Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 361 (2020) 112732
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data

Luning Sun[a,b], Han Gao[a,b], Shaowu Pan[c], Jian-Xun Wang[a,b,*]

[a] Department of Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame, IN, United States of America
[b] Center for Informatics and Computational Science, University of Notre Dame, Notre Dame, IN, United States of America
[c] Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, United States of America

Received 18 July 2019; received in revised form 4 November 2019; accepted 4 November 2019
Available online 21 November 2019

Abstract

Numerical simulations on fluid dynamics problems primarily rely on spatially or/and temporally discretization of the governing equation using polynomials into a finite-dimensional algebraic system. Due to the multi-scale nature of the physics and sensitivity from meshing a complicated geometry, such process can be computational prohibitive for most real-time applications (e.g., clinical diagnosis and surgery planning) and many-query analyses (e.g., optimization design and uncertainty quantification). Therefore, developing a cost-effective surrogate model is of great practical significance. Deep learning (DL) has shown new promises for surrogate modeling due to its capability of handling strong nonlinearity and high dimensionality. However, the off-the-shelf DL architectures, success of which heavily relies on the large amount of training data and interpolatory nature of the problem, fail to operate when the data becomes sparse. Unfortunately, data is often insufficient in most parametric fluid dynamics problems since each data point in the parameter space requires an expensive numerical simulation based on the first principle, e.g., Navier–Stokes equations. In this paper, we provide a physics-constrained DL approach for surrogate modeling of fluid flows *without* relying on any simulation data. Specifically, a structured deep neural network (DNN) architecture is devised to enforce the initial and boundary conditions, and the governing partial differential equations (i.e., Navier–Stokes equations) are incorporated into the loss of the DNN to drive the training. Numerical experiments are conducted on a number of internal flows relevant to hemodynamics applications, and the forward propagation of uncertainties in fluid properties and domain geometry is studied as well. The results show excellent agreement on the flow field and forward-propagated uncertainties between the DL surrogate approximations and the first-principle numerical simulations.

Keywords: Physics-informed machine learning; Label-free; Neural networks; Uncertainty quantification; Cardiovascular flows; Navier-Stokes

1. Introduction

Complex fluids are ubiquitous in natural and industrial processes, and accurately simulating the fluid flows is indispensable in many disciplines, e.g., aerospace, civil, and biomedical engineering. A fluid system is typically governed by the Navier–Stokes equations, which is a highly nonlinear partial differential equation (PDE) system.

* Corresponding author at: Department of Aerospace and Mechanical Engineering, University of Notre Dame, Notre Dame, IN, United States of America.
E-mail address: jwang33@nd.edu (J.-X. Wang).

https://doi.org/10.1016/j.cma.2019.112732
0045-7825/

Numerical simulation on fluid dynamics problems primarily relies on solving the PDE systems in a discretized form using, e.g., finite difference (FD), finite volume (FV), or finite element (FE) methods, which is known as the computational fluid dynamics (CFD) approach [1]. However, CFD simulations are often computationally cumbersome, especially for the flows with turbulence and complex geometries. Moreover, mesh generation also usually incurs a huge burden, in particular when moving boundary or large geometric variation is considered. The considerable computational expense greatly limits the use of principled CFD model to *real-time predictions* and *many-query analysis*, which are highly relevant to many scientific problems and real-life applications, e.g., timely clinical diagnosis and surgery planning for cardiovascular diseases, optimization design of aircraft with large parameter variations, and forward/inverse uncertainty quantification (UQ) in high-consequence systems. As an alternative, a cost-effective surrogate model is a computationally feasible way to tackle the aforementioned challenges.

A surrogate model only approximates the input–output relation of a system, which can be evaluated efficiently. Namely, given input parameters, e.g., initial/boundary/operational conditions, the quantities of interest (QoIs), such as velocity, pressure, shear stress, and their integrals can be obtained rapidly without conducting the principled CFD simulations. The existing surrogate modeling approaches can be roughly categorized into two classes: projection-based reduced order models (ROMs) and data-fit models [2]. In projection-based ROMs, a reduced basis is extracted from the simulation data using an unsupervised learning technique, e.g., proper orthogonal decomposition (POD, also known as principal component analysis) [3], and the full-order PDE operator is projected onto the subspace spanned by the reduced basis. As a result, the degrees of freedom of the system can be significantly reduced, and meanwhile, the underlying structure of the full-order model can be retained to a certain extent. Although holding some promises, the current projection-based ROM techniques have had limited impact on complex fluid dynamic problems mainly because of the stability and robustness issues [4,5]. Moreover, projection-based ROMs are highly code-intrusive and their speedup potential is limited when strong nonlinearity exists [6] though several remedies such as sparse sampling [7,8] exist. Another way to enable rapid simulations is to build a data-fit model, where a response surface of the system is learned from the simulation data in a supervised manner. Namely, a deterministic or probabilistic input–output mapping is constructed using, e.g., polynomial basis functions [9], radial basis functions [10], Gaussian process (GP) [11,12], and stochastic polynomial chaos expansion (PCE) [13–15], among others. All these models are built upon the CFD solutions of selected collocation points in parameter space without the need to modify the codes of the CFD solver. Because of the non-intrusive feature and ease of implementation, data-fit surrogates have been used for a wide range of forward and inverse uncertainty quantification (UQ) problems in fluid dynamics [14,16,17]. However, traditional data-fit models have a hard time handling the problems with strong nonlinearities and high dimensionality. Deep learning (DL), in particular, the deep neural network (DNN) has become a popular surrogate modeling approach and has shown great potential to deal with high-dimensional nonlinear UQ problems [18–20]. It has been shown that DNN as a universal function approximator [21] can overcome the curse of dimensionality in certain problems [22–24]. In broader scientific computing and physical modeling communities, machine learning (ML) has been receiving a lot of attentions [25–31]. However, the tremendous success of DL in the computer science, witnessed in areas of computer vision and image recognition [32], can be mainly attributed to the *availability of large-scale labeled data* (i.e., "big data") and the *interpolatory nature of their problems*. Unfortunately, labeled data for surrogate modeling of fluid systems are often *sparse and could be noisy*, since they are obtained from either principled CFD simulations or experimental observations, both of which are expensive to obtain. Therefore, in such "small data" regimes [33], the true power of DL cannot be fully exploited by naively using the off-the-shelf DL model in the computer science community as an end-to-end fashion [34] for a data-fit of surrogate modeling.

In conventional ML problems, the mechanism behind the system is usually unknown and thus can only be learned from the labeled data. In contrast, for modeling a physical system, the governing equations are usually known *a priori* but are difficult to solve efficiently. Instead of learning solely from the labeled data, e.g., solution of the states on certain points in parameter space, the known governing equations can be utilized to constrain (or even drive) the learning to compensate for the insufficiency of the data. Specifically, the training (optimization) of a DNN can be driven by minimizing the residual of the governing equations constructed by the DNN ansatz. This idea of physics-constrained learning is not new and was proposed back in the late '90s in the context of solving classic differential equations [35–37]. However, limited by the NN techniques and computational power at that time, this seminal work did not have a big impact. Recently, this idea has been revived because of the recent advances in

deep learning [32] combined with ever-increasing computational resources. Notably, the physics-informed neural network (PINN) proposed by Raissi et al. was used to solve a number of deterministic one-dimensional (1D) PDEs such as viscous Burger's equation, and two/three-dimensional (2D/3D) PDE-constrained inverse problems with a moderate amount of labeled data [33,38,39], e.g., measurements of the velocity field. A similar approach is also applied to learn the constitutive relationship in a Darcy flow [40]. The PINN approach has been recently extended to assimilate multi-fidelity training data [41], and its UQ analyses have been explored based on arbitrary polynomial chaos [42] and adversarial inference [43]. Similar ideas of using physical constraints to regularize the DNN training have also been investigated in [44–47]. In the aforementioned works, a moderate amount of labeled data either from simulations or experimental measurements are still needed for obtaining an approximation to the solution of the PDEs. In fact, if the initial and boundary conditions are well imposed thus the corresponding PDE problem is well-defined, in principle, the unique solution should be captured by the DNN via PDE-constrained learning without any labeled data. This is one of the main motivations of current work. Note that recently there have been several works on the concept of data-free DNNs, e.g., for solving a handful of computer vision problems [48], deterministic PDEs [33,49,50], high-dimensional stochastic partial differential equations (SPDE), and backward stochastic differential equations (BSDE) [51–55].

In the context of surrogate modeling, Nabian and Meidani [56] and Karumuri et al. [57] applied the PDE-constrained fully-connected neural network (FC-NN) for uncertainty propagation in steady heat equations. Zhu et al. [58] proposed a PDE-constrained, label-free DNN surrogate model for UQ in an elliptic PDE using both the FC-NN and convolutional neural networks (CNN). Moreover, both the deterministic and probabilistic formulations of physics-constrained learning were studied. Their results have shown a significant potential of using the physics-constrained DNN for surrogate modeling, where no labeled data are required during the training. Nonetheless, the success has only been demonstrated in a number of canonical problems with regular (rectangular) geometries, and it remains unclear if the physics-constrained learning can handle realistic fluid systems governed by the Navier–Stokes equations in a parametric setting. Significant work is still needed to further explore the real-world problems for broad impacts.

The *objective* of this paper is to develop a physics-constrained, data-free DNN for surrogate modeling of incompressible flows. A structured FC-NN is devised to approximate the solutions of parametric Navier–Stokes equations, where the initial/boundary conditions are enforced instead of being penalized together during training in previous works [33]. In addition, contrary to the existing data-driven surrogates, the training of our DNN is solely driven by minimizing the residuals of the governing PDEs (i.e., conservation laws), where *no expensive CFD simulation data* is used. The effectiveness and merits of the proposed method are demonstrated by investigating a number of internal flows relevant to cardiovascular applications. The contributions of the current paper are summarized as follows. First, this work explored the performance of label-free deep learning for parametric surrogate construction, while most existing PINN works focus on solving deterministic PDEs in a non-parametric setting [33]. Second, in this study a more challenging problem is tackled, where the system is governed by the full Navier–Stokes equations and with irregular geometries. Particularly, we demonstrated that the flow solutions to geometric variations can be accurately captured by the proposed deep learning approach, showing good promise for rapid geometric optimization and uncertainty quantification. To the best of authors' knowledge, this is the *first attempt* of using a single DNN structure to learn the solutions of Navier–Stokes equations in *a parametric setting without relying on any labeled training data*. Lastly, the boundary conditions are encoded into the DNN architecture in a hard manner, and the advantages of using "hard" boundary enforcement compared to the "soft" ones are demonstrated in data-free scenarios. The current work aims to push forward the PDE-constrained deep learning framework towards more realistic applications. The rest of the paper is organized as follows. The framework of structured FC-NN surrogate based on the physics-constrained label-free training is introduced in Section 2. Numerical results of surrogate modeling and uncertainty propagation on several vascular flows are presented in Section 3. The performance of soft and hard boundary enforcement approaches, different adaptive activation functions, and data-free/data-driven learning strategies are discussed in Section 4. Finally, conclusion is drawn in Section 5.

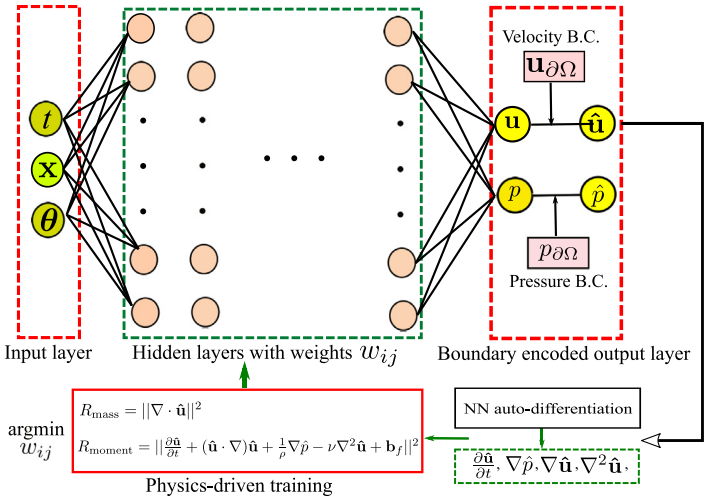


Fig. 1. A schematic diagram of the physics-constrained, data-free DL framework for surrogate modeling of fluid flows. A structured fully-connected neural network (FC-NN) is developed with the boundary conditions encoded by construction. The network is trained by minimizing the equation-based loss function and no CFD simulation data are needed.

2. Methodology

2.1. Overview

Most low-speed flows, e.g., blood flows in large or medium sized vessels, can be described by the incompressible Navier–Stokes equations given as:

$$\mathscr{F}(\mathbf{u}, p) = \mathbf{0} := \begin{cases} \nabla \cdot \mathbf{u} = \mathbf{0}, & \mathbf{x}, t \in \Omega_{f,t}, \boldsymbol{\theta} \in \mathbb{R}^d, \\ \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \dfrac{1}{\rho}\nabla p - \nu\nabla^2\mathbf{u} + \mathbf{b}_f = \mathbf{0}, & \mathbf{x}, t \in \Omega_{f,t}, \boldsymbol{\theta} \in \mathbb{R}^d \end{cases} \tag{1}$$

where t and $\mathbf{x}$ are time and space coordinates, respectively; $\Omega_{f,t} \triangleq \Omega_f \times [0, T]$; $\boldsymbol{\theta}$ is a d-dimensional parameter vector, including input and/or operational parameters such as fluid properties, inlets/outlets, and geometry of the domain; both velocity $\mathbf{u}$ and pressure p are functions of time t, space $\mathbf{x}$, variable parameters $\boldsymbol{\theta}$; ρ and ν represent density and viscosity of the fluid, respectively; $\mathbf{b}_f$ is the body force; $\Omega_f \subset \mathbb{R}^3$ denotes the fluid domain. The solutions of velocity $\mathbf{u}$ and pressure p can be uniquely determined when suitable initial and boundary conditions are prescribed,

$$\mathcal{I}(\mathbf{x}, p, \mathbf{u}, \boldsymbol{\theta}) = 0, \qquad \mathbf{x} \in \Omega_f, t = 0, \boldsymbol{\theta} \in \mathbb{R}^d, \tag{2a}$$

$$\mathcal{B}(t, \mathbf{x}, p, \mathbf{u}, \boldsymbol{\theta}) = 0, \qquad \mathbf{x}, t \in \partial\Omega_f \times [0, T], \boldsymbol{\theta} \in \mathbb{R}^d, \tag{2b}$$

where both $\mathcal{I}$ and $\mathcal{B}$ are general differential operators that define the initial and boundary conditions, respectively; $\partial\Omega_f$ denotes the boundary region. When a set of parameters $\boldsymbol{\theta}$ is given, the flow fields, i.e., $\mathbf{u}(t, \mathbf{x})$ and $p(t, \mathbf{x})$, can be solved numerically by discretizing the Eqs. (1) and (2) using FD/FV/FE methods. However, this process involves mesh generation and iteratively solving large linear/nonlinear systems, which is usually time-consuming. Therefore, propagating the parameter uncertainty or inferring the unknown parameters through the FD/FV/FE solver becomes intractable when it comes to parametric problems, e.g., some parameters of $\boldsymbol{\theta}$ are uncertain or unknown. Solving varying-geometry problems is especially challenging since any change of the geometry requires regeneration of the computational meshes.

To enable fast predictions in terms of UQ and optimization applications, a deep neural network (DNN) architecture is built to approximate the solutions of the Navier–Stokes equations in a parametric setting. The DNN-based surrogate is expected to provide a rapid online prediction of the flow field with any given set of parameters $\boldsymbol{\theta}$ after the offline training. A schematic diagram of the proposed framework is shown in Fig. 1. A FC-NN is devised

with the input layer composed of time t, spatial coordinates $\mathbf{x}$, and variable parameters $\boldsymbol{\theta}$. The raw outputs of the FC-NN (i.e., $\mathbf{u}$ and p) are used to construct the state variables (i.e., velocity $\hat{\mathbf{u}}$ and pressure $\hat{p}$) together with contribution from the particular solution that encodes initial/boundary conditions. The FC-NN is trained by minimizing residuals of the Navier–Stokes equations and no data from CFD simulations are needed. Therefore, the DNN predictions are expected to conform to the conservation laws of fluid flows and satisfy the specified initial/boundary conditions. Note that the Navier–Stokes equations will not be solved with any numerical discretization. The details of the physics-constrained training and boundary condition enforcement will be presented in the following subsections.

2.2. Deep neural network and physics-constrained training

Neural networks (NN) are a set of algorithms, inspired by the biological neural networks in brains, for classification and regression tasks. There are various types of NNs with different neuron connection forms and architectures, e.g., fully-connected neural networks (FC-NN), convolutional neural networks (CNN), and recurrent neural networks (RNN). In this work, the feedforward FC-NN is considered, where the neurons of adjacent layers are fully connected and outputs of each layer are fed forward as the inputs to the next layer. A FC-NN defines a mapping from the input layer $\mathbf{z}_0 \in \mathbb{R}^{n_0}$ to the output $\mathbf{z}_L \in \mathbb{R}^{n_L}$. The layers between the input and output layers are called hidden layers $\mathbf{z}_l$, where $l = 1, \ldots, L-1$. By convention, a neural network with more than one hidden layer is called a "deep" NN. Mathematically, two adjacent layers are connected as,

$$\mathbf{z}_l = \sigma_l(\mathbf{W}_l^T \mathbf{z}_{l-1} + \mathbf{b}_l), \tag{3}$$

where $\mathbf{W}_l \in \mathbb{R}^{n_{l-1} \times n_l}$ and $\mathbf{b}_l \in \mathbb{R}^{n_l}$ are the weight matrix and bias vector; the subscript l denotes the index of the layer; $\sigma_l(\cdot)$ is an activation function acting element-wise, for which a number of options can be chosen, e.g., sigmoids, rectified linear units (ReLU), and tanh functions. After training, the weights, bias, and activation function at each layer are determined, and the output prediction $\mathbf{z}_L$ (i.e., velocity and pressure) can be rapidly computed from any given input vector $\mathbf{z}_0$ (i.e., coordinates and parameters) based on the Eq. (3). Since this feedforward algorithm (Eq. (3)) only involves a few matrix multiplications, the computational cost for evaluating the trained FC-NN can be neglected compared to that of a CFD simulation.

Traditionally, to build a surrogate model for the CFD simulation of the solution $\mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta})$, one can simply consider a black-box surrogate, e.g., FC-NN, or CNN [18], as $\mathbf{z}_L(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b})$, i.e.,

$$\mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta}) \approx \tilde{\mathbf{f}}(t, \mathbf{x}, \boldsymbol{\theta}) \triangleq \mathbf{z}_L(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}), \tag{4}$$

where $\mathbf{f}$ is the solution vector as $\mathbf{f} = \begin{bmatrix} \mathbf{u} & p \end{bmatrix}^\intercal$, including velocity $\mathbf{u}$ and pressure p; $\mathbf{W}$ and $\mathbf{b}$ denote the weights and biases of the entire network. Generally, training of a DNN is purely data-driven, and it consists of finding a set of (sub)optimal DNN parameters $(\mathbf{W}, \mathbf{b})$ such that the mismatch between the training data $\mathbf{f}^d$ and the DNN predictions $\tilde{\mathbf{f}}$ is locally minimized. That is, one can formulate an optimization problem as,

$$\mathcal{L}_{data}(\mathbf{W}, \mathbf{b}) = \left\| \mathbf{f}^d(t, \mathbf{x}, \boldsymbol{\theta}) - \mathbf{z}_L(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}) \right\|_{\Omega_{f,t}}, \tag{5a}$$

$$\mathbf{W}^*, \mathbf{b}^* = \arg\min_{\mathbf{W},\mathbf{b}} \mathcal{L}_{data}(\mathbf{W}, \mathbf{b}), \tag{5b}$$

where the loss function $\mathcal{L}_{data}(\mathbf{W}, \mathbf{b})$ is named as "data-based loss", and $\|\cdot\|_{\Omega_{f,t}}$ is L_2 norm over $\Omega_{f,t}$; $\mathbf{W}^*, \mathbf{b}^*$ denote a set of (sub)optimal NN weights and biases obtained from the optimization.

However, as discussed above, this black-box surrogate modeling requires enormous training data $\mathbf{f}^d$, which is too expensive to obtain from a large number of CFD simulations. Instead, following previous PINN framework [33], we consider leveraging the governing PDEs in the loss function by minimizing the violation of the solution $\mathbf{z}_L$ in terms of the known governing PDEs for fluid dynamics over a domain of interests without the needs of solving these equations for each parameter with traditional numerical methods. Specifically, only the residuals of the Navier–Stokes equations are computed based on the FC-NN predictions and it corresponds to a constrained optimization as follows,

$$\mathcal{L}_{phy}(\mathbf{W}, \mathbf{b}) = \underbrace{\left\| \nabla \cdot \tilde{\mathbf{u}} \right\|_{\Omega_{f,t}}}_{\text{Mass conservation}} + \underbrace{\| \frac{\partial \tilde{\mathbf{u}}}{\partial t} + (\tilde{\mathbf{u}} \cdot \nabla)\tilde{\mathbf{u}} + \frac{1}{\rho}\nabla \tilde{p} - \nu \nabla^2 \tilde{\mathbf{u}} + \mathbf{b}_f \|_{\Omega_{f,t}}}_{\text{Momentum conservation}}, \tag{6}$$

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \mathcal{L}_{phy}(\mathbf{W}),$$
$$\text{s.t.} \begin{cases} \mathcal{I}(\mathbf{x}, \tilde{p}, \tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0, & t = 0, \text{ in } \Omega_f, \\ \mathcal{B}(t, \mathbf{x}, \tilde{p}, \tilde{\mathbf{u}}, \boldsymbol{\theta}) = 0, & \text{on } \partial\Omega_{f,t}, \end{cases} \tag{7}$$

where the loss function $\mathcal{L}_{phy}(\mathbf{W}, \mathbf{b})$ here is named as "physics-based loss"; To construct the PDE residuals in the loss function, several first and/or second derivative terms of $\tilde{\mathbf{u}}$ and $\tilde{p}$ with respect to time t and space coordinates $\mathbf{x}$ are required, which can be computed based on automatic differentiation (AD) [59]. AD is an accurate and efficient way to calculate derivatives in a computational graph, which has started to gain increasing attention in the machine learning community. The general idea of AD is to use the chain rule to back-propagate derivatives from the output layer to the inputs as the connection between each layer of a NN is analytically defined. Compared to numerical differentiation techniques, derivatives calculated from AD are much more accurate since they do not suffer from truncation or round-off errors. Most modern deep learning frameworks such as PyTorch [60], TensorFlow [61], and Theano [62] have the AD implemented. To solve the optimization problem defined in Eq. (6), stochastic gradient descent (SGD) algorithms are used, which are known to be a stochastic approximation of the gradient descent (GD) optimization. In SGD, only a subset of points are randomly sampled from the input space to calculate the direction of the gradient at each iteration. The SGD algorithms are known to work very well to escape bad local minima in the neural network training [63] under one point convexity property. Although the global minimum cannot be guaranteed for a non-convex optimization problem as defined in Eq. (6), an empirically good local minimum is usually found based on the SGD algorithms.

2.3. Boundary condition enforcement

If the physics-based loss $\mathcal{L}_{phy}$ becomes identically zero, the DNN predictions of velocity $\tilde{\mathbf{u}}$ and pressure $\tilde{p}$ will exactly satisfy the Navier–Stokes equations (Eq. (1)). Therefore, penalizing the PDE residuals can regularize the data-driven DNN solutions to be more physical. This idea is known as the physics-informed, weakly-supervised deep learning [33,38,45]. To make the problem well-posed, proper initial and boundary conditions (IC/BC) are needed and imposed as constraints (Eq. (7)) which are often treated in a "soft" manner by modifying the original loss function with penalty terms [33,64]. For example, the IC/BC can be imposed in a "soft" way by modifying Eq. (6) as,

$$\mathcal{L}^c_{phy}(\mathbf{W}, \mathbf{b}, \lambda_i, \lambda_b) = \underbrace{\mathcal{L}_{phy}(\mathbf{W}, \mathbf{b})}_{\text{Equation loss}} + \underbrace{\lambda_i \left\| \mathcal{I}(\mathbf{x}, \tilde{p}, \tilde{\mathbf{u}}, \boldsymbol{\theta}) \right\|_{\Omega_{f,t}}}_{\text{Initial loss}} + \underbrace{\lambda_b \left\| \mathcal{B}(t, \mathbf{x}, \tilde{p}, \tilde{\mathbf{u}}, \boldsymbol{\theta}) \right\|_{\partial\Omega_{f,t}}}_{\text{Boundary loss}}, \tag{8}$$

where λ_i and λ_b are penalty coefficients. However, the soft IC/BC enforcement methods have several major drawbacks: (1) there is no quantitative guarantee on how accurate the IC/BC being imposed and thus the solution could be unsatisfactory; (2) the optimization performance can depend on the relative importance of each term, but how to assign weight for each term can be difficult. Alternatively, we can impose the IC/BC in a "hard" manner, where a particular solution that solely satisfies the initial/boundary condition is added. Hence, the constraints on IC/BC are automatically fulfilled. A mixed enforcement on IC/BC is proposed in this work, where the Neumann and Dirichlet boundary conditions (BC) are treated separately: the Neumann BC are formulated into the equation loss, i.e., in a soft manner, while the IC and Dirichlet BC are encoded in a hard manner by constructing the DNN ansatz $\hat{\mathbf{u}}$ and $\hat{p}$ with a particular solution as follows,

$$\begin{aligned} \hat{\mathbf{u}}(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}) &= \mathbf{u}_{par}(t, \mathbf{x}, \boldsymbol{\theta}) + D(t, \mathbf{x}, \boldsymbol{\theta})\tilde{\mathbf{u}}(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}), \\ \hat{p}(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}) &= p_{par}(t, \mathbf{x}, \boldsymbol{\theta}) + D(t, \mathbf{x}, \boldsymbol{\theta})\tilde{p}(t, \mathbf{x}, \boldsymbol{\theta}; \mathbf{W}, \mathbf{b}), \end{aligned} \tag{9}$$

where $\mathbf{u}_{par}$ is a particular solution that just satisfies IC and BC: $\mathbf{u}_{par}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$, $p_{par}(\mathbf{x}, 0) = p_0(\mathbf{x})$ and $\mathbf{u}_{par}(\mathbf{x}, t)|_{\mathbf{x}\in\partial\Omega_f} = \mathbf{u}^b(\mathbf{x})$, $p_{par}(\mathbf{x}, t)|_{\mathbf{x}\in\partial\Omega_f} = p^b(\mathbf{x})$; $D(t, \mathbf{x}, \boldsymbol{\theta})$ is a globally defined smooth function from internal points to the "boundary" in $\Omega_{f,t}$, i.e., a space–time sense. That is, D is zero on the boundary $\partial\Omega_f \times [0, T]$ and $\Omega_f \times \{0\}$ while increases away from the boundary. For those problems where the IC/BC and the geometry of the domain in $\Omega_{f,t}$ is simple, the function D and particular solution can be written analytically. However, if the geometry is too complex to have an analytic form, e.g., a patient-specific artery, both the particular solution ($\mathbf{u}_{par}$, p_{par})

and smooth distance function (D) can be parameterized by several pre-trained NNs similarly as [50]. These pre-trained NNs can be designed to over-fit the boundary values, since they are only used to represent the particular solutions or distance function as differentiable forms. Finally, the constrained optimization problem in Eq. (6) can be reformulated as an unconstrained one, as shown in Eq. (10).

$$\mathcal{L}^c_{phy}(\mathbf{W},\mathbf{b}) = \underbrace{\left\|\nabla\cdot\hat{\mathbf{u}}\right\|_{\Omega_f}}_{\text{Mass conservation}} + \underbrace{\|\frac{\partial\hat{\mathbf{u}}}{\partial t}+(\hat{\mathbf{u}}\cdot\nabla)\hat{\mathbf{u}}+\frac{1}{\rho}\nabla\hat{p}-\nu\nabla^2\hat{\mathbf{u}}+\mathbf{b}_f\|_{\Omega_f}}_{\text{Momentum conservation}} \tag{10a}$$

$$\mathbf{W}^*,\mathbf{b}^* = \arg\min_{\mathbf{W},\mathbf{b}} \mathcal{L}^c_{phy}(\mathbf{W},\mathbf{b}), \tag{10b}$$

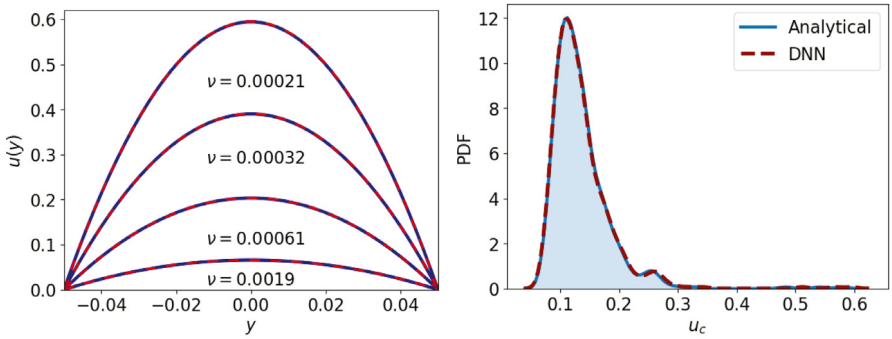
Note that since the Neumann BCs were not studied numerically in this work, the loss term associated with Neumann BC is omitted in Eq. (10) and thus the total loss only contains the L_2 residual of governing equations.

3. Numerical results

A number of 2D vascular flows with idealized geometries, including circular pipe flows, stenotic flows, and aneurysmal flows, are studied to evaluate the performance of the proposed data-free DL surrogate model in Eq. (10). Forward propagation of the uncertainties in the fluid properties and domain geometry to the flow QoIs (e.g., velocity, pressure, and wall shear stress) are investigated through the DNN surrogate model. In this study, only steady-state solutions are considered for proof-of-concept, thus the constraint of initial condition can be neglected in Eq. (9).

A composite FC-NN architecture is devised for the surrogate, which is composed of three sub-DNNs with an identical structure of 3 hidden layers with 20 neurons per layer. The Swish activation function [65], defined as $x\cdot\text{Sigmoid}(\beta x)$ with a fixed hyperparameter $\beta=1$, is employed in each layer except the last one, where a linear activation function is used. The three sub-DNNs share the same input layer and separately predict three scalar state variables, i.e., velocity u, v, and pressure p. All three sub-DNNs are trained simultaneously with a unified physics-based loss function. To solve the unconstrained optimization problem defined in Eq. (10), we used the Adam optimizer [66], a robust variant of the SGD method, where the learning rate is adaptively changed based on the estimates of the moments. The initial learning rate and mini-batch size are set as 1×10^{-3} and 50, respectively. Because of the adaptivity feature of the Adam optimizer, the hyperparameters in the training are robust to some extent and require little tuning. Properly initializing the DNN parameters is also important. After comparing several widely-used initialization schemes [67,68], we chose the He's normal initializer [68], where initial weights are drawn from a truncated normal distribution. In general, a good choice of the DNN architecture, including the number of layers, number of neurons per layer, activation function, and initialization schemes, is important to the learning performance but is still determined by trial and error. A rule of thumb is to achieve the desired learning performance using the "simplest" network structure, which is known as "Occam's razor" [69]. Namely, a DNN structure with the minimum number of layers and neurons still having the desired performance is preferable, because it usually enables efficient training and better generalizability. To search for such a structure, a series of tests were conducted. Specifically, we studied the ν-varying stenosis case using a group of DNN structures with different number of hidden layers and neurons, from the shallowest network with only 1 hidden layer of 10 neurons per layer to the largest one with 5 hidden layers of 40 neurons per layer. By comparing the test errors of all cases (Table 5), we found the test errors do not notably decrease with the DNN structures of more than 3 hidden layers and 20 neurons in each layer, which justified the current choice of the DNN structure. More details can be found in Appendix A. To demonstrate the robustness of the physics-constrained learning, the architecture and hyperparameters remain the same for all the cases throughout the paper. Note that a comprehensive parameter study and architecture optimization of the DNN are out of the scope of the current work.

The composite FC-NN is implemented in the PyTorch platform [60]. As discussed in Section 2, only the collocation points are required and they are uniformly sampled in the spatial $\mathbf{x}$ and parameter $\boldsymbol{\theta}$ spaces. Alternatively, one can choose a space-filling Latin hypercube sampling in the Ω_f [33]. In this work, the PDE residuals are extensively evaluated on a large number of collocation points to ensure learning quality. For all test cases, the training of about 10^6 SGD iterations are performed on an NVIDIA GeForce GTX 1080 Ti Graphics Processing Unit (GPU) card, and the cost is approximately 3.5 h. Note that the offline training cost can be potentially reduced by optimizing the DNN hyperparameters. To validate the prediction performance of the trained DNN surrogates, corresponding CFD simulations are also conducted using an open-source FV-based CFD solver, OpenFOAM [70]. Mesh convergence study is performed to ensure the solution accuracy. The code and datasets for this work will become available at https://github.com/Jianxun-Wang/LabelFree-DNN-Surrogate upon publication.



(a) Cross-section velocity profiles (b) Distribution of center velocity

Fig. 2. (a) DNN predicted cross-section pipe flow velocity profiles $u(y)$ of four different viscosity (ν) samples compared with the analytical solution; (b) probability density function of the center velocity $u_c = u(y = 0)$ propagated from a truncated Gaussian distributed ν with mean $\overline{\nu} = 10^{-3}$ and variance of $\sigma_\nu = 2.67 \times 10^{-4}$, using the trained DNN surrogate compared with the analytical solution.

3.1. Circular pipe flow

Flow in a pipe/tube is very common in physiological systems, e.g., blood in arteries or air flow in trachea. The pipe flow is often driven by the pressure difference between the two ends of a tube, or by the body force of gravity. In a cardiovascular system, the former one is more dominant since the blood flow is mainly governed by pressure drop due to the heart pumping. In general, simulating the fluid dynamics in a tube requires solving the full Navier–Stokes equations numerically, but if the tube is straight and has a constant circular cross section, analytical solution of the fully-developed steady-state flow is available, which is an ideal benchmark to validate the performance of the proposed method. As a result, we first study the flow in a 2D circular pipe (also known as the Poiseuille flow).

In this case, the pressure inlet and outlet are used to drive the flow since we only focus on the fully-developed regime, and no-slip wall boundary is prescribed on the tube walls. The boundary conditions are encoded into the surrogate model by constructing the DNN ansatz $\hat{u}$, $\hat{v}$, and $\hat{p}$ based on Eq. (9). The no-slip condition of velocity on the wall can be imposed by designing the $\hat{u}$, $\hat{v}$ as,

$$\hat{u} = \left(\frac{d^2}{4} - y^2\right)\tilde{u}, \qquad \hat{v} = \left(\frac{d^2}{4} - y^2\right)\tilde{v}, \tag{11}$$

where y is the radial distance, $d = 0.1$ is the diameter of the tube, the raw DNN output is denoted by $\tilde{u}$.

The pressure inlet $p_{in} = 0.1$ and outlet $p_{out} = 0$ are imposed by designing the $\hat{p}$ as,

$$\hat{p} = \frac{x - x_{in}}{x_{out} - x_{in}} p_{out} + \frac{x_{out} - x}{x_{out} - x_{in}} p_{in} + (x - x_{in})(x_{out} - x)\tilde{p}, \tag{12}$$

where x_{in} and x_{out} are coordinates of the two ends of the tube, and the raw DNN output is denoted by $\tilde{p}$. All three sub-DNNs are trained to capture the spatial flow fields with parameter variation in the fluid viscosity ν. Input (collocation) points in the parameter space of ν are uniformly sampled in the range $2 \times 10^{-4} \leq \nu \leq 1.9 \times 10^{-3}$, where the corresponding Reynolds number (Re) are moderate ($Re < 300$). After training, both velocity and pressure fields can be obtained immediately by evaluating the trained DNN with any given input ν and a spatial grid on $\mathbf{x}$. Hence, the DNN surrogate can be utilized to propagate the uncertainty in viscosity ν based on the Monte Carlo (MC) simulation, where a large number of samples are drawn from the ν distribution and propagated to the QoIs via the DNN surrogate. In the following test cases, 500 MC samples are used to compute desired statistics and distributions.

The DNN surrogate results (shown in Fig. 2) are compared against the analytical solution, which is given by

$$u_a = \frac{\Delta p}{2\nu\rho L}\left(\frac{d^2}{4} - y^2\right), \tag{13}$$

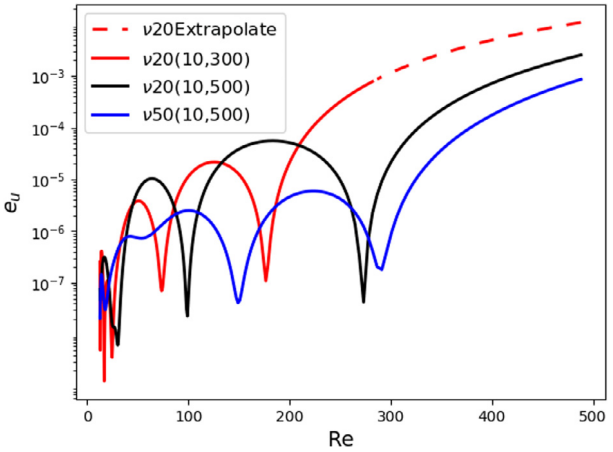


Fig. 3. Learning performance in different ranges of Reynolds number Re. Solid lines indicate interpolation (testing within the training range) and dashed lines indicate extrapolation; $\nu 20$ and $\nu 50$ indicate that the number of collocation points in ν is 20 and 50, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where y denotes the spanwise coordinate and Δp is the pressure difference. It can be observed from Fig. 2a that the DNN-predicted velocity profiles (red dashed lines) of four different ν samples almost exactly agree with the analytical solutions (blue solid lines), where the Reynolds numbers (Re) of the four cases are 283, 121, 33, and 3, respectively. Actually, the trained DNN is able to accurately predict the pipe flow field with any given viscosity, where the Reynolds number is moderate. Fig. 2b shows the uncertainty of the center velocity u_c propagated from a truncated Gaussian distributed ν with mean of $\overline{\nu} = 1 \times 10^{-3}$ and variance of $\sigma_\nu = 2.67 \times 10^{-4}$ to guarantee the viscosity ν is always positive random variable. Namely, the truncated Gaussian distribution is defined with in the range of $(0, +\infty)$, and the probability density function f is shown as,

$$f(\nu; \overline{\nu}, \sigma_\nu) = \frac{\frac{1}{\sigma_\nu}\mathcal{N}(\frac{\nu-\overline{\nu}}{\sigma_\nu})}{1-\phi(-\frac{\overline{\nu}}{\sigma_\nu})}, \nu > 0 \tag{14}$$

where $\mathcal{N}$ is the standard Gaussian density, and $\phi(x) = \frac{1}{2}\left(1 + \text{erf}(\frac{x}{\sqrt{2}})\right)$, with $\text{erf}(z) = \frac{2}{\sqrt{\pi}}\int_0^\pi e^{-t^2}dt$. It can be seen that the DNN-predicted probability density function (PDF) is almost identical to that of the analytical solutions, demonstrating the excellent performance of the physics-constrained learning for uncertainty propagation.

To comprehensively evaluate the learning performance, mean square errors of the DNN predictions are calculated and normalized by Δp. It is interesting to examine how well the predictions are across different Reynolds numbers Re in both the training range and the range of extrapolation. In current training setting, collocation points of ν are sampled from $[2 \times 10^{-4}, 1.9 \times 10^{-3}]$, corresponding to $Re \in [10, 300]$ as the training range. The trained DNN is also tested on $Re \in (300, 500]$, which is the range of extrapolation. Moreover, we also investigated the impact of Re on the learning performance by using a wider Re range for training. The mean square test error $e_{\mathbf{u}}$ for each case are plotted against the Reynolds number Re in Fig. 3, where solid lines indicate interpolation (testing within the training range) and dashed line indicates extrapolation.

It is clear that the test error in the interpolation range ($Re \in [10, 300]$, red solid) is relatively lower than that in the extrapolation range ($Re \in (300, 500]$, red dashed). If the training set includes $Re \in (300, 500]$, the test error can be largely reduced by nearly an order of magnitude (see the comparison of black solid line and red dashed line). This is expected since the success of deep learning largely relies on the interpolatory nature of the problem, and the extrapolation is always much more challenging. However, the generalization ability is not critical in the label-free learning framework since any collocation points can be freely drawn in the range of interest for training. Moreover, the effect of the total number of collocation points was studied and we found that the test error can be further reduced if more collocation points are sampled during the training. This can be observed by comparing the blue and black lines, which correspond to the cases trained with 50 and 20 collocation points of ν (or Re), respectively. Lastly, it can be seen that the test error tends to grow as the Re increases, regardless of interpolation or

extrapolation, which indicates that the flow solutions with higher Reynolds are more difficult to be captured by the DNN. It might be because steep gradients of flows with higher Re pose more challenges on learning. We refined ourselves to the laminar flow regime, and higher-Re turbulent flows is out of the scope of this work.

3.2. Blood flow with standardized geometry

Two types of canonical vascular flows, stenotic flow and aneurysmal flow, with standardized vessel geometries are studied. A stenotic flow refers to the flow through a vessel where there is narrowing and re-expansion of the vessel wall. This local restriction of the vessel is related to many cardiovascular diseases, e.g., arteriosclerosis, stroke, and heart attack [71]. The vascular flow within an aneurysm, which is the expansion of an artery due to the weakness of vessel walls, is called aneurysmal flow. The rupture of an aneurysm may cause life-threatening conditions, e.g., subarachnoid hemorrhage (SAH) due to cerebral aneurysm rupture [72], and investigation of the hemodynamics can improve the diagnosis and fundamental understanding of aneurysm progression and rupture [73].

Whereas realistic vascular geometries are usually irregular and complex, including sites of curvature, bifurcation and junctions, idealized stenosis and aneurysm models are studied here for proof-of-concept. Namely, both the stenotic and aneurysmal vessels are idealized as an axisymmetric tube with a varying cross-section radius, which is parameterized by the following function,

$$R(x) = R_0 - A\frac{1}{\sqrt{(2\pi\sigma^2)}}\exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{15}$$

where R_0 is the radius at the inlet, which is set as 0.05, and the sign of A determines if the vessel is stenotic or aneurysmal. Namely, a positive and negative sign correspond to the stenosis and aneurysm, respectively. Three control parameters A, μ, and σ define the shape of the stenotic (aneurysmal) vessel. The scale parameter A controls the curvature along the tube, and a larger $|A|$ leads to narrower stenosis (broader aneurysm). Parameter μ defines the streamwise location of the minimum (maximum) radius of the stenosis (aneurysm), and σ affects the steepness of the geometric variation. In this study, the latter two parameters μ and σ are fixed as 0.5 and 0.1, respectively. Only A is considered as a variable parameter to control the degree of the stenosis (aneurysm).

Similar to the pipe flow, the pressure inlet/outlet and no-slip wall boundary conditions are prescribed for both stenosis and aneurysm cases. The boundary-encoded sub-DNNs are constructed to learn the parametric flow solutions, where the wall BC is imposed using the geometric function $D(x, y) = R(x)^2 - y^2$. Contrary to the case studied above, where only the viscosity variation is considered, the DNN surrogate is also trained to capture the varying stenosis (aneurysm) geometry, which is known challenging for mesh-based CFD simulations. Specifically, the solutions of varying viscosity are learned for fixed vascular geometries ($A = 5 \times 10^{-3}$ for stenosis and $A = -5 \times 10^{-3}$ for aneurysm) in the first place, and then the performance for capturing geometry variations is examined at a fixed viscosity ($\nu = 1 \times 10^{-3}$). Moreover, the uncertainties from the flow viscosity and vessel geometry are propagated to the QoIs through the trained DNN surrogate using MC sampling, and the results are validated by CFD-based MC simulations.

3.2.1. Flow in idealized stenosis

The DNN is trained to parameterize the solutions of stenotic flows with varying viscosity, where collocation points are sampled in ν space within the range of $[5 \times 10^{-4}, 1 \times 10^{-2}]$ for physics-based training.

Fig. 4 shows the DNN-predicted flow fields of three different viscosity samples, i.e., $\nu = 6.4 \times 10^{-4}$, 1.85×10^{-3}, and 2.14×10^{-3}, at moderate Reynolds numbers. The corresponding CFD simulations are performed for comparison. It can be seen that the flow patterns of different ν are similar, where the fluid is accelerated streamwisely through the converging region and slows down passing the diverging part of the tube. However, the velocity magnitude reduces as the viscosity increases (left to right columns). As shown in Fig. 4a and b, the DNN-predicted velocity contours of streamwise and spanwise components (u and v) agree with the CFD solutions very well, though the magnitude in the case with the smallest viscosity ($\nu = 6.4 \times 10^{-4}$) is slightly underestimated. Moreover, the nonlinear pressure drops can be accurately captured by the DNN surrogate as the profiles of centerline pressure from the DNN and CFD are almost identical (Fig. 4c).

To learn the flow solutions with varying geometry, the DNN is trained on uniformly sampled points within the range of $0 \leq A \leq 1 \times 10^{-2}$.

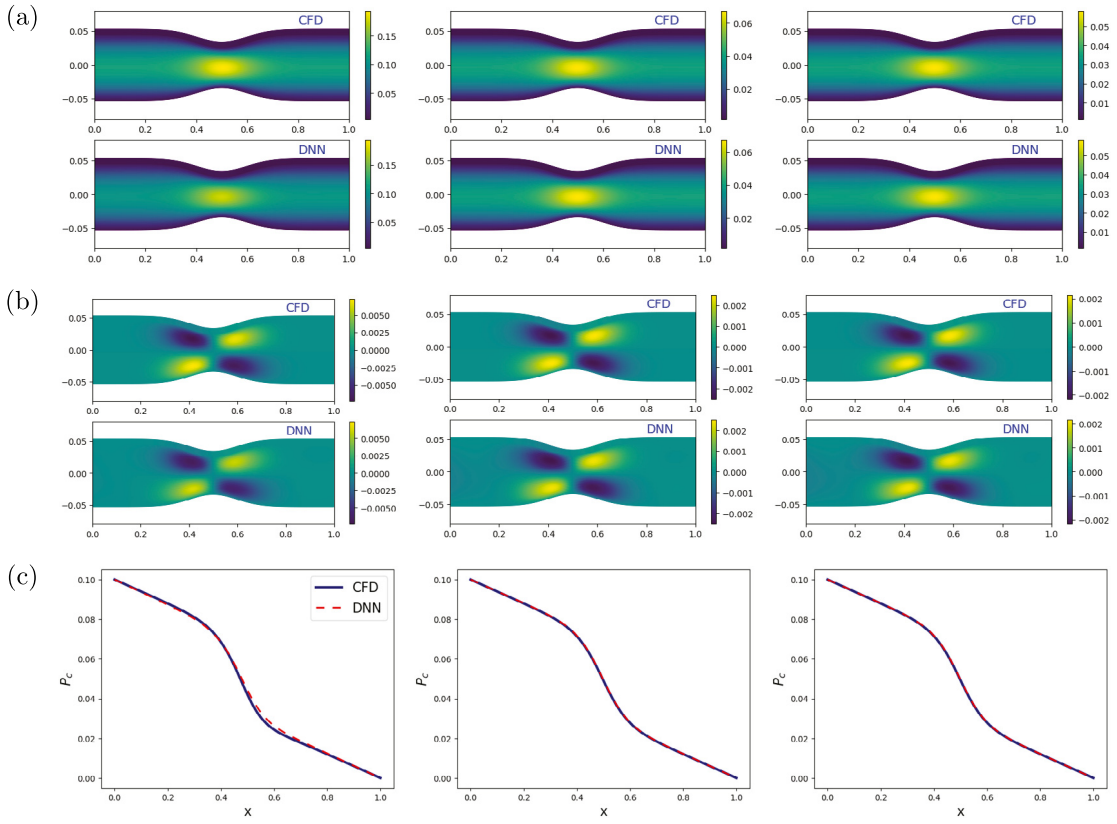


Fig. 4. Comparison between physics-constrained DNN predictions and CFD solutions of idealized stenotic flows at three different viscosity samples: (left) $\nu = 6.4 \times 10^{-4}$, (middle) $\nu = 1.85 \times 10^{-3}$, (right) $\nu = 2.14 \times 10^{-3}$. (a) Streamwise velocity component u (b) spanwise velocity component v (c) centerline pressure profile P_c.

We compare the flow fields predicted by the trained DNN against the CFD benchmarks, and the results of three different samples of stenosis geometries, $A = 2 \times 10^{-3}$, 4×10^{-3}, and 7×10^{-3}, are shown in Fig. 5. From left to right, the degree of stenosis increases and thus the total flow rate is reduced due to the increased resistance (Fig. 5a and b). The pressure drop becomes more nonlinear as the stenotic vessel turns to be narrower (Fig. 5c). We can see that the DNN predictions can capture these flow features and agree with the CFD benchmarks well. Admittedly, the streamwise velocity of the flow with the narrowest stenotic vessel is slightly underestimated, and the centerline pressure profiles predicted by the DNN and CFD has a small discrepancy. This might be because the increased nonlinearity due to the steep geometric variation poses a challenge on the learning.

After the training, the DNN surrogates are used to rapidly propagate uncertainties in viscosity and vessel geometry, and the effects on QoIs are investigated. Specifically, 500 MC samples are drawn from a truncated Gaussian distribution of viscosity ν and a Gaussian distribution of geometric parameter A, which are propagated to the center velocities u_c (at $x = 0.5$, $y = 0.0$) through the DNN surrogates. The probability distributions of u_c due to uncertain viscosity and vessel geometry are shown in Fig. 6a and b, respectively, where the propagated results through the CFD solver are also plotted for comparison.

It shows that the propagated distributions of u_c are non-Gaussian in both cases, which is due to the strong nonlinearity of the Navier–Stokes operator. As expected, the DNN-propagated uncertainties present a good agreement with the CFD-based benchmarks, especially in the case of viscosity uncertainty propagation (Fig. 6a), where the two PDF curves are almost overlapped with each other. As for the geometry uncertainty propagation, although the overall feature of the PDF is captured, the peak density is slightly underpredicted by the DNN. The reason behind this could be that the DNN surrogate tends to underestimate the velocity magnitude in particular for a steep geometry variation.

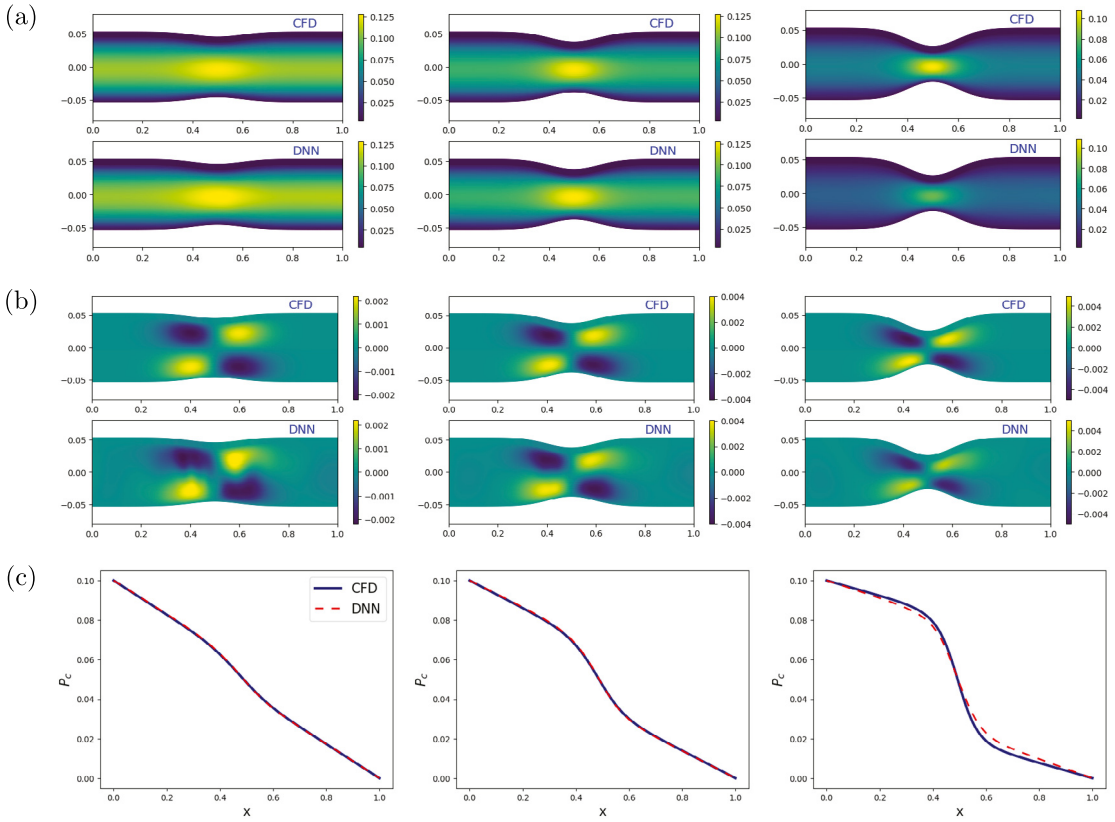
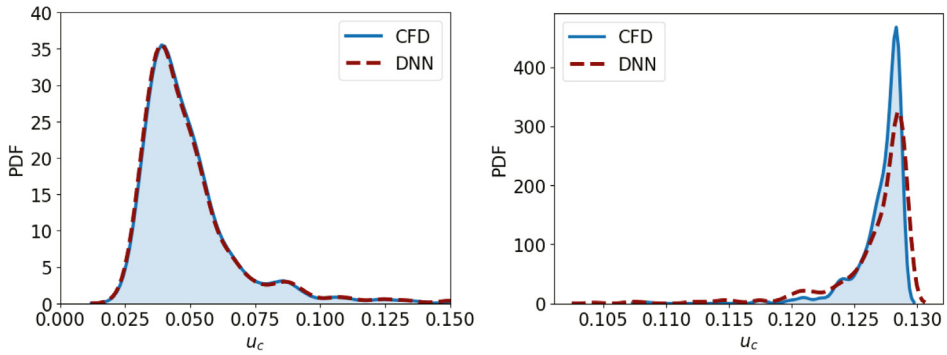


Fig. 5. Comparison between physics-constrained DNN predictions and CFD solutions of idealized stenotic flows with three different stenosis geometries: (left) $A = 2 \times 10^{-3}$, (middle) $A = 4 \times 10^{-3}$, (right) $A = 7 \times 10^{-3}$. (a) Streamwise velocity component u (b) spanwise velocity component v (c) centerline pressure profile P_c.



(a) Viscosity uncertainty propagation (b) Geometry uncertainty propagation

Fig. 6. Probability density of the center velocity u_c propagated from (a) a truncated Gaussian distributed viscosity ν and (b) a normally distributed geometric parameter A based on the trained DNN surrogate, compared against CFD-based MC solutions.

3.2.2. Flow in idealized aneurysm

We first learn the aneurysmal flows with varying viscosity, where the geometry of the aneurysm is fixed ($A = -5 \times 10^{-3}$).

Training is conducted by sampling the viscosity points ranging from 5×10^{-4} to 1×10^{-2}. Fig. 7 shows the DNN-predicted flow fields of three viscosity samples, i.e, $\nu = 6.4 \times 10^{-4}$, 1.85×10^{-3}, 2.14×10^{-3}, where the CFD

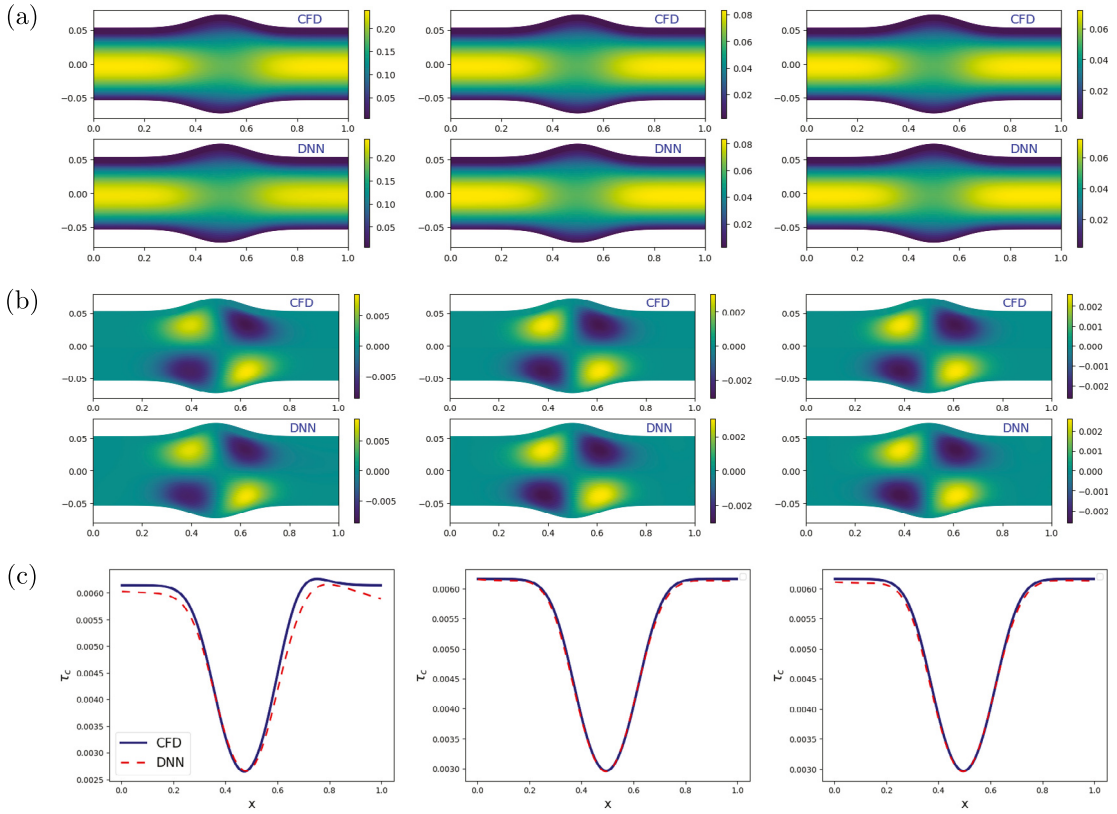


Fig. 7. Comparison between physics-constrained DNN predictions and CFD solutions of idealized aneurysmal flows at three different viscosity samples: (left) $\nu = 6.4 \times 10^{-4}$, (middle) $\nu = 1.85 \times 10^{-3}$, (right) $\nu = 2.14 \times 10^{-3}$. (a) Streamwise velocity component u (b) spanwise velocity component v (c) centerline wall shear profile τ_c.

benchmarks are plotted for comparison. In addition to the flow velocity and pressure, here we also investigate the wall shear stress (WSS) τ, which has been demonstrated as a critical factor affecting the aneurysm initialization, progression, and rupture [74]. It can be seen from Fig. 7 that the DNN-predicted flow QoIs are in a good agreement with the CFD solutions. The decrease of the flow velocity and the minimum WSS of the vessel are accurately captured by the DNN surrogate.

Fig. 8 shows the performance for learning the geometry-varying solutions of the aneurysmal flow, where the training is conducted by sampling the geometric parameter A, ranging from -2×10^{-2} to 0. The predicted flow fields of three different geometry samples are presented, where the size of the aneurysm increases from left to right. The flow decelerates through the expanded region of the vessel and the velocity at the center of the aneurysm is significantly reduced, in particular when the aneurysm becomes larger. It is observed from the contour comparisons (Fig. 8a and b), the DNN predictions agree with the CFD solutions pretty well. As for the WSS profile, its shape and magnitude vary as the geometry changes, which can be accurately captured by the DNN surrogate (Fig. 8c).

The uncertainty propagation using the trained DNN surrogates is then conducted.

Uncertainties in viscosity ν and geometric parameter A with a truncated Gaussian and Gaussian distributions are considered, and the QoIs are center velocity u_c and the minimum WSS τ_c, which are important for aneurysmal flows. As shown in Fig. 7c, the WSS remains invariant due to viscosity perturbation with a fixed vessel geometry, hence the distribution of u_c is studied for viscosity uncertainty propagation (shown in Fig. 9a). The PDF obtained by the DNN surrogate almost coincides with the CFD-based benchmark. As for the geometry uncertainty propagation, the center velocity u_c is not an interesting quantity since it almost remains the same as geometry changes. Instead, we investigate the propagated uncertainty in the minimum WSS τ_c, which is sensitive to geometry variation. It is observed from Fig. 9b that the probabilistic distribution of τ_c propagated by the DNN is in a favorable agreement with the CFD-based benchmark, though the peak of the density is slightly underestimated.

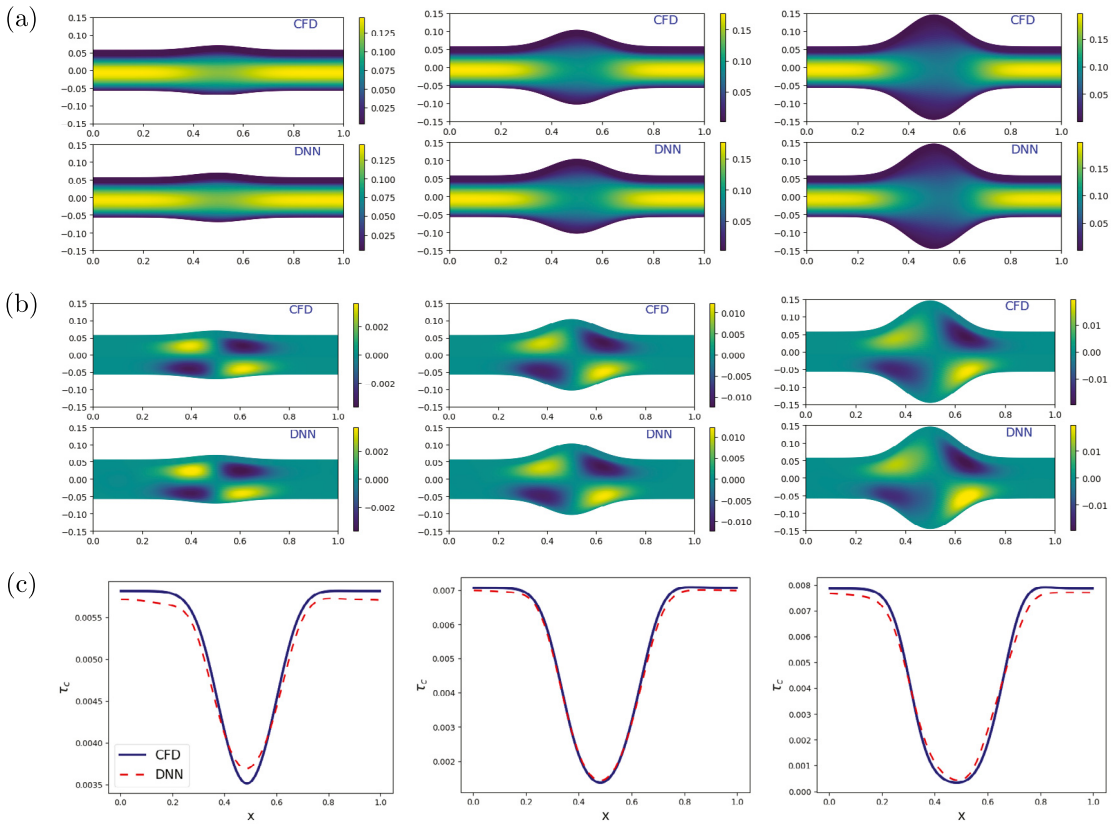
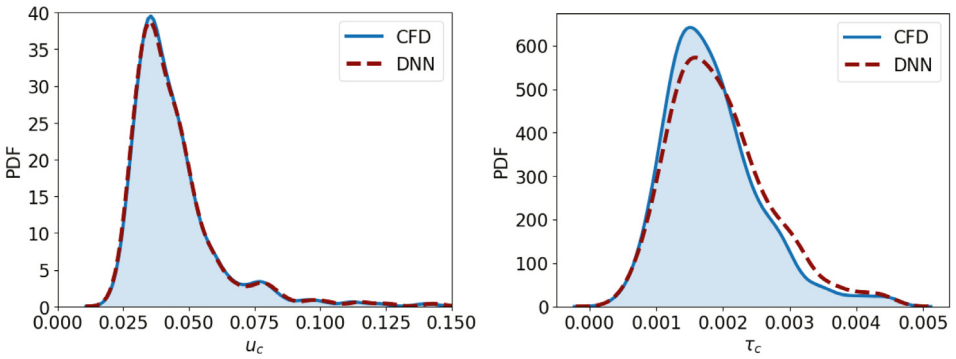


Fig. 8. Comparison between physics-constrained DNN predictions and CFD solutions of idealized aneurysmal flows of three different aneurysm geometries: (left) $A = -3 \times 10^{-3}$, (middle) $A = -1.2 \times 10^{-2}$, (right) $A = -2.2 \times 10^{-2}$. (a) Streamwise velocity component u (b) spanwise velocity component v (c) centerline wall shear profile τ_c.



(a) Viscosity uncertainty propagation (b) Geometry uncertainty propagation

Fig. 9. Probability density of the (a) center velocity u_c propagated from a truncated Gaussian distributed viscosity v and (b) the minimum wall shear τ_c propagated from a normally distributed geometric parameter A based on the trained DNN surrogate, compared against CFD-based MC solutions.

3.2.3. Summary of training and prediction performance

The training and testing performance on the vascular flow cases are summarized in Table 1, where the training loss is defined as the sum of L_2 norms of momentum and continuity equation residuals (see Eq. (10)) and test error

Table 1
Summary of learning/prediction performance of vascular flows with 20 parameter collocation points in parameter spaces for training.

	Stenosis		Aneurysm	
	Viscosity	Geometry	Viscosity	Geometry
Training loss	8×10^{-5}	1×10^{-3}	5.5×10^{-5}	2.9×10^{-5}
Test error e_u	8.18×10^{-5}	9.61×10^{-4}	9.2×10^{-5}	1.38×10^{-4}
Test error e_v	7.14×10^{-8}	1.76×10^{-6}	1.33×10^{-7}	1.15×10^{-6}
Test error e_p	2.33×10^{-5}	2.23×10^{-3}	5.81×10^{-6}	2.09×10^{-5}

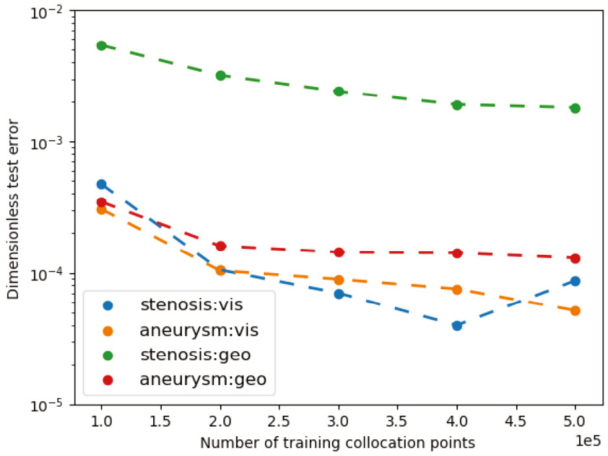


Fig. 10. Total test errors v.s. different number of training collocation points. The total collocation points is the multiplication of parameter point size (N_p) by geometric collocation point size ($N_g = 10^4$).

is defined as the normalized L_2 difference between DNN and CFD results, shown as,

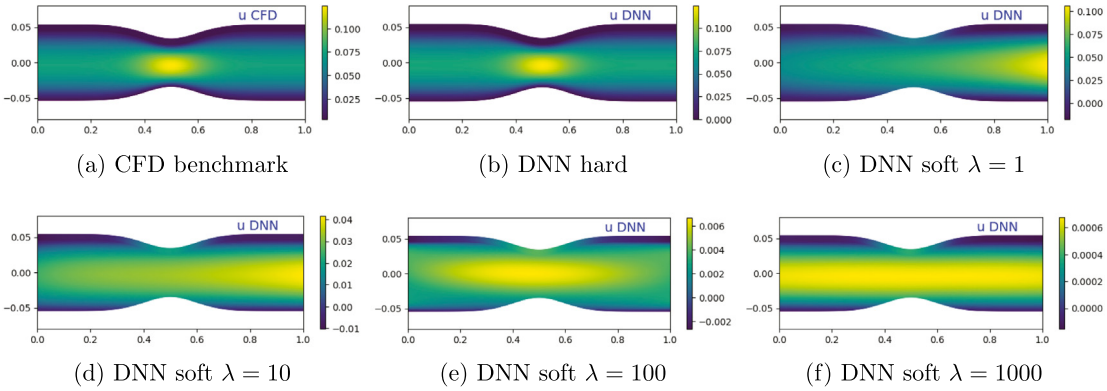
$$e_{\mathbf{u}} = \frac{\|\mathbf{u}_{DNN} - \mathbf{u}_{CFD}\|_2}{\Delta p} \tag{16a}$$

$$e_p = \frac{\|p_{DNN} - p_{CFD}\|_2}{(\Delta p)^2} \tag{16b}$$

where pressure drop $\Delta p = 0.1$ is used to normalize the errors as it has the same dimension as $\rho \mathbf{u}^2$ and ρ is set 1 in all cases. As a result, the prediction performance is evaluated by calculating these dimensionless mean square errors. It can be seen from Table 1 that both the training loss and test errors are reasonably small for all cases after training, and the test error in u is more dominant compared to that in v. Among the four scenarios, the geometric variation of the stenosis is the most challenging to learn since the training loss (i.e., equation residual) remains relatively large and the test errors are one-order bigger than the other cases. This is consistent with the notable discrepancy in the DNN-predicted PDF observed in Fig. 6b.

All the cases presented above are trained on $N_p = 20$ parameter collocation points. Namely, the equation residuals are minimized on 20 collocation points uniformly sampled from the parameter space. It is necessary to check if the size N_p of parameter collocation points is sufficiently large for the training. Therefore, we conduct a parameter study using different amounts of parameter collocation points for training. The total test errors (sum of test errors in u, v, and p) against different numbers of training collocation points for all four cases are presented in Fig. 10.

As expected, the errors generally decrease as the number of training collocation points increases. However, the error decreasing rate is quite mild. The test errors remain approximately unchanged when $N_p > 20$ for all cases while the training cost will increase as more collocation points are used, which justifies the sufficiency of the total collocation points used in this work. Detailed results of training cost and testing errors for all cases are summarized by Tables 4 and 5 in Appendix B.



(a) CFD benchmark (b) DNN hard (c) DNN soft $\lambda = 1$

(d) DNN soft $\lambda = 10$ (e) DNN soft $\lambda = 100$ (f) DNN soft $\lambda = 1000$

Fig. 11. Physics-constrained learning results for a stenotic flow ($A = 5 \times 10^{-3}$, $\nu = 1 \times 10^{-3}$) with (b) hard BC constraint, compared to those using soft BC constraints with (c) $\lambda = 1$, (d) $\lambda = 10$, (e) $\lambda = 100$, (f) $\lambda = 1000$.

Lastly, we briefly discuss the computational cost of the uncertainty propagation tasks presented above, which is determined by the forward model-evaluation time and number of MC samples. The online evaluation of the trained DNN surrogate is very fast, and the cost of each forward DNN evaluation is less than 2×10^{-2} CPU seconds. In contrast, the CFD simulation even for simple cases (e.g., stenotic and aneurysmal flows with a 2D mesh of 10^4 quadrilateral grids) takes around 40 CPU seconds. Therefore, remarkable speedups can be expected considering many-query applications like uncertainty propagation, and this advantage can be considerable when a larger number of forward evaluations or more complicated fluid systems are considered. Moreover, the FV/FE-based CFD simulations require mesh generation, which is often a manually-cumbersome and labor-intensive process, in particular for the flows with complex geometries and moving boundaries, e.g., patient-specific cardiovascular simulations. Therefore, the mesh-free feature of the proposed method is preferable in these applications.

4. Discussion

4.1. Pitfall on using soft boundary enforcement

The initial and boundary conditions (IC/BC) can be imposed in the physics-constrained learning either as a soft or hard constraint. When no labeled data is used in training, a properly enforced IC/BC is crucial to ensure the uniqueness of the learned PDE solutions. Although we have demonstrated the effectiveness of the hard enforcement approach in Section 3, it is still interesting to investigate the performance of soft enforcement method [33] in the purely PDE-driven training. Hence, all the test flows are studied again using physics-constrained learning, where BCs are imposed in a soft manner (as Eq. (17)). Namely, the BCs are formulated as a boundary loss component $\mathcal{L}_B$, which is incorporated into the physics-based loss function $\mathcal{L}_{phy}$ as,

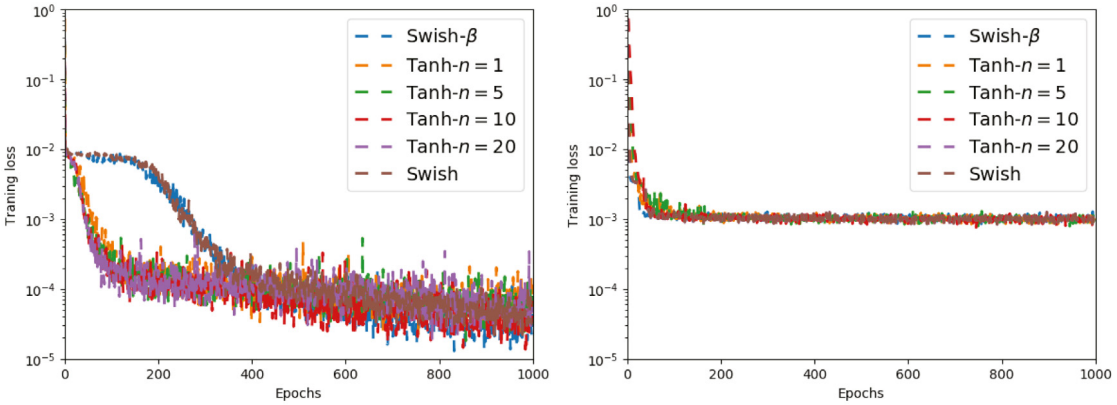
$$\mathcal{L}^c_{phy} = \mathcal{L}_{phy} + \lambda \mathcal{L}_B, \tag{17}$$

where λ is the penalty coefficient. For the circular pipe flow case, both the hard and soft BC constraints can lead to excellent learning and prediction performance (results not shown). However, when the radius varies along the tube as in the stenosis and aneurysm cases, the DNN with the soft BC enforcement does not perform well as the no-slip BC of the vessel wall is poorly imposed especially near the bottleneck in Fig. 11. Consequently, the solution to the flow field becomes inaccurate. For example, Fig. 11 shows the results for learning a stenotic flow ($A = 5 \times 10^{-3}$) using the soft constraint with different λ values, where the result with hard boundary constraints and CFD benchmark are plotted as well for comparison.

In contrast to the result with hard constraints (Fig. 11b), both the flow patterns and magnitudes predicted with the soft boundary enforcement with different λ (Fig. 11c–f) are completely wrong compared to the CFD benchmark (Fig. 11a). Moreover, the nonlinear behavior of the centerline pressure profile cannot be accurately captured (results not shown). The poor performance reflects the major drawbacks of the soft constraints in the physics-driven training as mentioned above. First, unique PDE solutions are determined by the IC/BC, which cannot

Table 2
Converged loss with different penalty coefficient λ.

Loss	$\lambda = 1$	$\lambda = 10$	$\lambda = 100$	$\lambda = 1000$
Boundary condition	1×10^{-3}	2×10^{-4}	7×10^{-6}	1×10^{-7}
x-Momentum equation	1×10^{-4}	3×10^{-3}	8×10^{-3}	1×10^{-2}



(a) Hard BC enforcement (b) Soft BC enforcement

Fig. 12. Learning curves of different activation functions on a stenotic flow ($A = 5 \times 10^{-3}$, $\nu = 1 \times 10^{-3}$) with (a) hard BC constraint, (b) soft BC constraint with the penalty coefficient $\lambda = 1$.

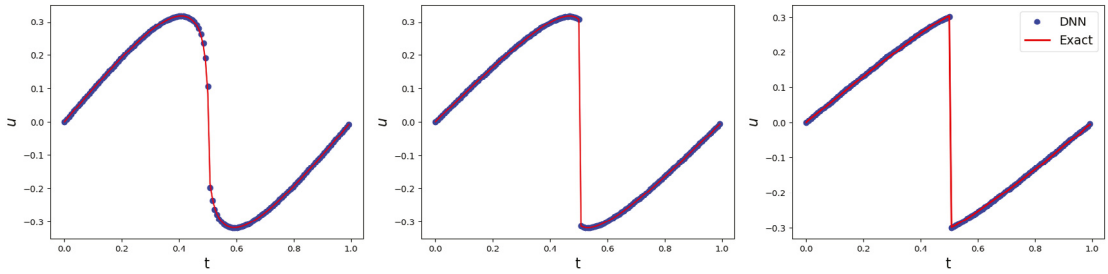
be guaranteed by simply penalizing the boundary loss. Moreover, it is difficult to assign the relative weight (λ) for different components in the loss function, and there could be "competing" effects between the equation loss and boundary loss, which makes the optimization difficult to converge. As shown in Table 2, by assigning a large weight for the boundary loss ($\lambda = 1000$), the boundary condition can be well prescribed but the PDE residual remains a large value and cannot be further reduced. On the other hand, when the weight for the boundary is small ($\lambda = 1$), the loss of x-momentum equation can be reduced to $O(10^{-4})$ but the BC fails to be imposed. It is important to note that none of these four λ leads to a physical stenotic flow pattern as shown in Fig. 11.

4.2. Role of activation function in physics-constrained learning

The performance of DNN training is affected by the activation function to a large degree. The widely used activation functions include ReLU, Sigmoid, Tanh, etc. [75]. However, these activation functions are not guaranteed to be optimized in terms of the convergence rate and accuracy. Recent studies [76,77] proposed to train an adaptive activation function as well as the neural network weights to achieve better convergence property. Notably, Ramachandran et al. [76] introduced an adaptive activation function called Swish, which is defined as $x \cdot \text{Sigmoid}(\beta x)$ and β is a trainable parameter. Jagtap and Karniadakis [77] presented a new adaptive function defined as $\text{Tanh}(nax)$, where a is an adaptive parameter to be learned and n is a scale factor that potentially speeds up the convergence.

In current work, the training process uses a Swish activation function with fixed $\beta = 1$. It is not clear how the adaptivity of activation function can affect the convergence rate and accuracy. Furthermore, as we discussed and highlighted the necessity of the hard BC enforcement in the physics-constrained data-free learning, it is also interesting to compare the relative importance of boundary condition enforcement and adaptive activation function on model performance. Therefore, we test the effects of different activation functions with/without hard boundary constraints in the stenosis case ($A = 5 \times 10^{-3}$, $\nu = 1 \times 10^{-3}$), and the resulting learning curves are shown in Fig. 12, where panels (a) and (b) show the convergence histories of different activation functions with and without a hard BC enforcement, respectively.

The legend "Swish-β" refers to the Swish function with adaptive β, while the "Swish" means the Swish function with a fixed $\beta = 1$. The results of adaptive Tanh activation function with different n [77] are plotted as well. It



(a) $t = 0.5$ (b) $t = 0.7$ (c) $t = 1.0$

Fig. 13. Performance of the proposed DNN for solving an inviscid Burgers equation, with a smooth initial solution turning into a shock at $t = 0.5$.

can be seen in Fig. 12a that the convergence rate of using adaptive Tanh functions is faster than that of the Swish activation functions within the first 400 epochs, but the training loss of all cases finally converge to the same order. Moreover, the convergence curves almost overlap with each other for the same type of activation function with different hyperparameters. When the BC is imposed in a soft manner (Fig. 12b), the solution can be quickly trapped in a bad local minimum as we discussed above and the final training loss is over one-order larger than that of the cases with hard BC constraints. Using different adaptive activation functions does not help to further decrease the loss and all the convergence curves are overlapped. Moreover, the convergence history of each trainable hyperparameter (i.e., β in Swish function and a in Tanh function) is monitored (see Fig. 14 in Appendix C), and we found that these hyperparameters are more likely to converge to the optimized values when the BCs are enforced in a hard manner. To sum up, the way of imposing BCs is found to be more critical than the adaptivity of activation function in our cases in terms of accuracy in the physics-constrained data-free learning.

4.3. Physics-constrained learning for discontinuous solution

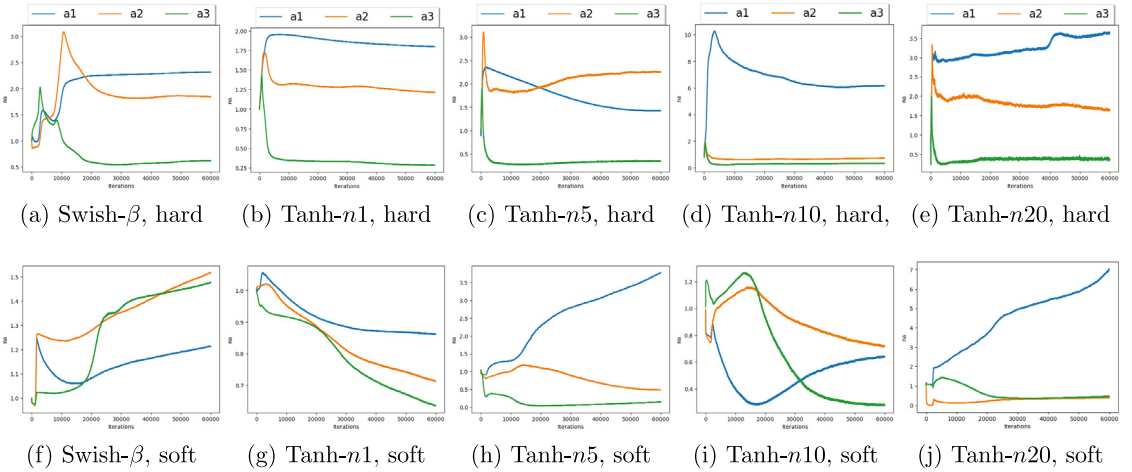
In this work, we mainly focused on incompressible fluid problems relevant to hemodynamics applications, where the flow solutions are usually smooth and continuous. To better explore the performance of the proposed IC/BC-encoded physics-constrained learning, canonical problems with discontinuous solutions were studied as well. Specifically, the proposed method was applied to capture the solutions of hyperbolic PDEs with discontinuities, e.g., inviscid Burger's equation, which is known to be notoriously difficult for most traditional numerical methods, especially the continuous Galerkin finite element methods. As an example, here we show the learning results for a one-dimensional inviscid Burger's equation,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \tag{18}$$

with a smooth initial condition as,

$$u_0 = \frac{1}{\pi}\sin(2\pi x), x \in [0, 1]. \tag{19}$$

The analytical solution is available, where the shock starts to form at $t = 0.5$ (i.e., $\frac{\partial u}{\partial x} \to \infty$ at $x = 0.5$) and becomes very sharp at $t = 1.0$, as shown in Fig. 13. These features can be well captured by our PDE-constrained DNN, where the IC and periodic BC were encoded into the DNN structure in a hard manner. It can be observed that even the sharpest discontinuity shown in Fig. 13c can be accurately predicted as the DNN approximation (blue dots) is almost overlapped with the exact solution (red line). The mean square test error for this case is 3.64×10^{-6}, and the training loss can be reduced to $O(10^{-6})$. The results demonstrate the good potential of physics-constrained learning for solving PDEs even with discontinuous behaviors. However, the loss function formulation here can be interpreted as a strong form of the PDE, and we expect that discontinuous Galerkin (DG) formulation in a weak form may help to further improve the learning performance for discontinuous problems, which is out of the scope of this work.



(a) Swish-β, hard (b) Tanh-$n1$, hard (c) Tanh-$n5$, hard (d) Tanh-$n10$, hard, (e) Tanh-$n20$, hard

(f) Swish-β, soft (g) Tanh-$n1$, soft (h) Tanh-$n5$, soft (i) Tanh-$n10$, soft (j) Tanh-$n20$, soft

Fig. 14. Convergence histories of trainable parameters (β for Swish and na for Tanh) of adaptive activation functions, where panels (a)–(e) are for cases with hard BC enforcement and panels (f)–(j) are for cases with soft BC enforcement. The legend $a1, a2, a3$ refer to three different layers.

4.4. Physics-constrained data-free learning vs. traditional data-driven learning

We have demonstrated that the solutions of parametric Navier–Stokes equations can be effectively learned by solely minimizing the PDE residuals without using any simulation data. To better evaluate the advantages and limitations of the physics-constrained learning, we also conducted a series of comparison studies between the physics-constrained data-free DNN and traditional data-driven DNN in terms of learning efficiency and prediction accuracy. Namely, all the vascular flow cases discussed above are learned again in a purely data-driven way, where the DNN architecture and hyperparameters remain the same and only the physics-based loss function (i.e., Eq. (10)) is replaced with the data-based one (i.e., Eq. (5)). A number of CFD simulations with different input parameters are conducted and the simulated velocity and pressure fields are collected as the training data. The detailed comparisons for stenotic and aneurysmal flows cases with varying viscosity and geometry are summarized by Tables 4 and 5 in Appendix B. In general, the prediction results from purely data-driven learning are slightly more accurate than those of physics-constrained learning, and the accuracy of both models improves as the parameter collocation points increase. However, data-driven learning requires additional offline CFD simulations and this computational overhead can quickly grow as more training points are sampled from the parameter space. In this paper, since the CFD cases considered here are not costly to simulate (e.g., each CFD simulation takes about 40 CPU seconds), the computational overhead due to the offline data generation process is not significant. Nonetheless, the advantage of the data-free feature in physics-constrained learning will become more notable when large-scale 3D flow problems are considered, where a single simulation run could be very expensive. It notes that when training data are ready for use, the cost of data-driven training process is approximately similar to that of the physics-constrained training (in the same order), though the latter is slightly slower due to the additional AD calculations for derivatives. To reduce the training cost in data-driven learning, one way is to reduce the spatial dimensionality by projecting the training data (i.e., velocity/pressure fields) onto the POD basis, and learning is performed on POD coefficients instead of spatial collocation points. We also conducted POD-based data-driven learning and found that the learning performance with the current shallow network structure is unsatisfactory (results are not shown here for conciseness).

Admittedly, the current form of physics-constrained DNN has its limitations, for example, the offline training process is still costly, the convergence cannot be guaranteed due to the non-convexity of DNN optimization, and scalability for high-dimensional complex problems is still challenging. The proposed PDE-constrained DNN is not expected to replace the classical CFD (numerical) solvers, which have been developed for decades. However, the development of PDE-constrained DNN for surrogate modeling shows strong promise. Particularly, the proposed method is mesh-free and thus does not require arduous mesh generation labor and intensive domain expertise in numerical modeling, which is suitable for, e.g., rapidly testing ideas in the design phase. We expect that the

Table 3
Summary of test errors on ν-varying stenosis case for different DNN architectures, i.e., networks with different number of hidden layers and neurons.

1 Hidden layer				
Neurons per layer	10	20	30	40
Test error e_u	7.44×10^{-3}	5.92×10^{-3}	5.39×10^{-3}	5.93×10^{-3}
Test error e_v	9.31×10^{-6}	6.83×10^{-6}	5.7×10^{-6}	7.08×10^{-6}
Test error e_p	3.26×10^{-3}	1.75×10^{-3}	1.46×10^{-3}	1.53×10^{-3}
2 Hidden layers				
Neurons per layer	10	20	30	40
Test error e_u	1.42×10^{-3}	8.92×10^{-4}	8.82×10^{-4}	2.75×10^{-4}
Test error e_v	1.05×10^{-6}	6.23×10^{-7}	6.72×10^{-7}	2.91×10^{-7}
Test error e_p	2.14×10^{-4}	1.48×10^{-4}	1.61×10^{-4}	6.10×10^{-5}
3 Hidden layers				
Neurons per layer	10	20	30	40
Test error e_u	9.91×10^{-5}	4.16×10^{-5}	2.91×10^{-5}	1.21×10^{-4}
Test error e_v	9.71×10^{-8}	3.72×10^{-8}	3.52×10^{-8}	3.67×10^{-8}
Test error e_p	1.96×10^{-5}	2.36×10^{-5}	8.40×10^{-6}	1.05×10^{-5}
4 Hidden layers				
Neurons per layer	10	20	30	40
Test error e_u	6.75×10^{-5}	3.35×10^{-5}	2.53×10^{-5}	6.05×10^{-5}
Test error e_v	5.33×10^{-8}	2.23×10^{-8}	1.86×10^{-8}	7.32×10^{-8}
Test error e_p	8.85×10^{-6}	3.22×10^{-6}	2.56×10^{-6}	6.65×10^{-6}
5 Hidden layers				
Neurons per layer	10	20	30	40
Test error e_u	2.82×10^{-5}	1.32×10^{-5}	2.50×10^{-5}	2.00×10^{-5}
Test error e_v	6.12×10^{-8}	3.04×10^{-8}	7.55×10^{-8}	2.86×10^{-8}
Test error e_p	4.00×10^{-5}	1.54×10^{-5}	1.19×10^{-5}	5.70×10^{-6}

effectiveness of surrogate model based on physics-constrained DNN will be significantly promoted along with the potential improvement of DNN training efficiency, e.g., a recent study has suggested that a novel photonic chip has a potential to be used to train deep neural networks 10-million of times more efficiently than current CPUs/GPUs do [78].

5. Conclusion

Surrogate modeling of fluid flows governed by the Navier–Stokes equations is significant for uncertainty quantification, optimization design, and inverse analysis in many engineering systems. As a universal function approximator, DNN is becoming a popular approach for surrogate modeling. However, training of a DNN often requires large number of labeled data, which are usually not available for efficiently developing surrogates since each data point requires an expensive CFD simulation. This paper presented a novel DNN surrogate for fluid simulations without using any labeled data (i.e., CFD simulation data). Specifically, a structured DNN architecture is devised to approximate the solutions of the parametric Navier–Stokes equations, where the initial/boundary conditions are satisfied automatically. Instead of using any simulation data, the DNN is trained by solely minimizing the violation of the mass and momentum conservation laws for fluid flows. Compared to the previous works of physics-constrained learning, this paper focuses on modeling of fluid systems governed by parametric Navier–Stokes equations. The proposed methods were tested on three flow cases relevant to cardiovascular applications, i.e., circular pipe flow, stenotic flow, and aneurysmal flow. The DNNs with equation-based loss were trained to learn the flow fields with parameter variations in, e.g., viscosity and domain geometry. Uncertainties in these parameters are propagated through the trained DNN surrogate and the results are validated against the CFD benchmarks. The comparisons

Table 4
Training and testing performance for stenosis case with varying viscosity/geometry.

Stenosis with varying viscosity (physics-constrained data-free)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	7598 s	12977 s	25988 s	28543 s	31996 s
Training minibatch loss	3.11×10^{-4}	8×10^{-5}	7.64×10^{-6}	4.6×10^{-5}	5.8×10^{-5}
Test error e_u	4.13×10^{-4}	8.18×10^{-5}	6.02×10^{-5}	3.29×10^{-5}	7.27×10^{-5}
Test error e_v	3.32×10^{-7}	7.14×10^{-8}	6.43×10^{-8}	3.86×10^{-8}	1.41×10^{-7}
Test error e_p	6.08×10^{-5}	2.33×10^{-5}	9.82×10^{-6}	6.87×10^{-6}	1.41×10^{-5}
Stenosis with varying viscosity (purely data-driven)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	6900 s	11967 s	19982 s	22497 s	30893 s
Simulation cost on CPU	400 s	800 s	1200 s	1600 s	2000 s
Training minibatch loss	5×10^{-7}	5×10^{-7}	1×10^{-7}	1×10^{-7}	1×10^{-7}
Test error e_u	1.49×10^{-5}	6.83×10^{-6}	2.43×10^{-6}	1.33×10^{-6}	2.64×10^{-6}
Test error e_v	7.84×10^{-8}	2.37×10^{-7}	8.84×10^{-8}	3.09×10^{-8}	3.01×10^{-8}
Test error e_p	6.42×10^{-7}	7.13×10^{-7}	1.56×10^{-6}	3.72×10^{-7}	1.65×10^{-6}
Stenosis with varying geometry (physics-constrained data-free)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	6458 s	11688 s	18239 s	24674 s	42814 s
Training minibatch loss	5.0×10^{-4}	1.0×10^{-3}	3.0×10^{-4}	7.0×10^{-4}	1.0×10^{-4}
Test error e_u	1.54×10^{-3}	9.61×10^{-4}	5.24×10^{-4}	3.46×10^{-4}	4.81×10^{-4}
Test error e_v	2.93×10^{-6}	1.76×10^{-6}	1.26×10^{-6}	7.43×10^{-7}	9.53×10^{-7}
Test error e_p	3.85×10^{-3}	2.23×10^{-3}	1.89×10^{-3}	1.57×10^{-3}	1.33×10^{-3}
Stenosis with varying geometry (purely data-driven)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	5498 s	11238 s	16285 s	21848 s	26396 s
Simulation cost on CPU	400 s	800 s	1200 s	1600 s	2000 s
Training minibatch loss	3×10^{-7}	1×10^{-7}	2×10^{-7}	1×10^{-7}	1×10^{-7}
Test error e_u	3.21×10^{-5}	2.18×10^{-5}	2.4×10^{-5}	2.74×10^{-5}	3.12×10^{-5}
Test error e_v	2.11×10^{-7}	1.37×10^{-7}	6.92×10^{-8}	9.33×10^{-8}	5.21×10^{-8}
Test error e_p	2.72×10^{-6}	2.88×10^{-7}	1.22×10^{-6}	3.41×10^{-7}	6.40×10^{-7}

indicate the excellent agreement between the physics-constrained DNN surrogate models and CFD simulations. Without using any labeled data in training, the DNN is able to accurately parameterize the velocity/pressure solutions with varying viscosity and geometries, which can be used to efficiently propagate uncertainties with enormous MC samples. Moreover, the performances of using hard and soft IC/BC enforcement approaches are compared and the issues of soft constraints in physics-constrained are discussed. We also investigated the influence of state-of-art adaptive activation functions and compared the present labeled-data-free learning approach with traditional data-driven learning approach in terms of accuracy and efficiency. In summary, the results have demonstrated the merit of the proposed method and suggest a great promise in developing DNN for surrogate fluid models without the need for CFD simulation data.

Acknowledgments

LS gratefully acknowledge partial funding of graduate fellowship from China Scholarship Council (CSC) in this effort. JXW would acknowledge support from the National Science Foundation (NSF contract CMMI-1934300) and Defense Advanced Research Projects Agency (DARPA) under the Physics of Artificial Intelligence (PAI) program (contract HR00111890034). The authors would like to thank Dr. Nicholas Zabaras and Mr. Yinhao Zhu for their helpful discussions during this work. The authors also thank the anonymous reviewers for their comments and suggestions, which helped improve the quality and clarity of the manuscript.

Table 5
Training and testing performance for aneurysm case with varying viscosity/geometry.

Aneurysm with varying viscosity (physics-constrained data-free)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	5731 s	14490 s	25350 s	24810 s	31595 s
Training minibatch loss	9.71×10^{-5}	5.50×10^{-5}	1.07×10^{-5}	2.33×10^{-5}	2.22×10^{-5}
Test error e_u	2.72×10^{-4}	9.2×10^{-5}	8.33×10^{-5}	6.4×10^{-5}	4.67×10^{-5}
Test error e_v	2.22×10^{-7}	1.33×10^{-7}	1.31×10^{-7}	5.96×10^{-8}	7.46×10^{-8}
Test error e_p	6.14×10^{-7}	1.18×10^{-5}	5.81×10^{-6}	1.12×10^{-5}	4.75×10^{-6}
Aneurysm with varying viscosity (purely data-driven)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	6851 s	11861 s	24580 s	24353 s	31116 s
Simulation cost on CPU	400 s	800 s	1200 s	1600 s	2000 s
Training minibatch loss	5×10^{-7}	5×10^{-7}	1×10^{-7}	1×10^{-7}	1×10^{-7}
Test error e_u	1.76×10^{-4}	5.11×10^{-6}	8.51×10^{-6}	5.30×10^{-7}	3.20×10^{-6}
Test error e_v	1.07×10^{-7}	6.31×10^{-8}	1.56×10^{-7}	6.95×10^{-8}	7.42×10^{-8}
Test error e_p	6.14×10^{-7}	2.38×10^{-7}	1.81×10^{-6}	1.06×10^{-6}	3.28×10^{-7}
Aneurysm with varying geometry (physics-constrained data-free)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	7755 s	12904 s	24645 s	26724 s	29515 s
Training minibatch loss	8.8×10^{-5}	2.9×10^{-5}	1.8×10^{-5}	7.8×10^{-6}	1.9×10^{-5}
Test error e_u	3.07×10^{-4}	1.38×10^{-4}	1.29×10^{-4}	1.31×10^{-4}	1.19×10^{-4}
Test error e_v	1.67×10^{-6}	1.15×10^{-6}	1.01×10^{-6}	9.95×10^{-7}	9.06×10^{-7}
Test error e_p	3.77×10^{-5}	2.09×10^{-5}	1.28×10^{-5}	8.99×10^{-6}	1.02×10^{-5}
Aneurysm with varying geometry (purely data-driven)					
Training viscosity points	10	20	30	40	50
Training cost on GPU	6322 s	11607 s	18582 s	23309 s	32124 s
Simulation cost on CPU	400 s	800 s	1200 s	1600 s	2000 s
Training minibatch loss	3×10^{-7}	2×10^{-7}	4×10^{-7}	1×10^{-7}	8×10^{-8}
Test error e_u	7.9×10^{-5}	8.63×10^{-5}	7.74×10^{-5}	8.69×10^{-5}	6.73×10^{-5}
Test error e_v	8.77×10^{-7}	5.34×10^{-7}	5.03×10^{-7}	6.24×10^{-7}	6.97×10^{-7}
Test error e_p	1.06×10^{-5}	8.94×10^{-6}	8.96×10^{-6}	9.24×10^{-6}	1.26×10^{-5}

Appendix A. Learning performance of different network structures

In order to search for a network with the "simplest structure" that has the desired predictive accuracy for better training efficiency and generalizability, a series of tests on the v-varying stenosis case were conducted using a group of DNNs with different number of hidden layers and neurons, from 1 hidden layer of 10 neurons per layer to 5 hidden layers of 40 neurons per layer. The test errors for u, v and p of all the different networks are listed in Table 3. In general, the test error decreases when the network becomes deeper (more hidden layers) and wider (more neurons per layer). We can see if the number of hidden layers is less than 3, adding one more layer can significantly reduce the error by nearly an order of magnitude. However, the accuracy stays in the same order when the networks have more than 3 layers. Furthermore, in the case of 3 hidden layers, there is no significant improvement by using more than 20 hidden neurons per layer.

Appendix B. Performance on learning idealized blood flows

See Tables 4 and 5.

Appendix C. Convergence of trainable parameters in activation functions

Fig. 14 shows the convergence histories of trainable parameters of different adaptive activation functions. The comparison is also made between cases where the BCs are imposed in hard and soft manners. It can be found that the parameters of activation function are more likely to converge with hard BC enforcement (panels (a)–(e)) compared to the cases with soft BC enforcement (panels (f)–(j)).

References

[1] J.D. Anderson, J. Wendt, Computational Fluid Dynamics, Vol. 206, Springer, 1995.
[2] P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems, SIAM Rev. 57 (4) (2015) 483–531.
[3] P. Benner, A. Cohen, M. Ohlberger, K. Willcox, Model Reduction and Approximation: Theory and Algorithms, Vol. 15, SIAM, 2017.
[4] T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza, Model order reduction in fluid dynamics: challenges and perspectives, in: Reduced Order Methods for Modeling and Computational Reduction, Springer, 2014, pp. 235–273.
[5] C. Huang, K. Duraisamy, C. Merkle, Challenges in reduced order modeling of reacting flows, in: 2018 Joint Propulsion Conference, 2018, p. 4675.
[6] S. Chaturantabut, D.C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764.
[7] B. Peherstorfer, Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling, arXiv preprint arXiv:1812.02094.
[8] B. Peherstorfer, Z. Drmač, S. Gugercin, Stabilizing discrete empirical interpolation via randomized and deterministic oversampling, arXiv preprint arXiv:1808.10473.
[9] W.-X. Ren, H.-B. Chen, Finite element model updating in structural dynamics by using the response surface method, Eng. Struct. 32 (8) (2010) 2455–2465.
[10] R.G. Regis, C.A. Shoemaker, A stochastic radial basis function method for the global optimization of expensive functions, INFORMS J. Comput. 19 (4) (2007) 497–509.
[11] M.C. Kennedy, A. O'Hagan, Predicting the output from a complex computer code when fast approximations are available, Biometrika 87 (1) (2000) 1–13.
[12] S. Atkinson, N. Zabaras, Structured bayesian gaussian process latent variable model: Applications to data-driven dimensionality reduction and high-dimensional inversion, J. Comput. Phys. 383 (2019) 166–195.
[13] D. Xiu, G.E. Karniadakis, The Wiener–Askey polynomial chaos for stochastic differential equations, SIAM J. Sci. Comput. 24 (2) (2002) 619–644.
[14] H.N. Najm, Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics, Annu. Rev. Fluid Mech. 41 (2009) 35–52.
[15] X. Yang, X. Wan, L. Lin, L. Huan, A general framework for enhancing sparsity of generalized polynomial chaos expansions, Int. J. Uncertain. Quantif. 9 (3) (2017) 221–243.
[16] O. Le Maître, O.M. Knio, Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics, Springer Science & Business Media, 2010.
[17] J.-X. Wang, C.J. Roy, H. Xiao, Propagation of input uncertainty in presence of model-form uncertainty: a multifidelity approach for computational fluid dynamics applications, ASCE-ASME J. Risk Uncertain. Eng. Syst. Part B: Mech. Eng. 4 (1) (2018) 011002.
[18] Y. Zhu, N. Zabaras, Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification, J. Comput. Phys. 366 (2018) 415–447.
[19] R.K. Tripathy, I. Bilionis, Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification, J. Comput. Phys. 375 (2018) 565–588.
[20] S. Mo, N. Zabaras, X. Shi, J. Wu, Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification, Water Resour. Res. 55 (5) (2019) 3856–3881.
[21] F. Scarselli, A.C. Tsoi, Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results, Neural Netw. 11 (1) (1998) 15–37.
[22] M. Hutzenthaler, A. Jentzen, T. Kruse, T.A. Nguyen, P. von Wurstemberger, Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations, arXiv preprint arXiv:1807.01212.
[23] P. Grohs, F. Hornung, A. Jentzen, P. Von Wurstemberger, A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of black-scholes partial differential equations, arXiv preprint arXiv:1809.02362.
[24] M. Hutzenthaler, A. Jentzen, T. Kruse, T.A. Nguyen, A proof that rectified deep neural networks overcome the curse of dimensionality in the numerical approximation of semilinear heat equations, arXiv preprint arXiv:1901.10854.
[25] S. Lee, N. Baker, Basic Research Needs for Scientific Machine Learning: Core Technologies for Artificial Intelligence, USDOE Office of Science (SC), United States, 2018.
[26] G. Carleo, M. Troyer, Solving the quantum many-body problem with artificial neural networks, Science 355 (6325) (2017) 602–606.
[27] J.-X. Wang, J.-L. Wu, H. Xiao, Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data, Phys. Rev. Fluids 2 (3) (2017) 034603.
[28] J. Ling, R. Jones, J. Templeton, Machine learning strategies for systems with invariance properties, J. Comput. Phys. 318 (2016) 22–35.

[29] P.E. Shanahan, D. Trewartha, W. Detmold, Machine learning action parameters in lattice quantum chromodynamics, Phys. Rev. D 97 (9) (2018) 094506.
[30] R. King, O. Hennigh, A. Mohan, M. Chertkov, From deep to physics-informed learning of turbulence: Diagnostics, arXiv preprint arXiv:1810.07785.
[31] S.L. Brunton, J.N. Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press, 2019.
[32] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[33] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[34] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al., Deep speech: Scaling up end-to-end speech recognition, arXiv preprint arXiv:1412.5567.
[35] H. Lee, I.S. Kang, Neural algorithm for solving differential equations, J. Comput. Phys. 91 (1) (1990) 110–131.
[36] I.E. Lagaris, A. Likas, D.I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw. 9 (5) (1998) 987–1000.
[37] I.E. Lagaris, A.C. Likas, D.G. Papageorgiou, Neural-network methods for boundary value problems with irregular boundaries, IEEE Trans. Neural Netw. 11 (5) (2000) 1041–1049.
[38] M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data, arXiv preprint arXiv:1808.04327.
[39] M. Raissi, Z. Wang, M.S. Triantafyllou, G.E. Karniadakis, Deep learning of vortex-induced vibrations, J. Fluid Mech. 861 (2019) 119–137.
[40] A.M. Tartakovsky, C.O. Marrero, D. Tartakovsky, D. Barajas-Solano, Learning parameters and constitutive relationships with physics informed deep neural networks, arXiv preprint arXiv:1808.03398.
[41] X. Meng, G.E. Karniadakis, A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse pde problems, arXiv preprint arXiv:1903.00104.
[42] D. Zhang, L. Lu, L. Guo, G.E. Karniadakis, Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems, arXiv preprint arXiv:1809.08327.
[43] Y. Yang, P. Perdikaris, Adversarial uncertainty quantification in physics-informed neural networks, arXiv preprint arXiv:1811.04026.
[44] R. Sharma, A.B. Farimani, J. Gomes, P. Eastman, V. Pande, Weakly-supervised deep learning of heat transport via physics informed loss, arXiv preprint arXiv:1807.11374.
[45] M.A. Nabian, H. Meidani, Physics-informed regularization of deep neural networks, arXiv preprint arXiv:1810.05547.
[46] K. Xu, E. Darve, The neural network approach to inverse problems in differential equations, arXiv preprint arXiv:1901.07758.
[47] J.R. Holland, J.D. Baeder, K. Duraisamy, Towards integrated field inversion and machine learning with embedded neural networks for rans modeling, in: AIAA Scitech 2019 Forum, 2019, p. 1884.
[48] R. Stewart, S. Ermon, Label-free supervision of neural networks with physics and domain knowledge, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 2576–2582.
[49] J. Sirignano, K. Spiliopoulos, Dgm: A deep learning algorithm for solving partial differential equations, J. Comput. Phys. 375 (2018) 1339–1364.
[50] J. Berg, K. Nyström, A unified deep artificial neural network approach to partial differential equations in complex geometries, Neurocomputing 317 (2018) 28–41.
[51] E. Weinan, J. Han, A. Jentzen, Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations, Commun. Math. Stat. 5 (4) (2017) 349–380.
[52] C. Beck, E. Weinan, A. Jentzen, Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations, J. Nonlinear Sci. (2017) 1–57.
[53] C. Beck, S. Becker, P. Grohs, N. Jaafari, A. Jentzen, Solving stochastic differential equations and Kolmogorov equations by means of deep learning, arXiv preprint arXiv:1806.00421.
[54] E. Weinan, B. Yu, The deep ritz method: A deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (1) (2018) 1–12.
[55] J. Han, A. Jentzen, E. Weinan, Solving high-dimensional partial differential equations using deep learning, Proc. Natl. Acad. Sci. 115 (34) (2018) 8505–8510.
[56] M.A. Nabian, H. Meidani, A deep neural network surrogate for high-dimensional random partial differential equations, arXiv preprint arXiv:1806.02957.
[57] S. Karumuri, R. Tripathy, I. Bilionis, J. Panchal, Simulator-free solution of high-dimensional stochastic elliptic partial differential equations using deep neural networks, arXiv preprint arXiv:1902.05200.
[58] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, P. Perdikaris, Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data, arXiv preprint arXiv:1901.06314.
[59] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: a survey, J. Mach. Learn. Res. 18 (2018) 1–43.
[60] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, in: NIPS Autodiff Workshop, 2017.
[61] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., Tensorflow: A system for large-scale machine learning, in: 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 16, 2016, pp. 265–283.

[62] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I.J. Goodfellow, A. Bergeron, N. Bouchard, Y. Bengio, Theano: new features and speed improvements. deep learning and unsupervised feature learning, in: Neural Information Processing Systems Workshop (NIPS), 2012, pp. 1–10.
[63] R. Kleinberg, Y. Li, Y. Yuan, An alternative view: When does SGD escape local minima? arXiv preprint arXiv:1802.06175.
[64] P. Márquez-Neila, M. Salzmann, P. Fua, Imposing hard constraints on deep networks: Promises and limitations, arXiv preprint arXiv:1706.02025.
[65] P. Ramachandran, B. Zoph, Q.V. Le, Swish: a self-gated activation function, arXiv preprint arXiv:1710.05941.
[66] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.
[67] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the International Conference on Artificial Intelligence and Statistics, AISTATS'10, Society for Artificial Intelligence and Statistics, 2010.
[68] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.
[69] A. Blumer, A. Ehrenfeucht, D. Haussler, M.K. Warmuth, Occam's razor, Inform. Process. Lett. 24 (6) (1987) 377–380.
[70] H. Jasak, A. Jemcov, U. Kingdom, Openfoam: A c++ library for complex physics simulations, in: International Workshop on Coupled Methods in Numerical Dynamics, IUC, 2007, pp. 1–20.
[71] S. Berger, L.-D. Jou, Flows in stenotic vessels, Annu. Rev. Fluid Mech. 32 (1) (2000) 347–382.
[72] J.L. Brisman, J.K. Song, D.W. Newell, Cerebral aneurysms, New Engl. J. Med. 355 (9) (2006) 928–939.
[73] J.R. Cebral, F. Mut, J. Weir, C.M. Putman, Association of hemodynamic characteristics and cerebral aneurysm rupture, Am. J. Neuroradiol. 32 (2) (2011) 264–270.
[74] N. Chalouhi, B.L. Hoh, D. Hasan, Review of cerebral aneurysm formation, growth, and rupture, Stroke 44 (12) (2013) 3613–3622.
[75] A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proceedings of the 30th Annual International Conference on Machine Learning, Vol. 30, 2013, p. 3.
[76] P. Ramachandran, B. Zoph, Q.V. Le, Searching for activation functions, arXiv preprint arXiv:1710.05941.
[77] A.D. Jagtap, G.E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, arXiv preprint arXiv:1906.01170.
[78] R. Hamerly, L. Bernstein, A. Sludds, M. Soljačić, D. Englund, Large-scale optical neural networks based on photoelectric multiplication, Phys. Rev. X 9 (2) (2019) 021032.