Simulation of Steady and Transient 3D Flows via Physics-Informed Deep Learning

Philipp Moser[a], Wolfgang Fenz[b], Stefan Thumfart[c], Isabell Ganitzer[d] and Michael Giretzlehner[e]

Research Department Medical Informatics, RISC Software GmbH, Softwarepark 32a, 4232 Hagenberg, Austria

Keywords: Physics-Informed Neural Networks, Fluids, Navier-Stokes, Deep Learning.

Abstract: Physics-Informed deep learning methods are attracting increased attention for modeling physical systems due to their mesh-free approach, their straightforward handling of forward and inverse problems, and the possibility to seamlessly include measurement data. Today, most learning-based flow modeling reports rely on the representational power of fully-connected neural networks, although many different architectures have been introduced into deep learning, each with specific benefits for certain applications. In this paper, we successfully demonstrate the application of physics-informed neural networks for modeling steady and transient flows through 3D geometries. Our work serves as a practical guideline for machine learning practitioners by comparing several popular network architectures in terms of accuracy and computational costs. The steady flow results were in good agreement with finite element-based simulations, while the transient flows proved more challenging for the continuous-time PINN approaches. Overall, our findings suggest that standard fully-connected neural networks offer an efficient balance between training time and accuracy. Although not readily supported by statistical/practical significance, we could identify a few more complex architectures, namely Fourier networks and Deep Galerkin Methods, as attractive options for accurate flow modeling.

1 INTRODUCTION

In recent years, the remarkable advances of deep learning (DL) in various domains (e.g., computer vision and natural language processing) have inspired vibrant research on applying DL to obtain solutions for physical models, which, in general, has become an integral part of modern science. While physics-based DL comprises a variety of conceptually different approaches (Karniadakis et al., 2021), we focus on physics-informed DL that encodes physical information (typically expressed as partial differential equations (PDEs) with suitable initial/boundary conditions) into the loss function of a neural network, hence, constraining the network's trainable parameters. An overview of Raissi's general framework of physics-informed neural networks (PINNs) (Raissi et al., 2019) is given in Section 1.1. Compared to conventional computational fluid dynamics (CFD) methods, PINNs operate without the need to mesh the simulation geometry and can easily handle forward simulations (predicting state or temporal evolution) or inverse problems (e.g., obtaining a parametrization for a physical system from observations).

Fluid mechanics is among the most active research topics due to (a) the complex underlying Navier-Stokes equations, (b) the lack of general analytic solutions and (c) the widespread applications (e.g., biomedical engineering, geophysics, aerospace). Selected PINN applications include the prediction of near-wall blood flow from sparse data (Arzani et al., 2021), direct turbulence modeling (Jin et al., 2021), solving the Reynolds-averaged Navier–Stokes equations (Eivazi et al., 2022), tackling ill-posed inverse fluid-mechanics problems (Raissi et al., 2020), or mixed-variable PINN schemes for 2D flows (Rao et al., 2020).

Deep learning is a fast-evolving field of research where many network architectures and add-ons (e.g., Fourier encoding layers, adaptive activation functions, skip connections) have been proposed (Choudhary et al., 2022; Markidis, 2021). Despite the broad

[a] https://orcid.org/0000-0002-9717-6197
[b] https://orcid.org/0000-0002-6143-3024
[c] https://orcid.org/0000-0001-9838-3641
[d] https://orcid.org/0000-0003-3600-0857
[e] https://orcid.org/0000-0003-1620-1002

Moser, P., Fenz, W., Thumfart, S., Ganitzer, I. and Giretzlehner, M.
Simulation of Steady and Transient 3D Flows via Physics-Informed Deep Learning.
DOI: 10.5220/0012078600003546
In *Proceedings of the 13th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2023)*, pages 243-250
ISBN: 978-989-758-668-2; ISSN: 2184-2841

architectural variety, to date the majority of PINN-based flow simulations have employed simple fully-connected neural networks (FCNNs) (Jin et al., 2021; Eivazi et al., 2022; Oldenburg et al., 2022; Arzani et al., 2021; Amalinadhi et al., 2022). Only few publications report the use of more complex network architectures, such as convolutional neural networks (CNNs) (Ma et al., 2022; Eichinger et al., 2021; Guo et al., 2016; Gao et al., 2021) or Deep Galerkin Methods (DGMs) (Matsumoto, 2021; Li et al., 2022). While this is related to the early development phase of PINNs and the initial focus on feasibility and proof-of-concept (Cuomo et al., 2022; Karniadakis et al., 2021), the increasing interest in PINNs motivates the integration and evaluation of recent DL developments that have shown promising results in other applications. Comparing results across publications is often difficult because of significant variations in simulation domains (2D or 3D), fluid characteristics and evaluation metrics.

In this paper, we fill this gap and provide the first comprehensive comparison of several popular network architectures for modeling steady and transient flows in 3D geometries. Besides standard FCNNs, we evaluate the computational costs and accuracy of (in total) seven different architectures using finite element-based CFD models as references. Our comparative analysis intends to highlight the potential but also the challenges of current PINN architectures for modeling fluid dynamics.

1.1 Principles of Physics-Informed Neural Networks

In general, a physical system is governed by a set of PDEs that describe the variation of some physical quantity $\mathbf{u}$ (e.g., velocity in the case of fluid dynamics) at points $\mathbf{x} \in \Omega$ (i.e., points within a geometry Ω). Most often, these PDEs are accompanied by (a) boundary conditions $\mathcal{B}$ which specify the values of $\mathbf{u}$ at points $\mathbf{x}$ on the boundary $\partial\Omega$ and (b) initial conditions I which define the values of $\mathbf{u}$ at $t = 0$.

Raissi et al. introduced PINNs (Raissi et al., 2019), which aim at approximating the complex (and nonlinear) solution function $\mathbf{u}$ via a deep neural network. Physics information (given by the underlying PDE), boundary constraints, initial conditions as well as sparse measurement data (if available) are integrated into the loss function of a neural network:

$$\mathcal{L}_{\text{tot}} = \mathcal{L}_{\text{PDE}} + \mathcal{L}_{\text{BC}} + \mathcal{L}_{\text{IC}} + \left[\mathcal{L}_{\text{data}}\right]. \quad (1)$$

$\mathcal{L}_{\text{PDE}}$ evaluates (via the two-norm $\|\cdot\|$) the PDE's residual on randomly sampled points within the geometry Ω

$$\mathcal{L}_{\text{PDE}} = \left\|\mathcal{N}(\mathbf{x},\mathbf{u},t)\right\|_{\Omega}, \quad (2)$$

with $\mathcal{N}$ being the PDE's nonlinear differential operator. $\mathcal{L}_{\text{BC}}$ is sampled on points on the boundary $\partial\Omega$ where boundary conditions $\mathcal{B}$ are known:

$$\mathcal{L}_{\text{BC}} = \|\mathcal{B}(\mathbf{x},\mathbf{u},t)\|_{\partial\Omega}, \quad (3)$$

while $\mathcal{L}_{\text{IC}}$ is calculated from initial conditions I with sampling points at $t = 0$:

$$\mathcal{L}_{\text{IC}} = \|I(\mathbf{x},\mathbf{u},t=0)\|. \quad (4)$$

The data loss term $\mathcal{L}_{\text{data}}$ is evaluated on discrete points Γ where additional measurement data $\mathcal{D}$ is available:

$$\mathcal{L}_{\text{data}} = \|\mathbf{u} - \mathcal{D}\|_{\Gamma}. \quad (5)$$

The PINN training process is depicted in Figure 1. A neural network generates estimates for $\mathbf{u}$ on a set of randomly sampled spatio-temporal points (x,y,z,t). Using auto-differentiation, all necessary derivatives are generated and allow for the calculation of the PDE-related (physics-informed) loss term $\mathcal{L}_{\text{PDE}}$. The total loss function $\mathcal{L}_{\text{tot}}$ is constructed by adding all four loss terms in Eq. 1. While training the network, the optimal set of weights and biases of the neural network is being sought, for which the total loss function is minimized and the resulting estimates for $\mathbf{u}$ approach the true solution.

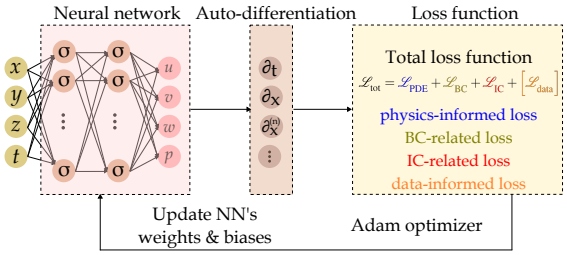

Figure 1: Schematic of a PINN. The physical quantities of interest are (u,v,w,p) and the spatio-temporal coordinates are (x,y,z,t). σ is the activation function.

2 METHODS

2.1 Simulation Model, Geometries and Ground Truth

The flow of incompressible Newtonian fluids is described by the Navier-Stokes and continuity equations:

$$\rho\left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right] - \mu\nabla^2\mathbf{u} + \nabla p = 0, \quad (6)$$

$$\nabla \cdot \mathbf{u} = \mathbf{0}. \quad (7)$$

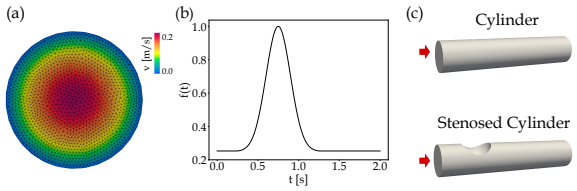

Figure 2: (a) The parabolic inlet velocity profile, (b) the time-varying inflow function $f(t)$, (c) the two studied geometries. The inflow is indicated by a red arrow.

$\mathbf{u} = \mathbf{u}(\mathbf{x},t) = [u(\mathbf{x},t), v(\mathbf{x},t), w(\mathbf{x},t)]^\intercal$ denotes the velocity vector and $p = p(\mathbf{x},t)$ is the pressure field. The inflow was set perpendicular to the inlet plane with a parabolic velocity profile:

$$u_{\text{inlet}}(r) = u_{\max}\left(1 - \frac{r^2}{R^2}\right), \tag{8}$$

with r being the radial distance from the inlet center and R being the radius of a cylindrical inlet. The central peak velocity $u_{\max}$ was set to 0.2 m/s, see Figure 2a. A pulsatile flow was generated by a temporally varying function $f(t)$ using a Gaussian-shaped pulse with a duration of 1.0 s, see Figure 2b. The entire time-varying inflow profile $u_{\text{inlet}}(r,t)$ is obtained by multiplying $u_{\text{inlet}}(r)$ (from Eq. 8) and $f(t)$ (depicted in Figure 2b). The total simulation time was set to 2.0 s.

We modeled two cylindrical geometries (Figure 2c) that frequently appear in various engineering domains (e.g., modeling healthy and diseased arteries in computational hemodynamics). Specifically, the first system was the laminar flow of a fluid (density $\rho = 1233$ kg/m^3, dynamic viscosity $\mu = 0.20393$ Pa·s, Reynolds number $Re = 1209$) through a regular cylinder (radius: 0.5 m, length: 5 m). The second geometry was a cylinder with a stenosis created by a sphere of radius of 0.5 m centered on its surface 1.25 m from the inlet. The fluid parameters ($\rho = 616.5$ kg/m^3, $\mu = 0.40786$ Pa·s, $Re = 302$) differed from the first system to showcase PINN's ability to handle various regimes of laminar fluid properties. We employed the following boundary conditions: no-slip conditions on the geometry walls ($u,v,w = 0$) and the pressure set to zero at the outlet ($p = 0$).

Ground truth (reference) distributions of velocity and pressure were generated via finite-element-based simulations using the framework MODSIM, an in-house CFD software suite (Fenz et al., 2010; Fenz et al., 2016; Gmeiner et al., 2018). Both CFD geometries had around 600k mesh points. All simulations (i.e., PINN and CFD modeling) were performed on a workstation with an Intel i9-11900 (CPU), NVIDIA GeForce RTX 3080 Ti (GPU) and 64 GB RAM.

2.2 Network Architectures

We implemented all PINN architectures within NVIDIA's Modulus framework v22.03 (Hennigh et al., 2021; NVIDIA, 2022). For comparison purposes, the same network size (6 layers with 256 neurons each), swish activation functions (β=1), exponentially decaying learning rates (initial learning rate: 0.0005, decay rate: 0.97, decay steps: 20005) and Adam optimizers were used across all network architectures. In the following, the seven studied architectures are briefly summarized. More information can be obtained in the referenced original publications.

2.2.1 Fully Connected Neural Networks

In the standard version, all neurons of one layer are connected to each neuron in the next layer. The output of an n-layer FCNN $\mathbf{u}_{\text{net}}(\mathbf{x};\theta)$ reads

$$\mathbf{u}_{\text{net}}(\mathbf{x};\theta) = \mathbf{W}_n\{\phi_{n-1} \circ \phi_{n-2} \circ \cdots \circ \phi_1\}(\mathbf{x}) + \mathbf{b}_n, \tag{9}$$

$$\phi_i(\mathbf{x}_i) = \sigma(\mathbf{W}_i\mathbf{x}_i + \mathbf{b}_i), \tag{10}$$

where ϕ_i is the i^{th} layer, $\mathbf{W}_i$ and $\mathbf{b}_i$ are the weights and biases of the i^{th} layer. θ refers to the trainable parameters $\{\mathbf{W}_i, \mathbf{b}_i\}_{i=1}^n$. Throughout the architecture descriptions, σ is the activation function and the vector notation $\mathbf{x}$ includes all spatial and temporal coordinates $\mathbf{x} = (x,y,z;t)$.

The second architecture (termed FCNNaa) is a variation of simple FCNNs and features adaptive activation functions as suggested by Jagtap et al. (Jagtap et al., 2020). A global, trainable, multiplicative parameter α is added to the activation functions and the i^{th} layer of the FCNN, hence, becomes

$$\phi_i(\mathbf{x}_i) = \sigma(\alpha \cdot (\mathbf{W}_i\mathbf{x}_i + \mathbf{b}_i)). \tag{11}$$

The third architecture (termed FCNNskip) is a further variation of FCNNs which includes skip connections every two hidden layers. Originally introduced into deep learning for addressing the vanishing gradient problem (He et al., 2016), skip connections feed the output of one layer as the input to the next layers (instead of only the subsequent layer).

2.2.2 Fourier Networks

The fourth architecture (termed FN) is a Fourier network which tries to reduce spectral biases (Rahaman et al., 2019; Tancik et al., 2020; Mildenhall et al., 2020). A FCNN architecture is modified by adding an initial Fourier encoding layer ϕ_E that includes a trainable frequency matrix $\mathbf{m}_v$:

$$\mathbf{u}_{\text{net}}(\mathbf{x};\theta) = \mathbf{W}_n\{\phi_{n-1} \circ \phi_{n-2} \circ \cdots \circ \phi_1 \circ \phi_E\}(\mathbf{x}) + \mathbf{b}_n, \tag{12}$$

$$\phi_E = [\sin(2\pi \mathbf{m}_v \cdot \mathbf{x});\ \cos(2\pi \mathbf{m}_v \cdot \mathbf{x})]^T. \quad (13)$$

The fifth architecture (termed modFN) is a modified Fourier network. It uses two additional layers that transform the Fourier features to a learned feature space and transfer the information flow to the other hidden layers via Hadamard multiplications (denoted by $\odot$) (Wang et al., 2021). Here, the i^{th} layer reads:

$$\phi_i(\mathbf{x}_i) = (1 - \sigma(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i)) \odot \sigma(\mathbf{W}_{T_1} \phi_E + \mathbf{b}_{T_1}) + \sigma(\mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i) \odot \sigma(\mathbf{W}_{T_2} \phi_E + \mathbf{b}_{T_2}) \quad \text{for } i > 1. \quad (14)$$

The additional trainable parameters $\{\mathbf{W}_{T_1}, \mathbf{b}_{T_1}, \mathbf{W}_{T_2}, \mathbf{b}_{T_2}\}$ belong to the two added transformation layers.

2.2.3 Multiplicative Filter Network (MFN)

The sixth architecture is a Fourier MFN which generates representational power through repeated multiplications of sinusoidal wavelet functions $\mathbf{f}(\mathbf{x},\ \xi_i)$ applied to the input (Fathony et al., 2021):

$$\begin{aligned} &\mathbf{f}(\mathbf{x}, \xi_i) = \sin(\omega_i \mathbf{x} + \phi_i) \quad \text{with} \quad \xi_i = (\omega_i,\ \phi_i), \\ &\phi_1 = \mathbf{f}(\mathbf{x}, \xi_1), \\ &\phi_{i+1} = \sigma(\mathbf{W}_i \phi_i + \mathbf{b}_i) \odot f(\mathbf{x},\ \xi_{i+1}), \forall i \in \{1, \ldots, n-1\}, \\ &\mathbf{u}_{\text{net}}(\mathbf{x}; \theta) = \mathbf{W}_n \phi_n + \mathbf{b}_n \end{aligned} \quad (15)$$

2.2.4 Deep Galerkin Method (DGM)

The DGM approach as proposed by Sirignano and Spiliopoulos is the seventh architecture. Inspired by the well-established Galerkin method, the DGM replaces the linear combination of basis functions to approximate the PDE's solution by a deep neural network. The architecture is rather complex and has some similarities to Long Short-Term Memory (LSTM) networks. For the explicit network structure, we refer to the original publication (Sirignano and Spiliopoulos, 2018).

2.3 Evaluation Metrics

To compare the accuracy of the PINN results against CFD simulations, we calculated mean absolute errors (MAE) and standard deviations of absolute errors (STD of AE) of the physical quantities of interest (i.e., absolute velocity and pressure): The PINNs were evaluated at the CFD mesh points and the error metrics were calculated based on the absolute differences of the PINN and CFD solutions.

Regarding statistical analysis, we calculated Cohen's effect size d (Lakens, 2013) to assess the practical significance of the differences in MAEs between the best-performing architecture and all others.

3 RESULTS

3.1 Steady Flow

For the steady flow simulations, the MAEs and STDs of AE for all used PINN architectures are summarized in Table 1. In the regular cylinder, the lowest MAE in absolute velocity was obtained for the FN architecture. However, all MAEs were close to each other with overlapping 95% confidence intervals and effect sizes d below 0.2 suggesting practically insignificant differences in accuracy among the architectures.

In the stenosed cylinder, the MAEs in absolute velocity had a broader range and the values were generally higher (which was expected due to the higher geometrical complexity), with modFN performing far worse than all other architectures. The DGM performed best, however, the effect sizes between DGM and the other architectures were small indicating rather low practical importance.

In both cylinder geometries, FCNNskip performed best in terms of pressure MAE, while modFN performed worst with effect sizes $d > 0.9$ suggesting considerably worse results.

Figure 3 depicts the predicted (i.e., learned) distributions of the absolute velocity and pressure as well as their absolute deviations from CFD references. At the inlet of the regular cylinder, a mean pressure of 3.38 Pa was obtained with FCNNskip, while the analytic values would be 3.26 Pa (calculated via the Hagen–Poiseuille equation which describes the flow pressure drop in a cylindrical pipe).

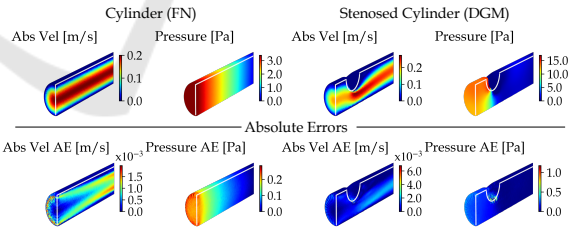

Figure 3: Steady flow simulations: The predicted absolute velocity and pressure distributions (first row), as well as their absolute errors (AEs) (second row). The shown architectures were chosen based on the best absolute velocity estimates.

The training times for one epoch are also compiled in Table 1 and show significant variations: The DGM, for example, required about a factor 3.9 more time to train than a standard FCNN. In absolute numbers, the FCNN training (with 350 epochs) of the steady flow system took around 9.6 hours on our hardware.

Table 1: Steady flow simulations: Errors metrics for the various PINN architectures. Note: The velocity errors are given in mm/s. The percentage values in the last column are relative to the FCNN architecture. Bold numbers indicate the best results.

	MAE and STD of AEs				Training time
	Cylinder		Stenosed Cylinder		per epoch [s]
	Abs Vel [mm/s]	Pressure [Pa]	Abs Vel [mm/s]	Pressure [Pa]	
FCNN	0.838±0.405	0.075±0.060	1.024±0.899	0.066±0.075	**99 (100%)**
FCNNaa	0.837±0.406	0.073±0.062	0.990±0.823	0.054±0.075	113 (114%)
FCNNskip	0.819±0.412	**0.068±0.058**	1.201±1.107	**0.052±0.054**	101 (102%)
FN	**0.801±0.387**	0.076±0.061	1.188±1.051	0.055±0.079	100 (101%)
modFN	0.813±0.486	0.342±0.194	25.514±24.163	1.696±1.226	178 (178%)
MFN	0.838±0.383	0.105±0.051	0.918±0.840	0.056±0.072	103 (104%)
DGM	0.843±0.387	0.076±0.046	**0.819±0.693**	0.053±0.075	384 (388%)

3.2 Transient Flow

For the transient flow simulations, we calculated MAEs every 0.05 seconds and visualize their temporal evolution in Figure 4. Overall, the lowest MAEs were recorded for the FCNN, FN and DGM architectures, while MFN and modFN performed significantly worse. MAEs of absolute velocity were generally lower than for the pressure estimates. For all architectures, the highest MAEs occured between $t = 0.5$ s and $t = 1.0$ s (which includes the peak of the Gaussian inflow profile and the region of fastest change in inflow). Still, differences regarding the shape of the temporal MAE curves were observed among architectures.

The predicted absolute velocity and pressure distributions at three specific points in time are visualized in Figure 5. The time point $t = 1.00$ s corresponds to the half maximum on the right side of the inflow profile (see Figure 4). Noticeable deviations were observed throughout the pressure estimates with relative AEs (i.e., AEs normalized by the absolute CFD reference) being highest in the flat part of the inflow profile.

4 DISCUSSION

An integral part of an effective and efficient learning framework involves the design and selection of the network architecture. The training process (i.e., the convergence and accuracy of the learned solution) can be positively influenced when a domain-intrinsic property is reflected within the architecture. CNNs, for example, unfold their strength in image-related tasks as convolution operators reflect the translational invariance of images. Unfortunately, no such intrinsic symmetry was identified for PINN-based fluid modeling, which results in PINNs mostly relying on the representational power of FCNNs. Also, our PINN approach of randomly sampling points within (irregular) geometries prohibited the use of convolutional architectures as these rely on structured grids (as is the case for images). Future research in geometric DL may provide ways of leveraging convolutions in non-Euclidean spaces.

In this paper, we have not only successfully demonstrated the modeling of steady and transient 3D flows, but also investigated the performance of various network architectures. To our knowledge a comparable evaluation has only been published for steady flows so far (Moser et al., 2023), while transient flows have not been investigated before. Overall, the learned solutions were in good agreement with CFD references, especially for the steady flows, while the increased complexity of time-dependent systems was obviously more challenging. The higher errors of the transient pressure estimates can be attributed to the networks struggling with multi-scale issues: The pressure distributions ranged over 4000 Pa, resulting in single-digit pressures being challenging to resolve.

Although benefits (improved convergence and solution accuracy) were reported for adaptive activation functions (Jagtap et al., 2020), we did not observe distinct improvements compared to standard FCNNs that would favor their usage, especially when taking into account their higher computational costs. Our findings regarding FCNNskip were mixed since the best pressure estimates were obtained with FCNNskip in the steady flow simulations, while the transient flow results lagged behind standard FCNNs at equally fast training times. FNs and DGMs achieved overall good results and could match/outperform FCNNs. We obtained good results with MFNs for the steady flow simulations, but their multiplicative way of generating representational power was insufficient for time-dependent problems.

Significant differences in computational costs were observed among the studied network architectures which are mainly dominated by the complexity of the architecture (e.g., the number of trainable

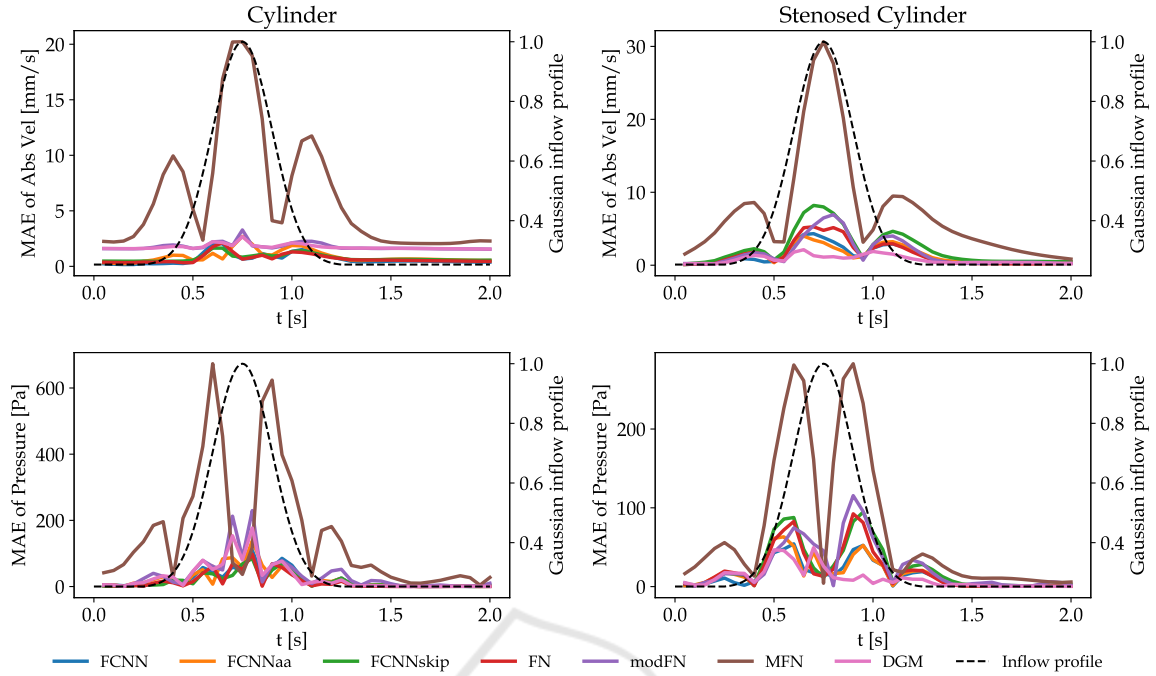

Figure 4: Transient flow simulations: The MAEs of the predicted absolute velocity (given in mm/s) and pressure distributions. Note: The dashed line is the Gaussian inflow profile which serves only as temporal guidance and is not the "target curve".

parameters). The DGM, for example, uses roughly eight times more parameters in each hidden layer than a usual dense layer (Al-Aradi et al., 2022), which significantly increases memory consumption and training time. While the presented relative computational times are generally applicable, the absolute training times are highly dependent on the available resources. Multi-GPU systems and efficient handling of tensor operations (e.g., NVIDIA's TensorFloat-32) should further accelerate the training.

Overall, our findings suggest that FCNNs (with or without skip connections) are an efficient (accuracy per unit training time) starting point due to their fast training and consistent mid-to-high-tier results. Keeping in mind that differences in accuracy were not always supported by (practical) significance, we could still identify a trend that some more complex architectures, such as FNs or DGMs, might be attractive options for complex flow modeling.

In this paper, we employed the continuous time approach for transient systems, i.e., the time is treated similar to the other continuous, spatial variables. While this is a straightforward extension of steady-state modeling, one potential limitation is the high amount of sampling points needed to enforce physics constraints in the entire spatio-temporal domain, especially in higher dimensions, longer simulation time spans and complex geometries. Other publications have discussed improvements of discrete time models (Runge-Kutta time-stepping schemes) (Raissi et al., 2019) or alternative network architectures (e.g., Recurrent Neural Networks) for modeling (longer) temporal dynamics (Wu et al., 2022).

In this paper, we restrict our simulations to laminar flows at low Reynolds numbers. However, the PINN approach could readily be applied to turbulent flows with higher Reynolds numbers since turbulences are inherently described by the Navier-Stokes equation. To adequately resolve (chaotic) changes in pressure and flow velocities, (a) significantly more sampling points would be necessary during training (increased memory consumption and computation times) or (b) alternative approaches such as super-resolution (Jiang et al., 2020) could be applied.

Future research may investigate the influence of network hyperparameters, such as network sizes and activation functions. Also, the generalizability of a trained network to changes in fluid parameters or geometry may involve fine-tuning/transfer-learning techniques, which remains an effort for future studies.

5 CONCLUSIONS

In this paper, we addressed the practically relevant question which network architectures in the rapidly evolving field of deep learning provide good learning

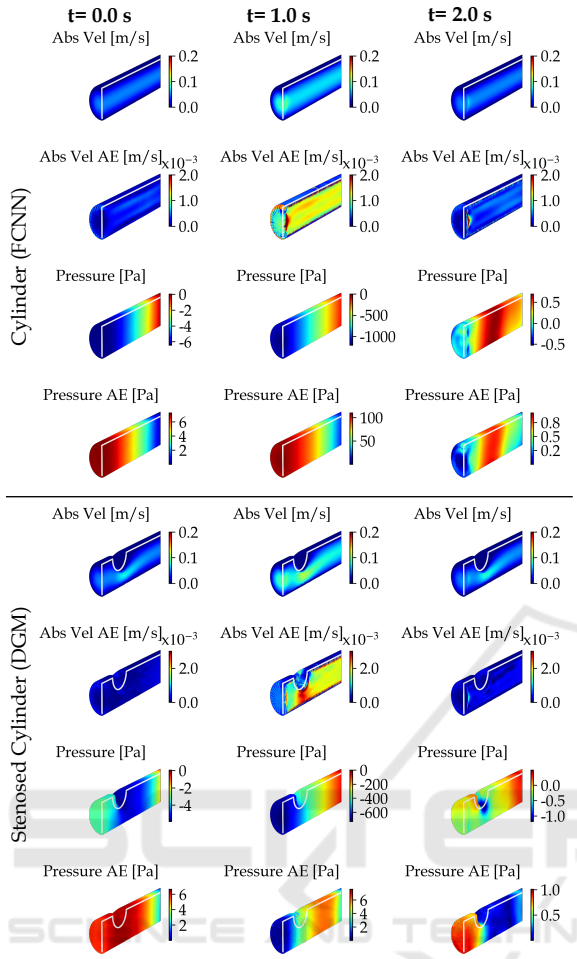

Figure 5: Transient flow simulations: The predicted absolute velocity and pressure distributions, as well as their absolute errors are shown for three distinct points in time. The network architectures with the best velocity estimates are shown. Note: The absolute velocity and their error distributions share a common color range.

abilities to model the dynamics of fluids in 3D geometries. While steady flows could be modeled with high accuracy, the transient flows were more challenging to resolve with our continuous-time PINNs. We conclude that while the investigated PINN implementations were in general capable of modeling the complex behavior of fluids, future architectures that intrinsically reflect some underlying physical property/symmetry and adequately handle multi-scale behaviors have the potential to outperform current approaches.

ACKNOWLEDGEMENTS

This project is financed by research subsidies granted by the government of Upper Austria within the research projects MIMAS.ai, MEDUSA (FFG grant no. 872604) and ARES (FFG grant no. 892166). RISC Software GmbH is Member of UAR (Upper Austrian Research) Innovation Network.

REFERENCES

Al-Aradi, A., Correia, A., Jardim, G., de Freitas Naiff, D., and Saporito, Y. (2022). Extensions of the deep galerkin method. *Applied Mathematics and Computation*, 430:127287.

Amalinadhi, C., Palar, P. S., Stevenson, R., and Zuhal, L. (2022). On physics-informed deep learning for solving navier-stokes equations. In *AIAA SCITECH 2022 Forum*.

Arzani, A., Wang, J.-X., and D'Souza, R. M. (2021). Uncovering near-wall blood flow from sparse data with physics-informed neural networks. *Physics of Fluids*, 33(7):071905.

Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C. W., Choudhary, A., Agrawal, A., Billinge, S. J. L., Holm, E., Ong, S. P., and Wolverton, C. (2022). Recent advances and applications of deep learning methods in materials science. *npj Computational Materials*, 8(1):1–26.

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., and Piccialli, F. (2022). Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88.

Eichinger, M., Heinlein, A., and Klawonn, A. (2021). Stationary flow predictions using convolutional neural networks. *Lecture Notes in Computational Science and Engineering, Numerical Mathematics and Advanced Applications ENUMATH 2019*, pages 541–549.

Eivazi, H., Tahani, M., Schlatter, P., and Vinuesa, R. (2022). Physics-informed neural networks for solving reynolds-averaged navier–stokes equations. *Physics of Fluids*, 34(7):075117.

Fathony, R., Sahu, A. K., Willmott, D., and Kolter, J. Z. (2021). Multiplicative filter networks. In *International Conference on Learning Representations*.

Fenz, W., Dirnberger, J., and Georgiev, I. (2016). Blood flow simulations with application to cerebral aneurysms. In *Proceedings of the Modeling and Simulation in Medicine Symposium*, pages 3:1–3:8. Society for Computer Simulation International.

Fenz, W., Dirnberger, J., Watzl, C., and Krieger, M. (2010). Parallel simulation and visualization of blood flow in intracranial aneurysms. In *11th IEEE/ACM International Conference on Grid Computing*, pages 153–160. ISSN: 2152-1093.

Gao, H., Sun, L., and Wang, J.-X. (2021). PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. *Journal of Computational Physics*, 428:110079.

Gmeiner, M., Dirnberger, J., Fenz, W., Gollwitzer, M., Wurm, G., Trenkler, J., and Gruber, A. (2018). Virtual cerebral aneurysm clipping with real-time haptic force feedback in neurosurgical education. *World Neurosurgery*, 112:e313–e323.

Guo, X., Li, W., and Iorio, F. (2016). Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 481–490.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hennigh, O., Narasimhan, S., Nabian, M. A., Subramaniam, A., Tangsali, K., Fang, Z., Rietmann, M., Byeon, W., and Choudhry, S. (2021). NVIDIA SimNet™: an AI-accelerated multi-physics simulation framework. In *Computational Science – ICCS 2021: 21st International Conference, Krakow, Poland, Proceedings, Part V*, page 447–461. Springer-Verlag.

Jagtap, A. D., Kawaguchi, K., and Karniadakis, G. E. (2020). Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404.

Jiang, C. M., Esmaeilzadeh, S., Azizzadenesheli, K., Kashinath, K., Mustafa, M., Tchelepi, H. A., Marcus, P., Prabhat, M., and Anandkumar, A. (2020). MESHFREEFLOWNET: A physics-constrained deep continuous space-time super-resolution framework. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15.

Jin, X., Cai, S., Li, H., and Karniadakis, G. E. (2021). NSFnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440.

Lakens, D. (2013). Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and ANOVAs. *Frontiers in Psychology*, 4.

Li, J., Yue, J., Zhang, W., and Duan, W. (2022). The deep learning galerkin method for the general stokes equations. *Journal of Scientific Computing*, 93(1):5.

Ma, H., Zhang, Y., Thuerey, N., null, X. H., and Haidn, O. J. (2022). Physics-driven learning of the steady navier-stokes equations using deep convolutional neural networks. *Communications in Computational Physics*, 32(3):715–736.

Markidis, S. (2021). The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *Frontiers in Big Data*, 4.

Matsumoto, M. (2021). Application of deep galerkin method to solve compressible navier-stokes equations. *Transactions of the Japan Society for Aeronautical and Space Sciences*, 64:348–357.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 405–421. Springer International Publishing.

Moser, P., Fenz, W., Thumfart, S., Ganitzer, I., and Giretzlehner, M. (2023). Modeling of 3d blood flows with physics-informed neural networks: Comparison of network architectures. *Fluids*, 8(2):46. Number: 2 Publisher: MDPI.

NVIDIA (2022). NVIDIA Modulus Framework. https://developer.nvidia.com/modulus. accessed 12/2022.

Oldenburg, J., Borowski, F., Öner, A., Schmitz, K.-P., and Stiehm, M. (2022). Geometry aware physics informed neural network surrogate for solving navier–stokes equation (GAPINN). *Advanced Modeling and Simulation in Engineering Sciences*, 9(1):8.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F. A., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97:5301-5310.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707.

Raissi, M., Yazdani, A., and Karniadakis, G. E. (2020). Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030.

Rao, C., Sun, H., and Liu, Y. (2020). Physics-informed deep learning for incompressible laminar flows. *Theoretical and Applied Mechanics Letters*, 10(3):207–212.

Sirignano, J. and Spiliopoulos, K. (2018). DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33:7537-7547. Curran Associates, Inc.

Wang, S., Teng, Y., and Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081.

Wu, B., Hennigh, O., Kautz, J., Choudhry, S., and Byeon, W. (2022). Physics informed RNN-DCT networks for time-dependent partial differential equations. In *Computational Science – ICCS 2022*, Lecture Notes in Computer Science, pages 372–379. Springer International Publishing.