Design of continuous structures using physics informed neural networks

Adrien Gallet, Ph.D[1] and Danny Smyl, Ph.D, P.E., M.ASCE[2]

[1]Structural Engineer, Unipart Construction Technologies, Advanced Manufacturing Park, Brunel Way, Sheffield, S60 5WG, UK

[2]Assistant Professor, School of Civil and Environmental Engineering, Georgia Institute of Technology, 790 Atlantic Drive, Atlanta, GA 30332-0355. Corresponding author email address: *danny.smyl@ce.gatech.edu*

ABSTRACT

Since their inception, physics informed neural networks (PINNs) have shown advantages over traditional data-driven only models by reducing the need for large training dataset sizes via the embedment of physical information in the training process targeting predictions consistent with the laws of physics. This work investigates the viability and advantages of PINN models used for designing continuous structures. To achieve this, local stiffness relations based on Timoshenko–Ehrenfest beam theory are used to constrain network output spaces for predicting variable member sizes in continuous beams. The results herein indicate that PINNs have improved performance relative to data-driven only design models, but the present correlation between data-losses and physics-losses is tenuous, limiting the verifiability of PINN models during inference. Future avenues of research to develop a extensively verifiable architectures are suggested.

INTRODUCTION

The use of physics informed neural networks (PINNs) in civil and structural engineering is beginning to gain attention since their introduction between 2017 to 2019 (Raissi et al. 2019), (Karpatne et al. 2017) and (Muralidhar et al. 2018). This observation also holds true for the numerous recent promising applications of data-driven machine learning (ML) approaches in civil and structural applications. For example, recently in (Kourehpaz and Molina Hutt 2022) a data-driven approach was successfully adopted for seismic risk assessments, in (Kabir and Garg 2023) the authors used convolutional networks to characterize contact angles and in (Pak et al. 2023) where Pak and coauthors employed knowledge transfer to predict shear capacity of deep beams. In contrast, a PINN differentiates itself from data-driven only artificial neural networks by supplementing a standard loss function, such as Mean Average Error (MAE) or Mean Average Percentage Error (MAPE) with a physical equation that relates the features and targets of the network based on physics that are known to apply to the problem at hand (Karniadakis et al. 2021).

One of the earliest applications of PINNs in civil and structural engineering literature was published in 2022 for a geometry identification problem in the field of continuum mechanics, a specific type of inverse problem (Zhang et al. 2022). In this study, the authors trained a neural network to minimise the difference between the predicted to measured boundary displacement and stress fields based on coordinate information, and thereby update the estimated material and geometric parameters of the square-shaped matrix material based residual of equilibrium equations. Meanwhile, authors in (Hu et al. 2022) showed promising results using PINNs for designing steel dampers. A different study in 2023 developed a PINN based topology optimisation (PINNTO) framework to numerically determine the displacement fields, and thereby entirely replace the Finite Element Analysis found in conventional structural topology optimisation approaches (Jeong et al. 2023). In this work, coordinate information are also used as inputs, and the optimised solution corresponds to the displacement field that minimises the total potential energy. A further work published in 2023 incorporated PINNs within a graph neural network structure to minimise the error in equilibrium equations to structurally analyse the response of frames subject to nodal forces (Song et al. 2023). Lastly, in early 2024, a unified solver-free structural optimiser was developed, in which a physical loss function based on complementary energy equations was minimised to determine the optimal truss structures (Mai et al. 2024). Their results compared favourably with low errors when compared to traditional FEA based optimisation approaches.

These examples demonstrate that PINNs have been successfully applied for various applications. From the perspective of structural analysis and design, and the various machine learned operators discussed in (Gallet et al. 2024b), two of the aforementioned pieces of work focused on building *ML optimisation solvers* using PINNs, in which the goal is to replace traditional finite elements solvers to arrive at optimised designs (Jeong et al. 2023) and (Mai et al. 2024), one piece of work used PINNs to build a *ML forward operator* to help analyse structures (Song et al. 2023), and only one investigated the application of PINNs for a *ML inverse operator*, albeit not for the purpose of structural design (Zhang et al. 2022).

The investigation presented here focuses on how PINNs could potentially be deployed to improve the performance of machine learned inverse operators, specifically focusing on comparing the performance of a physics-informed neural network to the data-only networks developed in (Gallet et al. 2024b). Aims herein are based on the current limitations of the inverse operator achieved in the previous investigation, namely: attempt to reduce the error and accuracy dispersion profiles by incorporating known physical information into the operator. This is substantially different to previous research conducted (Jeong et al. 2023) and (Mai et al. 2024), which created data-less operators only based on well defined physics equations, but closer to what was achieved in previous PINN based ML inverse operator research (Zhang et al. 2022). In that regard, more evidence for identifying structural design as an inverse operator is presented (Gallet et al. 2022). Moreover, we intend to theorise and test a potential verifiable architecture of the inverse operator through which the physical validity of the predicted results could be evaluated during inference of the model, namely outside of training and validation. In totality, this work aims to accomplish the following:

1. Propose PINN models for the design of continuous structures;
2. Investigate the efficacy of PINN design models compared to data-driven only models;

3. Test the concept of verifiable PINNs.

The paper is subsequently organized as follows. Section 2 introduces the general methodology propositioning how PINN based inverse operators can be built using local stiffness relations. Section 3 presents the main results. Section 4 discusses the outcomes and, lastly, Section 5 summarises the areas for future works.

METHODOLOGY

Physics informed neural networks

The general objective of supervised neural networks is to solve a generalised optimization problem by minimizing the empirical loss function defined by Equation 1:

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{y}_i, f\left(\boldsymbol{x}_i;\boldsymbol{\theta}\right)\right) + \lambda\Omega\left(\boldsymbol{\theta}\right) \tag{1}$$

where $\boldsymbol{y}_i$ are the target variables from a dataset containing n data points, $f\left(\boldsymbol{x}_i;\boldsymbol{\theta}\right)$ is the predicted target variable based on predictors $\boldsymbol{x}_i$ and $\boldsymbol{\theta}$ are the weights and biases of the model. $\mathcal{L}$ is a chosen loss measured between the true targets $\boldsymbol{y}_i$ and predicted targets $f\left(\boldsymbol{x}_i;\boldsymbol{\theta}\right)$, such as Mean Square Error (MSE) or the Mean Absolute Percentage Error (MAPE).

PINNs differentiate themselves from traditional neural networks via the last term $\lambda\Omega\left(\boldsymbol{\theta}\right)$, which is representative of a regulariser that penalises model parameters $\boldsymbol{\theta}$ that deviate significantly from a pre-set value (typically 0, although other expected values may be used) to ensure a non-complex model where no network nodes dominate the predictions. In PINNs, $\Omega\left(\boldsymbol{\theta}\right)$ of the regularisation term is replaced by physical equations, which create a functional relationship between the input and output variables based on known physical phenomena. Hence, in addition to the data loss term, there is a physics loss term, which evaluates if the inputs and output variables construe to physically meaningful solutions.

Normally, the physics loss equations are based on the well established ordinary or partial differential equations (ODEs/PDEs) such as the one-dimensional heat equation:

$$u_t = ku_{xx} + h(x,t) \tag{2}$$

where the variables x and t would be inputs to the network, u the singular output, and terms u_t and u_{xx} being evaluated through auto-differentiation during learning. By then generating a physics loss function in the form of:

$$\text{MSE}_{\text{Physics}} = \left(ku_{xx} + h(x,t) - u_t\right)^2 \tag{3}$$

one can evaluate the magnitude of prediction error, and back-propagate these errors through the network to achieve learned behaviour. The progressive minimisation of the physics loss function represents a convergence towards physically realistic results, and hence can guide the neural network during training on how to correctly update the parameters in respect to the loss landscape (or loss surface) (McClenny and Braga-Neto 2023).

The difficulty in developing a PINN for a structural design model is that structural design is not solely characterised by an ODE/PDE equation. This investigation will, however, expand upon and relate the input to the output space established in (Gallet et al. 2024b) (where inputs were span and uniformly distributed load (UDL) values, and outputs were the cross-section properties of a continuous beam element), by using non-discretised local stiffness relations.

Local stiffness matrix relations

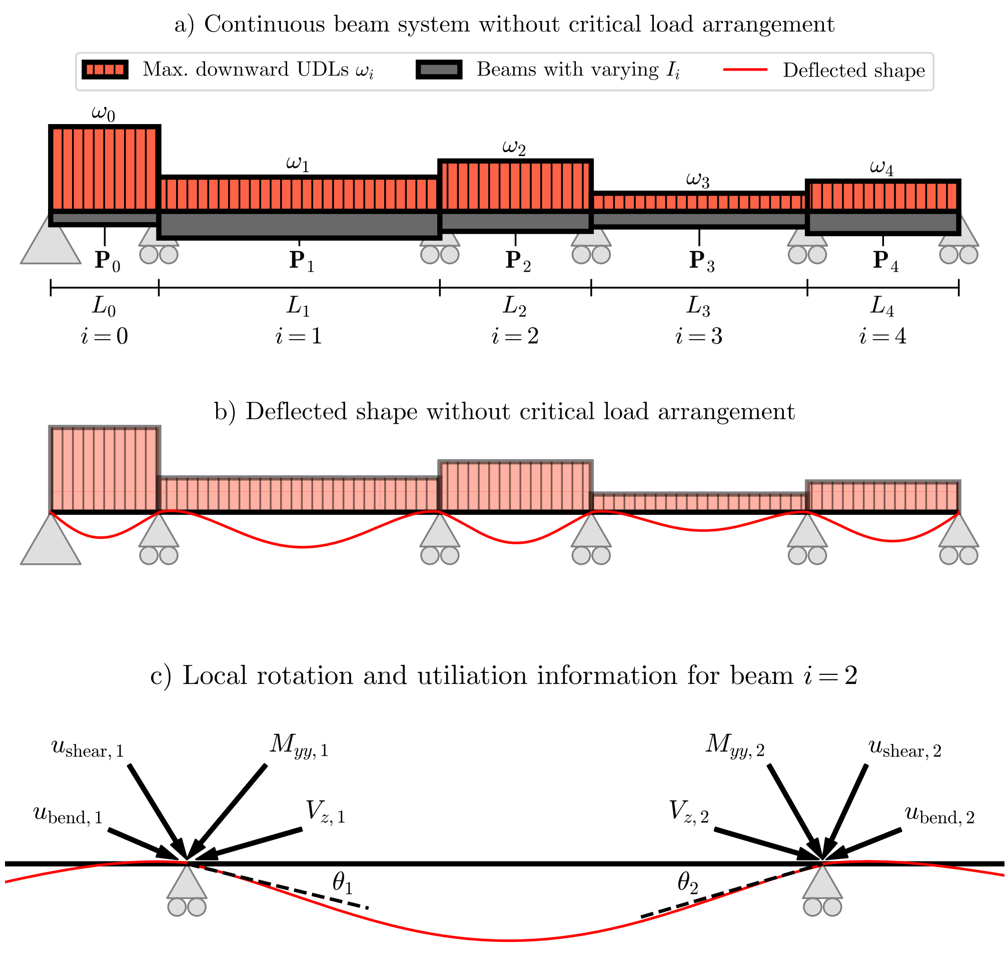

Fig. 1. Continuous beam system subjugated to uniformly distributed loads with b) deflected shape and c) resulting rotations θ_1 and θ_2. Physical response information in terms of bending u_{bend} and shear u_{shear} utilisation ratios and bending moments M_{yy} and shear force V_z can be identified over each support.

The relationship between the deformation and internal forces within a member are expressed here via the Timoshenko-Ehrenfest stiffness matrix equations. For this work, we assume a continuous beam system indexed by i with ω_i UDLs, L_i spans, with each beam having cross-section properties P_i as shown in Figure 1. In this case, the deformation vectors can be related to the internal reaction forces with end-force adjustments (due to the UDLs) using the local stiffness matrix relations as shown in equation 4:

$$\begin{bmatrix} V_{z,1} \\ M_{yy,1} \\ V_{z,2} \\ M_{yy,2} \end{bmatrix} + \begin{bmatrix} \frac{-\omega L}{2} \\ \frac{-\omega L^2}{12} \\ \frac{-\omega L}{2} \\ \frac{\omega L^2}{12} \end{bmatrix} = \frac{EI}{L^3(1+\phi)} \begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & (4+\phi)L^2 & -6L & (2-\phi)L^2 \\ -12 & -6L & 12 & -6L \\ 6L & (2-\phi)L^2 & -6L & (4+\phi)L^2 \end{bmatrix} \begin{bmatrix} \delta_1 \\ \theta_1 \\ \delta_2 \\ \theta_2 \end{bmatrix} \tag{4}$$

$$\text{where } \phi = \tfrac{12EI}{A_z G L^2}$$

with shear forces V_z and bending moments M_{yy}, Youngs Modulus E, shear modulus G, the second moment of area I, shear area A_z and rotations θ and displacements δ at the start and end of each continuous beam member numbered as location 1 and 2 respectively as shown in Figure 1c). Note that these locations (1 and 2) always correspond to areas over supports in this work, and that non-discretised members will be used in this investigation, meaning each local stiffness correlation will correspond to a single prismatic beam. Since there are no deflections occurring over the supports, $\delta_1 = 0$ and $\delta_2 = 0$ and Equation 4 can be simplified to the following:

$$\begin{bmatrix} V_{z,1} \\ M_{yy,1} \\ V_{z,2} \\ M_{yy,2} \end{bmatrix} + \begin{bmatrix} \frac{-\omega L}{2} \\ \frac{-\omega L^2}{12} \\ \frac{-\omega L}{2} \\ \frac{\omega L^2}{12} \end{bmatrix} = \frac{EI}{L^3(1+\phi)} \begin{bmatrix} 6L & 6L \\ (4+\phi)L^2 & (2-\phi)L^2 \\ -6L & -6L \\ (2-\phi)L^2 & (4+\phi)L^2 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}. \tag{5}$$

Equation 5 can therefore be broken down into the following four equations:

$$\begin{aligned} V_{z,1} =& \frac{EI_{yy}}{L^3(1+\phi)}(6L\theta_1 + 6L\theta_2) + \frac{\omega L}{2} \\ M_{yy,1} =& \frac{EI_{yy}}{L^3(1+\phi)}\Big((4+\phi)L^2\theta_1 + (2-\phi)L^2\theta_2\Big) + \frac{\omega L^2}{12} \\ V_{z,2} =& \frac{EI_{yy}}{L^3(1+\phi)}(-6L\theta_1 - 6L\theta_2) + \frac{\omega L}{2} \\ M_{yy,2} =& \frac{EI_{yy}}{L^3(1+\phi)}\Big((2-\phi)L^2\theta_1 + (4+\phi)L^2\theta_2\Big) - \frac{\omega L^2}{12}. \end{aligned} \tag{6}$$

The neural network developed in (Gallet et al. 2024b) related span L_i and UDL ω_i inputs to cross-sectional area output values, not internal forces V_z and M_{yy}. Using Eurocode 3 (European Committee for Standardisation 2015), it is possible to evaluate the shear and bending utilisation factors using the cross-sectional properties of an element,

$$
\begin{aligned}
\frac{|V_z|}{A_z \frac{f_y}{\sqrt{3}}} &= u_{\text{shear}} \\
\frac{\left|M_{yy}\right|}{W_{\text{pl},yy}\, f_y} &= u_{\text{bend}}
\end{aligned} \tag{7}
$$

which can be re-arranged:

$$
\begin{aligned}
\left|V_{z,\text{Ed}}\right| &= u_{\text{shear}}\, A_z \frac{f_y}{\sqrt{3}} \\
\left|M_{yy,\text{Ed}}\right| &= u_{\text{bend}}\, W_{\text{pl},yy} f_y .
\end{aligned} \tag{8}
$$

Consequently, it would then be possible to create a functional, physical relationship between input and output information by making the following substitutions into Equation 6:

$$
\begin{aligned}
u_{\text{shear},1}\, A_z \frac{f_y}{\sqrt{3}} &= \left|\frac{EI_{yy}}{L^3(1+\phi)}(6L\theta_1 + 6L\theta_2) + \frac{\omega L}{2}\right| \\
u_{\text{bending},1}\, W_{\text{pl},yy} f_y &= \left|\frac{EI_{yy}}{L^3(1+\phi)}\left((4+\phi)L^2\theta_1 + (2-\phi)L^2\theta_2\right) + \frac{\omega L^2}{12}\right| \\
u_{\text{shear},2}\, A_z \frac{f_y}{\sqrt{3}} &= \left|\frac{EI_{yy}}{L^3(1+\phi)}(-6L\theta_1 - 6L\theta_2) + \frac{\omega L}{2}\right| \\
u_{\text{bending},2}\, W_{\text{pl},yy} f_y &= \left|\frac{EI_{yy}}{L^3(1+\phi)}\left((2-\phi)L^2\theta_1 + (4+\phi)L^2\theta_2\right) - \frac{\omega L^2}{12}\right| .
\end{aligned} \tag{9}
$$

Note that the absolute signs are only needed because utilisation ratios are expressed as non-negative values. Finally, the re-arranged form of these equations becomes:

$$
\begin{aligned}
A_z &= \frac{\sqrt{3}}{u_{\text{shear},1}\, f_y} \left|\frac{EI_{yy}}{L^3(1+\phi)}(6L\theta_1 + 6L\theta_2) + \frac{\omega L}{2}\right| \\
W_{\text{pl},yy} &= \frac{1}{u_{\text{bending},1}\, f_y} \left|\frac{EI_{yy}}{L^3(1+\phi)}\left((4+\phi)L^2\theta_1 + (2-\phi)L^2\theta_2\right) + \frac{\omega L^2}{12}\right| \\
A_z &= \frac{\sqrt{3}}{u_{\text{shear},2}\, f_y} \left|\frac{EI_{yy}}{L^3(1+\phi)}(-6L\theta_1 - 6L\theta_2) + \frac{\omega L}{2}\right| \\
W_{\text{pl},yy} &= \frac{1}{u_{\text{bending},2}\, f_y} \left|\frac{EI_{yy}}{L^3(1+\phi)}\left((2-\phi)L^2\theta_1 + (4+\phi)L^2\theta_2\right) - \frac{\omega L^2}{12}\right| .
\end{aligned} \tag{10}
$$

The usefulness of Equation 10 can be understood in context of the previous structural design model developed in (Gallet et al. 2024b), namely these equations relate a portion of the input space of the network (span and UDL values) and the associated outputs (the cross section properties of an element within a continuous beam system) as long as the following additional terms are known or are predicted:

- Yield strength of steel f_y, Young's modulus E and shear modulus G.

- Utilisation ratios $u_{\text{shear},1}$ $u_{\text{shear},2}$, $u_{\text{bend},1}$ and $u_{\text{bend},2}$.
- Rotations θ_1 and θ_2.

Data and physics loss functions

It is now possible to formulate data and physics loss functions, L_{data} and L_{phy}, respectively to train a physics informed neural network. Both these loss functions are built assuming the same feature and target space as that from the data-driven approach from (Gallet et al. 2024b). Specifically, these included the span value L_0 and UDLs ω_0 from the beam to be designed along with the spans and UDLs of the five adjacently connected elements as inputs (leading to a total of 22 inputs), and the shear area A_z, second moment of area I, and the plastic modulus W_{pl} as outputs.

In general, the combined loss function for a PINN is written as:

$$L = L_{\text{data}} + \lambda L_{\text{phy}} \tag{11}$$

where λ is a hyperparameter introduced to control the extent to which physical constraints should be applied on the parameter space of the network $\boldsymbol{\theta}$. The data loss function L_{data} adopted here is equivalent to the MAPE loss function established in the previous data-driven work, and is chosen due to its improved performance compared to other loss functions tested. Consequently, L_{data} is defined as:

$$L_{\text{data}} = \frac{1}{3}\left(\left|\frac{\widehat{I} - I}{I + \epsilon}\right| + \left|\frac{\widehat{A_z} - A_z}{A_z + \epsilon}\right| + \left|\frac{\widehat{W_{\text{pl}}} - W_{\text{pl}}}{W_{\text{pl}} + \epsilon}\right|\right) \tag{12}$$

where $\widehat{I}$, $\widehat{A_z}$, $\widehat{W_{\text{pl}}}$ each represent the predicted cross-section properties. The physics loss function L_{phy} on the other hand will compare predicted cross-section property values (denoted by "pred") against the the physics derived cross-section properties introduced by Equation 10, whilst still being constructed as MAPE loss functions:

$$\begin{aligned} L_{\text{phy}} \rightarrow L_{\frac{\text{pred}-\text{phy}}{\text{pred}}} = \frac{1}{4}\Bigg(&\left|\frac{\widehat{A_z} - A_{z,\text{phy},1}}{\widehat{A_z} + \epsilon}\right| + \left|\frac{\widehat{W_{\text{pl}}} - W_{\text{pl,phy},1}}{\widehat{W_{\text{pl}}} + \epsilon}\right| \\ &+ \left|\frac{\widehat{A_z} - A_{z,\text{phy},2}}{\widehat{A_z} + \epsilon}\right| + \left|\frac{\widehat{W_{\text{pl}}} - W_{\text{pl,phy},2}}{\widehat{W_{\text{pl}}} + \epsilon}\right|\Bigg) \end{aligned} \tag{13}$$

where the physics derived cross-section properties $A_{z,\text{phy},1}$, $W_{\text{pl,phy},1}$, $A_{z,\text{phy},2}$ and $W_{\text{pl,phy},2}$ are defined by:

$$
\begin{aligned}
A_{z,\mathrm{phy},1} =& \frac{\sqrt{3}}{u_{\mathrm{shear},1}\, f_y} \left| \frac{E\widehat{I}}{L^3(1+\phi)}(6L\theta_1 + 6L\theta_2) + \frac{\omega L}{2} \right| \\
W_{\mathrm{pl},\mathrm{phy},1} =& \frac{1}{u_{\mathrm{bend},1}\, f_y} \left| \frac{E\widehat{I}}{L^3(1+\phi)} \Big((4+\phi)L^2\theta_1 + (2-\phi)L^2\theta_2 \Big) + \frac{\omega L^2}{12} \right| \\
A_{z,\mathrm{phy},2} =& \frac{\sqrt{3}}{u_{\mathrm{shear},2}\, f_y} \left| \frac{E\widehat{I}}{L^3(1+\phi)}(-6L\theta_1 - 6L\theta_2) + \frac{\omega L}{2} \right| \\
W_{\mathrm{pl},\mathrm{phy},2} =& \frac{1}{u_{\mathrm{bend},2}\, f_y} \left| \frac{E\widehat{I}}{L^3(1+\phi)} \Big((2-\phi)L^2\theta_1 + (4+\phi)L^2\theta_2 \Big) - \frac{\omega L^2}{12} \right| \\
& \text{where } \phi = \tfrac{12E\widehat{I}}{\widehat{A_z}GL^2}
\end{aligned}
\tag{14}
$$

It is worth drawing attention to the fact that the physics derived cross-section properties $A_{z,\mathrm{phy},1}$, $A_{z,\mathrm{phy},2}$, $W_{\mathrm{pl},\mathrm{phy},1}$ and $W_{\mathrm{pl},\mathrm{phy},2}$ as shown in Equation 14 are based on both the UDL ω and span L of the target beam, along with the *predicted* second moment of area $\widehat{I}$ and shear area $\widehat{A_z}$ values. In other words, there is now a partial physical relationship between the input and output space. In end effect, the transformation from Equation 4 to 14 has yielded a physical equation which evaluates what the physically consistent $A_{z,\mathrm{phy},1}$, $A_{z,\mathrm{phy},2}$, $W_{\mathrm{pl},\mathrm{phy},1}$ and $W_{\mathrm{pl},\mathrm{phy},2}$ values should be for a specific beam, and any deviation from those can be minimised with the L_{phy} loss function established by Equation 13.

As discussed previously, in order to be able to minimise L_{phy}, the remaining physical information needs to be known. This includes the Youngs and shear modulus E and G, the various utilisation factors $u_{\mathrm{shear},1}$ $u_{\mathrm{shear},2}$, $u_{\mathrm{bend},1}$ and $u_{\mathrm{bend},2}$, and the rotations θ_1 and θ_2. In this study, these values were initially extracted from structural analysis simulations based on all UDLs applied (that is, no pattern loading). For the first set of networks with a *standard architecture*, these additional values were simply provided to the physics loss function L_{phy} as known data, since this means the output space does not need to be changed, and therefore allows direct comparison with the results achieved in (Gallet et al. 2024b) to determine the advantages of the proposed PINN formulation. A second *verifiable architecture* will however also be tested in which the $u_{\mathrm{shear},1}$, $u_{\mathrm{shear},2}$, $u_{\mathrm{bend},1}$ and $u_{\mathrm{bend},2}$, and the two rotations θ_1 and θ_2 will be predicted in addition to the three cross-section parameters I, A_z, W_{pl}, resulting in a total output space of nine values. This allows the physical validity of the predictions to be tested during inference discussed in Section 4.

Physics loss functions of standard architectures

It is in fact possible to immediately arrive at three further physics loss functions L_{phy}. The first one introduced in equation 13 above compares predicted cross section values against physics derived ones, and subsequently divides by the predicted values to evaluate a percentage-like error estimation. However, it is relatively straightforward to build variations of the function:

$$L_{\frac{\text{true-phy}}{\text{true}}} = \frac{1}{4}\left(\left|\frac{A_z - A_{z,\text{phy},1}}{A_z + \epsilon}\right| + \left|\frac{W_{\text{pl}} - W_{\text{pl},\text{phy},1}}{W_{\text{pl}} + \epsilon}\right| \right.$$
$$\left. + \left|\frac{A_z - A_{z,\text{phy},2}}{A_z + \epsilon}\right| + \left|\frac{W_{\text{pl}} - W_{\text{pl},\text{phy},2}}{W_{\text{pl}} + \epsilon}\right|\right) \tag{15}$$

$$L_{\frac{\text{phy-pred}}{\text{phy}}} = \frac{1}{4}\left(\left|\frac{A_{z,\text{phy},1} - \widehat{A_z}}{A_{z,\text{phy},1} + \epsilon}\right| + \left|\frac{W_{\text{pl},\text{phy},1} - \widehat{W_{\text{pl}}}}{W_{\text{pl},\text{phy},1} + \epsilon}\right| \right.$$
$$\left. + \left|\frac{A_{z,\text{phy},2} - \widehat{A_z}}{A_{z,\text{phy},2} + \epsilon}\right| + \left|\frac{W_{\text{pl},\text{phy},2} - \widehat{W_{\text{pl}}}}{W_{\text{pl},\text{phy},2} + \epsilon}\right|\right) \tag{16}$$

$$L_{\frac{\text{phy-true}}{\text{phy}}} = \frac{1}{4}\left(\left|\frac{A_{z,\text{phy},1} - A_z}{A_{z,\text{phy},1} + \epsilon}\right| + \left|\frac{W_{\text{pl},\text{phy},1} - W_{\text{pl}}}{W_{\text{pl},\text{phy},1} + \epsilon}\right| \right.$$
$$\left. + \left|\frac{A_{z,\text{phy},2} - A_z}{A_{z,\text{phy},2} + \epsilon}\right| + \left|\frac{W_{\text{pl},\text{phy},2} - W_{\text{pl}}}{W_{\text{pl},\text{phy},2} + \epsilon}\right|\right). \tag{17}$$

Each of these PINN formulations differentiate themselves slightly. Whilst Equation 13 evaluates the percentage loss by dividing by the predicted cross-sectional properties, Equation 15 achieves this by dividing by the true cross-sectional values, and Equations 16 and 17 by the physics-evaluated properties via Equation 14. Furthermore, the numerators across all of these formulations differ, by either comparing the physics derived properties from Equation 14 with either predicted properties as in Equations 13 and 16, or true properties as is the case for Equations 15 and 17. Given the novelty of using the local stiffness relations to physically constrain a structural design model, all four variations will be tested in the standard architectures to determine which of these physics-loss functions L_{phy} perform best.

Physics loss functions for verifiable networks

One advantage of the $L_{\frac{\text{pred-phy}}{\text{pred}}}$ shown in Equation 13 is that it relies primarily on predicted cross-section values. If additionally, all further response information was predicted by the network, that is the various utilisation ratios $u_{\text{shear},1}$, $u_{\text{shear},2}$, $u_{\text{bend},1}$ and $u_{\text{bend},2}$, and the two rotation values θ_1 and θ_2 for each individual beam, then it should be possible to evaluate the physics derived cross-section properties using the input and output vectors alone. In this case, $\widehat{A_{z,\text{phy},1}}$, $\widehat{A_{z,\text{phy},2}}$, $\widehat{W_{\text{pl},\text{phy},1}}$ and $\widehat{W_{\text{pl},\text{phy},2}}$ are evaluated by:

$$
\begin{aligned}
\widehat{A_{z,\text{phy},1}} &= \frac{\sqrt{3}}{\widehat{u_{\text{shear},1}}\; f_y}\left|\frac{E\widehat{I}}{L^3(1+\phi)}(6L\widehat{\theta_1}+6L\widehat{\theta_2})+\frac{\omega L}{2}\right| \\
\widehat{W_{\text{pl},\text{phy},1}} &= \frac{1}{\widehat{u_{\text{bend},1}}\; f_y}\left|\frac{E\widehat{I}}{L^3(1+\phi)}\left((4+\phi)L^2\widehat{\theta_1}+(2-\phi)L^2\widehat{\theta_2}\right)+\frac{\omega L^2}{12}\right| \\
\widehat{A_{z,\text{phy},2}} &= \frac{\sqrt{3}}{\widehat{u_{\text{shear},2}}\; f_y}\left|\frac{E\widehat{I}}{L^3(1+\phi)}(-6L\widehat{\theta_1}-6L\widehat{\theta_2})+\frac{\omega L}{2}\right| \\
\widehat{W_{\text{pl},\text{phy},2}} &= \frac{1}{\widehat{u_{\text{bend},2}}\; f_y}\left|\frac{E\widehat{I}}{L^3(1+\phi)}\left((2-\phi)L^2\widehat{\theta_1}+(4+\phi)L^2\widehat{\theta_2}\right)-\frac{\omega L^2}{12}\right|
\end{aligned}
\tag{18}
$$

and used in the following verifiable physics loss function:

$$
\begin{aligned}
L_{\text{phy,verifiable}} = \frac{1}{4}\Bigg(&\left|\frac{\widehat{A_z}-\widehat{A_{z,\text{phy},1}}}{\widehat{A_z}+\epsilon}\right|+\left|\frac{\widehat{W_{\text{pl}}}-\widehat{W_{\text{pl},\text{phy},1}}}{\widehat{W_{\text{pl}}}+\epsilon}\right| \\
&+\left|\frac{\widehat{A_z}-\widehat{A_{z,\text{phy},2}}}{\widehat{A_z}+\epsilon}\right|+\left|\frac{\widehat{W_{\text{pl}}}-\widehat{W_{\text{pl},\text{phy},2}}}{\widehat{W_{\text{pl}}}+\epsilon}\right|\Bigg).
\end{aligned}
\tag{19}
$$

In this case, the only required values would be the material properties, namely the Yield strength f_y, Young's modulus E and shear modulus G. These can be easily established, as for example would be the case using the *CBeamXP* dataset (Gallet and Smyl 2023).

Standard and verifiable neural network architectures

Two distinct physics loss functions are to be tested, a standard and verifiable physics loss function, each requiring a different output size. The standard physics loss functions from Equations 13, 15, 16 and 17 will be tested with the exact same architectures as those established in (Gallet et al. 2024b), namely 50-50 and 600-600-600 layered architectures using $a_{\text{in,ReLU}}$ (rectified linear unit, ReLU, activation) and $a_{\text{out,exp}}$ (exponential, exp) activation functions. The 600-600-600 was specifically chosen due to its optimal performance profile evaluated empirically in (Gallet et al. 2024b). The only difference will be the addition of the λL_{phy} term to enforce the physical constraints on the learning parameters. The verifiable loss function from Equation 19 will contain the exact same internal architecture (that is the same hidden layer structure), yet be tested tested on an expanded output size of size 9 to be able to predict the utilisation ratios and rotations of the local beam element to be designed. These predicted values can then be used in Equation 19 to evaluate the $L_{\text{phy,verifiable}}$. Different outer activation functions will be applied to certain outputs to match their ranges as shown in Figure 2.

Overview of neural network training details

Both architectures will require extraction of the utilisation ratio and rotation information during training. This was easily achieved by using the CBeamXP dataset to model the individual systems in *Grasshopper* (Robert McNeel &

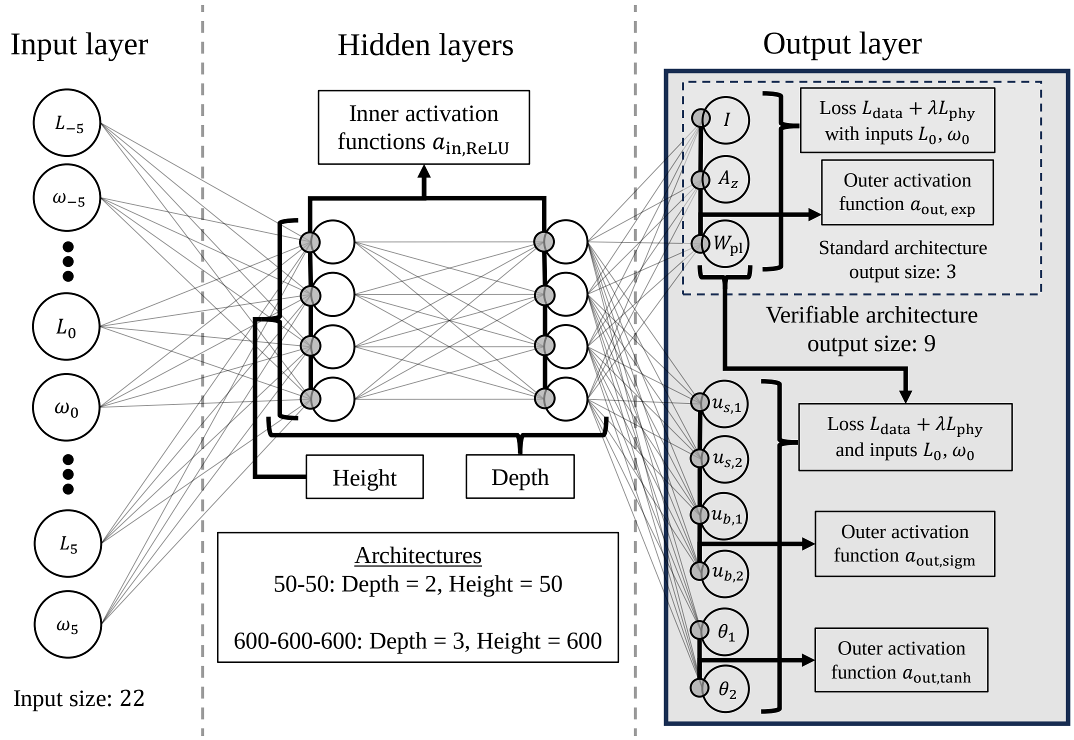

Fig. 2. Overview of the standard and verifiable neural network architectures to be investigated with hidden layer sizes 50-50 and 600-600-600. In the standard architecture, physical information (utilisation ratios u and rotations θ) are passed directly into the physics function L_{phy} as true values to allow comparison with previous research results. In the verifiable architecture, the physical information itself is predicted to allow for verification at inference. Notice that u_{shear} and u_{bending} have been abbreviated to u_{s} and u_{b} for visualisation purposes.

Associates 6 09) with *Karamba3D* (Preisinger 2013). Importantly, data for design members was generated using the concept of the *influence zone* detailed in (Gallet et al. 2024a) and implemented in (Gallet et al. 2024b). Adopting the influence zone structural analysis approach was critical in this work as it negates functionally non-influential data from members loaded outside of the influence zone having k_{max} influential adjacent neighbours. Moreover, this approach, in a broader picture, permits design of individual spans for structures containing a arbitrary number of members – thus, fixing the network input size to a computed influence zone defined by k_{max}. Furthermore, the utilisation ratios and rotations over each support were then extracted for each individual continuous system each containing 11 beams. All other neural network training details were kept identical to those from the previous machine learning model developed in (Gallet et al. 2024b), which were obtained based on empirical tests, and are briefly summarised in Table 1 below.

RESULTS

Various L_{phy} loss functions were tested using standard architectures only. The purpose of these tests was to identify which physics loss functions are best suited for the problem at hand. Additionally, in order to study the impact of dataset size (since PINNs have been shown to require less data to achieve the similar performance in the past (Raissi et al. 2019)), 50-50 architectures were tested with three separate training dataset sizes, whilst varying λ from 0 to 1000

Neural network training option	Chosen parameter
Optimiser	Nadam
Learning rate	$\alpha = 0.0005$
Initialiser	Gaussian with $\mu = 0$, $\sigma = 0.05$
Batch-size	1024

TABLE 1. Overview of neural network learning parameters and hyperparameters during standard and verifiable PINN architecture testing.

in increasing orders of magnitudes. These results are shown numerically in Tables 2, 3, 4, and graphically in Figure 3.

Performance comparison of various L_{phy} loss function formulations for standard architectures

The following three Tables 2, 3 and 4 compare the performance of data-driven models only (when $\lambda = 0$) against PINNs with increasing physics constraints. In each table L_{data} presented the data-derived MAPE for each network, with both training and validation results indicated, with the specified physics loss function $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$, $L_{\frac{\text{phy}-\text{true}}{\text{true}}}$, $L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$ or $L_{\frac{\text{true}-\text{phy}}{\text{phy}}}$ errors shown as well.

10k training dataset size

	Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy}-\text{true}}{\text{true}}}$			
	L_{data}		$L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$		L_{data}		$L_{\frac{\text{phy}-\text{true}}{\text{true}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.166	0.181	0.394	0.417	0.166	0.181	0.288	0.358
0.0001	0.160	0.177	0.352	0.375	0.165	0.181	0.286	0.355
0.001	0.203	0.219	0.423	0.472	0.166	0.180	0.288	0.355
0.01	0.149	0.169	0.290	0.361	0.166	0.180	0.270	0.353
0.1	0.184	0.199	0.281	0.374	0.208	0.224	0.244	0.400
1	0.233	0.244	0.199	0.343	0.299	0.316	0.289	0.465
10	0.345	0.345	0.223	0.346	0.411	0.428	0.323	0.544
100	0.312	0.340	0.175	0.325	0.459	0.471	0.349	0.575
1000	0.331	0.340	0.186	0.322	0.459	0.500	0.340	0.564
	Training with $L_{\text{data}} + \lambda L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{true}-\text{phy}}{\text{phy}}}$			
	L_{data}		$L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$		L_{data}		$L_{\frac{\text{true}-\text{phy}}{\text{phy}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.166	0.181	0.597	0.459	0.166	0.181	1.132	0.745
0.0001	1.018	0.983	1.806	1.889	0.816	0.816	16.18	17.83
0.001	1.132	1.053	1.483	1.996	25.76	25.08	1.402	3.997
0.01	42.56	41.88	5.867	8.135	50.72	49.70	1.425	3.862
0.1	12.93	12.49	7.234	13.73	9.803	8.957	1.148	2.011
1	69.49	68.50	4.229	8.497	15.39	14.69	1.354	2.626
10	101.7	101.4	2.281	4.123	80.00	79.14	1.120	1.535
100	55.50	54.56	2.826	4.307	70.21	69.56	1.074	1.680
1000	79.90	78.62	3.520	4.131	153.6	152.0	0.974	1.280

TABLE 2. Performance profiles for standard 50-50 architectures at epoch 1000 with various $L_{\text{data}} + \lambda L_{\text{phy}}$ loss function combinations using 10k training and 150k validation sets.

100k training dataset size

	Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy-pred}}{\text{pred}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy-true}}{\text{true}}}$			
	L_{data}		$L_{\frac{\text{phy-pred}}{\text{pred}}}$		L_{data}		$L_{\frac{\text{phy-true}}{\text{true}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.086	0.091	0.225	0.179	0.086	0.091	0.197	0.167
0.0001	0.090	0.094	0.287	0.203	0.080	0.082	0.179	0.162
0.001	0.092	0.095	0.203	0.206	0.076	0.081	0.176	0.142
0.01	0.082	0.083	0.181	0.170	0.080	0.084	0.150	0.149
0.1	0.078	0.087	0.121	0.176	0.106	0.106	0.164	0.190
1	0.104	0.112	0.132	0.186	0.166	0.174	0.190	0.267
10	0.190	0.207	0.165	0.223	0.227	0.223	0.234	0.324
100	0.175	0.173	0.131	0.203	0.267	0.260	0.263	0.309
1000	0.243	0.286	0.154	0.224	0.307	0.330	0.289	0.349
	Training with $L_{\text{data}} + \lambda L_{\frac{\text{pred-phy}}{\text{phy}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{true-phy}}{\text{phy}}}$			
	L_{data}		$L_{\frac{\text{pred-phy}}{\text{phy}}}$		L_{data}		$L_{\frac{\text{true-phy}}{\text{phy}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.086	0.091	0.293	0.283	0.086	0.091	0.361	0.336
0.0001	0.588	0.571	4.831	4.731	0.462	0.462	18.12	18.636
0.001	0.802	0.802	4.189	4.216	1.144	2.939	5.336	4.114
0.01	44.28	44.74	6.339	6.122	10.72	10.91	2.630	2.617
0.1	71.70	72.70	2.838	3.825	53.62	54.34	1.550	1.811
1	22.33	23.687	7.739	7.587	38.61	39.24	1.448	2.500
10	0.989	0.993	0.935	0.954	27.28	27.46	2.166	2.288
100	691.7	695.0	3.856	4.068	599.0	610.8	0.965	0.960
1000	121.5	122.9	2.662	3.240	1.819	1.863	3.051	3.803

TABLE 3. Performance profiles for standard 50-50 architectures at epoch 1000 with various $L_{\text{data}} + \lambda L_{\text{phy}}$ loss function combinations using 100k training and 150k validation sets.

700k training dataset size

Visual comparison of results

The results from Tables 2, 3 and 4 can also be visualised for ease of comparison as shown in Figure 3. This figure indicates the impact which increasing the dataset size has on the overall training performance in terms of the pure data loss L_{data} and physics loss L_{physics}. The first observation is the deterioration in performance when the physics loss function uses the physics-estimated cross-section properties within the denominator as is the case for $L_{\frac{\text{pred-phy}}{\text{phy}}}$ and $L_{\frac{\text{true-phy}}{\text{phy}}}$. This may be due to the volatility associated with dividing the losses through the physics estimated cross-section properties, which is a more volatile measure than dividing by the true cross-section properties, such as $L_{\frac{\text{phy-true}}{\text{true}}}$ or predicted cross-section properties $L_{\frac{\text{phy-pred}}{\text{pred}}}$.

Furthermore, increased dataset size generally increased the smoothness of the results, yet from the remaining

	Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{phy}-\text{true}}{\text{true}}}$			
	L_{data}		$L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$		L_{data}		$L_{\frac{\text{phy}-\text{true}}{\text{true}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.066	0.065	0.171	0.157	0.066	0.065	0.150	0.133
0.0001	0.065	0.065	0.159	0.159	0.062	0.063	0.125	0.118
0.001	0.057	0.058	0.137	0.129	0.067	0.066	0.150	0.134
0.01	0.064	0.064	0.138	0.135	0.065	0.065	0.136	0.134
0.1	0.060	0.055	0.120	0.117	0.062	0.055	0.117	0.100
1	0.084	0.077	0.134	0.134	0.081	0.072	0.124	0.115
10	0.119	0.112	0.131	0.137	0.126	0.110	0.166	0.143
100	0.139	0.133	0.116	0.121	0.135	0.124	0.179	0.170
1000	0.299	0.184	0.117	0.130	0.152	0.140	0.172	0.170

	Training with $L_{\text{data}} + \lambda L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$				Training with $L_{\text{data}} + \lambda L_{\frac{\text{true}-\text{phy}}{\text{phy}}}$			
	L_{data}		$L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$		L_{data}		$L_{\frac{\text{true}-\text{phy}}{\text{phy}}}$	
λ	Train.	Val.	Train.	Val.	Train.	Val.	Train.	Val.
0	0.066	0.065	0.262	0.193	0.066	0.065	0.343	0.258
0.0001	0.582	0.457	8.432	2.182	0.386	0.387	18.68	18.63
0.001	0.702	0.704	2.256	2.471	0.803	0.738	7.411	8.957
0.01	0.703	0.704	1.190	1.166	0.778	0.775	6.887	6.777
0.1	0.494	0.494	0.457	0.462	17.02	29.60	6.022	4.665
1	0.560	0.557	0.270	0.275	0.743	0.741	18.59	18.60
10	13660	6713	142.1	23.33	1.369	1.364	13.91	16.23
100	37.09	31.86	0.954	0.955	0.823	0.827	20.46	18.55
1000	0.564	0.562	0.227	0.227	29.25	46.78	3.435	1.635

TABLE 4. Performance profiles for standard 50-50 architectures at epoch 1000 with various $L_{\text{data}} + \lambda L_{\text{phy}}$ loss function combinations using 700k training and 150k validation sets.

physics loss function, $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ was the only loss function in which physics losses continued to decrease with increasing λ values. The fact that data losses appear to deteriorate significantly for values larger than $\lambda = 1$ was also an interesting result that will be discussed in further in Section 4. For theses reasons, the remaining training runs conducted in this study used $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ as the physics loss function only.

Detailed performance profiles of standard architectures with $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ and 700k training set

Results for λ variations with standard architectures

An in-depth breakdown of the performance profile of the PINNs using the $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ physics loss function and 700k training set are provided in Table 5. These include showing the accuracy percentile profiles achieved on the validation set containing 150k data points for both the 50-50 and 600-600-600 architectures.

Visual representation of results for λ variations for standard architectures

The key trends of this performance profile can also be shown visually with Figures 4 and 5. These results indicate that PINNs formulated with the $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ loss function appear to provide some benefit to smaller/simpler architectures such as with 50-50 hidden layers, demonstrated by the reduction in the physics loss L_{phy} and a tightening of the accuracy percentiles for lambda values up to $\lambda = 1$. For the 600-600-600 architecture, there is no visible advantage using PINNs. In both cases the performance deteriorates for lambda values greater than $\lambda > 1$.

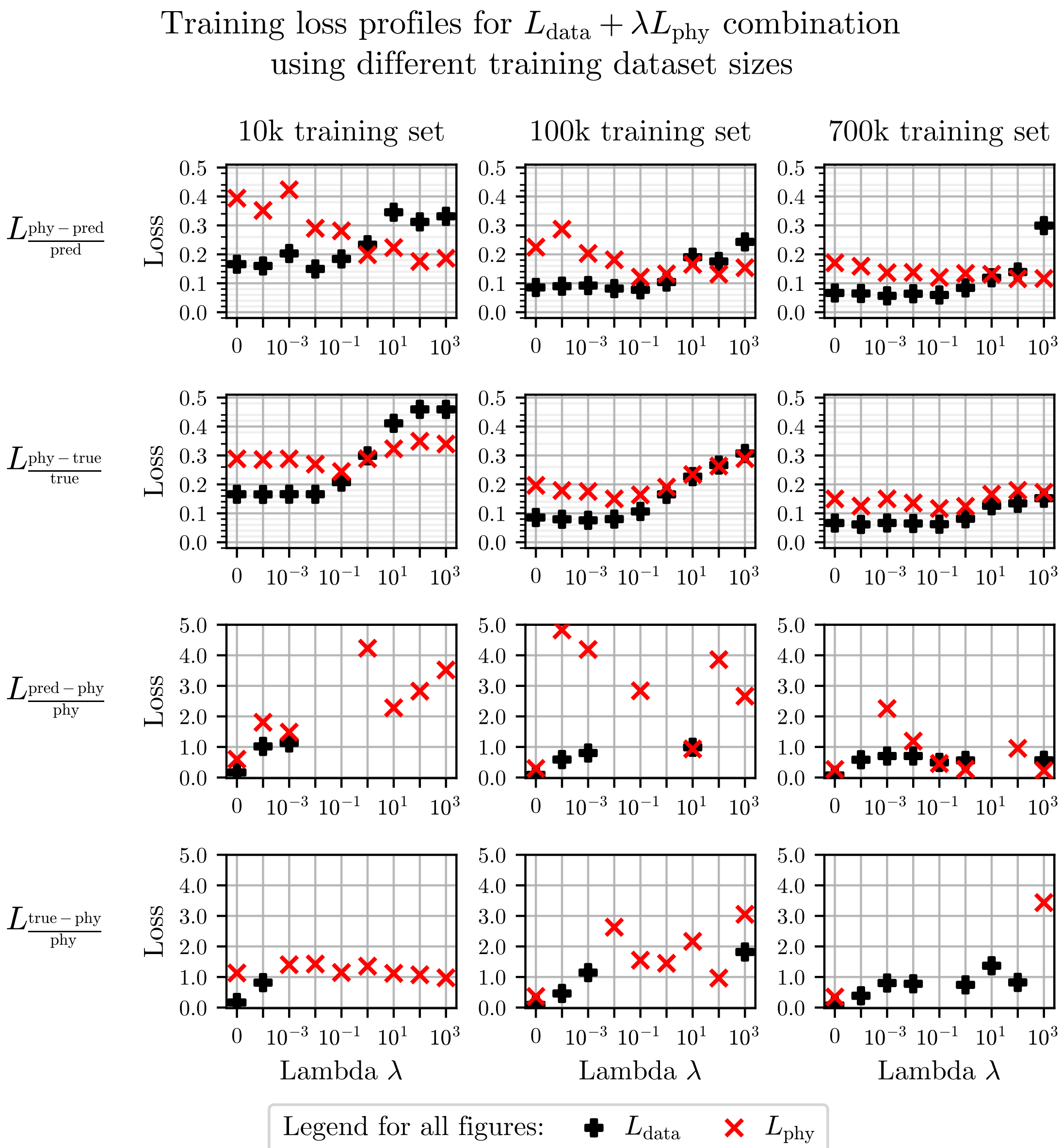

Fig. 3. Training losses for standard 50-50 architecture at epoch 1000 for various $L_{\text{data}} + \lambda L_{\text{phy}}$ loss functions and training dataset sizes.

Performance comparison for training, validation and testing data for $\lambda = 0.1$ models with full training graphs for standard architectures

Lastly, a full set of results of accuracy profiles and L_{data} errors trained with $\lambda = 0.1$ and the $L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function are shown in Table 6, with training profiles shown in Figures 6 and 7. As a general observation, which will be discussed fully in Section 4, PINNs tend to increase the performance of smaller standard architectures such as 50-50 slightly, yet there is no substantial performance improvement when applied to larger architectures.

	Standard 50-50 architecture											
	L_{data}		$L_{\frac{\text{phy-pred}}{\text{pred}}}$		Validation Accuracies Percentiles							
λ	Train.	Val.	Train.	Val.	Min	0.5%	2.5%	Median	97.5%	99.5%	Max
0	0.066	0.065	0.171	0.157	0.032	0.523	0.785	0.994	1.185	1.458	19.93
0.0001	0.065	0.065	0.159	0.159	0.031	0.510	0.803	1.006	1.187	1.454	44.61
0.001	0.057	0.058	0.137	0.129	0.055	0.548	0.799	0.983	1.143	1.402	14.79
0.01	0.064	0.064	0.138	0.135	0.037	0.539	0.804	0.996	1.191	1.489	33.09
0.1	0.060	0.055	0.120	0.117	0.049	0.596	0.835	1.001	1.166	1.440	29.31
1	0.084	0.077	0.134	0.134	0.037	0.579	0.802	1.001	1.279	1.750	26.29
10	0.119	0.112	0.131	0.137	0.041	0.401	0.689	1.006	1.444	2.492	108.9
100	0.139	0.133	0.116	0.121	0.108	0.594	0.774	1.012	1.506	3.053	235.4
1000	0.299	0.184	0.117	0.130	0.064	0.473	0.735	1.008	1.503	3.351	1420
	Standard 600-600-600 architecture											
	L_{data}		$L_{\frac{\text{phy-pred}}{\text{pred}}}$		Validation Accuracies Percentiles							
λ	Train.	Val.	Train.	Val.	Min	0.5%	2.5%	Median	97.5%	99.5%	Max
0	0.009	0.016	0.021	0.030	0.275	0.920	0.968	1.006	1.058	1.133	26.81
0.0001	0.009	0.016	0.021	0.032	0.220	0.891	0.946	0.994	1.036	1.099	15.87
0.001	0.009	0.016	0.022	0.033	0.159	0.896	0.948	0.993	1.035	1.102	26.36
0.01	0.010	0.016	0.022	0.033	0.316	0.896	0.949	0.994	1.037	1.108	7.800
0.1	0.014	0.018	0.034	0.039	0.232	0.884	0.942	0.991	1.032	1.094	8.878
1	0.032	0.027	0.064	0.052	0.178	0.838	0.921	0.997	1.081	1.188	11.90
10	0.034	0.029	0.046	0.057	0.181	0.828	0.916	1.001	1.090	1.237	24.86
100	0.053	0.041	0.051	0.058	0.149	0.816	0.910	1.000	1.110	1.316	531.4
1000	0.128	0.071	0.049	0.056	0.189	0.815	0.908	1.003	1.118	1.447	2043

TABLE 5. Performance profiles for standard 50-50 and 600-600-600 architectures at epoch 1000 with $L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function combination using 700k training and 150k validation sets.

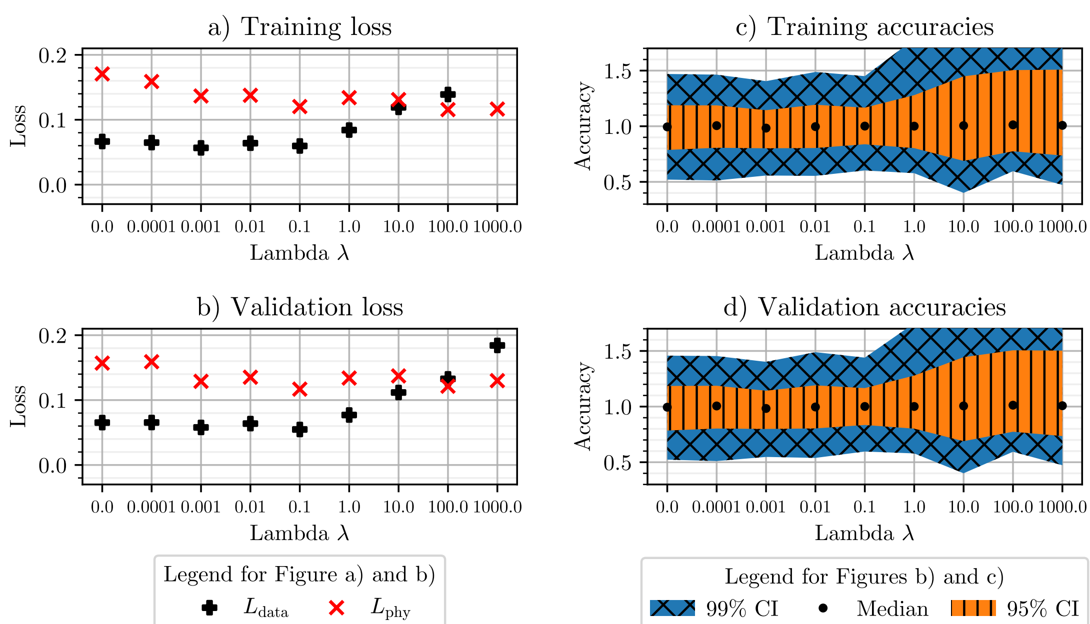

Fig. 4. Visual performance profiles for the standard 50-50 architecture at epoch 1000 with $L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function using 700k training and 150k validation sets.

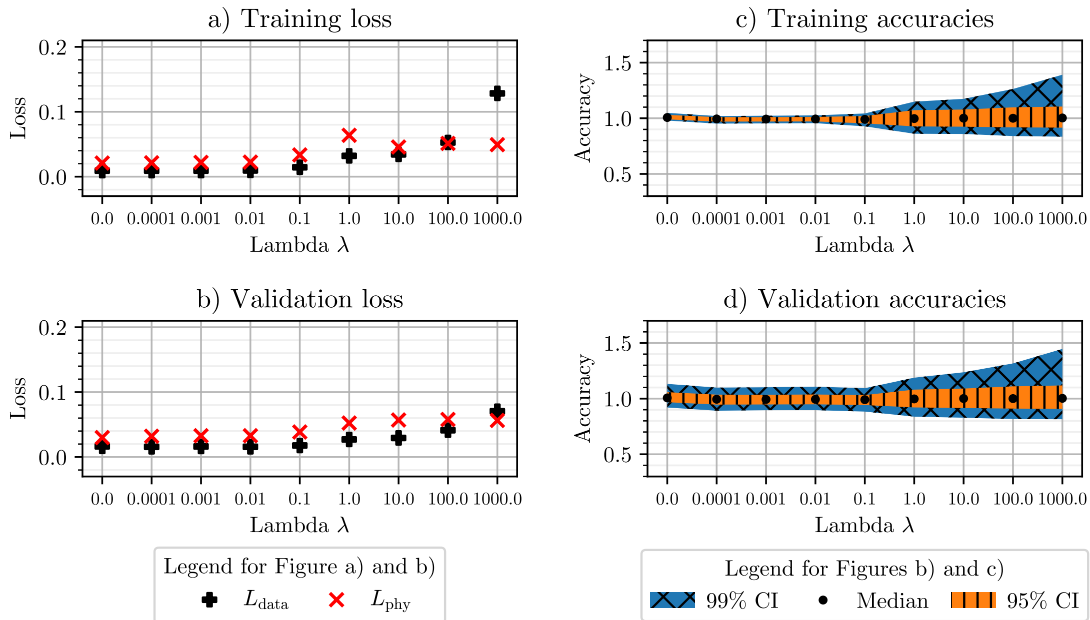

Fig. 5. Visual performance profiles for the standard 600-600-600 architecture at epoch 1000 with $L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function using 700k training and 150k validation sets.

Model	Dataset	L_{data}	Accuracy Percentiles						
			Min	0.5%	2.5%	Median	97.5%	99.5%	Max
Standard 50-50 model with $\lambda = 0.0$	Train.	0.066	0.022	0.520	0.787	0.994	1.188	1.470	23.33
	Val.	0.065	0.032	0.523	0.785	0.994	1.185	1.458	19.94
	Test.	0.066	0.032	0.514	0.785	0.994	1.189	1.477	18.43
Standard 50-50 model with $\lambda = 0.1$	Train.	0.060	0.020	0.603	0.838	1.001	1.167	1.450	14.32
	Val.	0.055	0.049	0.596	0.835	1.001	1.166	1.440	29.31
	Test.	0.054	0.070	0.601	0.837	1.001	1.169	1.470	12.91
Standard 600-600-600 model with $\lambda = 0.0$	Train.	0.009	0.802	0.980	0.990	1.007	1.030	1.048	1.272
	Val.	0.016	0.275	0.920	0.968	1.006	1.058	1.133	26.81
	Test.	0.016	0.313	0.917	0.967	1.006	1.058	1.138	12.28
Standard 600-600-600 model with $\lambda = 0.1$	Train.	0.014	0.488	0.927	0.954	0.990	1.020	1.043	5.289
	Val.	0.018	0.232	0.884	0.942	0.991	1.032	1.094	8.878
	Test.	0.018	0.163	0.881	0.942	0.990	1.032	1.095	10.36

TABLE 6. Loss and accuracy profiles for standard architectures at epoch 1000 trained with $\lambda = 0.1$ and the $L_{\frac{\text{phy-pred}}{\text{pred}}}$ with 700k training, 150k validation sets and 150k testing set.

Verifiable PINNs with $L_{\frac{\text{phy-pred}}{\text{pred}}}$

Presented here are all the results associated with the verifiable architecture trained with the $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$ loss function combination. Since only one physics function is applicable here, and since there was no advantage to training with reduced dataset sizes, the verifiable results presented here were all training with the 700k dataset.

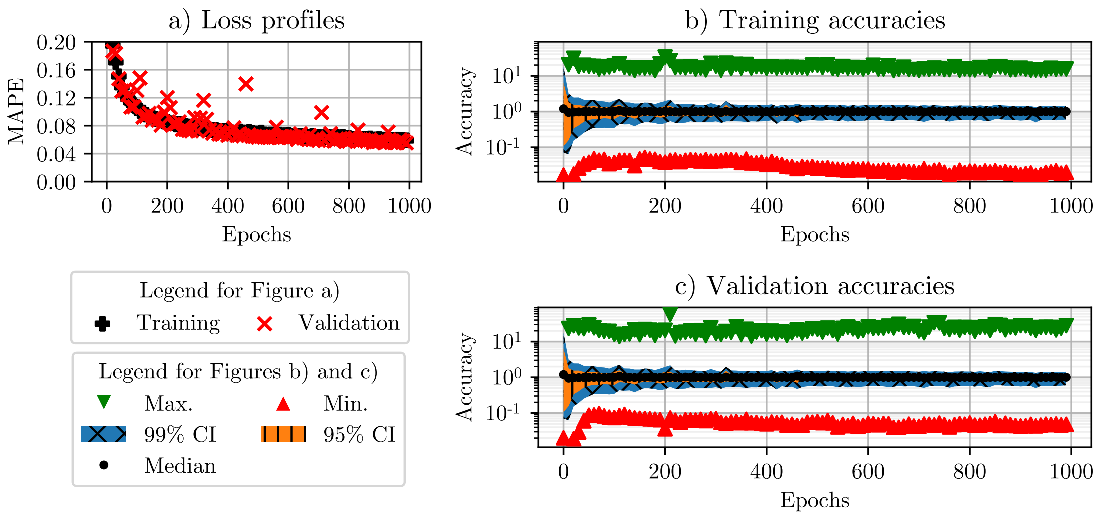

Fig. 6. Visual performance profiles for standard 50-50 architecture at epoch 1000 with with λ = 0.1 and the $L_{\frac{\text{phy-pred}}{\text{pred}}}$ using 700k training and 150k validation sets.

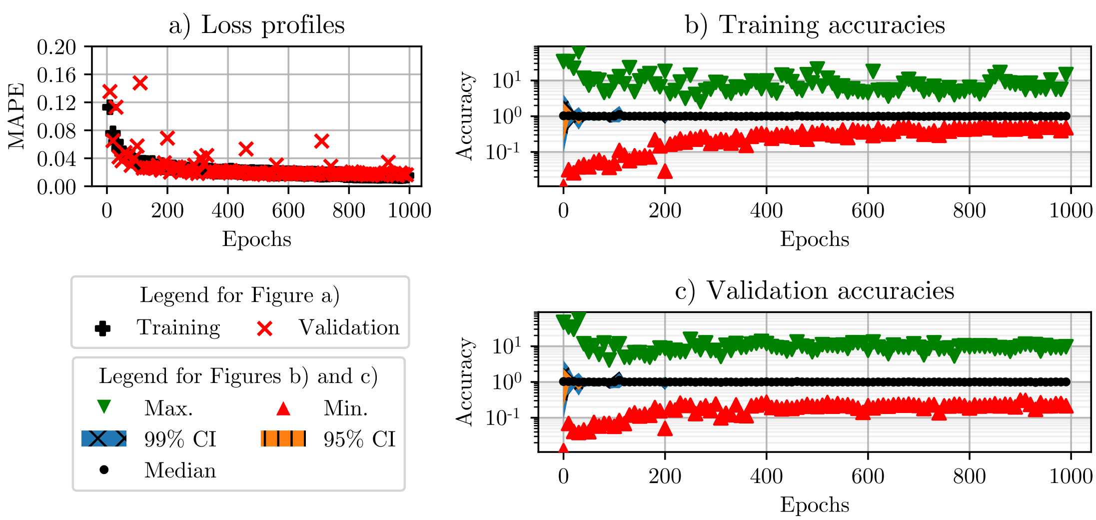

Fig. 7. Visual performance profiles for standard 600-600-600 architecture at epoch 1000 with with λ = 0.1 and the $L_{\frac{\text{phy-pred}}{\text{pred}}}$ using 700k training and 150k validation sets.

Results for λ variations with verifiable architectures

Here, we tabulate results for verifiable architectures for variable λ in Table 7 These results are also visualised for the verifiable 50-50 architecture as shown in Figure 8 and the verifiable 600-600-600 architecture as shown in Figure 9. A deterioration of performance for $\lambda > 1$ is once again clearly visible.

	Verifiable 50-50 architecture										
	L_{data}		$L_{\text{phy,verifiable}}$		Validation Accuracies Percentiles						
λ	Train.	Val.	Train.	Val.	Min	0.5%	2.5%	Median	97.5%	99.5%	Max
0	0.064	0.064	224.9	229.0	0.026	0.531	0.788	0.997	1.185	1.444	25.08
0.0001	0.063	0.063	1.347	1.478	0.029	0.520	0.790	1.000	1.180	1.433	27.67
0.001	0.064	0.063	0.452	0.514	0.045	0.528	0.767	0.997	1.170	1.377	18.26
0.01	0.063	0.063	0.186	0.188	0.042	0.503	0.796	0.999	1.185	1.463	63.53
0.1	0.067	0.067	0.097	0.102	0.034	0.542	0.798	0.996	1.195	1.500	47.62
1	0.140	0.141	0.023	0.023	0.024	0.193	0.427	0.987	1.244	1.615	37.52
10	0.694	0.691	0.010	0.011	0.012	0.095	0.203	1.492	2.310	2.752	12.99
100	1.068	1.063	0.006	0.006	0.002	0.031	0.086	1.852	3.036	3.945	23.76
1000	2.388	2.377	0.004	0.004	0.023	0.100	0.226	3.258	7.088	11.73	88.62
	Verifiable 600-600-600 architecture										
	L_{data}		$L_{\text{phy,verifiable}}$		Validation Accuracies Percentiles						
λ	Train.	Val.	Train.	Val.	Min	0.5%	2.5%	Median	97.5%	99.5%	Max
0	0.010	0.014	98.66	109.6	0.272	0.906	0.954	0.992	1.020	1.066	12.41
0.0001	0.010	0.013	0.383	0.316	0.279	0.911	0.956	0.993	1.023	1.068	19.21
0.001	0.010	0.013	0.136	0.161	0.170	0.934	0.976	1.006	1.046	1.102	14.15
0.01	0.009	0.014	0.066	0.058	0.145	0.902	0.953	0.993	1.019	1.062	8.888
0.1	0.010	0.014	0.033	0.034	0.200	0.909	0.954	0.993	1.022	1.069	14.72
1	0.035	0.038	0.013	0.013	0.195	0.645	0.675	0.998	1.034	1.093	25.38
10	0.076	0.076	0.007	0.007	0.000	0.405	0.641	0.987	1.056	1.124	3.099
100	0.863	0.860	0.004	0.004	0.054	0.389	0.558	1.708	2.641	2.893	23.27
1000	2.089	2.085	0.002	0.002	0.051	0.398	0.614	3.002	5.473	6.665	61.02

TABLE 7. Performance profiles of verifiable PINNs with 50-50 and 600-600-600 architectures for various λ values at epoch 1000 with the $L_{\text{phy,verifiable}}$ loss function using 700k training and 150k validation sets.

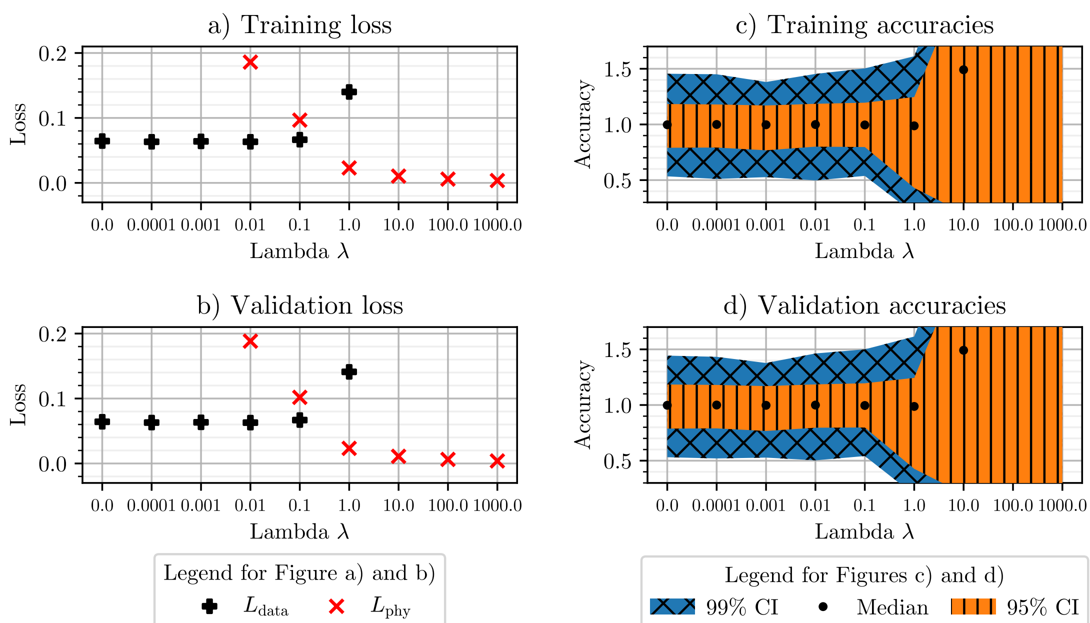

Fig. 8. Visual performance profiles for verifiable 50-50 architecture at epoch 1000 with various $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$ loss function using 700k training and 150k validation sets.

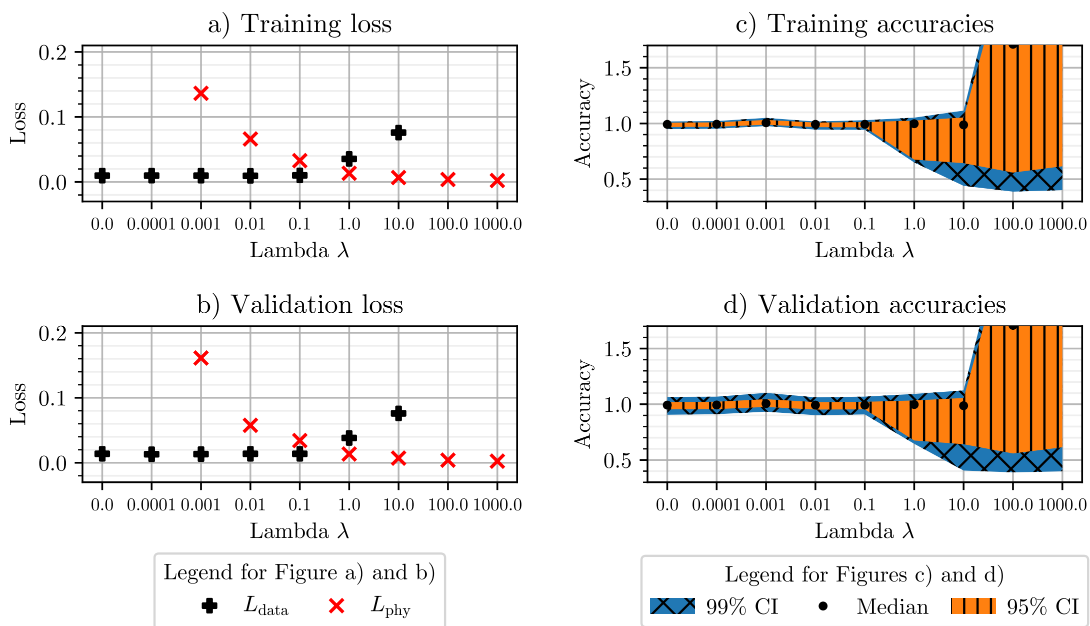

Fig. 9. Visual performance profiles for verifiable 600-600-600 architecture at epoch 1000 with various $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$ loss function using 700k training and 150k validation sets.

Performance comparison for training, validation and testing data for $\lambda = 0.1$ models with full training graphs for verifiable architectures

Similarly to before, the best performing PINN in terms of balancing physics and data losses occurred close to $\lambda = 0.1$ values for both the 50-50 and 600-600-600 verifiable architectures. The full performance profile of a data-only architecture ($\lambda = 0$) and the verifiable architecture with $\lambda = 0.1$ is shown in Table 8, with performance profiles shown in Figure 10 and 11.

Model	Dataset	L_{data}	Accuracy Percentiles						
			Min	0.5%	2.5%	Median	97.5%	99.5%	Max
Verifiable 50-50 model with $\lambda = 0.0$	Train.	0.064	0.023	0.534	0.791	0.997	1.183	1.455	21.83
	Val.	0.064	0.026	0.531	0.788	0.997	1.185	1.444	25.09
	Test.	0.064	0.025	0.532	0.792	0.997	1.185	1.470	18.63
Verifiable 50-50 model with $\lambda = 0.1$	Train.	0.067	0.035	0.538	0.797	0.996	1.195	1.503	19.059
	Val.	0.067	0.034	0.542	0.798	0.996	1.195	1.500	47.621
	Test.	0.066	0.037	0.539	0.796	0.995	1.198	1.510	21.85
Verifiable 600-600-600 model with $\lambda = 0.0$	Train.	0.010	0.548	0.949	0.968	0.992	1.008	1.018	1.809
	Val.	0.014	0.272	0.906	0.954	0.992	1.020	1.066	12.42
	Test.	0.013	0.248	0.907	0.954	0.992	1.020	1.066	3.069
Verifiable 600-600-600 model with $\lambda = 0.1$	Train.	0.010	0.639	0.943	0.965	0.992	1.013	1.029	1.345
	Val.	0.014	0.200	0.909	0.954	0.993	1.022	1.069	14.73
	Test.	0.013	0.199	0.908	0.954	0.993	1.022	1.070	5.232

TABLE 8. Loss and accuracy profiles for 600-600-600 $a_{\text{in,ReLU}}$ and $a_{\text{out,exp}}$ network at epoch 1000 with $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$, 700k training, 150k validation sets and 150k testing set.

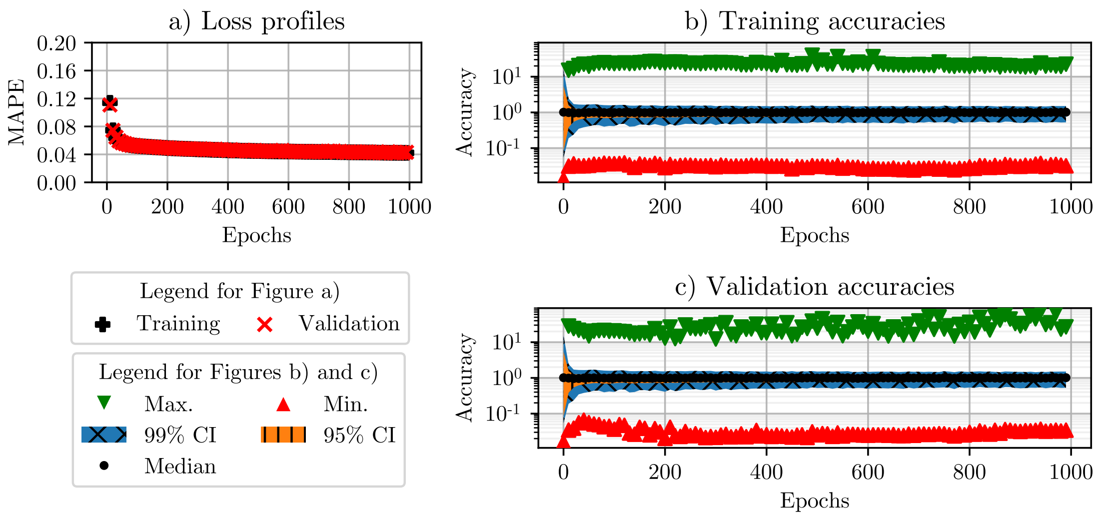

Fig. 10. Visual performance profiles for verifiable 50-50 architecture at epoch 1000 with $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$ loss function combinations using 700k training and 150k validation sets with $\lambda = 0.1$.

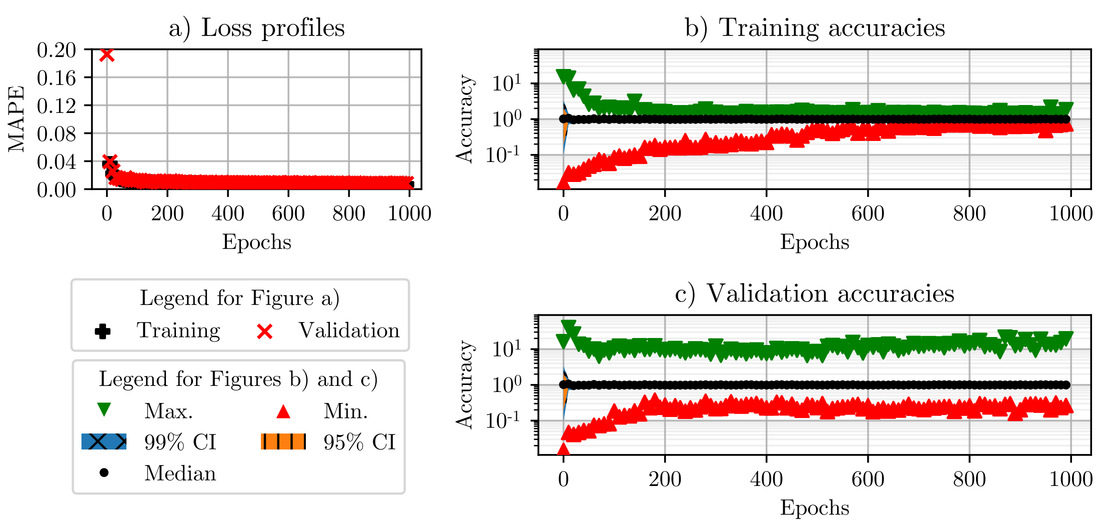

Fig. 11. Visual performance profiles for verifiable 600-600-600 architecture at epoch 1000 with $L_{\text{data}} + \lambda L_{\text{phy,verifiable}}$ loss function combinations using 700k training and 150k validation sets with $\lambda = 0.1$.

DISCUSSION

Identifying appropriate λ values

One key theme across all networks trained is that the performance of the PINNs tend to deteriorate when λ exceeds a value of 1. This pattern was clearly observable in Figure 3 across most physics loss functions, including $L_{\frac{\text{phy}-\text{true}}{\text{true}}}$ and $L_{\frac{\text{pred}-\text{phy}}{\text{phy}}}$, demonstrating this was not an issue related to the $L_{\frac{\text{phy}-\text{pred}}{\text{pred}}}$ or $L_{\text{phy,verifiable}}$ loss equations identified by Equations 13 and 19. The deterioration in performance was identified in both the standard and verifiable architectures, however with slightly different behaviour. In both cases, values with $\lambda > 1$ resulted in a broadening of the accuracy percentiles, which is representative of worse performance. However, whilst both L_{data} and L_{phy} losses worsened with increasing values of λ for standard architectures as shown in Figure 6 and 7, increasing λ only increased the L_{data} errors, yet decreased the physics-based $L_{\text{phy,verifiable}}$ errors for the verifiable architectures.

This behaviour may be attributable to the fact that the physics loss function used to constrain this structural design problem is highly ill-posed resulting from significant non-uniqueness, which is a core issue of inverse problems in general (Gallet et al. 2022). When large λ values are used to constrain this structural design problem, the increased relative magnitude of the L_{phy} error is likely to encourage the neural network parameters to find the closest viable physics solution to the problem, without necessarily finding the one identified by L_{data} that maintains high fidelity with respect to the true data. This suggests that the L_{data} term is critical in "nudging" the solution to the correct answer. Alternatively, more constraints might be required (which did exist in the original data generation stage of the CBeamXP dataset, namely in terms of class of cross-sections allowed and strict dimensional ratios between the flanges and webs of the I-sections). This highlights the central theme of inverse structural design problems: that physical models alone provide only partial prior information.

This would also ultimately explain why the PINNs with $\lambda = 0.1$ had in general the best performance, since this approached the limit where the physics losses L_{phy} informed the neural network slightly towards physically realistic predictions without dominating the loss magnitude to the extent that it would completely diverge from the true data-point. This is both suggestive of the limits of using a local stiffness based formulation, but also helps to identify avenues for improvement, by providing either more constraints or perhaps moving towards a global formulation.

Compared to literature, it is worth noting that since the majority of previously identified literature has predominantly dealt with data-less PINN formulations (Jeong et al. 2023), (Mai et al. 2024) and (Song et al. 2023) and therefore did not face an issue with choosing an appropriate λ value. The only example of a PINN based ML inverse operator that did use λ values (Zhang et al. 2022) tested a completely different PINN formulation based on a data term that is combined with residual of equilibrium partial differential equations and boundary condition constraints, with λ fixed for the PDE term at a value of one and varied between $\lambda = 1, 3, 10$ for the boundary conditions. The impact of varying λ terms in that particular research was minimal. Further research is required to therefore understand the a correct choice for the

λ value for PINNs based on the local stiffness relation presented here.

Performance improvements of standard architectures

One of the fundamental aims of this investigation was to evaluate if the proposed PINN formulation improves the performance of ML structural design models when compared against its data-driven only counterpart (that is when $\lambda = 0$). The results based on the standard architecture with $\lambda = 0.1$ shown in Table 5 indicate that the local formulation only decreases the L_{data} error (which is equivalent to the MAPE error reported in (Gallet et al. 2024b)) slightly for the standard 50-50 architecture from 6.6% to 5.4% for the testing dataset. Whilst there are also improvements observed in the accuracy percentile values, these improvements are not as large as anticipated.

Furthermore, for the larger standard 600-600-600 architecture, providing the physical constraints actually deteriorated both the L_{data} and accuracy bounds, with only slight improvements measured at the 97.5% and 99.5% percentile bounds. This is also reflective in the testing dataset when subjected to the decile analysis conducted in the previous data-driven work as shown in Figure 12 below. Whilst there is some improvement in terms of the standard deviation of standard deviations $\sigma(\sigma_{\text{Deciles}})$, it is still the largest for data within the first decile of total load $\omega_0 \times L_0$. The heatmap shown in Figure 13 reveals a similar distribution of errors when compared to (Gallet et al. 2024b).

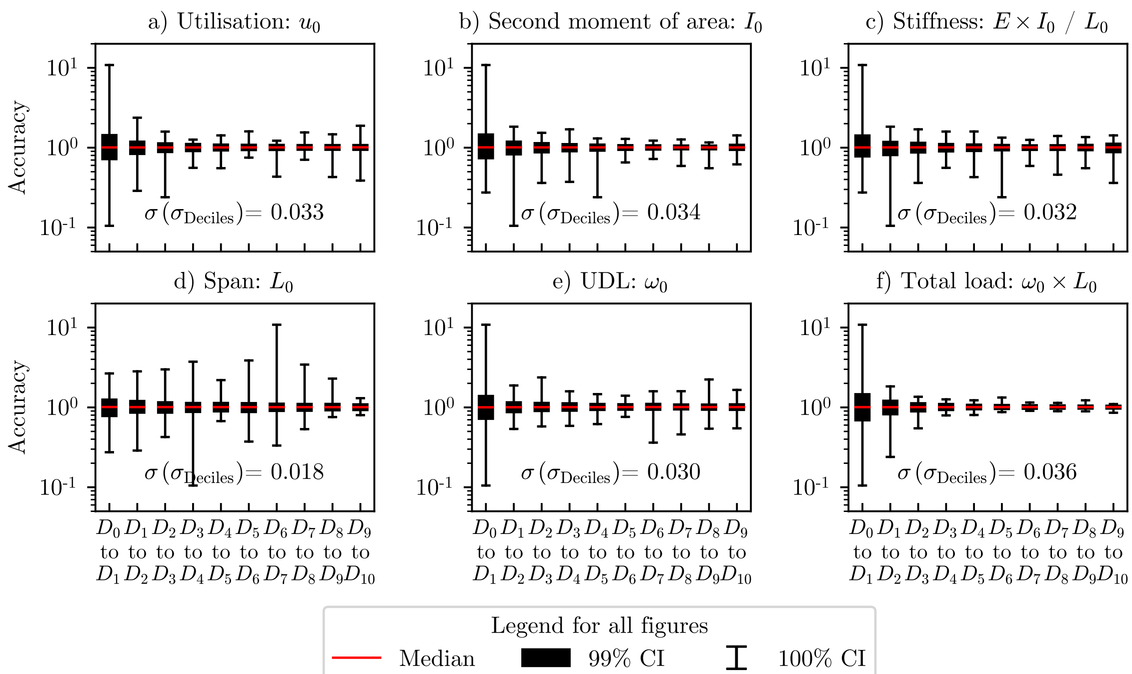

Fig. 12. Decile analysis of 150k testing dataset with the standard 600-600-600 architecture at epoch 1000 with $L_{\text{data}} + L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function and $\lambda = 0.1$.

Whilst the improvement of the accuracy bounds for the standard 50-50 architecture is an encouraging sign, it is clear that supplementing physics constraints to a previously designed architecture is not by itself sufficient to drastically

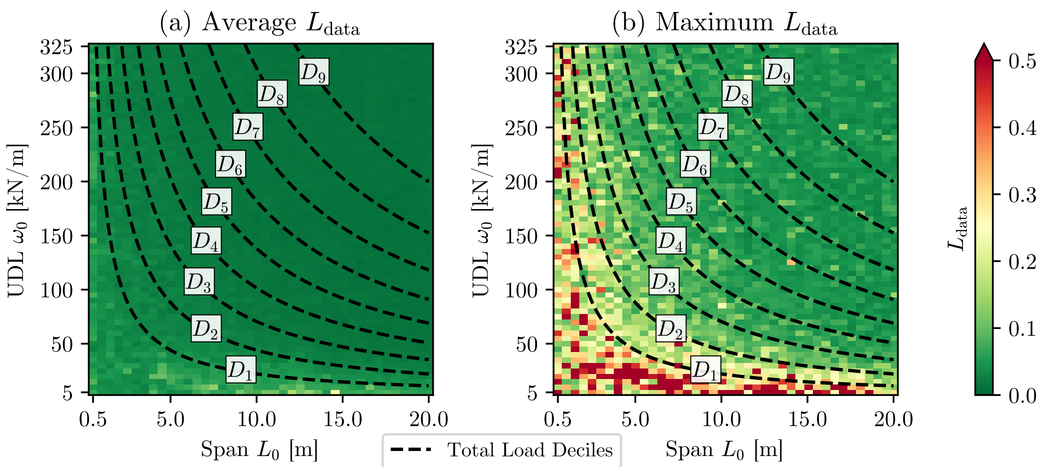

Fig. 13. Heatmap for 150k testing dataset performance with standard 600-600-600 architecture at epoch 1000 with $L_{\text{data}} + L_{\frac{\text{phy-pred}}{\text{pred}}}$ loss function and $\lambda = 0.1$.

improve its performance. This may perhaps be another indication of the limitations of using local stiffness relations only to physically inform the network. It is possible that a global formulation is instead desirable, especially when considering the fact that the influence zone of such a system has been shown to be up to $k_{\max} = 5$ (Gallet et al. 2024a). Global stiffness relations that reflect the adjacent stiffness information of the system may, resultingly, constrain the solution space better, and thereby possibly help address the error dispersions associated with low total load values.

Viability of verifiable architectures

The second aim of this investigation was to propose a verifiable PINN architecture that would allow a user to calculate the physical consistency of a prediction based on the local stiffness relations at inference. Similar to the standard architectures, the best performing PINNs based on the verifiable architecture was with $\lambda = 0.1$ as clearly demonstrated by Figures 8 and 9. However, there was no performance improvement (although no deterioration either) in terms of data error L_{data} or accuracy profiles with $\lambda = 0.1$ compared to $\lambda = 0$ as shown in Table 8. This is further evidence that the local stiffness relations may be non-optimal for the problem at hand.

Since the verifiable architecture predicts all of the required structural information including utilisation ratios and the rotations at the support of each beam, it is possible to evaluate the difference between the predicted cross-sectional properties of the network with the physics-derived ones via Equation 14. This means that some insight on how physically consistent a prediction can now be evaluated for *any* input. Whilst the level of physical consistency is of interest, in an ideal scenario, physical consistency should also correlate with lower prediction error. To this end, the correlation between the 150k training L_{data} and $L_{\text{phy,verifiable}}$ losses for each verifiable 50-50 and 600-600-600 architecture and for all λ values tested were evaluated, with results of this analysis shown in Figure 16.

Figure 16 indicates that while the correlation between these two sets of errors grows as λ approaches a value of 1, it never exceeds a coefficient of determination of $R^2 = 0.12$. The fact that the correlation does increase within this range is encouraging. Additional trials and a increase range of λ values could result in lower error margins. However, overall, increasing λ values decreases $L_{\text{phy,verifiable}}$ at the cost of increasing L_{data} losses. For similar reasons as explained before, this may simply indicate the limitations of the local stiffness relation employed here.

Finally, a decile analysis of the error dispersion for the verifiable 600-600-600 architecture with $\lambda = 0.1$ when tested against the 150k testing dataset is shown in Figure 14. Interestingly, a decrease in dispersion of dispersions $\sigma(\sigma_{\text{Deciles}})$ is identified here, with a maximum value of $\sigma(\sigma_{\text{Deciles}}) = 0.025$, which is less than from the standard architectures as shown in Figure 12 and when compared to the previous investigation as presented in (Gallet et al. 2024b). Whilst the lowest total load combination $\omega_0 \times L_0$ still contains the largest errors, the verifiable architecture does appear to have clearly contributed to reducing the maximum L_{data} errors when compared to that achieved in (Gallet et al. 2024b).

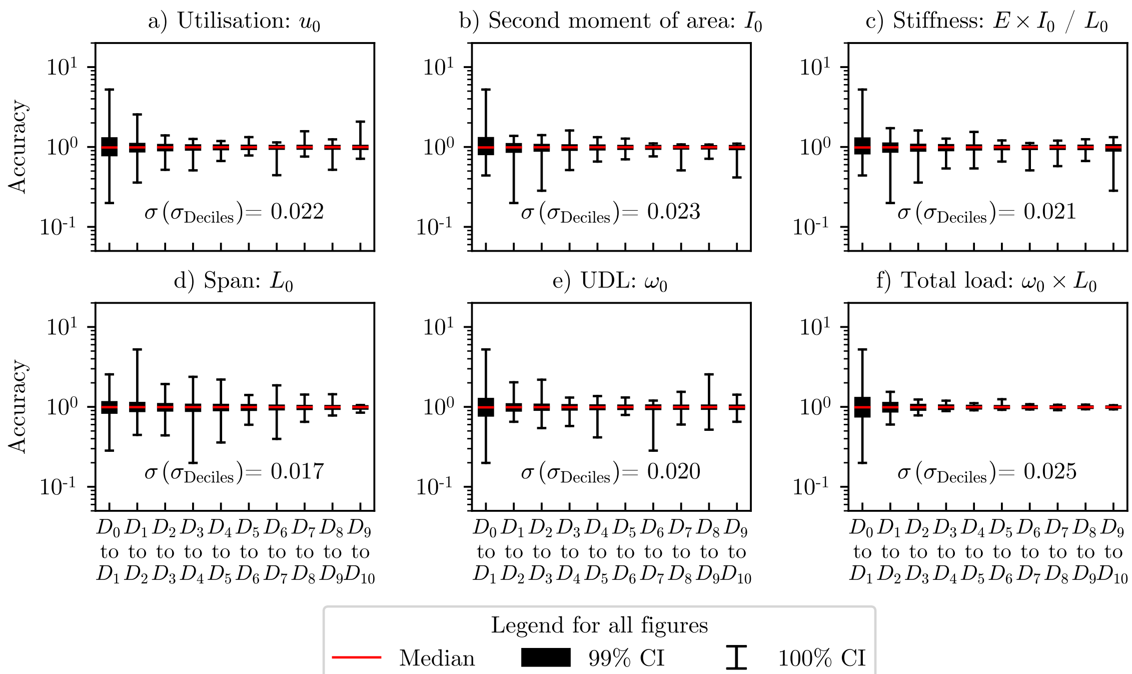

Fig. 14. Decile analysis of 150k testing dataset with the verifiable 600-600-600 architecture at epoch 1000 with $L_{\text{data}} + L_{\text{phy,verifiable}}$ loss function and $\lambda = 0.1$.

Limitations and scope for future works

PINNs based on the local stiffness formulation appear to generate only a small increase in performance when compared their data-only equivalents. The discussion has already identified that the existence of non-unique solutions necessitates the data-term unlike previous investigations in literature with data-less formulations (Jeong et al. 2023), (Mai et al. 2024) and (Song et al. 2023). Furthermore, the local stiffness formulation here does not constrain the solution based on the spans and UDL load values from adjacent members, which with knowledge of the influence zone size of the system $k_{\max} = 5$ would point towards the need for a global formulation instead. This is identified as a likely

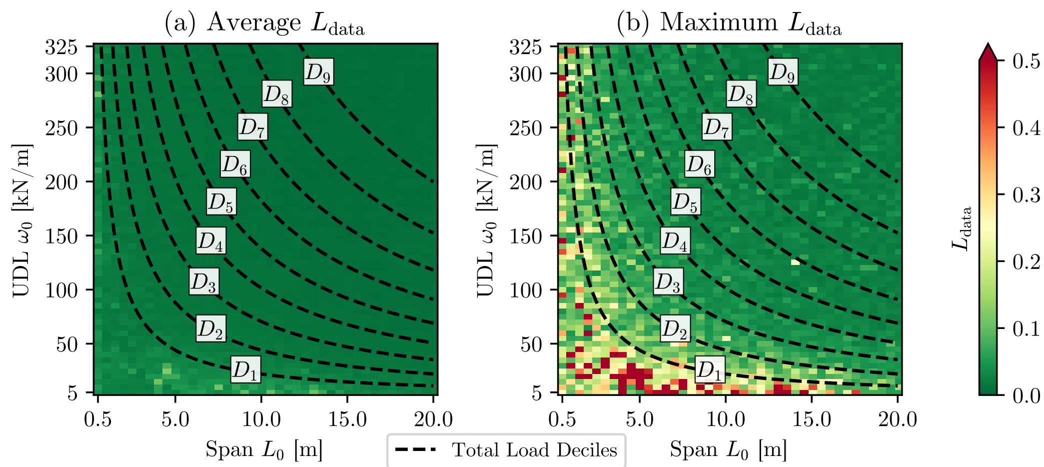

Fig. 15. Heatmap for 150k testing dataset performance with verifiable 600-600-600 architecture at epoch 1000 with $L_{\text{data}} + L_{\text{phy,verifiable}}$ loss function and $\lambda = 0.1$.

scope for future works.

Another limitation of the methodology employed here is that the rotational and utilisation ratio information is evaluated for the full, non-patterned load arrangement, which is never the critical load arrangement. In this investigation, this was chosen to avoid having to potentially factor end-beam reaction forces based on whether the critical load arrangement had an active UDL on the member under investigation, however future studies could investigate if a proper acknowledgement of the critical load arrangement, especially for potential global stiffness relations, lead to increased performance in the system. Furthermore, if a successful PINN formulation, specifically one in which physics losses correlate with data losses can be identified, then the investigation should be expanded to different structural systems to evaluate the versatility of the methodology employed here.

It is possible that the dataset is already sufficiently balanced and well-spaced to negate the need for a PINN formulation. Previous research suggest that PINNs excel in data-scarce environments, specifically when needing to interpolate across large data gaps (Pannell et al. 2022). The extent to which this similar behaviour arises within PINNs developed for ill-posed problems remains to be investigated. Lastly, it is worth identifying that PINNs also carry an associated increased computational cost. The total computational cost of this study was just over 13 days as shown by Table 9. When compared to data-only structural design models, a PINN formulation will only be worth the additional computational cost if it yields to significantly improved results.

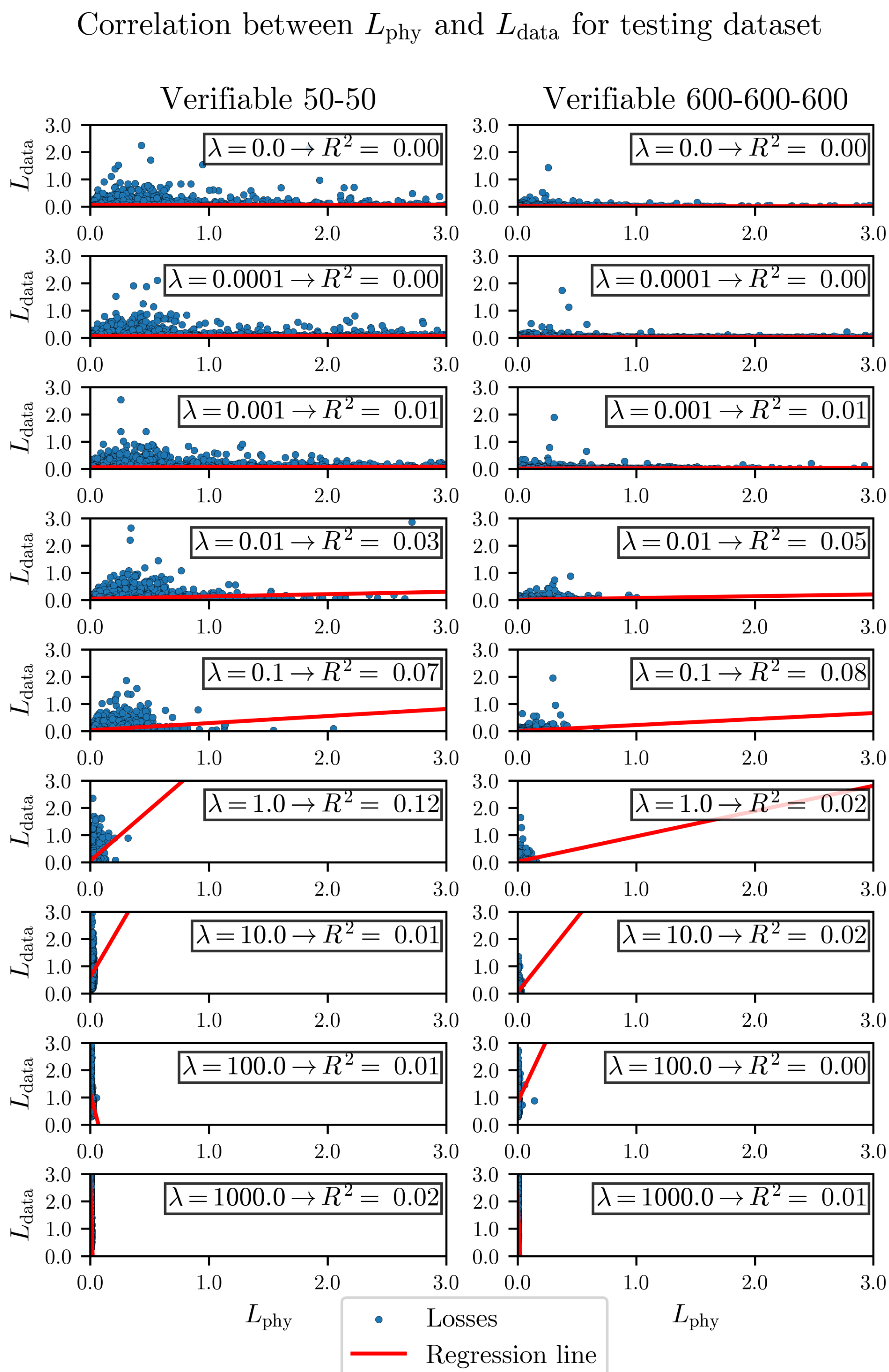

Fig. 16. Coefficients of determinations between L_{data} and $L_{\text{phy,verifiable}}$ for various verifiable architectures and λ values.

Section	Stage	Computation time		Proportion [%]
		In hours [h]	In days [D]	
Section 3	All standard L_{phy} variations	58.54	2.44	18.7
Section 3	with 10k training dataset	5.64	0.23	
Section 3	with 100k training dataset	9.93	0.41	
Section 3	with 700k training dataset	42.97	1.79	
Section 3	All standard architecture training	88.14	3.67	28.1
Section 3	Single 50-50 training run	1.25	0.05	
Section 3	Single 600-600-600 training run	8.55	0.36	
Section 3	All verifiable architecture training	166.79	6.95	53.2
Section 3	Single 50-50 training run	1.30	0.05	
Section 3	Single 600-600-600 training run	17.54	0.73	
Total computation time (Sections 3, 3, 3):		313.47	13.06	100

TABLE 9. Computation time for each physics informed neural network development stage.

CONCLUSIONS

This paper aimed to first introduce various physics informed neural network formulations based on local stiffness relations with two explicit aims. The second aim was to investigate if PINNs are suitable to create structural design models for continuous beam systems with lower prediction errors and error dispersions than of those achieved from data-only models. Whilst the PINNs developed in this investigation did reduce such errors slightly, they were highly sensitive to the λ hyperparameter which controlled the extent to which the physical constraints were enforced during network training. The third aim was to test a structural design model in which design predictions could be verified during inference, which would allow prediction verifications including for unknown inputs. Whilst such a neural network was successfully trained, the physics losses only had a weak correlation with data losses. Several avenues of future works were identified, the most notable being to develop an equivalent PINN with global stiffness relations.

Data Availability Statement

Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgements

AG would like to acknowledge the support of the Engineering & Physical Sciences Research Council. DS would like to acknowledge the support from Georgia Tech and the Air Force Faculty Fellowship Program.

REFERENCES

European Committee for Standardisation (2015). *BS EN 1993-1-1:2005+A1:2014 Eurocode 3. Design of Steel Structures. General Rules and Rules for Buildings*. BSI British Standards.

Gallet, A., Liew, A., Hajirasouliha, I., and Smyl, D. (2024a). “Influence zones of continuous beam systems.” *Structures*, 68, 107069.

Gallet, A., Liew, A., Hajirasouliha, I., and Smyl, D. J. (2024b). “Machine learning for structural design models of continuous beam systems via influence zones.” *Inverse Problems*.

Gallet, A., Rigby, S., Tallman, T. N., Kong, X., Hajirasouliha, I., Liew, A., Liu, D., Chen, L., Hauptmann, A., and Smyl, D. (2022). “Structural engineering from an inverse problems perspective.” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 478(2257), 20210526.

Gallet, A. and Smyl, D. (2023). “CBeamXP: Continuous Beam Cross-section Predictors dataset, <https://figshare.com/s/c84c620925c1c36e648c>.

Hu, Y., Guo, W., Long, Y., Li, S., et al. (2022). “Physics-informed deep neural networks for simulating s-shaped steel dampers.” *Computers & Structures*, 267, 106798.

Jeong, H., Bai, JS., Batuwatta-Gamage, CP., Rathnayaka, C., Zhou, Y., and Gu, YT. (2023). “A Physics-Informed Neural Network-based Topology Optimization (PINNTO) framework for structural optimization.” *Engineering Structures*, 278.

Kabir, H. and Garg, N. (2023). “Machine learning enabled orthogonal camera goniometry for accurate and robust contact angle measurements.” *Scientific reports*, 13(1), 1497.

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.

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., and Kumar, V. (2017). “Theory-guided data science: A new paradigm for scientific discovery from data.” *IEEE Transactions on knowledge and data engineering*, 29(10), 2318–2331.

Kourehpaz, P. and Molina Hutt, C. (2022). “Machine learning for enhanced regional seismic risk assessments.” *Journal of Structural Engineering*, 148(9), 04022126.

Mai, H. T., Mai, D. D., Kang, J., Lee, J., and Lee, J. (2024). “Physics-informed neural energy-force network: a unified solver-free numerical simulation for structural optimization.” *Engineering with Computers*, 40(1), 147–170.

McClenny, L. D. and Braga-Neto, U. M. (2023). “Self-adaptive physics-informed neural networks.” *Journal of Computational Physics*, 474, 111722.

Muralidhar, N., Islam, M. R., Marwah, M., Karpatne, A., and Ramakrishnan, N. (2018). “Incorporating prior domain knowledge into deep neural networks.” *2018 IEEE international conference on big data (big data)*, IEEE, 36–45.

Pak, H., Leach, S., Yoon, S. H., and Paal, S. G. (2023). “A knowledge transfer enhanced ensemble approach to predict the shear capacity of reinforced concrete deep beams without stirrups.” *Computer-Aided Civil and Infrastructure Engineering*, 38(11), 1520–1535.

Pannell, J. J., Rigby, S. E., and Panoutsos, G. (2022). “Physics-informed regularisation procedure in neural networks: An application in blast protection engineering.” *International Journal of Protective Structures*, 13(3), 555–578.

Preisinger, C. (2013). “Linking Structure and Parametric Geometry.” *Architectural Design*, 83(2), 110–113.

Raissi, M., Perdikaris, P., and Karniadakis, G. (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.

Robert McNeel & Associates (2021-06-09). “Grasshopper. Version 1.0.0007.

Song, L.-H., Wang, C., Fan, J.-S., and Lu, H.-M. (2023). “Elastic structural analysis based on graph neural network without labeled data.” *Computer-Aided Civil and Infrastructure Engineering*, 38(10), 1307–1323.

Zhang, E., Dao, M., Karniadakis, G. E., and Suresh, S. (2022). “Analyses of internal structures and defects in materials using physics-informed neural networks.” *Science Advances*, 8(7).