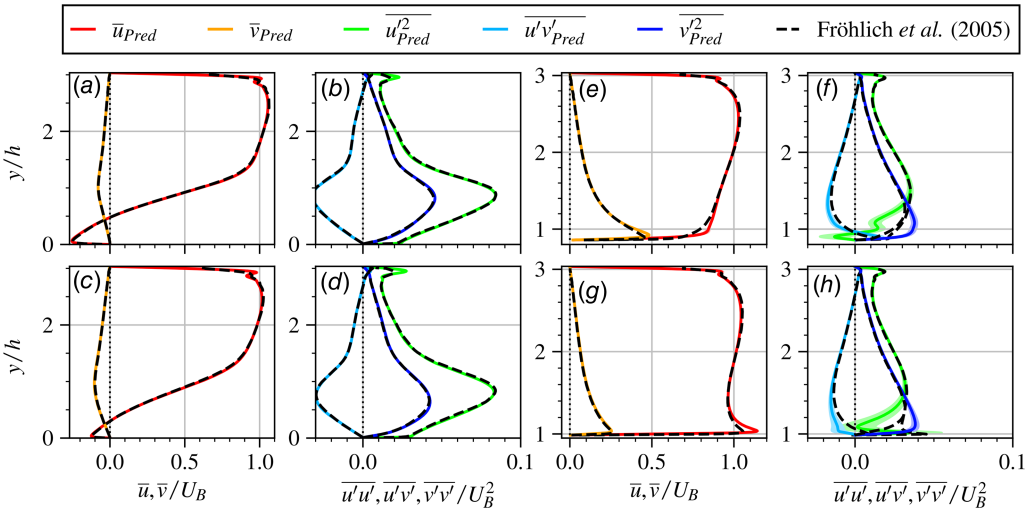

Fig. 4 A summary of the profiles of the periodic hills at ((*a*) and (*b*)) $x/h = 2.5$, ((*c*) and (*d*)) $x/h = 3.5$, ((*e*) and (*f*)) $x/h = 8.5$, and ((*g*) and (*h*)) $x/h = 8.75$. The profiles represent the mean ensemble value, and the banded region represents the standard deviation of the ensemble.

tensor to the wall-normal and spanwise stress components, through the pressure-strain mechanism [27]. The network predictions of the PBC at $x/h = 9$ compare well with the reference data of Frölich et al. [27]. This has led us to the conclusion that the network predictions of the boundary layer have degraded due to a lack of training data in a region of strong acceleration.

The curves of skin-friction C_f of the prediction and the reference case are shown in Fig. 5, and this is defined as

$$C_f = \frac{2(\partial \boldsymbol{u}/\partial y)_{\text{wall}}}{U_b \text{Re}_b} \tag{22}$$

where $(\partial \boldsymbol{u}/\partial y)_{\text{wall}}$ is a linear approximation of the wall tangential velocity gradient. The network has predicted the separation point and the reattachment point remarkably well considering that these are regions where the accuracy of RANS simulations traditionally becomes erroneous [31]. There are two regions where the PINN makes incorrect predictions of the C_f curve: downstream of the separation point and at the windward side of the hill. Downstream of the separation point, the PINN predicts a brief reattachment and separation (i.e., $C_f > 0$), which was not observed in the original study [27]. At $x/h > 8$, the network was unable to predict the peak magnitude of C_f, and this is due to the aforementioned lack of training data in this region.

The ensemble averaging of the network predictions provides some quantification of the reliability of the network predictions of the C_f curve. The mean prediction minimizes the stochastic effects from training and the noise along the boundary of the domain. The current formulation of PINNs is nonconservative [21], and this effect is observed along the wall of the domain. The ensemble average of larger network widths is found to improve the C_f prediction and decrease the associated standard deviations in the predictions as well; however, it did not offer substantial improvements after $x/h > 8$.

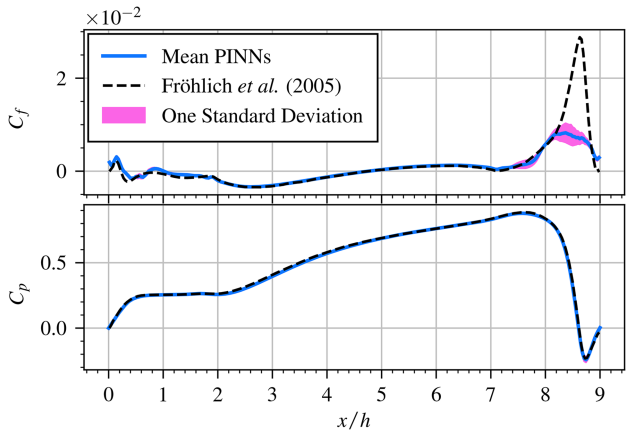

Fig. 5 The C_f and C_p on the bottom wall of the periodic hills

4 T106C Low-Pressure Turbine

The T106C LPT is a high lift design that is prone to forming separation bubbles on the aft portion of the suction side of the blade, due to the large pressure gradients and low levels of turbulent structures surrounding the blade when operating at cruising engine conditions (lower Re). However, the separation bubble is suppressed when operating at take-off conditions (higher Re). The reference data come from the highly resolved large eddy simulation of Rosenzweig et al. [32], which was run on HiPSTAR, a high-performance finite difference solver [33]. The blade has an angle of attack of 34 deg at $\text{Re}_{is} = 80,000$ and $\text{Ma}_{is} = 0.65$ based on isentropic exit conditions. The statistics are processed after 55 convective time units of the simulation, where a convective time unit is defined as the blade axial chord length divided by the streamwise velocity component at the outlet of the domain. The dataset includes two different inlet turbulent intensities (TI), which are TI = 0.0% and TI = 0.9%. We seek to understand if the PINN can model sensitive features such as the shift in the separation point due to the changes in TI. This problem is an excellent test case for PINNs, as the problem includes compressibility effects and flow features that typically require a turbulence model, a transition model, and a heat-flux model to run the compressible RANS calculation.

4.1 Numerical Setup. The residual points of the problem are extracted from the mesh of Ref. [32], which consists of a background h-grid mesh and an o-grid mesh surrounding the blade. The o-grid mesh consists of 80 points in the wall-normal direction and 800 points around the blade, and the h-grid mesh consists of 190 points in the y-coordinate and 468 points in the x-coordinate. The compressible PINN is applied to four different problem configurations, investigating the wake region, the boundary layer around the blade, the full domain with a TI = 0.0%, and a full domain with TI = 0.9%. A diagram of the different test cases is shown in Fig. 6. We use a PINN to model the region downstream of the blade to see whether improved wake loss profiles can be obtained. RANS simulations with the k–ω shear stress transport and Spalart–Allmaras

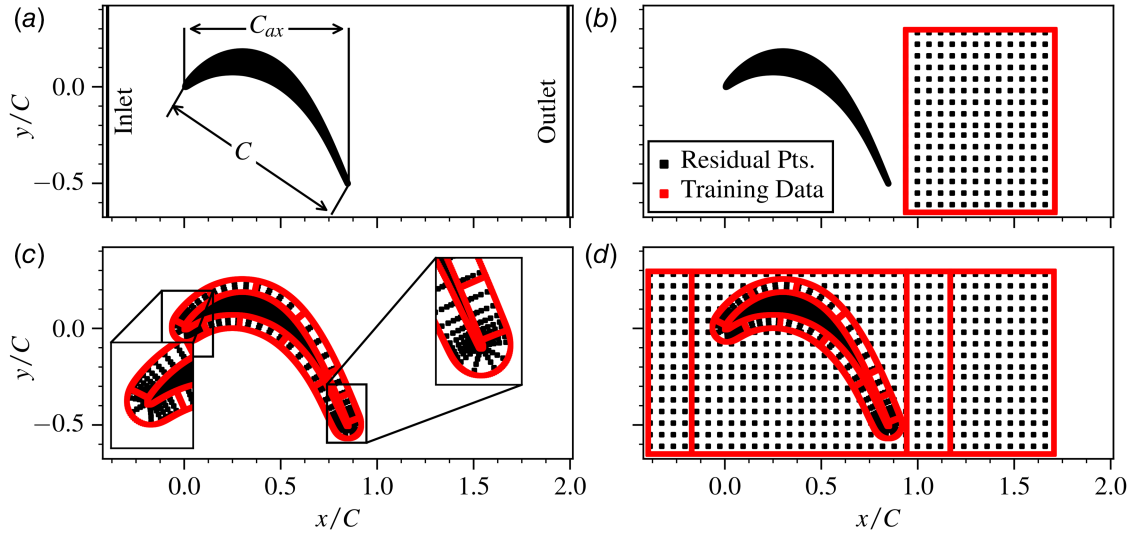

Fig. 6 The numerical setup of the T106C cascade. Numerical grid was reduced by a factor of 144 for visualization. (*a*) The problem schematic, (*b*) the wake test case, (c) the boundary layer test case, and (*d*) the full domain test case.

turbulence models typically over-predict the peak value in the wake loss profiles and also underestimate the width of the profiles [2]. This test case uses a subsection of the reference domain, between $0.94 \leq x/C \leq 1.71$ (which is $0.1 \leq x/C_{ax} \leq 1$ downstream of the trailing edge), and only uses training data around the perimeter of the test domain, as shown in Fig. 6(*b*). Next, to understand the network's abilities to model the boundary layer and laminar-to-turbulent transition effects around the blade, the o-grid is used as a second test case. The training data in this case are the data on the surface of the blade, the outer region of the o-grid, as well as 11 wall-normal profiles of data located around the blade. The boundary layer model is shown as the second test case in Fig. 6(*c*). The final two test cases use a larger domain size as shown in Fig. 6(*d*). This setup combines the training data of the previous boundary layer and wake test cases, along with profiles at $x/C = -0.4, -0.17, 0.94, 1.17, 1.71$. These test cases are run at TI = 0.0% and TI = 0.9%, which allows a comparison between the C_f curves around the blade to understand how PINNs respond to the shift in the separation point on the blade.

For each test case, the network is provided with $\tilde{u}$, $\tilde{v}$, $\overline{p}$, $\overline{-\rho u'^2}$, $\overline{-\rho u'v'}$, $\overline{-\rho v'^2}$, $\log_e(\tilde{T})$, $\overline{\rho u'T'}$, $\overline{\rho v'T'}$ at each location of training data. All compressible PINN models are trained with the Adam optimizer for 50, 000 epochs at $Lr = 0.0005$, and the L-BFGS optimizer for a further 127, 000 epochs. Eight hidden layers are used for all compressible PINN test cases, and further details about the numerical setup are summarized in Table 1, where the wake model is denoted as *wake*, the boundary layer model is denoted as *BL*, and the two models of the complete domain are denoted as TI = 0.0% and TI = 0.9%. All networks are trained with double precision. All results plotted in this article come from a network ensemble; with the wake and boundary layer cases consisting of five runs of the PINN and the full domain T106C cases consisting of ten PINN predictions consisting of ten network predictions.

Training the PINN is significantly more computationally expensive for the compressible flow when compared to the periodic hills. This is due to the large number of residual points in the problem and the larger network sizes required to represent the additional predicted quantities. Undertaking a hyperparameter study for this test case was more challenging due to the high computational costs, and only one run of the network could be undertaken for each hyperparameter. The network required more epochs of training, and this may be due to the inclusion of the additional heat-flux terms and the energy equation, or the calculations of $\overline{\rho}$ and $\overline{\mu}$ during training. The hyperparameter study was undertaken by assessing the loss of the o-grid and h-grid meshes separately. It was observed that the mean square error of the h-grid mesh was relatively stable with network widths of $w \geq 60$ neurons. The o-grid mesh demonstrated different convergence behavior, where the larger network sizes continued to improve the network prediction quality around the blade. Due to computational cost, the hyperparameter study was limited to a network width of $w = 60$ neurons wide. The hyperparameter study identified that $\lambda_{\text{data}} = 1.0$, $\lambda_{\text{res}} = 0.001$, and $\lambda_{sr} = 0.0001$ were suitable for quality network predictions. A discussion of the network training dynamics and convergence of the equation residuals of the T106C case are included in Appendix B2.

4.2 Results and Discussion

4.2.1 Wake Region. For the wake test case, the network makes excellent predictions of velocities, pressure, and temperature in the wake region. A typical assessment of turbulence model quality in turbomachinery applications is the wake loss profile Ω_w at $x/C_{ax} = 40\%$ downstream of the trailing edge. The wake loss and the total pressure are defined as

$$\Omega_w = \frac{\overline{p}_{t,1} - \overline{p}_{t,2}(y)}{\overline{p}_{t,1} - \overline{p}_2} \tag{23}$$

$$\overline{p_t} = \frac{\overline{p}}{\left(1 + 0.5(\gamma - 1)\left(\sqrt{\tilde{u}^2 + \tilde{v}^2}\right)/a\right)^{(-\gamma/(\gamma-1))}} \tag{24}$$

where the speed of sound is defined as $a = \gamma \overline{p}/\overline{\rho}$. The wake loss profiles in Fig. 7 show that the network can correctly predict the magnitude and width of the profile in a region far away from training data. The inferred Reynolds-stresses generally match those of the high-fidelity dataset with the correct magnitude and shape in the wake region. The $\tilde{u}$ and $\overline{\rho u'v'}$ fields are shown in Fig. 8. In small regions of the domain, the normal Reynolds-stress components yield small, positive predictions. These terms should be strictly negative (as defined in Eq. (12)), and enforcing a negative condition for this term could be achieved with a simple mathematical function. This hard enforcement of Reynolds-stress realisability should be considered for future work. The quality of the turbulent heat fluxes $\overline{\rho u'T'}$ and $\overline{\rho v'T'}$ is mixed, with acceptable quality predictions in most of the domain, and over predictions in between the wakes.

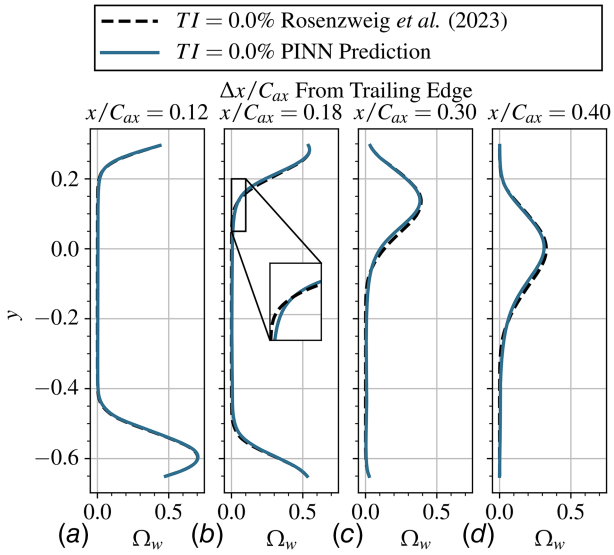

Fig. 7 The wake loss profiles Ω of the wake loss test case in axial chord units downstream of the linear cascade: (*a*) $x/C = 0.96$, (*b*) $x/C = 1.01$, (c) $x/C = 1.11$, and (*d*) $x/C = 1.20$. The banded region represents one standard deviation of the network ensemble.

This effect is more prominent in the $\overline{\rho u' T'}$ field. We conclude that more data are needed in the range of $0.94 \leq x/C \leq 1.71$ and an additional profile of data is included for the full domain test case at $x/C = 1.17$ to improve the heat-flux predictions.

4.2.2 Boundary Layer Region. We now focus our attention on the predictive capabilities of the PINN in the blade boundary layer. The network makes excellent predictions of the boundary layers around the blade, and the peak velocity on the suction side was also well modeled (Fig. 9). The largest differences between the reference data and the prediction are found close to the leading and trailing edges of the blade. These regions have very high gradients in both x and y, and we assume that more data are required in this region to make more accurate predictions. $\tilde{T}$ and $\overline{P}$, as well as the calculation of $\overline{\rho}$ are all of excellent quality. The Reynolds-stress fields are well predicted, with the same exceptions as in the wake modeling case previously discussed: there are some regions of unphysical normal Reynolds-stress components in the boundary layer, and the magnitude of the peak Reynolds-stress is underpredicted by the network. The turbulent heat-flux terms are well modeled (Fig. 9) and only have large errors at the trailing edge. This is an encouraging result, as the closure of the heat-flux terms has been inferred by the network, rather than with a mathematical model. The discussion of the boundary layer quantities ($\delta^\star$, θ, H_{12}, C_p, and C_f) will be undertaken in the context of the full domains, where comparisons will be made between inlet conditions of varying turbulent intensities, and the effect of inlet conditions on the quality of these predictions.

4.2.3 Full Domain With Varying Inlet Conditions. Based on the conclusions of the wake and boundary layer studies, an additional profile of data was included upstream and downstream of the cascade to improve the prediction quality (Fig. 6(*d*)). This resulted in excellent PINN predictions for all quantities as shown in Fig. 10. The $\overline{\rho u' v'}$ stress field is well predicted with minor discrepancies at the leading and trailing edges. Perhaps the most interesting result is the turbulent heat-flux predictions in the wake region, as shown in Fig. 10. The predictions of $\overline{\rho u' T'}$ and $\overline{\rho v' T'}$ have improved considerably from the wake test case due to the additional profile of training data added to the wake region. Profiles of the velocity vector, the Reynolds-stress tensor, and the heat-flux vector are plotted in Fig. 11, and these components have been rotated to be in wall-normal and wall-tangent coordinates. The profiles show that the network has typically produced excellent velocity predictions around the blade, and reasonable Reynolds-stress predictions as well. All predictions were observed to break down at the trailing edge (location c), and the Reynolds-stress and heat-flux terms also broke down on the pressure side of the trailing edge (location d).

Studying the boundary layer is the primary interest of this work, and we will now assess the PINN capabilities at different inlet turbulent intensities of TI = 0.0% and TI = 0.9%. The boundary layer displacement thickness ($\delta^\star$), the momentum thickness (θ), and the boundary layer shape factor (H_{12}) are plotted in Figs. 12(*a*)–12(*c*). These quantities capture the turbulence transition and separation on the suction side of the blade and are defined as

$$\delta^\star = \int_0^\infty \left(1 - \frac{\tilde{u}}{\widetilde{u_\infty}}\right) dn \tag{25}$$

$$\theta = \int_0^\infty \frac{\tilde{u}}{\widetilde{u_\infty}} \left(1 - \frac{\tilde{u}}{\widetilde{u_\infty}}\right) dn \tag{26}$$

$$H_{12} = \frac{\delta^\star}{\theta} \tag{27}$$

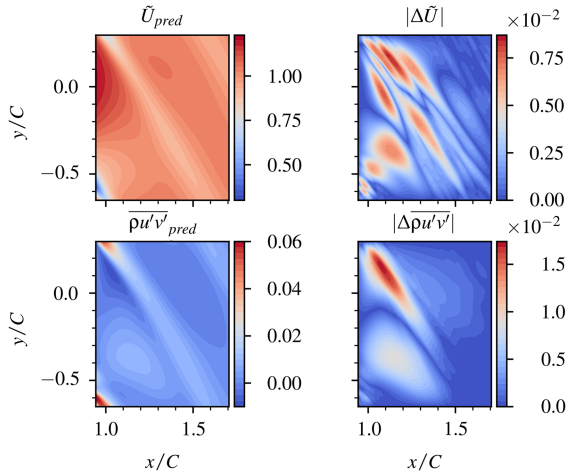

Fig. 8 Network predictions and the absolute difference for $\tilde{u}$ and $\overline{\rho u' v'}$ for the wake test case

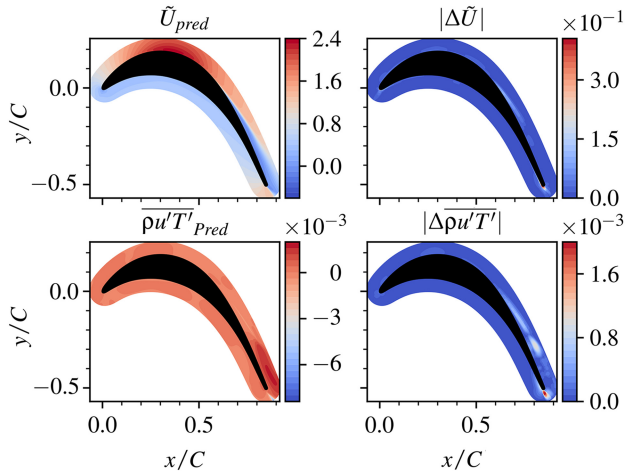

Fig. 9 Network predictions and the absolute difference for $\tilde{u}$ and $\overline{\rho u' T'}$ for the boundary layer test case

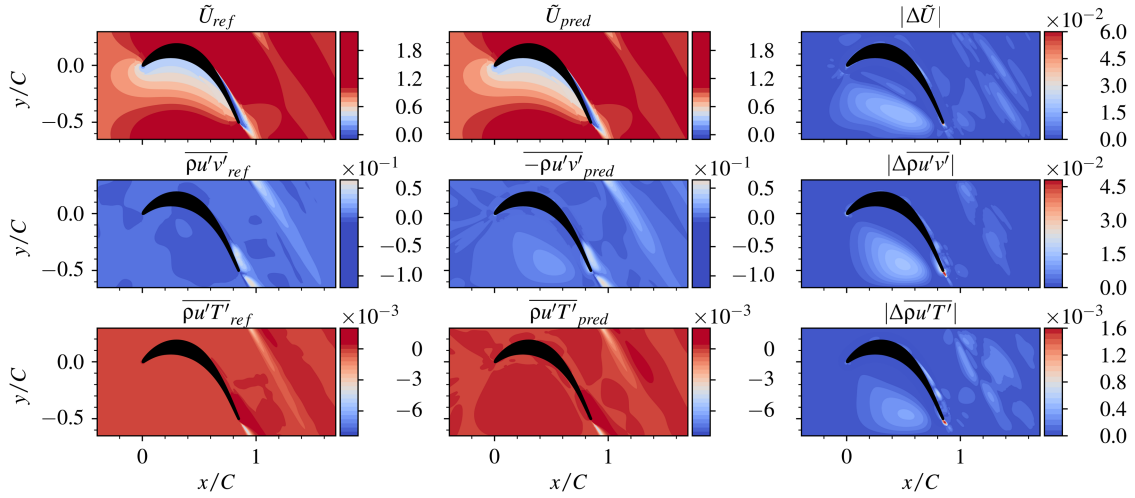

Fig. 10 A comparison of the $\tilde{u}$, $\overline{\rho u'v'}$, and $\overline{\rho u'T'}$ fields of the T106C cascade

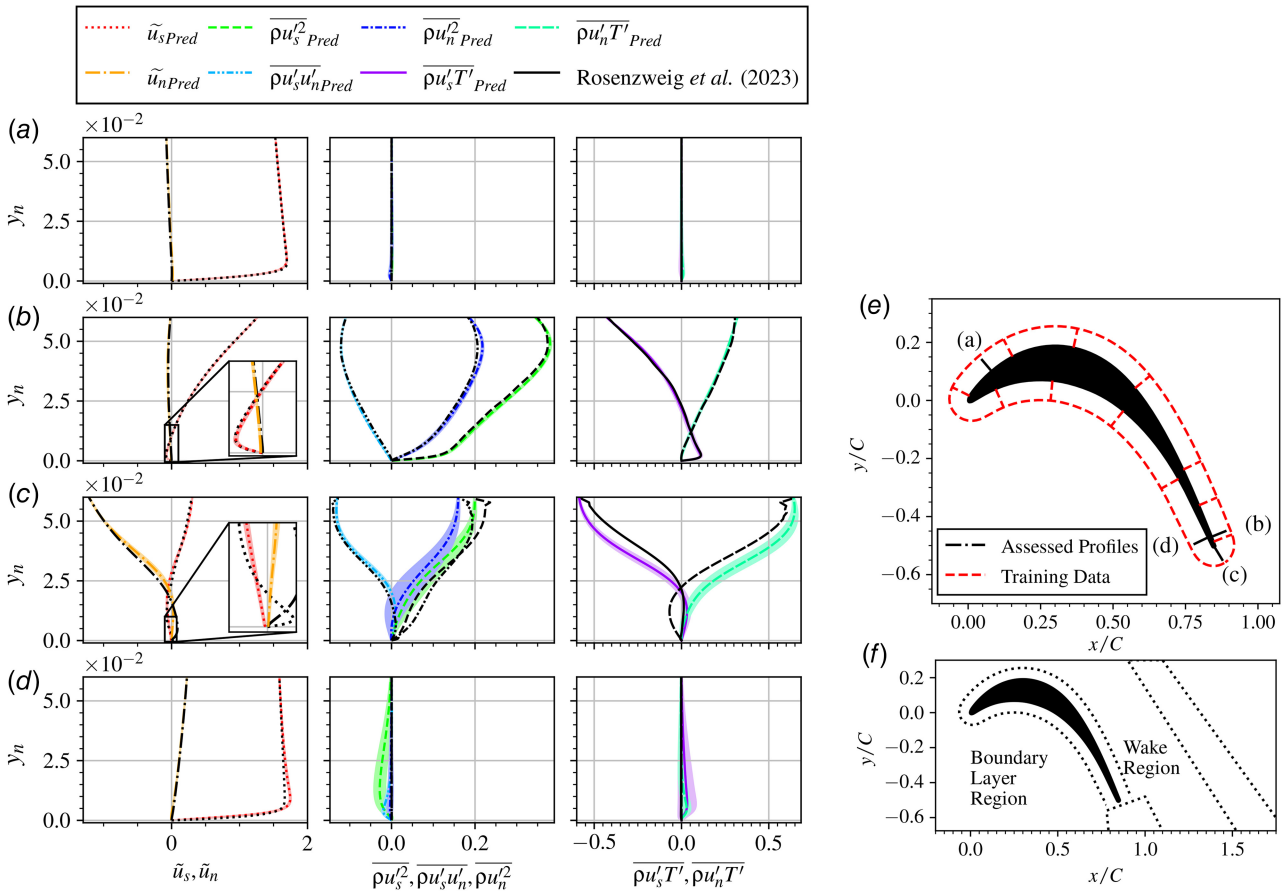

Fig. 11 (*a*)–(*d*) Wall-normal and wall-tangent profiles of the velocity vector, the Reynolds-stress tensor, and the heat-flux vector around the T106C blade. All locations listed in "unwrapped blade coordinates" correspond to (*e*). (*a*) 0.19, (*b*) 0.81, (c) −1.0, and (*d*) −0.8. The regions where the generalization error is assessed for the turbine cases, as shown in Fig. S3(*b*).

For the calculations of the boundary layer quantities, the boundary layer edge locations are extracted from the high-fidelity data, and this is used for the integration limit of the PINN and the reference curves. The predictions of $\delta^\star$, θ, and H_{12} are well predicted and provide meaningful assessments of the boundary layer formation and development around the blade. Similar comparisons can be made with C_f, as this indicates when the flow has separated in a region around the blade.

The introduction of inlet turbulence is known to delay the onset of separation or even completely suppress the separation bubble on a turbine blade, and a comparison between the C_f curves (TI = 0.0% and TI = 0.9%) is shown in Fig. 12(*d*). The PINN predictions show the same trend as the reference data, with the

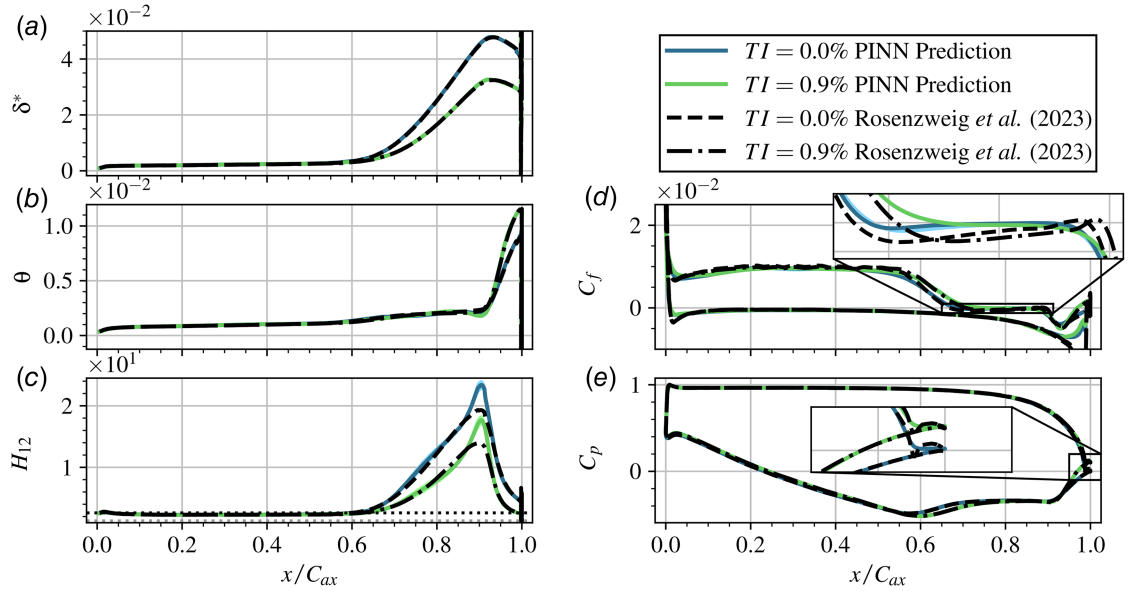

Fig. 12 Boundary layer and wall-bounded quantities of the full domain cases with differing inlet turbulence intensities. (*a*) Boundary layer thickness, (*b*) momentum thickness, (*c*) H factor, (*d*) friction coefficient, and (*e*) pressure coefficient. The banded region represents one standard deviation of the network ensemble.

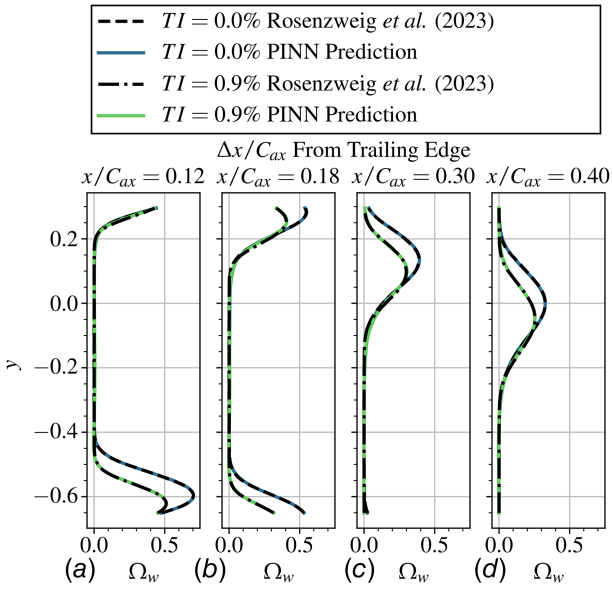

Fig. 13 Two wake loss profiles for the full domain cases with TI = 0.0% and TI = 0.9% inlet conditions. (*a*) $x/C = 0.96$, (*b*) $x/C = 1.01$, (*c*) $x/C = 1.11$, and (*d*) $x/C = 1.20$. The banded region represents one standard deviation of the network ensemble.

turbulent inlet conditions sustaining higher values of C_f for longer. In this section, the reference data and the network predictions have calculated C_f with the same fourth-order finite difference scheme. For the laminar case, the network predicted a separation point at $x/C_{ax} \approx 0.682$, and for the case with TI = 0.9%, the network predicted a separation point at $x/C_{ax} \approx 0.749$. The network was thus able to predict a delay in separation for the TI = 0.9% test case. In the reference high-fidelity data, two separation bubbles form near the trailing edge, with a small region of reattachment occurring between the two. The PINNs solution in this case only predicts one separation bubble with the separation point closely matching the location of the first bubble, and the reattachment point closely matching that of the second one.

Differences between the reference and the prediction are more prominent on the pressure side of the trailing edge as shown in Figs. 11(*d*) and 12(*d*). Training data are not provided in this region—as data are only applied to the suction side—and this is shown in the insert in Fig. 6(*c*). RANS solvers can adequately predict the flow on the pressure side of the blade up to the trailing edge, and this region is not of primary interest in this work. We consider a good prediction of the suction side of the blade to be an encouraging outcome for PINNs. Based on the hyperparameter study of the periodic hills in Sec. 3.2, it is expected that the near-wall velocity gradients close to the wall should improve with increasing network size, and we expect the magnitude of the predicted C_f curve to improve with larger network sizes. The C_p curve is also shown in Fig. 12(*e*), and the curve has been well represented on both sides of the blade. The Ω_w profiles are plotted in Fig. 13, and the networks are shown to represent the different wake profile shapes with the two inlet turbulence intensities.

5 Conclusion

PINNs are applied to the periodic hills at $\mathrm{Re}_b = 10,595$ and a T106C cascade at $\mathrm{Re}_{is} = 80,000$ in this work. The problems were selected since they feature flow physics such as separation and the wake region downstream of the cascade; both of which are challenging to simulate with baseline RANS turbulence models. The objective of this study was to demonstrate the performance of PINNs when inferring the Reynolds-stress and turbulent heat-flux fields. For the periodic hills case, the network made excellent predictions of the bulk quantities in the range of $0.05 < x/h < 8$, and the separation and C_f curve were also well modeled in this region with no turbulence closures.

For the T106C cascade, the network was applied to four test cases to understand the capability of PINNs when applied to different features of industrial flows. The network modeled the wake region between $0.1 \leq x/C_{ax} \leq 1$ downstream of the trailing edge, with the predictions of the wake loss profile approaching the quality of the high-fidelity dataset. The network was also found to model the suction side of the boundary layer region well, and the C_f captured the effects of separation and transition. The prediction of C_f was found to be sensitive to inlet turbulence intensity, and the PINN captured these trends in the separation bubble on the trailing edge of the blade. When provided with high-quality data located

near the features of interest, the PINN predictions can exceed the quality of RANS simulations [31]. PINNs are capable of reconstructing flows with sparse data, but the best predictions typically require training data close to features of interest such as the boundary layer or the separation bubble—regions that are challenging to measure in turbomachinery-related experiments. In this work, the highest quality PINN predictions were prioritized over computational costs and resources. This means that in this study, high-quality structured meshes were used with large network sizes to ensure the best possible predictions. Furthermore, ensemble averaging was used to provide a statistical assessment of the reported results. This study of data assimilation with "vanilla" PINNs (the architecture first proposed by Raissi et al. [10]) demonstrates the advantages and shortcomings of using PINNs to infer closure of the Reynolds-stress tensor in aerodynamics and fluid dynamics problems. The methodology of Eivazi et al. [16] is capable of reconstructing flow fields to a quality comparable to high-fidelity methods, given there is adequate data in regions of changing flow physics. The shortcoming of the methodology is that the network predictions can degrade in regions that have an absence of training data that feature changes in flow physics (i.e., the boundary layer in the periodic hills in $8 \leq x/h \leq 9$ and the pressure side of the trailing edge of the turbine blade). The current architecture is a meshless, non-conservative methodology [21] with weakly enforced Dirichlet boundary conditions. Future research on this topic should only consider the conservative PINN architectures [21] in combination with a distance function to exactly enforce Dirichlet boundary conditions of the domain [34]. The combination of the two methodologies should ensure that quantities are conserved between regions of sparse training data, and that Dirichlet boundary conditions are exactly enforced. Moving away from the initial architecture of Raissi et al. [10] to the architecture of Jagtap et al. [21] is the logical next step to improve the quality of data assimilation of sparse experimental data with PINNs.

Acknowledgment

SKH wishes to acknowledge the support of the Australian Government Research Training Program (RTP) scholarship. The authors would like to thank Associate Professor Ricardo Vinuesa and Mr. Hamidreza Eivazi for kindly providing their incompressible PINNs code, that has formed the basis for the present PINNs research. The authors would like to acknowledge the assistance of Mr. Marco Rosenzweig and Dr. Massimiliano Nardini, who provided assistance with postprocessing the T106C datasets. RDS acknowledges support from the Australian Research Council (ARC). This research was undertaken using the LIEF HPC-GPGPU facility hosted at the University of Melbourne. This facility was established with the assistance of LIEF Grant LE170100200.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

Nomenclature

p = pressure
q = heat-flux
u = stream-wise velocity
v = wall-normal velocity
C = chord length
T = temperature
$\mathbf{S}$ = mean-strain-rate tensor
$\mathbf{T}$ = supervised training data
$\mathbf{U}$ = network prediction
s, n = tangent and wall-normal coordinates
x, y = Cartesian coordinates

Greek Symbols

δ = Kronecker delta function
ε = residual
ν = fluid kinematic viscosity
ρ = density

Superscripts and Subscripts

b = bulk value
c = airfoil chord
h = hidden solution
t = stagnation quantity
R = closure term
ax = axial
bc = boundary condition
is = isentropic
res = residual
ref = reference value
Su = Sutherland temperature
Ref = reference quantity
1,2 = inlet, outlet quantities
∞ = freestream value

Calculated Quantities

Ma = Mach number, $u/\sqrt{(\gamma p/\rho)}$
Re = Reynolds number, $\left(uL/\nu\right)_{\text{ref}}$
TI = turbulence intensity, $\sqrt{2k/3}/|\mathbf{u}|$

Accents

$\overline{\cdot}$ = time-averaged
$\cdot'$ = fluctuation
$\widetilde{\cdot}$ = Favre-averaged
$\hat{\cdot}$ = network prediction

Acronyms

DNN = deep neural network
PBC = periodic boundary condition
PINN = physics-informed neural network
RANS = Reynolds-averaged Navier–Stokes

Appendix A: Overfitting Datasets With PINNs?

In the field of data science, there is a common discussion about the overfitting of datasets with neural networks. Because PINNs use a modified architecture that features a loss function that weakly regularizes the governing equations, we believe that there is a value in describing what overfitting means for PINN applications.

In practical terms, the definition of overfitting a neural network is where the network solution function $\hat{\mathbf{U}}$ is more complex than the underlying hidden solution $\mathbf{U}_h$. By having this unnecessary complexity, fluctuations can occur in the network solution between points of training data. These fluctuations in $\hat{\mathbf{U}}$ are identified with the test error L_{test}, and this represents the mean squared error of the network prediction at locations where data were not provided to the network in training. In an assessment of overfitting a DNN, the L_{data} term continues to decrease while L_{test} has reached a minimum and begins to increase again. This occurs because the solution function $\hat{\mathbf{U}}$ prioritizes the data $\mathbf{T}_{ij}$ over the hidden solution $\mathbf{U}_h$. The concept of overfitting is of particular importance to neural networks applied to generalized predictive problems, and identifying this minima of L_{test} is an indicator of the correct network

hyperparameters and training schedule for the generalizable predictive model.

Physics-informed neural networks are weakly regularized with residual points scattered between the training datapoints, and this means that (in principle) the solution function $\hat{\mathbf{U}}$ is also forced to fit the residual points during training. If network derivatives are included in the residual loss term, it also means that the residual points are also weakly enforcing the derivatives at each residual point. This further enhances the robustness of the PINN in training. To demonstrate the capabilities of PINNs to avoid overfitting a priori problems, a DNN and a PINN solve a simple solution function of $y = \sin(x)$. Both networks have eight hidden layers and 20 neurons in width with $\sigma = \arctan(\cdot)$, and both networks are trained with Adam for 500 epochs. The DNN is trained on four equidistant points of training data in the range of $x \in [0, 2\pi]$, and the PINN is trained with the boundary conditions of $\mathbf{U}_h(0) = \mathbf{U}_h(2\pi) = 0$, $\partial\mathbf{U}_h(0)/\partial x = \partial\mathbf{U}_h(2\pi)/\partial x = 1$, and $\partial^2\mathbf{U}_h(0)/\partial x^2 = \partial^2\mathbf{U}_h(2\pi)/\partial x^2 = 0$, as well as the associated residual functions for

$$L_{\text{res}} = \hat{\mathbf{U}} - \sin(\mathbf{x}_1) + \frac{\partial\hat{\mathbf{U}}}{\partial x} - \cos(\mathbf{x}_1) + \frac{\partial^2\hat{\mathbf{U}}}{\partial x^2} + \sin(\mathbf{x}_1) \tag{A1}$$

The $\mathbf{x}_1$ mesh contains 12 equidistant residual points between $x \in [0, 2\pi]$, and after training, both networks are probed with a high-resolution mesh $\mathbf{x}_2$, which contains 1200 points between $x \in [0, 2\pi]$. The network predictions of the DNN and the PINN are shown in Supplemental Fig. S1 available in the Supplemental Materials on the ASME Digital Collection. Because of the coarse training data, the DNN is unable to infer the underlying solution to the problem. When probed with the $\mathbf{x}_2$ mesh, the solution function is shown to deviate significantly from the solution $y(x)$. By the commonly accepted definition of overfitting, the network was correctly trained after 125 epochs, as this is the point of inflection of the L_{test} term. The PINN has been able to learn the $\mathbf{U}_h$, $\partial\mathbf{U}_h/\partial x$, and $\partial^2\mathbf{U}_h/\partial x^2$ terms from the L_{res}. For the PINNs case, L_{test} is solved in a similar manner to Eq. (A1), where the high-fidelity $\mathbf{x}_2$ mesh is used to sample the network. We can see that the L_{test} error term of the PINN continues to converge, and the network is not producing oscillations in the network solution in the $\mathbf{x}_2$ mesh. Clearly, in this example, the PINN is not overfitted.

There are two situations where the hyperparameter studies should be thoroughly scrutinized with PINNs. PINNs have been demonstrated to be capable of filtering noise from experimental datasets, and this was demonstrated by Cai et al. [35]. Experimental data have measurement and positional errors, and a well-tuned set of network hyperparameters are required to constrain the functional complexity of the network to correctly fit the underlying hidden solution $\mathbf{U}_h$. This was observed in the work of Hanrahan et al. [17], where small network sizes were required to prevent overfitting of the periodic hills experimental datasets.

As shown above, PINNs can be sampled with other residual point meshes after training, meaning that the network could be trained with a coarse mesh, before being upscaled to be compared to an LES or direct numerical simulation dataset. While the example of the $y = \sin(\mathbf{x}_1)$ function was shown to be robust, we still advise caution when using this method of *upscaling*, as the residual points have changed locations, meaning that the enforcement of the governing equations on the residual points located at $\mathbf{x}_1$ may not have the same adherence for the $\mathbf{x}_2$ mesh.

The learning from this example can be directly applied to the hyperparameter study of the PINNs with embedded RANS equations. If reasonably dense mesh is used, and the derivatives of the predicted quantities are included in the residual loss term, it can be expected that the network solution function to converge, and not overfit $\mathbf{U}_h$. In the present study, both PINNs have been trained on high-quality numerical data, and the mesh resolution has not been modified after training. Based on the simple demonstration, we see overfitting as a somewhat trivial issue for the problems studied in the present work, as we focus on the convergence of the residual functions. The hyperparameter studies in Secs. 3.2 and 4.2 were undertaken to identify the correct weightings λ and network widths to ensure that the PINN would train on the governing equations and to prevent underfitting of the solution.

Appendix B: Network Training Statistics

B.1 Convergence of the Periodic Hills. The convergence of the periodic hills network loss function is shown in Fig. S2(*a*) where each component of the loss function is inclusive of the corresponding weighting λ. The data loss, residual loss, and boundary conditions all show good convergence. The residual of the continuity and momentum equations is shown in Fig. S2(*b*) and the equation residuals effectively plateau by the end of training.

B.2 Convergence of the T106C Full Domain at TI = 0.0%. For the T106C full domain case, further network statistics have been extracted for analysis, and these are shown in Fig. S3. The evolution of the loss functions in Fig. S3(*a*) shows the convergence of the loss function for the problem. The evolution of the loss function changes after 50, 000 epochs and smoothly descends to a loss function minima. This coincides with the transition from the Adam optimizer to the LBFGS optimizer and highlights the value of using an LBFGS methodology in PINNs training. The test error L_{test} of the boundary layer and wake regions of the domain is shown in Fig. S3(*b*). This has been calculated with all available data in the boundary layer and the wake region (see regions in Fig. 11(*e*)). This data is compared with the network solution at each epoch of training with Eq. (7) and show a steady reduction throughout training. By assessing the continued convergence of the L_{test} curves shown in Fig. S3(*b*) and the governing equation residuals plotted in Fig. S3(*c*), it is clear that the network has not overfitted the problem. The final percentage error of each case of the network is included in Table S1 where low errors are achieved among the velocity, pressure, temperature, and density terms. The tabulated values of the Reynolds-stress and heat-flux terms have larger errors, however, these are very small quantities of interest, and this may lead to large errors in certain regions of the flow. Based on this consideration, we refer the reader to the velocity, Reynolds-stress, and heat-flux predictions around the blade, shown in Figs. 11(*a*)–11(*d*).

Finally, the value of a^d of the L-LAAF has been plotted in Fig. S3(*d*). L-LAAF is intended to increase the gradients of the hidden layers of the network during training to improve learning efficiency, and the figure shows how the value of a^d increases during training of the network, typically with the hidden layers closer to the network inputs gaining larger values of a^d. This suggests that it is challenging to train the earlier hidden layers $1 \leq d \leq 4$ of the DNN without a gradient tuning process such as L-LAAF [24], and this process has increased the magnitude of the gradients in this region of the network during training.

References

[1] Bush, R. H., Chyczewski, T. S., Duraisamy, K., Eisfeld, B., Rumsey, C. L., and Smith, B. R., 2019, "Recommendations for Future Efforts in RANS Modeling and Simulation," AIAA Scitech 2019 Forum, San Diego, CA, Jan. 7–11, p. 0317.

[2] Akolekar, H. D., Weatheritt, J., Hutchins, N., Sandberg, R. D., Laskowski, G., and Michelassi, V., 2019, "Development and Use of Machine-Learnt Algebraic Reynolds Stress Models for Enhanced Prediction of Wake Mixing in Low-Pressure Turbines," ASME J. Turbomach., **141**(4), p. 041010.

[3] Wilcox, D. C., 2006, *Turbulence Modeling for CFD*, 3rd ed., DCW Industries Inc, La Cañada, CA.

[4] Weatheritt, J., and Sandberg, R. D., 2016, "A Novel Evolutionary Algorithm Applied to Algebraic Modifications of the RANS Stress–Strain Relationship," J. Comput. Phys., **325**, pp. 22–37.

[5] Weatheritt, J., and Sandberg, R. D., 2017, "The Development of Algebraic Stress Models Using a Novel Evolutionary Algorithm," Int. J. Heat Fluid Flow, **68**, pp. 298–318.

[6] Sandberg, R. D., and Michelassi, V., 2022, "Fluid Dynamics of Axial Turbomachinery: Blade- and Stage-Level Simulations and Models," Annu. Rev. Fluid Mech., **54**(1), pp. 255–285.

[7] Ling, J., Kurzawski, A., and Templeton, J., 2016, "Reynolds Averaged Turbulence Modelling Using Deep Neural Networks With Embedded Invariance," J. Fluid Mech., **807**, pp. 155–166.
[8] Akolekar, H. D., Zhao, Y., Sandberg, R. D., and Pacciani, R., 2021, "Integration of Machine Learning and Computational Fluid Dynamics to Develop Turbulence Models for Improved Low-Pressure Turbine Wake Mixing Prediction," ASME J. Turbomach., **143**(12), p. 121001.
[9] Clark Di Leoni, P., Agarwal, K., Zaki, T. A., Meneveau, C., and Katz, J., 2023, "Reconstructing Turbulent Velocity and Pressure Fields From Under-Resolved Noisy Particle Tracks Using Physics-Informed Neural Networks," Exp. Fluids, **64**(5), p. 95.
[10] Raissi, M., Perdikaris, P., and Karniadakis, G. E., 2019, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," J. Comput. Phys., **378**, pp. 686–707.
[11] Raissi, M., Yazdani, A., and Karniadakis, G. E., 2020, "Hidden Fluid Mechanics: Learning Velocity and Pressure Fields From Flow Visualizations," Science, **367**(6481), pp. 1026–1030.
[12] Jin, X., Cai, S., Li, H., and Karniadakis, G. E., 2021, "NSFnets (Navier-Stokes Flow Nets): Physics-Informed Neural Networks for the Incompressible Navier–Stokes Equations," J. Comput. Phys., **426**, p. 109951.
[13] Kag, V., Seshasayanan, K., and Gopinath, V., 2022, "Physics-Informed Data-Based Neural Networks for Two-Dimensional Turbulence," Phys. Fluids, **34**(5), p. 055130.
[14] Chuang, P., and Barba, L. A., 2022, "Experience Report of Physics-Informed Neural Networks in Fluid Simulations: Pitfalls and Frustration," Proceedings of the 21st Python in Science Conference, Austin, TX, July 11–17, pp. 28–36.
[15] Wang, S., Sankaran, S., and Perdikaris, P., 2024, "Respecting Causality for Training Physics-Informed Neural Networks," Comput. Methods Appl. Mech. Eng., **421**, p. 116813.
[16] Eivazi, H., Tahani, M., Schlatter, P., and Vinuesa, R., 2022, "Physics-Informed Neural Networks for Solving Reynolds-Averaged Navier–Stokes Equations," Phys. Fluids, **34**(7), p. 075117.
[17] Hanrahan, S. K., Kozul, M., and Sandberg, R. D., 2023, "Studying Turbulent Flows With Physics-Informed Neural Networks and Sparse Data," Int. J. Heat Fluid Flow, **104**, p. 109232.
[18] Pioch, F., Harmening, J. H., Müller, A. M., Peitzmann, F.-J., Schramm, D., and Moctar, O. E., 2023, "Turbulence Modeling for Physics-Informed Neural Networks: Comparison of Different RANS Models for the Backward-Facing Step Flow," Fluids, **8**(2), p. 43.
[19] Jagtap, A. D., Mao, Z., Adams, N., and Karniadakis, G. E., 2022, "Physics-Informed Neural Networks for Inverse Problems in Supersonic Flows," J. Comput. Phys., **466**, p. 111402.
[20] Post, P., Winhart, B., and di Mare, F., 2022, "Investigation of Physics-Informed Neural Networks Based Solution Techniques for Internal Flows," Proceedings of ASME Turbo Expo 2023: Turbomachinery Technical Conference and Exposition, No. 80960 in Turbo Expo: Power for Land, Sea, and Air, Rotterdam, June 13–17.
[21] Jagtap, A. D., Kharazmi, E., and Karniadakis, G. E., 2020, "Conservative Physics-Informed Neural Networks on Discrete Domains for Conservation Laws: Applications to Forward and Inverse Problems," Comput. Methods Appl. Mech. Eng., **365**, p. 113028.
[22] Hasanuzzaman, G., Eivazi, H., Merbold, S., Egbers, C., and Vinuesa, R., 2023, "Enhancement of PIV Measurements Via Physics-Informed Neural Networks," Meas. Sci. Technol., **34**(4), p. 044002.
[23] Wilcox, D. C., 1998, "Reassessment of the Scale-Determining Equation for Advanced Turbulence Models," AIAA J., **26**(11), pp. 1299–1310.
[24] Jagtap, A. D., Kawaguchi, K., and Karniadakis, G. E., 2020, "Locally Adaptive Activation Functions With Slope Recovery for Deep and Physics-Informed Neural Networks," Proc. R. Soc. A: Math. Phys. Eng. Sci., **476**(2239), p. 20200334.
[25] Markidis, S., 2021, "The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers?" Front. Big Data, **4**, p. 669097.
[26] Baydin, A. G., Pearlmutter, B. A, Radul, A. A., and Siskind, J. M., 2018, "Automatic Differentiation in Machine Learning: A Survey," J. Mach. Learn. Res., **18**, pp. 1–43.
[27] Frölich, J., Mellen, C. P., Rodi, W., Temmerman, L., and Leschziner, M. A., 2005, "Highly Resolved Large-Eddy Simulation of Separated Flow in a Channel With Streamwise Periodic Constrictions," J. Fluid Mech., **526**, pp. 19–66.
[28] Kingma, D. P., and Ba, J., 2015, "Adam: A Method for Stochastic Optimization," 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, May 7–9, Conference Track Proceedings.
[29] Liu, D. C., and Nocedal, J., 1989, "On the Limited Memory BFGS Method for Large Scale Optimization," Math. Program., **45**(1), pp. 503–528.
[30] Goodfellow, I., Bengio, Y., and Courville, A., 2016, *Deep Learning*, 1st ed., MIT Press, Cambridge, MA.
[31] Jakirlic, S., 2012, "Assessment of the RSM, URANS, and Hybrid Models With Respect to the Different Roadmaps Including the Industrial Challenges," ATAAC Final Report D3.2-36, Technical University of Darmstadt.
[32] Rosenzweig, M., Giaccherini, S., Pinelli, L., Kozul, M., Sandberg, R., Marconcini, M., and Pacciani, R., 2023, "Best-Practice Guidelines for High-Fidelity Simulations Based on Detailed Analysis of a Highly-Loaded Low-Pressure Turbine Cascade," Proceedings of ASME Turbo Expo 2023 Turbomachinery Technical Conference and Exposition, Boston, MA, June 26–30, p. 102697.
[33] Sandberg, R. D., Michelassi, V., Pichler, R., Chen, L., and Johnstone, R., 2015, "Compressible Direct Numerical Simulation of Low-Pressure Turbines—Part I: Methodology," ASME J. Turbomach., **137**(5), p. 051011.
[34] Sukumar, N., and Srivastava, A., 2022, "Exact Imposition of Boundary Conditions With Distance Functions in Physics-Informed Deep Neural Networks," Comput. Methods Appl. Mech. Eng., **389**, p. 114333.
[35] Cai, S., Wang, Z., Fuest, F., Jeon, Y. J., Gray, C., and Karniadakis, G. E., 2021, "Flow Over an Espresso Cup: Inferring 3-D Velocity and Pressure Fields From Tomographic Background Oriented Schlieren Via Physics-Informed Neural Networks," J. Fluid Mech., **915**, p. A102.