IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 July 2023, accepted 1 August 2023, date of publication 7 August 2023, date of current version 17 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3302892

RESEARCH ARTICLE

Data vs. Physics: The Apparent Pareto Front of Physics-Informed Neural Networks

FRANZ M. ROHRHOFER[1], STEFAN POSCH[2], CLEMENS GÖßNITZER[2], AND BERNHARD C. GEIGER[1], (Senior Member, IEEE)

[1]Know-Center GmbH, Research Center for Data-Driven Business and Big Data Analytics, 8010 Graz, Austria
[2]LEC GmbH, Large Engines Competence Center, 8010 Graz, Austria

Corresponding author: Franz M. Rohrhofer (frohrhofer@acm.org)

The authors would like to acknowledge the financial support of the "COMET Module LEC HybTec" within the "COMET - Competence Centers for Excellent Technologies" Programme of the Austrian Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology (BMK), the Austrian Federal Ministry of Labour and Economy (BMAW) and the Province of Styria. The COMET Programme is managed by the Austrian Research Promotion Agency (FFG).

ABSTRACT Physics-informed neural networks (PINNs) have emerged as a promising deep learning method, capable of solving forward and inverse problems governed by differential equations. Despite their recent advance, it is widely acknowledged that PINNs are difficult to train and often require a careful tuning of loss weights when data and physics loss functions are combined by scalarization of a multi-objective (MO) problem. In this paper, we aim to understand how parameters of the physical system, such as characteristic length and time scales, the computational domain, and coefficients of differential equations affect MO optimization and the optimal choice of loss weights. Through a theoretical examination of where these system parameters appear in PINN training, we find that they effectively and individually scale the loss residuals, causing imbalances in MO optimization with certain choices of system parameters. The immediate effects of this are reflected in the apparent Pareto front, which we define as the set of loss values achievable with gradient-based training and visualize accordingly. We empirically verify that loss weights can be used successfully to compensate for the scaling of system parameters, and enable the selection of an optimal solution on the apparent Pareto front that aligns well with the physically valid solution. We further demonstrate that by altering the system parameterization, the apparent Pareto front can shift and exhibit locally convex parts, resulting in a wider range of loss weights for which gradient-based training becomes successful. This work explains the effects of system parameters on MO optimization in PINNs, and highlights the utility of proposed loss weighting schemes.

INDEX TERMS Multi-objective optimization, Pareto front, physics-informed neural networks, system parameters.

I. INTRODUCTION

Recent developments in scientific computing have led to deep learning approaches that can model the dynamics of physical systems governed by differential equations. State-of-the-art methods often infer the dynamics during model training by leveraging data that embodies the fundamental laws of physics [1], [2], [3]. In contrast, physics-informed neural networks (PINNs) directly encode the governing differential equations as soft constraints via a physics loss function [4], [5]. PINNs enable a seamless integration of data and physics with their respective losses often considered as multi-objective (MO). The large-scale flexibility of PINNs together with their time-continuous, mesh-independent, and unsupervised encoding of differential equations has propelled PINNs into a vast number of multi-scale and multi-physics applications [6], [7]. Today, PINNs are widespread in diverse scientific and engineering disciplines, such as bio-engineering [8], [9], aerodynamics [10], [11], and materials science [12], [13].

The associate editor coordinating the review of this manuscript and approving it for publication was Francesco Piccialli.

Setting up a well-working PINN application, however, is not straightforward. In particular, the vanilla implementation of PINNs is known to be prone to training failures that often lead to inaccurate and nonphysical predictions [14]. The discussion on training failures in PINNs is diverse, and each problem setup seems to present its own unique challenges for PINN optimization [15]. This diversity makes it difficult to choose the right remedy in the face of certain optimization issues. In general, any improvement in the robustness and generalizability of PINNs requires an understanding of the optimization complexity of the physics loss function. As a general rule, its complexity increases as the physical system and governing differential equations become more complex. This is true of systems with highly-nonlinear [16], chaotic [17], or multiscale dynamics [18]. To cope with complex systems and geometries, domain decomposition or sequence-to-sequence methods have been developed that divide the original problem into smaller subdomains [16], [19]. Each subdomain is then tackled by a separate PINN, resulting in an overall lower optimization complexity. Furthermore, soft attention mechanisms were introduced to focus the physics loss optimization on regions which are typically hard to resolve, such as discontinuities [20] and stiff dynamics [21]. For systems that are slightly more complex than a previously solved problem, curriculum regularization provides a simple starting point for the PINN optimization, which gradually becomes more complex as the PINN is trained [16]. Yet even with simple systems, the optimization may converge to suboptimal solutions that describe trivial solutions to the physics loss function [22]. In this regard, learning in sinusoidal space [25] or methods that respect causality [26] provide a potential remedy.

While all the above mentioned circumstances and proposed modifications particularly apply to optimization issues related to the physics loss function, a major class of discussions addresses issues related to the MO optimization in PINNs. Loss weighting schemes are arguably the most frequently used modifications to the vanilla PINN framework.

A. LOSS WEIGHTING SCHEMES

Loss weighting schemes originate from issues related to MO optimization [27]. In PINNs, MO optimization is inherently specified by multiple loss functions which are related to either data or physics. In this particular context, we use the term ''data'' to specify the use of loss functions with labeled data. Labeled data is typically used in PINNs to encode initial and boundary conditions of forward problems or to impose additional data constraints, e.g., with data coming from experiments. With the term ''physics'', we refer to the use of loss functions that encode the governing differential equations. These types of loss functions do not use any labeled data, as further discussed in Section II. Although the total number of loss functions can be reduced, e.g., by using hard constraints [23], [24], most PINN applications involve the use of multiple loss functions. The standard approach to minimizing them is gradient-based optimization of a linear scalarized MO problem, given by

$$\min_{\theta} \sum_{i} \alpha_i \mathcal{L}_i(\theta), \tag{1}$$

with θ denoting the network weights being optimized, $\mathcal{L}_i$ the constituent loss functions and α_i their respective loss weights. The vanilla PINN framework uses an unweighted scalarization, hence $\alpha = 1$ for any given loss function [5]. However, it has been frequently reported that this simple linear combination often leads to optimization failures, which are characterized by a stalled minimization of one loss and high prediction errors. Consequently, it has become a standard procedure in PINNs to adjust loss weights, which are trimmed (often in a trial-and-error procedure) until a sufficiently accurate prediction is obtained. Several adaptive loss weighting schemes have been introduced to overcome the cumbersome tuning of manual loss weights. These schemes focus on what is observed during the PINN training and rely on mean gradient statistics [27], [28], inverse average gradient magnitudes [29], or maximum likelihood estimations [30]. Recently, formulations of a constrained optimization problem that use Lagrangian multipliers have also appeared in the literature [31].

B. CONTRIBUTION

The use of loss weighting schemes in PINNs has become an essential approach to solving convergence issues related to MO optimization and/or refining the accuracy of final network predictions. Yet when and why the need for loss weighting schemes arises in the first place has rarely been discussed in terms of the investigated physical system and its properties. In this work, we therefore focus on how system parameters, such as characteristic length and time scales, the size of the computational domain, and coefficients of differential equations, affect MO optimization in PINNs. We observe that the choice of system parameters influences the absolute scale of both data and physics residuals. For certain choices of system parameters, this can result in imbalanced scales of loss residuals. This has an immediate effect on MO optimization and the ''apparent'' Pareto front, which we define as the set of loss values that are achievable with gradient-based training. By analyzing the apparent Pareto front and final PINN predictions, we find that loss weights can be used to compensate for the scaling effects of system parameters, with their optimal choice following the prescribed trend of which loss residuals dominate the MO optimization. Furthermore, we find that the apparent Pareto front of more balanced residuals can exhibit locally convex parts, which results in a wider range of loss weights with which gradient-based training becomes successful. Our results provide valuable insights for MO optimization in PINNs and contribute to a better understanding of the influence of system parameters.

The remainder of this paper is arranged as follows: In Section II, we discuss the fundamental working method of vanilla PINNs and provide the details of MO optimization

and the apparent Pareto front. In Section III, we introduce two physical systems on which we perform our analysis. The two systems employ the diffusion equation and Navier-Stokes equations, which are well-known differential equations frequently used in the PINN literature. In Section IV, we theoretically analyze the use of feature scaling to demonstrate how and where system parameters appear in the PINN training and affect the scale of loss residuals. In Section V, we empirically study the apparent Pareto front and trend of optimal loss weights while altering the system parameterization. Section VI provides a discussion of our findings and Section VII a conclusion.

II. PHYSICS-INFORMED NEURAL NETWORKS

To briefly demonstrate the fundamental working method of PINNs, we take the originally proposed PINN framework [5] and consider a parameterized partial differential equation (PDE) of the general form

$$\frac{\partial}{\partial t}u(t,x) + \mathcal{F}[u;\lambda] = 0, \quad x \in \Omega, t \in [0,T], \tag{2}$$

where $u(t,x)$ is the solution function, $\mathcal{F}[\cdot;\lambda]$ represents an arbitrary potentially nonlinear differential operator with parameterization λ, and Ω represents the spatial computational domain which is a subset of $\mathbb{R}^D$. We proceed by approximating the solution function $u(t,x)$ with a fully-connected neural network $u_\theta(t,x)$ with weights θ.

A. DATA AND PHYSICS LOSS FUNCTIONS

This paper only considers well-posed problems, i.e., problems with a unique solution function u, which is attained by imposing sufficient initial and boundary conditions (IC and BC). The IC and BC define the solution function at the boundary of the computational domain, i.e., at $t = 0$ and $x \in \partial\Omega$, respectively, and provide a basically infinite set of labeled training data that can be sampled once prior to network training or anew at each training epoch. The premise for encoding the IC and BC, this dataset is used by PINNs in the data loss function

$$\mathcal{L}_\mathcal{D}(\theta) = \frac{1}{N_\mathcal{D}}\sum_{i=1}^{N_\mathcal{D}} |e_i|^2 \text{, with} \tag{3a}$$

$$e_i = u_\theta(t_i,x_i) - u_i, \tag{3b}$$

where $\mathcal{L}_\mathcal{D}$ denotes the mean squared error loss (MSE) function given by the data residuals e_i which are determined from the labeled dataset $\{(t_i,x_i),u_i\}_{i=1}^{N_\mathcal{D}}$. Details of the exact definition of IC and BC in our experiments are given in Section III.

To encode the governing differential equation, PINNs make use of automatic differentiation [32] that retrieves (partial) derivatives of the neural network function at specified coordinates, commonly called collocation points. Derivatives can be evaluated up to the order to which the neural network activation function is differentiable. Activation functions commonly used in PINNs are thus the hyperbolic tangent (tanh), the sinusoidal (sin), or the swish. With the neural network derivatives evaluated at the collocation points, the physics loss function is given by

$$\mathcal{L}_\mathcal{F}(\theta) = \frac{1}{N_\mathcal{F}}\sum_{i=1}^{N_\mathcal{F}} |f_i|^2 \text{, with} \tag{4a}$$

$$f_i = \frac{\partial}{\partial t}u_\theta(t_i,x_i) + \mathcal{F}[u_\theta(t_i,x_i);\lambda], \tag{4b}$$

where $\mathcal{L}_\mathcal{F}$ denotes the MSE loss function given by the physics residuals f_i which are determined from the unlabeled dataset $\{(t_i,x_i)\}_{i=1}^{N_\mathcal{F}}$. In comparison to conventional PDE solvers that use a predefined computational mesh, the choice of collocation points in PINNs is not restricted, i.e., they can be randomly sampled from inside the function domain once or at each epoch, e.g., by using hyper-cube sampling.

B. THE APPARENT PARETO FRONT

Vanilla PINNs and many variants thereof handle data and physics loss functions in a MO manner. The standard approach in PINNs is to simultaneously minimize both by defining a single or total loss function via linear scalarization of the MO problem previously introduced by (1). The particular use of loss weights and their optimal choice differs in the literature (see Section I-A): they depend on the total number of loss functions in use and are tuned manually or in an adaptive weighting scheme.

From a theoretical perspective, any choice of loss weight in a scalarized problem selects a Pareto optimal solution to the MO optimization problem [33]. The Pareto front can then be seen as the set of all possible Pareto optima that are attained by continuous adjustment of the loss weights. In fact, any use of loss weights in a well-posed problem would converge to the same Pareto optima as the true solution of (2), since a perfectly approximated solution function yields zero for all losses. However, finite-sized networks and highly complex loss functions are used in practice, so the gradient decent optimization eventually converges to local optima. Hence these optima do not represent true Pareto optimal solutions but approximations that are found along the optimization path as selected by the particular choice of loss weights. For our analysis, we thus define the "apparent" Pareto front as the set of loss values that are achievable with gradient-based optimization. To empirically obtain the apparent Pareto front, we train several PINN instances with different loss weights and observe where the gradient-based optimization has converged (see Section V for further details).

For the sake of simplicity and demonstration purposes, this paper only distinguishes between data and physics loss functions and introduces a single parameter that is manually tuned and simultaneously trades both by

$$\mathcal{L}(\theta;\alpha) := \alpha\mathcal{L}_\mathcal{D}(\theta) + (1-\alpha)\mathcal{L}_\mathcal{F}(\theta), \tag{5}$$

where $\alpha \in (0,1)$. According to this definition, $\alpha \to 1$ favors low data losses, while $\alpha \to 0$ favors low physics losses.

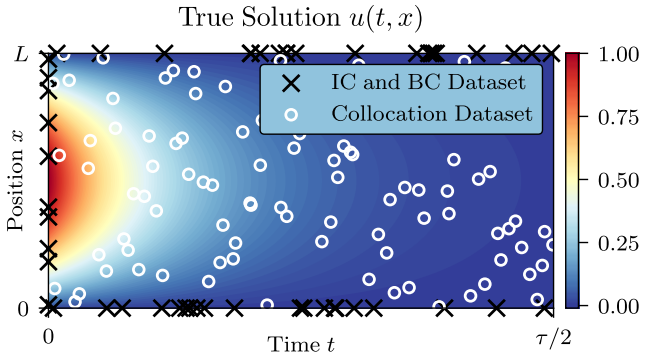

FIGURE 1. Geometrical setup and reference solution for the diffusion example. Representative sample of training data is shown as black crosses (IC and BC) and white circles (collocation).

A loss weight of $\alpha = 0.5$ relates to unweighted MO optimization as used by vanilla PINNs.

III. EXPERIMENTAL SETUP

A. DIFFUSION EXAMPLE

The diffusion equation, along with variants thereof, is among the most widely studied parabolic PDEs with applications in many fields of science, pure mathematics, and engineering. In this work we study the heat equation, a special case of the diffusion equation in the context of engineering, specifically considering the cooling of a one-dimensional rod with an initial temperature distribution and Dirichlet boundary conditions on both rod ends. The dynamics of the system are described by the equation

$$\frac{\partial}{\partial t}u(t,x) = \kappa \frac{\partial^2}{\partial x^2}u(t,x), \tag{6}$$

where the solution function u represents the temperature of the rod at position x and time t, and κ is the thermal diffusivity. We define the initial (IC) and boundary (BC) conditions as

$$\text{IC}: \; u(0,x) = \sin\left(\pi \frac{x}{L}\right) \quad x \in [0, L], \tag{7}$$

$$\text{BC}: \; u(t,0) = u(t,L) = 0 \quad t \in [0, T], \tag{8}$$

with L denoting the length of the rod and T the simulation time. For our later analysis and the sake of simplicity, we introduce the characteristic (diffusive) time scale

$$\tau := \frac{L^2}{\kappa}, \tag{9}$$

and consider the simulation time as multiples of it, i.e., $T = \lambda\tau$ with $\lambda \in \mathbb{R}_*^+$.

The problem stated above is well-posed and has a unique solution

$$u(x,t) = \sin\left(\pi \frac{x}{L}\right) e^{-\pi^2 t/\tau}. \tag{10}$$

Fixing λ while adjusting L and κ leaves the intrinsic diffusion process and solution function unaffected. This gives us a controllable setting of different system parameterizations all described by the same solution function (cf. Fig. 1). In an engineering context, this is known as dynamic similitude.

B. NAVIER-STOKES EXAMPLE

The Navier-Stokes equations are the governing PDEs in the study of fluid flow. Along with the continuity equation, they describe the conservation of momentum and mass of viscous fluids. Their importance in scientific and engineering modeling is indisputable as they are applied to many physical phenomena occurring in weather forecasting, blood flow through vessels, or flow around obstacles. For our study, we limit their application to a two-dimensional incompressible steady-state flow where the solution function to be approximated by the PINN is given by the vector-valued function $(x, y) \mapsto (u, v, p)$. The governing equations for the conservation of momentum and mass, respectively, are

$$(\mathbf{u}\cdot\nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu\nabla^2\mathbf{u} \tag{11a}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{11b}$$

where $\mathbf{u} = (u, v)$ and p denote fluid velocity in the x- and y-directions and pressure, respectively, and ρ and ν are the fluid density and viscosity. The continuity equation (11b) can be hard coded with PINNs (see [5]) which leaves behind the conservation of momemtum (11a) to be encoded in the physics loss function.

Analytical solutions in fluid dynamics are rare. Therefore, we consider the laminar fluid flow in the wake of a two-dimensional grid which has been solved analytically and is better known as Kovasznay flow [34]. With L as the characteristic spacing of the grid, u_0 the mean velocity in the x-direction and assuming constant density ρ, the analytical solution to this problem is given by

$$u(x,y) = u_0\left(1 - e^{\gamma\frac{x}{L}}\cos\left(2\pi\frac{y}{L}\right)\right), \tag{12a}$$

$$v(x,y) = \frac{u_0\gamma}{2\pi}e^{\gamma\frac{x}{L}}\sin\left(2\pi\frac{y}{L}\right), \tag{12b}$$

$$p(x,y) = u_0^2 e^{2\gamma\frac{x}{L}} + C, \tag{12c}$$

where C is a constant and

$$\gamma = \frac{1}{2\nu} - \sqrt{\frac{1}{4\nu^2} + 4\pi^2}. \tag{13}$$

To study this problem with different system parameterizations, we again make use of the concept of dynamic similitudes and consider a fixed Reynolds number $\text{Re} = Lu_0/\nu$ for the above mentioned system. Under these conditions, we adjust the mean velocity by $u_0 = 1/L$ (neglecting physical units) so that any selected grid spacing L represents the same fluid flow. To establish geometric similarity, the spatial extension is assumed to be equal in both directions, i.e., $x \in [0, L]$ and $y \in [0, L]$ (cf. Fig. 2).

In this example, the PINN is trained on boundary data sampled from the analytical solution of the flow velocities (12a) and (12b) at the boundary. Since the flow is steady state, there is no IC. The pressure is entirely learned by the PINN and inferred from the set of governing PDEs (11a).

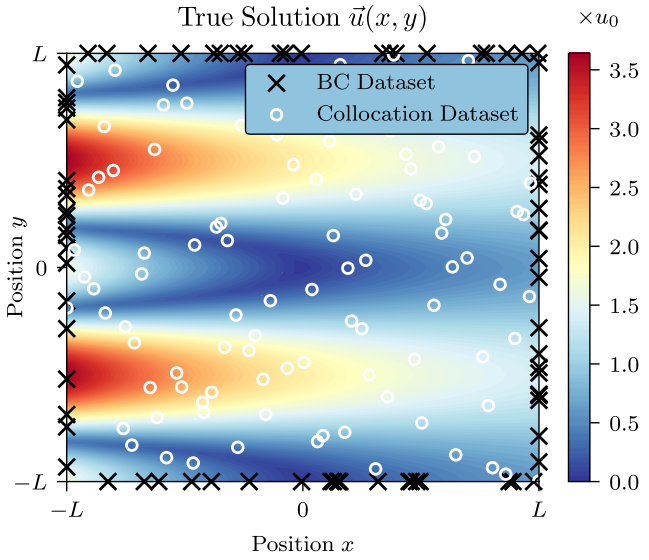

FIGURE 2. Geometrical setup and reference solution for the Navier-Stokes example. Representative sample of training data is shown as black crosses (IC and BC) and white circles (collocation).

IV. EFFECTS OF SYSTEM PARAMETERS ON LOSS RESIDUALS

In this section, we show that the absolute scale of data and physics residuals depends on the underlying system parameters. We discuss the effects of feature scaling, which is a necessary step to bring the input dimensions into a range suitable for gradient descent optimization. Our focus is on min-max feature scaling since its input ranges are well-defined and given by the computational domain.

A. DIFFUSION EXAMPLE

For the diffusion equation, we consider the input ranges $(t, x) \in [0, T] \times [0, L]$ scaled to the unit interval $[0, 1]^2$

$$\hat{t} = \frac{t}{T}, \quad \text{and} \quad \hat{x} = \frac{x}{L}, \tag{14}$$

where $\hat{t}$ and $\hat{x}$ denote the scaled input variables. It is noteworthy that applying feature scaling can be seen as scaling the network function to the physical system, thus adapting to characteristic time and length scales as apparent by (14). The physics residuals for the diffusion equation can thus be written in terms of the scaled and characteristic quantities:

$$f_i = \frac{1}{T}\frac{\partial}{\partial \hat{t}} u_\theta(\hat{t}_i, \hat{x}_i) - \frac{\kappa}{L^2}\frac{\partial^2}{\partial \hat{x}^2} u_\theta(\hat{t}_i, \hat{x}_i), \tag{15}$$

where the additional pre-factors in the equation appear when the chain rule of derivatives is applied. We again consider the simulation time as multiples of the characteristic diffusive time scale, i.e., $T = \lambda\tau = \lambda L^2/\kappa$, and rearrange the physics residuals according to this definition to yield the following:

$$f_i = \frac{\kappa}{L^2}\left(\frac{1}{\lambda}\frac{\partial}{\partial \hat{t}} u_\theta(\hat{t}_i, \hat{x}_i) - \frac{\partial^2}{\partial \hat{x}^2} u_\theta(\hat{t}_i, \hat{x}_i)\right). \tag{16}$$

It is now apparent that applying feature scaling re-parameterizes the diffusion equation encoded in the physics loss function and effectively scales the physics residuals by $f_i \sim \kappa/L^2$. In contrast, the data residuals are not affected by those system parameters and stay in the range of $e_i \sim 1$ due to the chosen IC (7). Consequently, data and physics residuals scale differently according to the scale ratio given by

$$f_i/e_i \sim \frac{\kappa}{L^2}. \tag{17}$$

This shows how a certain choice of L and κ overbalances their scale. While for $\kappa/L^2 \gg 1$ the scale of physics residuals is predominant, for $\kappa/L^2 \ll 1$ the scale of data residuals is predominant. This scaling in turn directly affects the loss values in (5) and respective gradients with measurable consequences for MO optimization [27].

B. NAVIER-STOKES EXAMPLE

For the Navier-Stokes system, we consider the input ranges $(x, y) \in [-L, L]^2$ scaled to the interval $[-1, 1]^2$

$$\hat{x} = \frac{x}{L}, \qquad \hat{y} = \frac{y}{L}, \quad \text{thus} \quad \hat{\nabla} = L\nabla. \tag{18}$$

This time, the solution function will be affected by the particular choice of L, since $u_0 = 1/L$ (see experimental setup III-B). Hence, we consider the velocities and pressure in terms of scaled quantities that are commonly used in the nondimensionalization of the Navier-Stokes equations:

$$\hat{\mathbf{u}} := \frac{\mathbf{u}}{u_0}, \quad \text{and} \quad \hat{p} := \frac{p}{\rho u_0^2}. \tag{19}$$

Rewriting the physics residuals in terms of the scaled quantities yields

$$f_i = \frac{u_0^2}{L}\left(\hat{\mathbf{u}}_\theta \cdot \hat{\nabla}\right)\hat{\mathbf{u}}_\theta + \frac{u_0^2}{L}\hat{\nabla}\hat{p}_\theta - \nu\frac{u_0}{L^2}\hat{\nabla}^2\hat{\mathbf{u}}_\theta, \tag{20}$$

where for clarity we have omitted the spatial dependencies $\hat{\mathbf{u}}_\theta(\hat{x}, \hat{y})$ and $\hat{p}_\theta(\hat{x}, \hat{y})$. After $u_0 = 1/L$ is used and rearranged,

$$f_i = \frac{1}{L^3}\left((\hat{\mathbf{u}}_\theta \cdot \hat{\nabla})\hat{\mathbf{u}}_\theta + \hat{\nabla}\hat{p}_\theta - \nu\hat{\nabla}^2\hat{\mathbf{u}}_\theta\right) \tag{21}$$

it is again apparent that the physics residuals are scaled by the factor $f_i \sim 1/L^3$, while the data residuals are scaled by $e_i \sim 1/L$ since $e_i = u_0(\hat{\mathbf{u}}_\theta(\hat{x}_i, \hat{y}_i) - \hat{\mathbf{u}}_i)$. Consequently, data and physics residuals are scaled differently with their scale ratio given by

$$f_i/e_i \sim \frac{1}{L^2}. \tag{22}$$

Again, this demonstrates that a certain choice of L overbalances their scale with consequences for MO optimization.

V. STUDYING THE APPARENT PARETO FRONT AND OPTIMAL CHOICE OF LOSS WEIGHTS

In this section, we empirically analyze the scaling effects of system parameters on MO optimization of data and physics. To understand the qualitative trend of optimal loss weights, we analyze the apparent Pareto front with different system parameterizations and study which loss weights achieve a

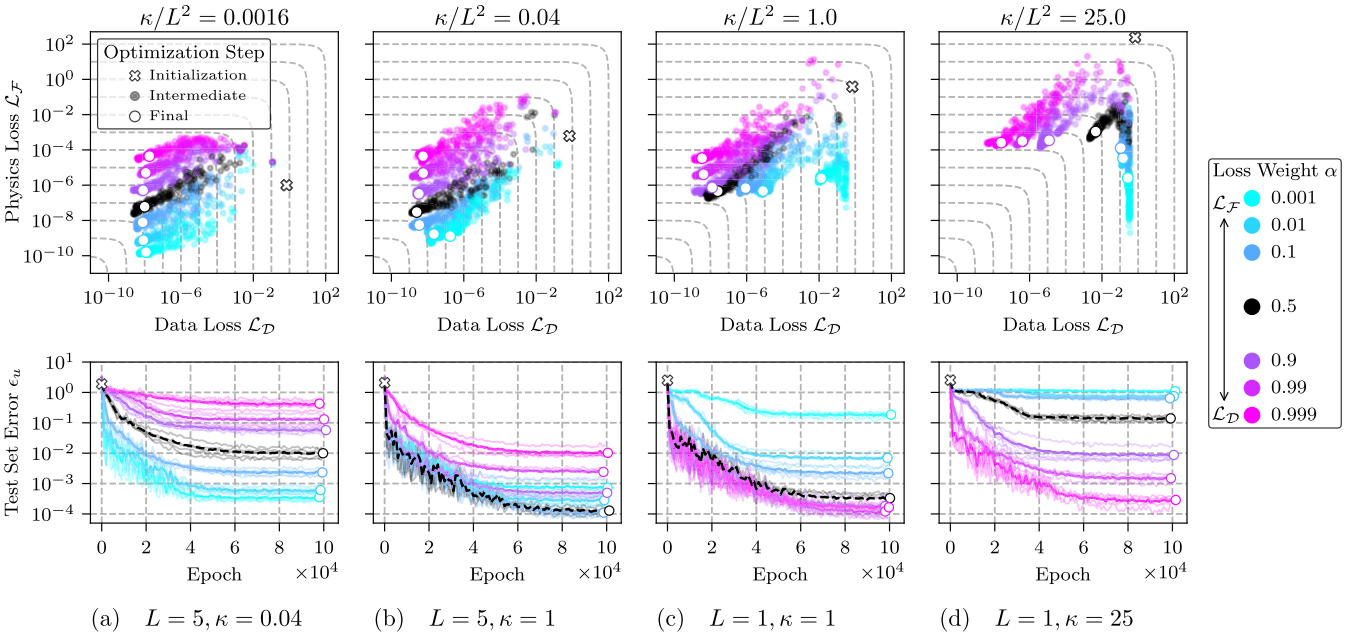

FIGURE 3. Data vs. physics loss (top row) and test set errors (bottom row) for the diffusion equation. Different system parameterizations are arranged as columns. The system parameters determine the residual scale ratio $f_i/e_i \sim \kappa/L^2$, where for $\kappa/L^2 \gg 1$ the scale of physics residuals is predominant and for $\kappa/L^2 \ll 1$ that of data residuals. Final optimization steps (empty circles) in the top row outline the apparent Pareto front.

good PINN performance in terms of the relative ℓ^2 error. We use the concept of dynamic similitudes (see experimental setup III) to study one and the same physical system with individual sets of system parameters, which causes different scaling effects captured by the residual scale ratio f_i/e_i.

A. PINN SETUP

We optimize PINNs with four hidden layers and 50 neurons per layer, employing a hyperbolic tangent (tanh) activation function for the hidden layers and a linear activation for the output layer.[1] Network weights are initialized using the Glorot uniform [35] initializer. We use Adam [36] to minimize the MO loss (5) with a learning rate of 0.01. Training is performed for 10^5 epochs to ensure that the optimization has converged to a solution, where measured quantities have stabilized and no longer change substantially. Training data is sampled anew at each epoch with $N_\mathcal{F} = 1024$ (unlabeled collocation points) and $N = 128$ (labeled data points) at each boundary, thus $N_\mathcal{D} = 384$ for the diffusion example (cf. Fig. 1) and $N_\mathcal{D} = 512$ for the Navier-Stokes example (cf. Fig. 2).

To evaluate the accuracy of predictions, we measure the relative ℓ^2 error on an independent test set, which is randomly sampled inside the computational domain with $N = 1024$. The relative ℓ^2-error is given by

$$\epsilon_u \equiv \frac{\|u - u_\theta\|_2}{\|u\|_2}, \tag{23}$$

where u is obtained from the analytical solutions (10) and (12). For qualitative analysis of the apparent Pareto front, we train PINN instances with loss weights $\alpha = \{0.001, 0.01, 0.1, 0.5, 0.9, 0.99, 0.999\}$. Test runs are repeated with five unique seeds for data sampling and network initialization.

Note that we have also performed tests with different activation functions (sin, swish), network architectures (2×30, 6×100), learning rates (0.001, 0.0001), and training set sizes, but the results had a similar qualitative outcome as presented in the subsection. For the sake of simplicity, the following subsections are concerned with only the setting presented above.

B. DIFFUSION EXAMPLE

The diffusion example is parameterized by L, κ, and λ, where for a fixed value of λ any choice of L and κ represents the same underlying system dynamics as with the analytical solution given by (10), cf. Fig. 1. In this example, the residual scale ratio is determined by the system parameters through $f_i/e_i \sim \kappa/L^2$. To study the effects on the apparent Pareto front and the optimal choice of loss weights, we thus choose $\lambda = 0.5$ for our experiment and take $(L, \kappa) \in \{(5, 0.04), (5, 1), (1, 1), (1, 25)\}$ to cause a respective scaling by $\kappa/L^2 \in \{0.0016, 0.04, 1, 25\}$. We have also performed a test with $\lambda \in \{0.1, 1, 10\}$, but its results revealed qualitative behavior similar to what is presented in this section.

Fig. 3 shows the results of this experiment; the different system parameterizations are arranged by column. The top row displays the history of loss tuples ($\mathcal{L}_\mathcal{D}$ and $\mathcal{L}_\mathcal{F}$) in each of the trained PINN instances as recorded during the training.

[1] All code is available on GitHub at https://github.com/frohrhofer/PINN_pareto

To increase clarity and comprehension, initial (epoch 0) and final (epoch 10^5) optimization steps are averaged across the five different PINN runs and highlighted as empty crosses and circles, respectively. It is clear that each PINN instance starts close to the marked initialization step and eventually converges to different regions as determined by the loss weight α. The apparent Pareto front can thus be seen as a theoretical interpolation curve of the final loss values indicated by the empty circles. Note that as a result of utilizing a stochastic optimization method, it is possible for those points to have slightly worse values than those found in intermediate optimization steps. The bottom row of the figure shows the respective test set errors (ϵ_u) as a function of the optimization step. The test set errors are again averaged over the five PINN instances using different seeds, which are presented by thick lines while individual runs are thin. The black dashed line highlights the unweighted MO, and the empty circles at the end of the test error curves are randomly shifted vertically to reduce overlap.

Looking at the test set error in the bottom row of the figure, we observe a common trend from left to right: in the presence of predominant data residuals (Fig. 3a), a higher weighting of the physics loss ($\alpha \to 0$) yields low test set errors. On the contrary, when predominantly physics residuals are present (Fig. 3d), a higher weighting of the data loss ($\alpha \to 1$) is necessary to achieve comparable low errors. Intermediate scale factors (Fig. 3b and 3c) follow this trend where the unweighted optimization ($\alpha = 0.5$) already achieves comparably low errors due to the more balanced residual scales. Yet when $\kappa/L^2 = 1$, performance is slightly better when a higher weight is given to data loss.

The apparent Pareto front in the top row provides further insights: in general, we observe the apparent Pareto front substantially changing its shape with different system parameterizations. In Fig. 3b and 3c, we find that the apparent Pareto front in the lower left region locally exhibits convex parts (not explicitly highlighted), where generally a wider range of loss weights achieves similarly accurate results, i.e., MO optimization is less sensitive to the particular choice of the loss weight α. In contrast are the apparent Pareto fronts in the presence of unbalanced residual scales (Fig. 3a and 3d), where only a particular choice of α, here either $\alpha \to 0$ or $\alpha \to 1$, yields accurate results.

C. NAVIER-STOKES EXAMPLE

The Navier-Stokes setup has a single system parameter L that simultaneously affects both residual scales; their scale ratio is determined by $f_i/e_i \sim 1/L^2$. We thus choose $L \in \{25, 5, 1, 0.2\}$ to cause a respective scaling by $1/L^2 \in \{0.0016, 0.04, 1, 25\}$.

The history of loss tuples ($\mathcal{L}_\mathcal{D}$ and $\mathcal{L}_\mathcal{F}$), and test set errors (ϵ_p) for this experiment is found in Fig. 4. The same explanation of the figure that was provided in the previous subsection also applies here. Note that since the relative ℓ^2 error is dependent on the absolute range of the target value and the range in this example varies with regard to different system parameterizations, a direct quantitative comparison of test errors between different settings of L is not possible. Furthermore, since the velocity errors (ϵ_u and ϵ_v) exhibit behavior qualitatively similar to that of the pressure field, their display is omitted for the sake of clarity.

Similar to the previous example, a common trend is observed from left to right: in the presence of predominant data residuals (Fig. 4a), a higher weighting of the physics loss ($\alpha \to 0$) achieves lower prediction errors. In contrast, the setting with predominantly physics residuals (Fig. 4d) requires higher weighting of the data loss ($\alpha \to 1$) to yield accurate results. The trend is similar with intermediate settings (Fig. 4b and 4c), where the unweighted MO already achieves accurate results when $1/L^2 = 0.04$, while a slightly higher weighting of the data loss works better when $1/L^2 = 1$.

Regarding the apparent Pareto front, we once again observe that its shape is substantially influenced by the chosen parameterization. As observed in the previous example, locally convex parts emerge in the lower left corner of the apparent Pareto front (here in Fig. 4a-c), and this coincides with a wider range of loss weights that achieve similar accuracy.

VI. DISCUSSION

It has become widely known that the success of PINN training in the sense of obtaining the physically correct solution depends on a myriad of factors, and hence requires a variety of mitigation methods. In general, these factors can be categorized as depending on model parameters and system parameters, though we believe that this separation is less clear than in classical deep learning problems.

This paper has shed more light on the effect of system parameters. Specifically, Section IV shows mathematically with two systems of differential equations that changing the size of the computational domain, e.g., by altering the system parameterization, affects the loss residuals used in MO optimization. Our experiments in Section V are in line with our mathematical analysis and show how varying the system parameters leads to different scaling factors of loss residuals, which in turn requires different loss weights α for PINN training to be successful. Altering the system parameterization thus effectively converts one specific PINN training instance to another that is characterized by different loss weights.[2] This insight resonates with and mathematically supports the literature that claims that successful PINN training requires carefully selected loss weights or proposes automatic loss weighting schemes. While proposed methods are usually tailored to specific cases and directly address issues as they arise during training, our results offer a more comprehensive view and focus on potential root causes of issues that may arise from inappropriate system parameterizations.

From the perspective of differential equations, the computational domain is rarely assumed to be fixed. Instead and especially in computational fluid dynamics problems

[2] Additional investigations, which are not reproduced here, have revealed that the new PINN training instance may also be characterized by differently parameterized differential equations.

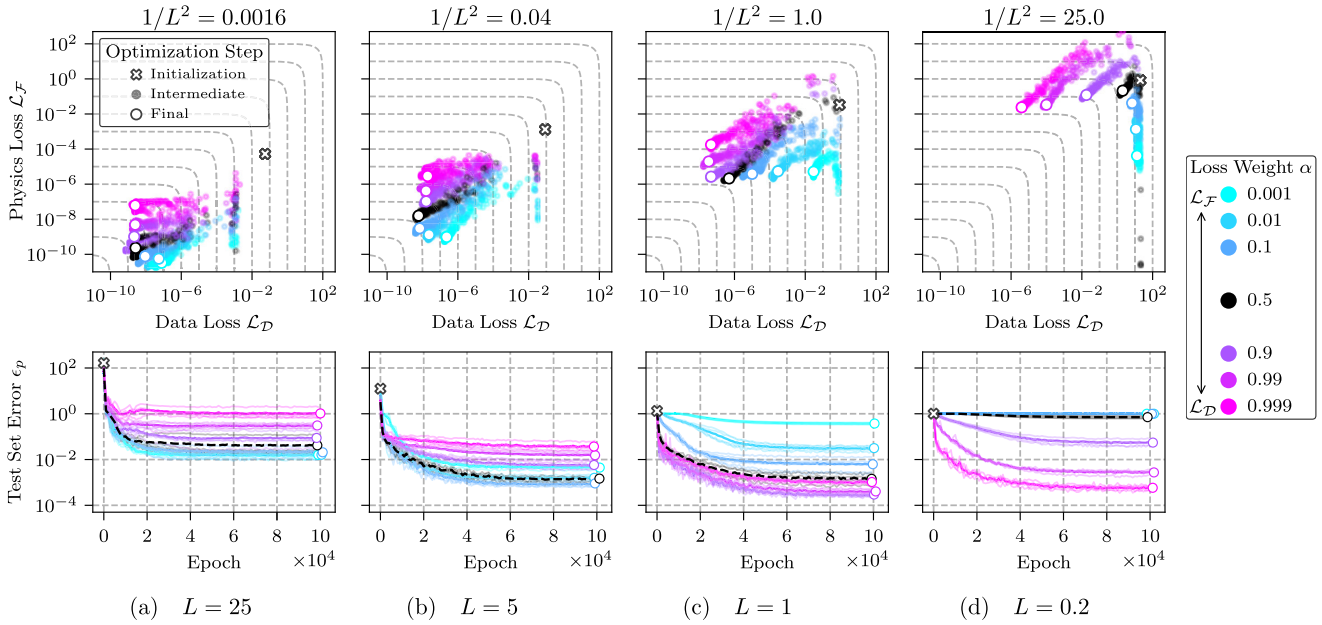

FIGURE 4. Data vs. physics loss (top row) and test set errors (bottom row) for the Navier-Stokes equations. Different system parameterizations are arranged as columns. The system parameters determine the residual scale ratio $f_i/e_i \sim 1/L^2$, where for $1/L^2 \gg 1$ the scale of physics residuals is predominant and for $1/L^2 \ll 1$ that of data residuals. Final optimization steps (empty circles) in the top row outline the apparent Pareto front.

are often nondimensionalized or scaled. In other words, the parameters of the system under consideration are reduced to their minimum number, and often normalized with regard to intrinsic quantities (e.g., the spatial domain is normalized with regard to a characteristic length scale). Nondimensionalization therefore suggests a certain set of system parameters and subsequently a set of ideal loss weights. Note that the ideal loss weights still have to be discovered, even those in nondimensionalized systems: In Fig. 4 (*c*), it can be seen that the best test set error is achieved with a loss weight of $\alpha = 0.9$.

From a practical perspective, we believe that the insights from this work can help to improve PINN training. For example, suppose that a parameterization (nondimensionalized or not) and PINN configuration (network architecture, activation function, loss weights, etc.) are known that achieve satisfactory training performance with a given system of differential equations. Such configurations may be taken from existing literature that has successfully applied PINNs to problems of practical relevance. To solve an instance of the same problem with different system parameters, the results presented here can now be taken to study how the individual loss terms of the new problem scale relative to the original problem as a function of the changed system parameters. This can then be used to adjust the loss weights of the new PINN instance. (Yet note that with this loss weight adjustment, the parameters of the differential equation may also change relative to the original and new systems.) Supposing that each differential equation and each dimension of the IC and BC losses is assigned a separate loss weight, a certain number of system parameter changes can be compensated for. If this does not suffice to ensure successful PINN training, one can resort to other methods that directly affect the system parameters (see Section I-A). These include domain decomposition (which either explicitly or indirectly, via adaptive collocation point sampling, affects the effective size of the computational domain), and curriculum learning (which starts PINN training at "simple" parameterizations and slowly changes the parameters to those of the new system). Moreover, when dealing with solution functions that have challenging regions to learn, such as discontinuities or stiff systems, the physics residuals can become comparably larger in these regions and exceed the scaling effects of system parameters. Therefore, it may become necessary to weight individual collocation points to handle such situation effectively. Future research should investigate whether such a general procedure makes PINN training more reliable; this paper has taken an important step in this direction.

VII. CONCLUSION

In this paper, we highlighted the importance of understanding the role of system parameters in the training of PINNs. We demonstrated that system parameters such as characteristic length and time scales, the computational domain, and coefficients of differential equations influence the absolute scale of data and physics residuals. We theoretically and empirically verified that this has a consequence for PINN training, especially when data and physics loss functions are minimized through a scalarized MO optimization problem. As measured by the set of loss values that are achievable with gradient-based training and defined as the "apparent" Pareto front, we observed that its shape and the optimal choice of loss weights are determined by the chosen system parameters. Moreover, we showed by altering the system parameterization that the apparent Pareto front can exhibit locally convex parts, resulting in a wider range of loss weights

for which gradient-based training is successful. These findings provide important insights into the MO optimization of PINNs and suggest that the common practice of nondimensionalization can have significant implications for PINN training. Ultimately, this work contributes to the development of more effective and efficient loss weighting schemes, which take into account fundamental properties of physical systems governed by differential equations.

REFERENCES

[1] S.-M. Udrescu and M. Tegmark, "AI Feynman: A physics-inspired method for symbolic regression," *Sci. Adv.*, vol. 6, no. 16, Apr. 2020, Art. no. eaay2631.
[2] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 15, pp. 3932–3937, Apr. 2016.
[3] H. Shi, Q. Yao, and Y. Li, "Learning to simulate crowd trajectories with graph networks," in *Proc. ACM Web Conf.*, Apr. 2023, pp. 8459–8468.
[4] I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 987–1000, Jan. 1998.
[5] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Feb. 2019.
[6] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations," *Science*, vol. 367, no. 6481, pp. 1026–1030, Feb. 2020.
[7] Y. Chen, L. Lu, G. E. Karniadakis, and L. Dal Negro, "Physics-informed neural networks for inverse problems in nano-optics and metamaterials," *Opt. Exp.*, vol. 28, no. 8, p. 11618, 2020.
[8] F. Sahli Costabal, Y. Yang, P. Perdikaris, D. E. Hurtado, and E. Kuhl, "Physics-informed neural networks for cardiac activation mapping," *Frontiers Phys.*, vol. 8, p. 42, Feb. 2020.
[9] G. Kissas, Y. Yang, E. Hwuang, W. R. Witschey, J. A. Detre, and P. Perdikaris, "Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks," *Comput. Methods Appl. Mech. Eng.*, vol. 358, Jan. 2020, Art. no. 112623.
[10] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," *Comput. Methods Appl. Mech. Eng.*, vol. 360, Mar. 2020, Art. no. 112789.
[11] A. Dourado and F. A. C. Viana, "Physics-informed neural networks for missing physics estimation in cumulative damage models: A case study in corrosion fatigue," *J. Comput. Inf. Sci. Eng.*, vol. 20, no. 6, pp. 1–12, Dec. 2020.
[12] Q. He, D. Barajas-Solano, G. Tartakovsky, and A. M. Tartakovsky, "Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport," *Adv. Water Resour.*, vol. 141, Jul. 2020, Art. no. 103610.
[13] M. Yin, X. Zheng, J. D. Humphrey, and G. E. Karniadakis, "Non-invasive inference of thrombus material properties with physics-informed neural networks," *Comput. Methods Appl. Mech. Eng.*, vol. 375, Mar. 2021, Art. no. 113603.
[14] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 26548–26560.
[15] S. Monaco and D. Apiletti, "Training physics-informed neural networks: One learning to rule them all?" *Results Eng.*, vol. 18, Jun. 2023, Art. no. 101023.
[16] O. Fuks and H. A. Tchelepi, "Limitations of physics informed machine learning for nonlinear two-phase transport in porous media," *J. Mach. Learn. Model. Comput.*, vol. 1, no. 1, pp. 19–37, 2020.
[17] S. Steger, F. M. Rohrhofer, and B. C. Geiger, "How PINNs cheat: Predicting chaotic motion of a double pendulum," in *Proc. Symbiosis Deep Learn. Differ. Equ. II*, 2022, pp. 1–13.
[18] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, no. 6, pp. 422–440, 2021.
[19] A. D. Jagtap and G. E. Karniadakis, "Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations," in *Proc. AAAI Spring Symp.*, 2021, pp. 2002–2041.
[20] C. Wu, M. Zhu, Q. Tan, Y. Kartha, and L. Lu, "A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks," *Comput. Methods Appl. Mech. Eng.*, vol. 403, Jan. 2023, Art. no. 115671.
[21] L. D. McClenny and U. M. Braga-Neto, "Self-adaptive physics-informed neural networks," *J. Comput. Phys.*, vol. 474, Feb. 2023, Art. no. 111722.
[22] F. M. Rohrhofer, S. Posch, C. Gößnitzer, and B. C. Geiger, "On the role of fixed points of dynamical systems in training physics-informed neural networks," *Trans. Mach. Learn. Res.*, vol. 10, no. 2, pp. 1–10, 2023.
[23] W. Peng, W. Zhou, J. Zhang, and W. Yao, "Accelerating physics-informed neural network training with prior dictionaries," 2020, *arXiv:2004.08151*.
[24] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, and S. G. Johnson, "Physics-informed neural networks with hard constraints for inverse design," *SIAM J. Sci. Comput.*, vol. 43, no. 6, pp. B1105–B1132, Jan. 2021.
[25] J. Cheng Wong, C. Ooi, A. Gupta, and Y.-S. Ong, "Learning in sinusoidal spaces with physics-informed neural networks," *IEEE Trans. Artif. Intell.*, early access, Jul. 19, 2022, doi: 10.1109/TAI.2022.3192362.
[26] S. Wang, S. Sankaran, and P. Perdikaris, "Respecting causality is all you need for training physics-informed neural networks," 2022, *arXiv:2203.07404*.
[27] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM J. Sci. Comput.*, vol. 43, no. 5, pp. A3055–A3081, Jan. 2021.
[28] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations," *J. Comput. Phys.*, vol. 426, Feb. 2021, Art. no. 109951.
[29] S. Maddu, D. Sturm, C. L. Müller, and I. F. Sbalzarini, "Inverse Dirichlet weighting enables reliable training of physics informed neural networks," *Mach. Learn., Sci. Technol.*, vol. 3, no. 1, Mar. 2022, Art. no. 015026.
[30] Z. Xiang, W. Peng, X. Liu, and W. Yao, "Self-adaptive loss balanced physics-informed neural networks," *Neurocomputing*, vol. 496, pp. 11–34, Jul. 2022.
[31] H. Son, S. W. Cho, and H. J. Hwang, "Enhanced physics-informed neural networks with augmented Lagrangian relaxation method (AL-PINNs)," 2022, *arXiv:2205.01059*.
[32] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *J. Marchine Learn. Res.*, vol. 18, pp. 1–43, Jan. 2018.
[33] C. L. Hwang and A. S. M Masud, *Multiple Objective Decision Making—Methods and Applications: A State-of-the-Art Survey*. Cham, Switzerland: Springer, 2012.
[34] L. I. G. Kovasznay, "Laminar flow behind a two-dimensional grid," *Math. Proc. Cambridge Phil. Soc.*, vol. 44, no. 1, pp. 58–62, Jan. 1948.
[35] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

FRANZ M. ROHRHOFER received the B.Sc. degree in physics from the University of Graz, Austria, in 2016, and the Dipl.-Ing. degree (Hons.) in technical physics from the Graz University of Technology, Austria, in 2019, where he is currently pursuing the Ph.D. degree in computer science. From 2019 to 2020, he was a Project Assistant with the Institute of Theoretical and Computational Physics, Graz University of Technology. In this position, his work included the development of machine learning surrogates for crystal structure prediction, with a special focus on hybrid and theory-assisted feature engineering. In 2020, he joined the Know-Center GmbH, Graz, Austria, as a Research Scientist. His current research interests include the development of physics-informed deep-learning approaches for computational fluid mechanics and reactive flow simulations.

STEFAN POSCH received the B.Sc., M.Sc., and Ph.D. degrees in mechanical engineering from the Graz University of Technology, Austria, in 2011, 2013, and 2017, respectively. He was a Senior Engineer with Midea Austria GmbH, where he was responsible for simulation tasks in the field of hermetic compressors. Since 2019, he has been with the Large Engines Competence Center GmbH, Graz, Austria, as a Senior Scientist and a Team Leader in system simulation and AI integration. His current research interests include the combination of numerical simulation and data-driven approaches.

CLEMENS GÖßNITZER received the bachelor's, master's, and Ph.D. degrees in chemical and process engineering from the Vienna University of Technology, Austria, in 2014, 2016, and 2019, respectively. After the Ph.D. degree, he joined the Large Engines Competence Center GmbH, Graz, Austria, as a Senior Engineer, where he worked on ignition modeling and combustion simulation in the context of internal combustion engines. Since 2021, he has led the CFD Team and is responsible for the physical modeling and simulation of complex combustion phenomena. His current research interests include the simulation of reactive flows with conventional and emerging fuels, physical modeling, and the integration of machine learning approaches in physical simulations.

BERNHARD C. GEIGER (Senior Member, IEEE) received the Dipl.-Ing. degree (Hons.) in electrical engineering and the Dr.Techn. degree (Hons.) in electrical and information engineering from the Graz University of Technology, Austria, in 2009 and 2014, respectively. In 2009, he joined the Signal Processing and Speech Communication Laboratory, Graz University of Technology, as a Project Assistant, where he became a Research and Teaching Associate, in 2010. He was a Senior Scientist and an Erwin Schrödinger Fellow with the Institute for Communications Engineering, Technical University of Munich, Germany, from 2014 to 2017, and a Postdoctoral Researcher with the Signal Processing and Speech Communication Laboratory, Graz University of Technology, from 2017 to 2018. He is currently a Senior Researcher with the Know-Center GmbH, Graz, Austria, where he also leads the Methods & Algorithms for AI Area. His current research interests include information theory for machine learning, theory-assisted machine learning, and information-theoretic model reduction for Markov chains and hidden Markov models.