Surrogate-data-enriched Physics-Aware Neural Networks

Raphael Leiteritz[1], Patrick Buchfink[2], Bernard Haasdonk[2], and Dirk Pflüger[1]

[1]Institute for Parallel and Distributed Systems, University of Stuttgart, Germany
[2]Institute of Applied Analysis and Numerical Simulation, University of Stuttgart, Germany

Abstract

Neural networks can be used as surrogates for partial differential equation (PDE) models. They can be made physics-aware by penalizing underlying equations or the conservation of physical properties in the loss function during training. Current approaches allow to additionally respect data from numerical simulations or experiments in the training process. However, this data is frequently expensive to obtain and thus only scarcely available for complex models. In this work, we investigate how physics-aware models can be enriched with computationally cheaper, but inexact, data from other surrogate models like Reduced-Order Models (ROMs). In order to avoid trusting too-low-fidelity surrogate solutions, we develop an approach that is sensitive to the error in inexact data. As a proof of concept, we consider the one-dimensional wave equation and show that the validation error is decreased by two orders of magnitude when inexact data from ROMs is incorporated.

1 Introduction

Design, optimization or control of complex phenomena are tasks that are critical for applications such as CO_2 storage, e.g. [6], or biomechanical simulations, e.g. [15]. For computationally expensive high-fidelity simulations, such tasks are prohibitive. With computationally efficient surrogate models these tasks can be carried out in an approximative fashion.

Funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 – 390740016. We acknowledge the support by the Stuttgart Center for Simulation Science (SimTech).

Machine Learning techniques can be used to derive such surrogates. Neural networks are one class of such data-driven methods that have successfully been applied to the solution of partial differential equations (PDEs) in various settings [7, 11, 12]. In a recent work [8], it has even been demonstrated that a data-driven method can outperform a numerical method in both, accuracy and runtime, to solve an inverse uncertainty quantification problem.

Already a few decades ago, methods have been proposed that use neural networks to solve PDEs by constraining the loss function with the underlying equations [7]. In recent years, this original idea has seen a renaissance in the form of so-called Physics-Informed Neural Networks (PINNs) [12]. These have meanwhile been successfully applied to a variety of problems such as reconstructing pressure and velocities from visual flow data, simulating blood-flow in cardiovascular structures [13] or subsurface flow [14]. In contrast to methods learning directly on simulation data [4] using e.g. CNNs [10] or LSTMs [9], PINNs add a term to the loss function which penalizes predictions that do not satisfy the underlying PDE. In the scope of this paper, we differentiate these loss terms by their nature. The *physical (or boundary value problem) loss* aims to minimize (a) the PDE residual on interior data points and (b) the (initial) boundary data on boundary data points. PINNs featuring an additional *data loss* on interior data points are referred to as *data-enriched* PINNs.

So far data-enriched PINNs in literature are based on expensive measurements which are either obtained from real or numerical experiments. An example is [12] where PINNs are trained on experimental data and afterwards are used to estimate model parameters of the PDE to solve this inverse

https://doi.org/10.7557/18.6268

problem.

In this work, we aim to integrate comparably cheap data from surrogate models in the data loss of the PINN. In a slightly more general view, we call this data *inexact data* as solutions from surrogates are approximative solutions. We additionally assume that this inexact data is provided with an error bound that quantifies the error with respect to the exact solution. This setting is quite natural for many surrogate models like e.g. Reduced-Order Models (ROMs). As our main contribution, we propose the notion of *error-sensitive* PINNs (see Fig. 1) as a generalization of data-enriched PINNs. The core idea is that the error bound from the inexact data is taken into account during the training by relaxing the optimization goal if the error with respect to the inexact data is smaller than the error bound. This approach comes with two key advantages: Firstly, the additional knowledge on the solution within the solution domain may provide a boost to training times as well as prediction accuracy as it now offers the optimizer more data to find a correct solution. This is crucial since, in a simulation setting, data is usually scarce due to their high computational costs. Secondly, since surrogates can be of low fidelity, the error-sensitive part does not force the PINN to fit the inexact data but instead gives it leeway to improve over the inexact data. Thereby, PINNs are encouraged to refine the given inexact data.

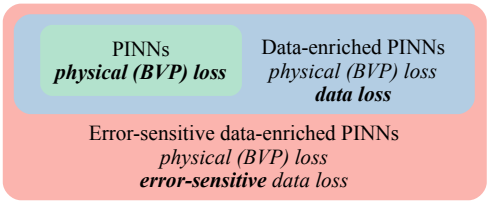

Figure 1: Illustration of different PINN approaches within this paper.

Previous studies for PINNs have shown [16, 17] that the prediction accuracy is sensitive to the weighting of the different loss terms. In order to apply [16] to data-enriched PINNs, we generalize this idea to the case of more than two loss terms. Moreover, we include a comparison of the different weightings in our numerical experiments.

The performance of PINNs on inexact data and the new error-sensitive approach are studied in a numerical experiment based on the one-dimensional linear wave equation. We show that the error-sensitive PINN outperforms a standard non-data-enriched PINN. Moreover, the experiments show that weighting the losses correctly is essential.

The rest of the paper is structured as follows: In Section 2, we introduce the essentials of scientific machine learning for PDEs using PINNs. Subsequently, we present the loss weighting strategies, the error-sensitive PINNs and ROM-data-enriched PINNs in Section 3. The new methods are compared to classical approaches in Section 4 in a numerical experiment for the one-dimensional linear wave equation. Section 5 concludes the paper and provides an outlook to future work.

2 Prerequisites

For the neural network architecture, we restrict ourselves to conventional fully-connected neural networks. We introduce these as a concatenation of n_l different layers f_i

$$\Phi(x;\theta) := (f_{n_l} \circ \ldots \circ f_2 \circ f_1)(x) \tag{1}$$

where $\theta \in \mathbb{R}^{n_\theta}$ represents the vector of all trainable parameters, such as the weights and biases.

The neural networks are tailored towards a specific goal by minimizing a cost functional, the so-called loss (functional), $l(\theta) : \mathbb{R}^{n_\theta} \to \mathbb{R}_{\geq 0}$ over the set of all possible network parameters θ. Typically, some form of stochastic gradient descent method such as ADAM [5] is used for the optimization of $l(\theta)$. The classical example for a loss to learn an input–output mapping from n_D input–output pairs $\{(x_{\mathrm{D},i}, y_{\mathrm{D},i})\}_{i=1}^{n_\mathrm{D}}$ is

$$l(\theta) = l_\mathrm{D}(\theta) := \frac{1}{n_\mathrm{D}} \sum_{i=1}^{n_\mathrm{D}} \left(y_{\mathrm{D},i} - \Phi(x_{\mathrm{D},i};\theta)\right)^2, \tag{2}$$

which is known as Mean Squared Error (MSE) loss. In the following, we call this loss term the *data loss*.

The Physics-Informed Neural Networks (PINNs) modify the loss function to inform the network about the underlying physics [12]. In the scope of this paper, the desired physical property is a boundary value problem (BVP) of the following type: find an unknown function $u : \Omega \to \mathbb{R}$ with

$\Omega \subset \mathbb{R}^d$ such that

$$\mathcal{N}[u] = 0 \quad \text{in } \Omega, \qquad \mathcal{B}[u] = 0 \quad \text{on } \partial\Omega, \tag{3}$$

where $\mathcal{N}$ is some, potentially non-linear, differential operator and $\mathcal{B}$ is an operator prescribing the boundary conditions. In the scope of this paper, time-dependent problems are of particular interest. In that case, $x = (t, \xi) \in \Omega := \Omega_t \times \Omega_\xi$ is composed of the time t and a spatial coordinate ξ. Compared to numerical simulations, which aim to ensure that the laws of physics are not violated, a PINN does not strictly guarantee a physically valid solution. Instead, it encourages the solution $\Phi(\theta) \approx u$ to satisfy the BVP in selected collocation points $x_{\mathrm{I},i}$, $x_{\mathrm{B},i}$ with

$$\begin{aligned} l_\mathrm{I}(\theta) &:= \frac{1}{n_\mathrm{I}} \sum_{i=1}^{n_\mathrm{I}} \left(\mathcal{N}[\Phi(\theta)](x_{\mathrm{I},i})\right)^2 \\ l_\mathrm{B}(\theta) &:= \frac{1}{n_\mathrm{B}} \sum_{i=1}^{n_\mathrm{B}} \left(\mathcal{B}[\Phi(\theta)](x_{\mathrm{B},i})\right)^2 \end{aligned} \tag{4}$$

and adds these terms as penalty terms to the loss function where the derivatives in $\mathcal{N}$ are evaluated via Automatic Differentiation [1]. The PINN loss then reads

$$l(\theta) = \sum_{j \in \mathcal{J}} \lambda_j l_j(\theta), \qquad \mathcal{J} := \{\mathrm{D}, \mathrm{I}, \mathrm{B}\}, \tag{5}$$

with weights $\lambda_j \in \mathbb{R}_{\geq 0}$ and the loss contributions $l_j(\theta) \geq 0$ from (2) and (4). We call $l_\mathrm{I}(\theta)$ the interior or PDE residual loss and $l_\mathrm{B}(\theta)$ the boundary loss. Both these terms together are referred to as the BVP losses.

3 Data-enriched PINNs

Theoretically, PINNs work without the data loss ($\lambda_\mathrm{D} = 0$). For *data-enriched* PINNs ($\lambda_\mathrm{D} > 0$), we would like to additionally use $y_{\mathrm{D},i} = u(x_{\mathrm{D},i})$, but the exact solution u is frequently not available or too expensive to compute. In the scope of this paper, we investigate how data-enriched PINNs behave, if the target function in the data loss is provided by inexact data, e.g. with an approximate solution $y_{\mathrm{D},i} = \widetilde{u}(x_{\mathrm{D},i}) \approx u(x_{\mathrm{D},i})$.

3.1 Loss Weighting for PINNs

For the case of non-data-enriched PINNs, it has been observed in previous studies that the choice of weights λ_j in the loss function is crucial for the training speed and quality [16, 17].

The *Learning Rate Annealing for PINNs* (LRA) in [17] is motivated by a stiffness-phenomenon in the gradient flow dynamics. It uses the statistic of the gradient to balance the interplay of all loss contributions l_j.

The *Optimal Loss Weight* (OPT) in [16] is a heuristic approach that tries to balance the losses based on the assumption that the relative error in the derivatives of the neural network can be bounded uniformly. We generalize this idea to more than two losses. In this formulation it chooses the loss based on characteristic quantities M_j of the loss l_j for each $j \in \mathcal{J}$, e.g. $M_\mathrm{D}[u] \approx \|u\|^2_{L_2(\Omega)} / |\Omega|$ for the data loss. The resulting weights are

$$\lambda_j[u] = \left(\sum_{k \in \mathcal{J}} M_j[u] / M_k[u] \right)^{-1}.$$

Note that the factors $M_j[u]$ may depend on the exact solution which is not available. In our numerical experiment, we compute the weights from the exact solution for the sake of simplicity. For practical applications however, one could use the inexact data $\widetilde{u}$ to compute the factors.

3.2 Error-Sensitive PINNs

The following section focuses on time-dependent problems with $x = (t, \xi)$ in the sense of Section 2. For all functions $w(x)$, we define the short-hand notation $w(t) := w(t, \cdot)$ for each $t \in \Omega_t$. The analysis is formulated in terms of a non-discrete analogue to the data loss from (2),

$$\mathcal{L}_\mathrm{D}(\theta) = \|u - \Phi(\theta)\|^2_{L_2(\Omega)},$$

where Monte-Carlo integration is used to approximate $\mathcal{L}_\mathrm{D}(\theta) / |\Omega| \approx l_\mathrm{D}(\theta)$ with n_D sampling points $x_{\mathrm{D},i} \in \Omega$ and $y_{\mathrm{D},i} = u(x_{\mathrm{D},i})$ for $1 \leq i \leq n_\mathrm{D}$. Moreover, the data loss $\mathcal{L}_\mathrm{D}(\theta)$ is sampled separately in time and space. For the sake of simplicity, we consider an equidistant sampling in space and time in the following, e.g. for $\xi \in \Omega_\xi \subset \mathbb{R}$ and Δ_t, $\Delta_\xi > 0$,

$$t_i = t_0 + i\Delta_t, \qquad \xi_j = \xi_0 + j\Delta_\xi,$$

with $1 \leq j \leq n_\xi$, $1 \leq i \leq n_t$. If we would know the exact solution u, the equidistantly sampled data

loss would read

$$l_{\mathrm{D}}(\theta) := \frac{1}{n_t}\sum_{i=1}^{n_t}\left(i_\xi[u](t_i;\theta)\right)^2,$$
$$\left(i_\xi[u](t_i;\theta)\right)^2 := \frac{1}{n_\xi}\sum_{j=1}^{n_\xi}\left|(\Phi(\cdot;\theta)-u)(t_i,\xi_j)\right|^2. \tag{6}$$

As u is not available, we use the inexact data $\widetilde{u}$ instead, for which we assume that the error can be quantified for each $t \in \Omega_t$ with

$$\|u(t)-\widetilde{u}(t)\|_{L_2(\Omega_\xi)} \leq \varepsilon(t) \tag{7}$$

with an error bound $\varepsilon : \Omega_t \to \mathbb{R}_+$. The idea of the *error-sensitive* data-enrichment is to trust the inexact data only up to the error bound $\varepsilon(t)$. To this end, consider the open $\varepsilon(t)$-ball around $\widetilde{u}(t)$

$$B_\varepsilon(\widetilde{u};t) := \left\{w \in L_2(\Omega_\xi)\middle|\, \|w-\widetilde{u}(t)\|_{L_2(\Omega_\xi)} < \varepsilon(t)\right\}$$

which can also be interpreted as a tube around $\widetilde{u}$ over time t. In order to differentiate whether $\Phi(t)$ lies in $B_\varepsilon(\widetilde{u};t)$, we define $w : \Omega_t \to \overline{B}_\varepsilon(\widetilde{u};t)$, the projection of $\Phi(t)$ onto the closed $\varepsilon(t)$-ball around $\widetilde{u}(t)$, see Fig. 2.

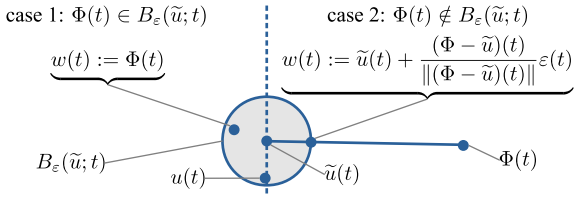

Figure 2: Illustration of the definition of $w(t)$.

The data loss $\mathcal{L}_{\mathrm{D}}(\theta)$ can then be bounded with

$$\mathcal{L}_{\mathrm{D}}(\theta) \leq 2\left(\|u-w(t)\|^2_{L_2(\Omega)} + \|w(t)-\Phi\|^2_{L_2(\Omega)}\right).$$

The first term on the right side, can be bounded with $4\varepsilon(t)^2\,|\Omega_t|$ since both, $u(t)$ and $w(t)$, are elements in the ball $B_\varepsilon(\widetilde{u};t)$ from which follows that their distance is bounded by the diameter of the ball, 2ε. This yields

$$\mathcal{L}_{\mathrm{D}}(\theta)/\,|\Omega| \leq 2\mathcal{L}_{\mathrm{D,ES}}(\theta)/\,|\Omega| + 8\varepsilon(t)^2/\,|\Omega_\xi|$$
$$\mathcal{L}_{\mathrm{D,ES}}(\theta) := \int_{\Omega_t} \underbrace{\|w(t)-\Phi(t;\theta)\|^2_{L_2(\Omega_\xi)}}_{=:(I_{\xi,\mathrm{ES}}(t;\theta))^2}\,\mathrm{d}t \tag{8}$$

which describes how the error from (7) propagates through the training. Moreover, this estimate guarantees that training with the error-sensitive loss $\mathcal{L}_{\mathrm{D,ES}}(\theta)$ improves the networks quality with respect to the original data loss $\mathcal{L}_{\mathrm{D}}(\theta)$ (as long as $\varepsilon(t)$ is small enough).

Due to the choice of $w(t)$, the term $I_{\xi,\mathrm{ES}}(t;\theta)$ in (8) is (a) zero if the error-sensitive PINN solution $\Phi(t;\theta)$ is in $B_\varepsilon(\widetilde{u};t)$ and (b) equal to the distance between $\Phi(t;\theta)$ and the closest point on the boundary $\partial B_\varepsilon(\widetilde{u};t)$ otherwise. This can be expressed with

$$I_{\xi,\mathrm{ES}}(t;\theta) = \mathtt{ReLu}\left(\|\Phi(t;\theta)-\widetilde{u}(t)\|_{L_2(\Omega_\xi)} - \varepsilon(t)\right),$$
$$\mathtt{ReLu}(x) := \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}.$$

This results in the the error-sensitive loss

$$\mathcal{L}_{\mathrm{D,ES}}(\theta)/\,|\Omega| \approx l_{\mathrm{D,ES}}(\theta) := \frac{1}{n_t}\sum_{i=1}^{n_t}\left(i_{\xi,\mathrm{ES}}(t_i;\theta)\right)^2,$$
$$i_{\xi,\mathrm{ES}}(t;\theta) := \mathtt{ReLu}\left(i_\xi[\widetilde{u}](t;\theta) - \varepsilon(t)/\,|\Omega_\xi|^{1/2}\right).$$

Note that the presented loss is readily implementable in the major ML frameworks due to usage of the well-established `ReLu` function. Moreover, the non-error-sensitive approach is a special case of error-sensitive data-enrichment if the data is fully trusted, i.e. $\varepsilon \equiv 0$.

3.3 ROM-data-enriched PINNs

A prominent example for inexact data certified with an error bound is data obtained from so-called Reduced Order Models (ROMs). ROMs are constructed to flexibly trade accuracy for efficiency by choosing different sizes of the reduced basis. At the same time many ROMs provide an error bound that can be evaluated efficiently. An example for ROMs with a time-dependent error bound of the assumed form (7) is derived in [2] for the linear wave equation. Technically, we additionally assume that the underlying finite element method (FEM) approximation space is rich enough to approximate u and thus, that the error between the FEM solution and the exact solution is negligible.

4 Experiments

We consider the wave equation in a one-dimensional spatial domain $\Omega_\xi := (-1, 1)$ over the time interval $\Omega_t := (0, 2)$, for which an analytical solution is available for validation. The PDE on the spatio-temporal domain $\Omega := \Omega_t \times \Omega_\xi$ is

$$\partial_t^2 u(t,\xi) - \partial_\xi^2 u(t,\xi) = 0 \qquad (t,\xi) \in \Omega$$

with homogeneous Dirichlet boundary conditions and zero initial velocity $v_0 \equiv 0$. The initial data and the solution are visualized in Fig. 3.

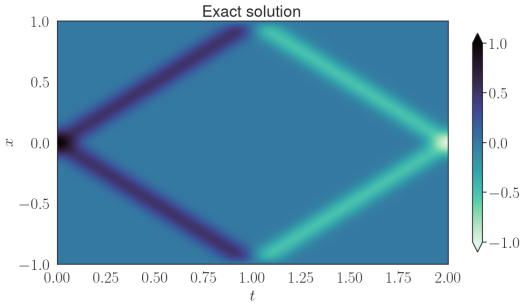

Figure 3: Visualization of the exact solution. A single bump in the middle of the domain at $t = 0$ travels outwards and is reflected at the boundaries at $t = 1$.

The different approaches are compared with respect to the mean squared error

$$\mathrm{MSE}[y] := \|u - y\|^2_{L_2(\Omega)} / |\Omega| \,. \tag{9}$$

Additionally, experiments are repeated r times and averaged to account for random initial weight configurations which we denote with $\overline{\mathrm{MSE}}_r[\Phi]$.

The network used is a conventional fully connected feed-forward network with *tanh* activation functions. Its architecture and hyperparameters were chosen with a hyperparameter optimization for the non-data-enriched PINN over 407 individual runs resulting in $n_l = 5$ layers, $n_n = 20$ neurons per layer, and a learning rate of $\alpha = 1\mathrm{e}-3$. The optimizer is ADAM with default parameters. The network parameters are initialized using the truncated Xavier initialization [3]. The number of sampling points of the different losses varies for each experiment and is depicted in Table 1 as an overview.

Table 1: Number of sampling points.

experiment	$\mathbf{n}_\mathrm{I}$	$\mathbf{n}_\mathrm{B}$	$\mathbf{n}_\mathrm{D}$
non-data-enriched	30,000	3,000	0
data-enriched	15,000	3,000	15,000

4.1 Baseline: Non-Data-Enriched

Before enriching the loss of the network with a data loss, we first establish a non-data-enriched PINN baseline, i.e. this experiment only features the interior loss $l_\mathrm{I}(\theta)$ and the boundary loss $l_\mathrm{B}(\theta)$.

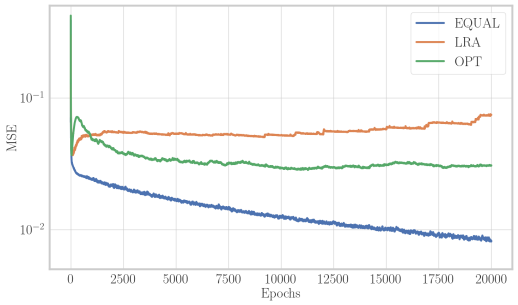

Figure 4: Validation error $\overline{\mathrm{MSE}}_{30}[\Phi]$ plotted over 20,000 training epochs of the non-data-enriched PINNs with EQUAL, LRA and OPT weighting.

For the weighting of the loss contributions (see Section 3.1), we consider equal weighting (EQUAL), i.e. $\lambda_j = 1$ for all $j \in \mathcal{J}$, in addition to the LRA and OPT approach.

Fig. 4 shows the training progress of all three weighting methods in terms of the validation error $\overline{\mathrm{MSE}}_{30}[\Phi]$ over the number of epochs. Neither of the three approaches was able to reliably capture the true solution, with minimum validation errors going only as low as $1\mathrm{e}-2$. This is in accordance with the observations in [18] that the one-dimensional wave equation is a very challenging problem for (non data-enriched) PINNs. Note that the equal weighting approach performs best while the other, more informed, weightings result in bad outliers shifting the mean curve $\overline{\mathrm{MSE}}_{30}[\Phi]$ upwards. This certainly poses a strong case for including additional data during training as presented in the following.

4.2 Data-Enriched: Exact Data

As a second baseline, we investigate how well data-enriched PINNs can train if the explicit solution is used in the data loss l_{D} from (6). Note that this is a clearly unrealistic scenario but a good "ideal" method indicating the performance limits that we may expect with the ROM-data-enriched approaches.

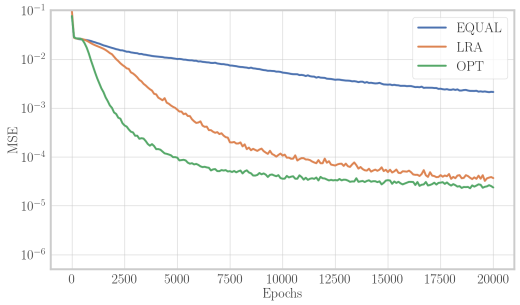

Figure 5: Validation error $\overline{\mathrm{MSE}}_{30}[\Phi]$ of the data-enriched PINNs (enriched with data of the explicit solution) plotted over 20,000 training epochs.

Fig. 5 shows the $\overline{\mathrm{MSE}}_{30}[\Phi]$ for this data-enriched training for the three different weighting approaches. The data-enrichment clearly improves the training performance as the validation error drops below $1\mathrm{e}-4$. Additionally, it can clearly be seen that the OPT and LRA methods show a much quicker convergence behavior than EQUAL, with the OPT performing best overall. Thus, we restrict the experiments in the following section to the OPT weighting.

4.3 ROM-data-enriched PINNs

Next, we replace the explicit solution in the data loss with a surrogate solution $\widetilde{u}$, i.e. $y_{\mathrm{D},i} = \widetilde{u}(x_{\mathrm{D},i})$ in (2), where a ROM is used to compute the surrogate solution (see Section 3.3). To this end, the PDE is discretized with the Finite Element Method (Lagrangian elements on an equidistant grid, piecewise constant in time and piecewise linear in space, 3000 discretization points in each dimension) which we refer to as Full-Order Model (FOM). The error of the FOM is $\mathrm{MSE}[u_{\mathrm{FOM}}] = 1.46\mathrm{e}-6$. Based thereon, model order reduction is applied to derive three different ROMs of reduced sizes $n \in \{4, 8, 12\}$ which varies the accuracy of the different ROMs. The maximal dimension is set to 12, which results in a reduction error of $\mathrm{MSE}[\widetilde{u}] = 3.85\mathrm{e}-6$. Note that by construction of this experiment the ROMs are based on much more accurate data (FOM snapshots) than the data-enriched PINNs (only ROM snapshots).

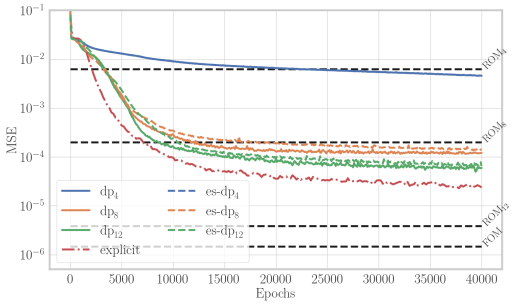

Figure 6: Validation error $\overline{\mathrm{MSE}}_{30}[\Phi]$ of the (error-sensitive) ROM-data-enriched PINNs with the weighting methods OPT for three different ROM sizes $\{4, 8, 12\}$. Horizontal bars depict the respective error of the FOM and ROM solution.

Fig. 6 shows the results for all six of these configurations, namely the ROM-data-enriched PINNs (dp_n, solid lines) and the error-sensitive variant ($\mathrm{es\text{-}dp}_n$, dashed lines) for $n \in \{4, 8, 12\}$. For comparison purposes, the data-enriched PINN run using the explicit solution data from Section 4.2 is depicted (red line) and the MSE of the FOMs and ROMs are included as horizontal lines. The graph shows that the ROM-data-enriched PINNs perform much better in terms of the overall predictive power when the ROM data is good enough ($n \geq 8$) achieving validation errors close to the model trained on explicit solution data. Even more noteworthy, the ROM-data-enriched PINNs are able to improve over the error in the ROM for $n \leq 8$. This can be seen as the respective MSE curves fall below the ROM_4 and ROM_8 markers. The error-sensitive data-enrichment, however, does not improve the accuracy in this example. This is assumed to be accounted to the fact that the non-data-enriched PINN model itself is not able to achieve a reasonable prediction as described in Section 4.1. The MSE (9) and corresponding standard deviation for the best validation error over all epochs are summarized in Table 2.

Table 2: MSE from (9) and the corresponding standard deviation (mse ± std) of the lowest validation error over all training epochs for the ROM-data-enriched (**dp**) and error-sensitive (**es-dp**) variants for different ROM sizes ($\boldsymbol{n}$). For reference this measure for training on the explicit solution data is 2.38e-05±1.59e-05.

$\boldsymbol{n}$	**dp**	**es-dp**
4	5.18e-03±8.22e-05	5.21e-03±1.39e-03
8	1.21e-04±1.01e-05	1.43e-04±1.76e-05
12	5.96e-05±1.12e-05	6.54e-05±1.41e-05

5 Conclusion

Our approach proved that it is beneficial to combine physics-aware neural networks with inexact data obtained from surrogate models. We have shown that ROM-data-enriched PINNs can outperform both, conventional PINNs and ROMs. The results presented here serve as a proof-of-concept, studying a problem for which the exact solution is known. We expect that error-sensitive PINNs will show their real benefit in higher-dimensional, parametrized simulation settings, which is subject of future work. In a parametric setting, it will be interesting to see how well the error-sensitive PINNs generalize to unseen parameters, avoiding prohibitively expensive simulation runs.

References

[1] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.*, 18(1):5595–5637, jan 2017. ISSN 1532-4435.

[2] S. Glas, A. T. Patera, and K. Urban. A reduced basis method for the wave equation. *International Journal of Computational Fluid Dynamics*, 34(2):139–146, 2020. doi: 10.1080/10618562.2019.1686486. URL `https://doi.org/10.1080/10618562.2019.1686486`.

[3] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010. URL `http://proceedings.mlr.press/v9/glorot10a.html`.

[4] Y. Khoo, J. Lu, and L. Ying. Solving parametric PDE problems with artificial neural networks. *European Journal of Applied Mathematics*, page 1–15, 2020. doi: 10.1017/S0956792520000182.

[5] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

[6] M. Köppel, F. Franzelin, I. Kröker, S. Oladyshkin, G. Santin, D. Wittwar, A. Barth, B. Haasdonk, W. Nowak, D. Pflüger, and C. Rohde. Comparison of data-driven uncertainty quantification methods for a carbon dioxide storage benchmark scenario. *Computational Geosciences*, 23(2):339–354, Apr. 2019. ISSN 1573-1499. doi: 10.1007/s10596-018-9785-x. URL `https://doi.org/10.1007/s10596-018-9785-x`.

[7] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998. doi: 10.1109/72.712178.

[8] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=c8P9NQVtmnO`.

[9] A. T. Mohan and D. V. Gaitonde. A deep learning based approach to reduced order modeling for turbulent flow control using LSTM neural networks. *arXiv:1804.09269*, 2018. doi: 10.48550/arXiv.1804.09269.

[10] O. Obiols-Sales, A. Vishnu, N. Malaya, and A. Chandramowliswharan. CFDNet: A deep

learning-based accelerator for fluid simulations. In *Proceedings of the 34th ACM*, ICS '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379830. doi: 10.1145/3392717.3392772. URL https://doi.org/10.1145/3392717.3392772.

[11] D. C. Psichogios and L. H. Ungar. A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, 38(10):1499–1511, 1992. doi: 10.1002/aic.690381003. URL https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003.

[12] M. Raissi, P. Perdikaris, and G. Karniadakis. 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, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.045. URL http://www.sciencedirect.com/science/article/pii/S0021999118307125.

[13] M. Raissi, A. Yazdani, and G. E. Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020. ISSN 0036-8075. doi: 10.1126/science.aaw4741. URL https://science.sciencemag.org/content/367/6481/1026.

[14] A. M. Tartakovsky, C. O. Marrero, P. Perdikaris, G. D. Tartakovsky, and D. Barajas-Solano. Physics-informed deep neural networks for learning parameters and constitutive relationships in subsurface flow problems. *Water Resources Research*, 56(5), 2020. doi: 10.1029/2019WR026731. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026731.

[15] J. Valentin, M. Sprenger, D. Pflüger, and O. Röhrle. Gradient-based optimization with B-splines on sparse grids for solving forward-dynamics simulations of three-dimensional, continuum-mechanical musculoskeletal system models. *International Journal for Numerical Methods in Biomedical Engineering*, 34(5):e2965, 2018. doi: 10.1002/cnm.2965. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.2965.

[16] R. van der Meer, C. W. Oosterlee, and A. Borovykh. Optimally weighted loss functions for solving pdes with neural networks. *Journal of Computational and Applied Mathematics*, 405:113887, 2022. ISSN 0377-0427. doi: 10.1016/j.cam.2021.113887. URL https://www.sciencedirect.com/science/article/pii/S0377042721005100.

[17] S. Wang, Y. Teng, and P. Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021. doi: 10.1137/20M1318043. URL https://doi.org/10.1137/20M1318043.

[18] S. Wang, X. Yu, and P. Perdikaris. When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, page 110768, 2021. ISSN 0021-9991. doi: 10.1016/j.jcp.2021.110768. URL https://www.sciencedirect.com/science/article/pii/S002199912100663X.