Journal of Nonlinear Modeling and Analysis
Volume 6, Number 2, June 2024, 485–513
http://jnma-online.com
DOI:10.12150/jnma.2024.485

# Meta-Learning-Based Physics-Informed Neural Network: Numerical Simulations of Initial Value Problems of Nonlinear Dynamical Systems Without Labeled Data and Correlation Analyses

Worrawat Duanyai[1], Weon Keun Song[2,†], Thanadol Chitthamlerd[3] and Girish Kumar[4]

**Abstract** There are several main challenges in solving nonlinear differential equations with artificial neural networks (ANNs), such as a nonlinear system's sensitivity to its initial values, discretization, and strategies for incorporating physics-based information into ANNs. As for the first issue, this paper addresses the initial value problems of nonlinear dynamical systems (a Duffing oscillator and a Burger's equation), which cause large global truncation errors in sub-domains with a significant reduction in the influence of initial constraints, using meta-learning-based physics-informed neural networks (MPINNs). The MPINNs with dual learners outperform physics-informed neural networks with a single learner (no fine reinitialization capability). As a result, the former approach improves solution convergence by 98.83% in the sub-time domain (III) of a Duffing oscillator, and by 85.89% at $t = 45$ in a Burger's equation problem, compared to the latter one. Model accuracy is highly dependent on the adaptability of the initial parameters in the first hidden layers of the meta-models. From correlation analyses, it is obvious that the parameters become less (the Duffing oscillator) or more (the Burger's equation) correlated during fine reinitialization, as the update manner differs or is similar to the one used in pre-initialization. In the first example, the MPINN achieves both the mitigation of model sensitivity to its output and the improvement of model accuracy. Conversely, the second example shows that the proposed approach is not enough to solve both issues simultaneously, as increased model sensitivity to its output leads to higher model accuracy. The application of transfer learning reduces the number of iterative pre-meta-trainings.

[†]the corresponding author.
Email address: worrawatduanyai@gmail.com(W. Duanyai), bauman98@naver.com(W K. Song), thanadolps@gmail.com(T. Chitthamlerd), girish.kumar154@gmail.com(G. Kumar)
[1]Department of Robotic and Computational Intelligence System, School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Chalong Krung 1 Rd.,10520 Bangkok, Thailand
[2]Department of Robotic and AI Engineering, School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Chalong Krung 1, 10520 Bangkok, Thailand
[3]Department of Computer Engineering, Chulalongkorn University, Phaya Thai Rd, Wang Mai, 10330 Bangkok, Thailand
[4]Department of Mechanical Engineering, Delhi Technological University, Shahbad Daulatpur, Bawana Road, 110042 Delhi, India

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

**Keywords** Physics-informed neural network, meta-learning, fine reinitialization, transfer learning, initial value problem, nonlinear dynamical systems, correlation analyses

**MSC(2010)** 65L99, 65M99.

**Nomenclature**

| Symbol | Definition |
|---|---|
| $\boldsymbol{a}^{[l]}$, $a_n^{[l]}$ | Output vector of the $l$-th hidden layer and the $n$-th component of $\boldsymbol{a}^{[l]}$ |
| $\boldsymbol{b}^{[l]}$ | Bias vector of the $l$-th hidden layer |
| $c$ | Constant parameter |
| $cn, dn, sn$ | Jacobi elliptic functions |
| $d$ | Constant parameter |
| $\boldsymbol{d}_k$ | Search direction vector at the $k$-th iteration |
| $\boldsymbol{D}_{qu}^{(i)}$, $\boldsymbol{D}_{su}^{(i)}$ | Query and support datasets, respectively, for the $i$-th query task and the $i$-th support one. |
| $\boldsymbol{D}_{te}, \boldsymbol{D}_{tr}$ | Test and training datasets, respectively |
| $f$ | Constant parameter |
| $f(\boldsymbol{u})$ | Function value at $\boldsymbol{u}$ |
| $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ | PINN |
| $f_{\boldsymbol{\theta}}^m(\boldsymbol{x})$ | MPINN |
| $f_{\boldsymbol{\theta}^{(i)}}^m(\boldsymbol{x}_{su}^{(i)})$ | A model for the $i$-th support task (it is regarded as the $i$-th support task) |
| $f_{\boldsymbol{\theta}^{(i)},'}^m(\boldsymbol{x}_{qu}^{(i)})$ | A model for the $i$-th query task (it is regarded as the $i$-th query task) |
| $f_{\boldsymbol{\theta}^*}^m(\boldsymbol{x}_{te})$ | MPINN evaluated on $\boldsymbol{D}_{te}$ |
| $\boldsymbol{g}_k$ | Gradient difference vector at the $k$-th iteration |
| $\boldsymbol{H}^{(I)}$, $\boldsymbol{H}_k$ | Initial Hessian matrix and Hessian matrix at the $k$-th iteration |
| $iter$, $iter^m$ | Iterative numbers, respectively, for the PINN (or each inner learner of the MPINN) and the outer learner of the MPINN |
| $iter^{MAX}$, $iter^{mMAX}$ | Maximum iterative numbers, respectively, for the PINN (or each inner learner of the MPINN) and the outer learner of the MPINN. |
| $k$ | Constant parameter |
| $L(\boldsymbol{\theta},\ \boldsymbol{D}_{tr})$ | Total PINN loss |
| $L^{(i)}(\boldsymbol{\theta}^{(i)}, \boldsymbol{D}_{su}^{(i)})$, $L^{(i)}(\boldsymbol{\theta}^{(i),'}, \boldsymbol{D}_{qu}^{(i)})$ | Support and query losses, respectively, for the $i$-th task |
| $\|\|\nabla L(\boldsymbol{\theta}, \boldsymbol{D}_{tr})\|\|$ | Norm of the projected gradient of $L(\boldsymbol{\theta}, \boldsymbol{D}_{tr})$ |
| $m$ | Constant parameter |
| $MSE_f, MSE_I$, $MSE_B$, $MSE_d$ | Losses, respectively, of residual, initial constraint, boundary constraint, and labeled data |
| $N_d, N_B, N_f, N_I$ | Number of labeled data for $MSE_d$ and number of training samples, respectively, for $MSE_B$, $MSE_f$ and $MSE_I$ |
| $NT$ | Number of test samples |
| $NQ$, $NS$ | Numbers, respectively, of query and support tasks |
| $t$, $\triangle t$ | Time coordinate and time increment |
| $u$, $u_n$ | A single output of $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ and a discrete solution of u at the $n$-th sampling point |
| $\boldsymbol{u}$ | Output vector of $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ |
| $\bar{u}$, $\bar{u}_n$ | Analytical (exact) solution and a discrete solution of $\bar{u}$ at the $n$-th sampling point |
| $\boldsymbol{u}_{su}^{(i)}$, $\boldsymbol{u}_{qu}^{(i),'}$, $\boldsymbol{u}_{te}$ | Output vectors, respectively, evaluated by $f_{\boldsymbol{\theta}^{(i)}}^m(\boldsymbol{x}_{su}^{(i)})$, $f_{\boldsymbol{\theta}^{(i)},'}^m(\boldsymbol{x}_{qu}^{(i)})$, and $f_{\boldsymbol{\theta}^*}^m(\boldsymbol{x}_{te})$ |
| $\bar{\boldsymbol{u}}_{su}^{(i)}$, $\bar{\boldsymbol{u}}_{qu}^{(i)}$, $\bar{\boldsymbol{u}}_{te}$ | Actual output vectors, respectively, for $\boldsymbol{D}_{su}^{(i)}$, $\boldsymbol{D}_{qu}^{(i)}$, and $\boldsymbol{D}_{te}$ ($\bar{\boldsymbol{u}}_{su}^{(i)}$ and $\bar{\boldsymbol{u}}_{qu}^{(i)}$ require no labeled data) |
| $D_{te}^n$, $\partial_x^n$, $\partial_t^n$ | $n$-th order differential operators |
| $x$ | Space coordinate |
| $\boldsymbol{x}$ | Input vector |

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

**Nomenclature**

| Symbol | Definition |
|---|---|
| $\boldsymbol{x}_{su}^{(i)}$, $\boldsymbol{x}_{qu}^{(i)}$, $\boldsymbol{x}_{te}$ | Input vectors, respectively, for $\boldsymbol{D}_{su}^{(i)}$, $\boldsymbol{D}_{qu}^{(i)}$, and $\boldsymbol{D}_{te}$ |
| $\boldsymbol{z}^{[l]}$, $z_n^{[l]}$ | Output vector of the l-th hidden layer before taking an activation and the $n$-th component of $\boldsymbol{z}^{[l]}$. |
| $\alpha$ | Scalar |
| $\alpha_k$ | Iterative step size |
| $\beta$ | Learnable parameter for swish |
| $\varepsilon$, $\varepsilon^m$, $\varepsilon^H$ | Convergence tolerances, respectively, for the PINN (or each inner learner of the MPINN), the outer learner of the MPINN, and loss reduction tolerance |
| $\boldsymbol{\theta}$ | Parameter matrix of the PINN and MPINN |
| $\boldsymbol{\theta}_k$, $\boldsymbol{\theta}_{mk}^{(i),\prime}$ | Parameter matrices updated, respectively, at the $k$-th iteration by the PINN (or each inner learner of the MPINN) and at the $mk$-th meta-iteration by the outer learner of the MPINN |
| $\boldsymbol{\theta}^{(I)}$ | Initial parameter matrix |
| $\boldsymbol{\theta}^{(i)}$ | Parameter matrix trained on $\boldsymbol{D}_{su}^{(i)}$ |
| $\boldsymbol{\theta}^{(i),\prime}$ | Parameter matrix trained on $\boldsymbol{D}_{qu}^{(i)}$ |
| $\boldsymbol{\theta}^{[l-1]}$ | Parameter matrix that maps from $\boldsymbol{a}^{[l-1]}$ to $\boldsymbol{z}^{[l]}$ |
| $\theta_{j,i}^{[l-1]}$ | Parameter that controls a function mapping from $a_i^{[l-1]}$ to $z_j^{[l]}$ |
| $\boldsymbol{\theta}^*$ | Optimal parameter matrix achieved by the meta-training for a meta-test phase |
| $\lambda_f$, $\lambda_I$, $\lambda_B$, $\lambda_d$ | Parameters, respectively, for $MSE_f$, $MSE_I$, $MSE_B$, and $MSE_d$ |
| $\omega$ | Constant parameter |
| $\mu$ | Constant parameter |
| $\boldsymbol{\sigma}^{[l]}$ | Activation of the $l$-th hidden layer |

# 1. Introduction

Physical phenomena of dynamical systems are modeled by governing differential equations, and the analysis of their solutions provides insight into them. That is why it is critical to find precise solutions in science and engineering. Due to the absence of analytical solutions in many practical cases, numerical approaches such as finite difference, finite volume, and finite element methods have dominated over the last seven decades. Nonetheless, classical solvers have disadvantages that require a lot of computational efforts, especially for multi-scale or multi-physics-based nonlinear systems. Researchers continue to face challenges in numerical solver design with the balance between accuracy and robustness. Recently, various artificial neural network (ANN)-based solvers among many alternatives have gained large attention because of the prospect of replacing classical solvers. The advantages of ANNs for solving governing differential equations are as follows: (i) powerful modeling for interdisciplinary problems; (ii) simple modeling for strong nonlinear relationships; (iii) efficient forward evaluation of a trained model for real-time applications; (iv) automatic gradient-based optimization due to analytically differentiable modeling; (v) meshless modeling for discretization. ANNs have been applied to solve dynamical systems governed by not only ordinary differential equations (ODEs) [1–5] but partial differential equations (PDEs) [2, 6–19]. Consistent challenges [20–22] were reported for performance improvement during 1990 – 2000, but there was no significant progress. Raissi et al. [7] proposed a new conceptual neural network known as a physics-informed neural network (PINN) in 2019. They developed a physics-informed fully connected neural network (PIFCNN) with high

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

model accuracy, where a PDE residual loss was incorporated into data-driven functional representation as one of constraint conditions, in small data regimes. Their success in the research has led many researchers to explore PINNs. There are several types of neural networks in modeling PINNs, and these two are among them: a PIFCNN [2, 3, 5–11, 13, 15–17, 19, 23] and a physics-informed convolutional neural network (PICNN) [4, 8, 12, 14]. Several studies have revealed that PIFCNNs perform well with small data [4, 7, 11] or even in the absence of data [8, 9, 12]. A literature review by us supports the claim that PINNs have addressed various problems in engineering, science, and mathematics: structural dynamics [1], nonlinear dynamical systems in applied mathematics [2, 3, 6, 10, 12–14, 23], seismic response [4], quantum mechanics, reaction-diffusion, and propagation of water wave [7], flow [7–9, 11, 16], thermal-transfer related problems [15, 18], composite material [17]. Studies on differential equations have been mostly engaged in boundary value problems (BVPs) [2, 8, 13–15] and initial boundary value problems (IBVPs) [2, 3, 6, 7, 9–12, 16–19]. When selecting a research subject, the complexity of boundary conditions can motivate researchers to choose it as a highly challenging topic. However, in many cases, solution convergence with some degree of accuracy or even higher accuracy can be achieved by adding intricate boundary constraints without labeled data. While initial value problems (IVPs) [3, 5, 23] have received less attention despite their poor solution convergence in sub-domains with a sharp decline in the influence of initial constraints. To address this fundamental issue, several studies introduced noticeable but limited approaches. Meng et al. [3] proposed a parallel PINN to decompose a long-time domain into multiple independent sub-domains with shorter time interval. This strategy resulted in faster convergence and reasonable solution predictions. However, the optimal number of sub-domains to be applied, depending on the characteristics of dynamic systems, was not discussed. The study by Florio et al. [5] highlighted the combination of a shallow PINN and the theory of functional connections and an extreme learning process. They succeeded in solving the stiff systems of ODEs by the proposed approach without reducing the stiffness of the problems. Nonetheless, the use of labeled data was exposed as a crucial limitation because of expensive data generation. Robinson et al. [23] used the injection of partially known physical information at an intermediate layer in a PINN during training for solving benchmark problems in the literature including oscillations. Despite the focus, the relationship between the injection and the nature of models has remained inconclusive. This implies that discovering a universal method for diverse dynamical systems is a huge challenging task. This study represents the first application of both meta-learning and transfer learning, which involve the fine initialization of parameters, to PINNs for solving the IVPs of a nonlinear Duffing oscillator and a nonlinear Burger's equation without labeled data. A Duffing oscillator is a type of nonlinear second-order ODE that describes the behavior of a damped-driven harmonic oscillator. It is a type of dynamical system that exhibits complex behavior, including chaos, resonance, and bifurcations due to its nonlinear nature. Applications include electronic circuits, mechanical systems, and biological systems, while Burgers' equation is a fundamental nonlinear PDE that governs fluid flow. It is used in various areas of fluid dynamics, including the study of turbulence, shock waves, and nonlinear wave propagation. Additionally, it has applications in other fields, such as traffic flow modeling and nonlinear acoustics. In a broad sense, meta-learning may be extended to include the selection of a suitable algorithm by taking into consideration the features of

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

input datasets, exploring the topology of neural network architectures, and tuning a set of hyperparameters. In a narrow sense, it is the algorithm to fine-tune a set of model parameters. Our current study is limited to the latter, but we will extend it to the former. A meta-learning strategy for a new task was proposed by Peng et al. [24] who were one of the pioneers to discover a suitable learning algorithm. A typical strategy involves training existing models on a new target dataset and then selecting the model that performs best on it. Such a strategy needs big datasets to achieve adequate model accuracy. However, in many real circumstances, a lack of data would limit the ineffective traditional method to laboratory experiments. A new requirement in machine learning is to discover new algorithms to imitate how humans learn new tasks based on their prior knowledge. As humans learn new similar tasks based on it, a set of fine meta-tuned initial parameters is similar in function to it. This means that meta-learning-based algorithms can learn new analogous features rapidly and precisely with fewer data. Therefore, meta-learning is one of the strategies for how an ANN acquires useful prior knowledge by adjusting initial parameters. Learning rules of meta-learning were proposed by a series of studies in 1991 [25] and 1997 [26]. Hochreiter et al. [27] and Younger et al. [28] marked a turning point in gradient descent-based deep meta-learning that belongs to one of the three categories: metric-based, model-based, and optimization (gradient)-based techniques. Many trials by the third method update a meta-model using a bi-level optimization process with multi-inner learners for tuning support models and a single outer learner for meta-learning. Model-agnostic meta-learning (MAML), proposed by Finn et al. [29], showed the feasibility of an application of the dual learners to few-shot problems for classification, regression, and reinforcement learning using gradient-based backpropagation. Their application was successful in adapting a meta-model to new tasks. Several ideas [30–33] on meta-learning have been proposed to improve the training efficiency of PINNs. This paper discusses the model accuracy of classical PINNs and meta-learning-based PINNs (MPINNs) for the IVPs of both examples. Non-uniform random sampling (NURS) is applied to both in training. Evaluations on the convergence of the solutions to the problems are made across entire domains. Experimental data show that the MPINNs outperform the PINNs in tracking their solutions in all domains as well as in the sub-time domain (III) of the Duffing oscillator and at $t = 45$ in the Burger's equation problem. The meta-learning with the dual learners makes a large contribution to improving the solution convergence of the MPINNs, as it mostly updates the initial values of the first hidden layers of the meta-models. Correlation analyses assist us in gaining a deeper comprehension of how the initial parameters become adaptable to inputs and responsive to outputs throughout the meta-training, depending on the nature of the dynamical systems. The Duffing oscillator achieves both the mitigation of the MPINN's sensitivity to its output and the enhancement of the model accuracy, but regarding the Burger's equation, the proposed model is insufficient to address both issues simultaneously, as the model's sensitivity to its output increases during the meta-tuning. This means that it could be less robust. In the MPINN algorithm, the application of transfer learning to support tasks minimizes the number of iterative pre-meta-tunings.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

## 2. Problem statement and examples

Using PINNs for solving IVPs may result in significant truncation errors due to their limited capacity to accurately adjust initial parameters. Figure 3 and 4 show two of the cases. This paper employs the MPINNs to tackle the issue at hand for two case studies regarding the nonlinear Duffing oscillator and the nonlinear Burger's equation. The IVPs and the BVPs (or the IBVPs) are both treated to examine the characteristics of the examples. When dealing with IVPs, the complexity of initial conditions can cause an initial constraint effect to persist throughout an entire domain. This can lead to insufficient global truncation error occurrence and poor observation of the reduction of the error. Additionally, as neural network modeling becomes more intricate, there can be more factors that influence model accuracy. To avoid these potential factors and any ambiguity in identifying the sources of errors, this study employs the simple examples and concentrates on the analysis of the impact of initial parameters on the solution accuracy of the MPINNs. In addition, it focuses on the correlation analysis between them before and after the fine reinitialization. For the IVPs, the initial condition of the Duffing oscillator is two-point, whereas in the Burger's equation case, it is one-dimensional. Duffing oscillators governed by a second-order nonlinear ODE can capture the relationship between the input and the output of nonlinear dynamical systems under various conditions such as steady-state, transient, and other dynamic scenarios. The behaviour of the exact solution to Eq.(2.1), including amplitude and phase, depends on the parameter value, $1/4$ of $cn(2t, 1/4)$. As it deviates from zero, the function deviates from a simple cosine wave, and the shape and characteristics of the periodic oscillation change accordingly. In other words, the modulus can range from 0 to 1, with values being close to 0 corresponding to simple cosine-like oscillations, and values being close to 1 corresponding to more complex and highly nonlinear oscillations. Figure 3 demonstrates the phenomena mentioned above well.

$$\begin{aligned} &m \cdot D_t^2 \bar{u} + c \cdot D_t \bar{u} + k \cdot \bar{u} + f \cdot \bar{u}^3 = d \cdot \cos(\omega t),\ \forall t \in [0, 9] \\ &\text{with } \bar{u}(0) = 1 \text{ and } D_t \bar{u}(0) = 0. \end{aligned} \tag{2.1}$$

A Burger's equation (see Eq.(2.2)) is a nonlinear PDE governing the behaviour of traffic, liquid, and gas flow as well as the flow of materials during material processing operations such as casting and forging. It has numerous applications in physics, engineering, and applied mathematics.

$$\begin{aligned} &\partial_t \bar{u} + \bar{u} \cdot \partial_x \bar{u} = \mu \cdot \partial_x^2 \bar{u},\ \forall (t, x) \in [0, 90] \times [-4, 4] \\ &\text{with } \bar{u}(0, x) = 2x, \end{aligned} \tag{2.2}$$

here, $m = 1$, $c = 0$, $k = f = 2$, $d = 0$, $\omega = 0$ and $\mu = 1$. Eq.(2.1) has the exact solution of $\bar{u}(t) = cn(2t, \frac{1}{4})$. The exact solution to Eq.(2.2) is $\bar{u}(t, x) = \frac{2x}{1+2t}$. For the BVP of the Duffing oscillator, the boundary conditions are enforced to Eq.(2.1) instead of the given initial conditions: $\bar{u}(0) = 1$ and $D_t \bar{u}(0) = 0$, and $\bar{u}(9) = cn(18, \frac{1}{4})$ and $D_t \bar{u}(9) = -2dn(18, \frac{1}{4})\ sn(18, \frac{1}{4})$. For the IBVP of the Burger's equation, the boundary condition of $\bar{u}(90, x) = \frac{2x}{181}$ is enforced additionally to Eq.(2.2).

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

# 3. PINN

This section introduces a model representation of a classical PINN (see Figure 1) and discusses the outputs from the IVPs and the BVP (or the IBVP) including the limitation in applying the PINN model to the IVPs.

## 3.1. PINN representation

A PINN is a form of multi-layer perceptron that is interpreted as a functional approximator. It can be defined by

$$\bar{\boldsymbol{u}} = f_\theta(\boldsymbol{x}) \tag{3.1}$$

for the Duffing oscillator,

$$\boldsymbol{x} = \{t\} \text{ and } \boldsymbol{u} = \{u\} \tag{3.2}$$

and in the case of the Burger's equation,

$$\boldsymbol{x} = \begin{Bmatrix} t \\ x \end{Bmatrix} \text{ and } \bar{\boldsymbol{u}} = \{u\}. \tag{3.3}$$

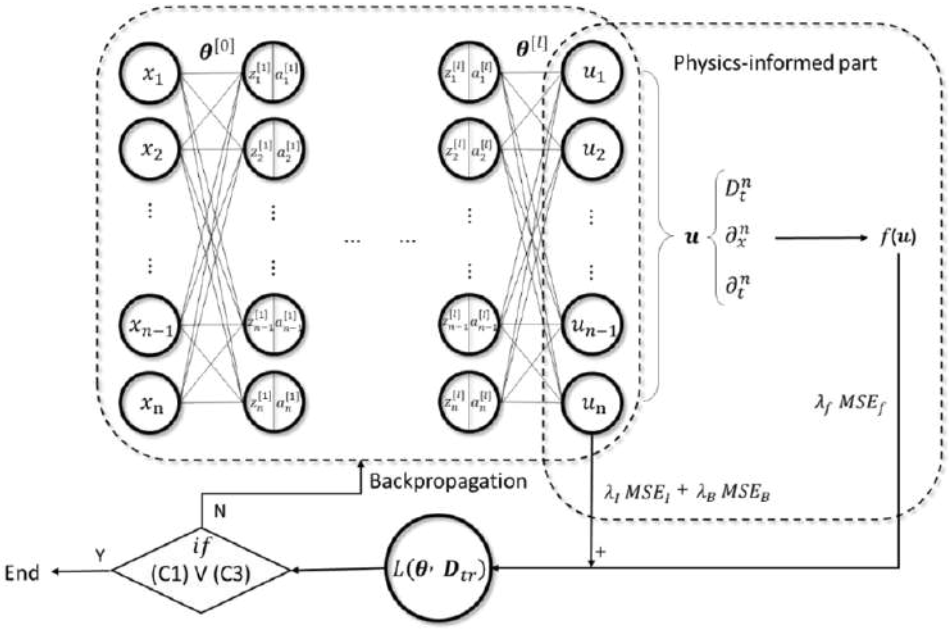



**Figure 1.** A schematic algorithm of a classical PINN.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

An input vector for the first hidden layer is $\boldsymbol{x}$ (see Eq.(3.2) and(3.3)) and each hidden layer's output is represented in a forward propagation phase by

$$\begin{aligned}
\boldsymbol{a}^{[0]} &= \boldsymbol{x}, \\
\boldsymbol{z}^{[1]} &= \boldsymbol{\theta}^{[0]} \cdot \boldsymbol{a}^{[0]} + \boldsymbol{b}^{[1]}, \\
\boldsymbol{a}^{[1]} &= \boldsymbol{\sigma}^{[1]}(\boldsymbol{z}^{[1]}), \\
&\vdots \\
\boldsymbol{z}^{[l]} &= \boldsymbol{\theta}^{[l-1]} \cdot \boldsymbol{a}^{[l-1]} + \boldsymbol{b}^{[l]}, \\
\boldsymbol{a}^{[l]} &= \boldsymbol{\sigma}^{[l]}(\boldsymbol{z}^{[l]}), \\
\boldsymbol{u} &= \boldsymbol{\theta}^{[l]} \cdot \boldsymbol{a}^{[l]} + \boldsymbol{b}^{[l+1]}.
\end{aligned} \tag{3.4}$$

The PINN has a single input layer, a single output one, and $l$ hidden ones. In this study, a Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm is used for optimization. It is in the family of Quasi-Newton methods that update an approximate $\boldsymbol{H}_k$ based on $\boldsymbol{g}_k$ across iterations. The optimizer has three different stopping criteria which are listed below, and it stops when one of those is met: (C1) $iter = iter^{MAX}$; (C2) $L(\boldsymbol{\theta}, \boldsymbol{D}_{tr}) < \varepsilon$; (C3) $||\nabla L(\boldsymbol{\theta}, \boldsymbol{D}_{tr})|| < \varepsilon^H$. Here, $iter$, $iter^{MAX}$ and $\varepsilon$ are replaced, respectively, with $iter^m$, $iter^{mMax}$, and $\varepsilon^m$for the meta-training. $L(\boldsymbol{\theta}, \boldsymbol{D}_{tr})$ is also switched with $L^{(i)}(\boldsymbol{\theta}^{(i)}, \boldsymbol{D}_{su}^{(i)})$ and $\sum_{i=1}^{NQ} L^{(i)}(\boldsymbol{\theta}^{(i),'}, \boldsymbol{D}_{qu}^{(i)})$, respectively, for each inner learner and the outer learner of the MPINN. The losses are defined by the same combination of the individual losses of Eq.(3.5). (C2) is applied only to the outer learner, not to each inner learner and the PINN with a single learner. The simulations in the study use the options: $iter^{mMax}$ is 5,000; $iter^{mMax}$ is 50; $\varepsilon^m$ is $8.0 \times 10^{-6}$; $\varepsilon^H$ is $2.22 \times 10^{-9}$; the maximum number of line search steps per iteration is 50. The PINN updates $\boldsymbol{\theta}$ by minimizing Eq.(3.5) that includes four individual losses presented below. $\bar{u}_s$ can be acquired through measurements or physical laws. In this research, the $MSE_d$ is not considered.

$$L(\boldsymbol{\theta}, \boldsymbol{D}_{tr}) = \lambda_f MSE_f + \lambda_I MSE_I + \lambda_B MSE_B + \lambda_d MSE_d, \tag{3.5}$$

where $\lambda_f = \lambda_I = \lambda_B = \lambda_d = 1$ and

$$MSE_f = \frac{1}{N_f} \sum_{p=1}^{N_f} [f(u_p)]^2, \tag{3.6a}$$

$$MSE_I = \frac{1}{N_I} \sum_{q=1}^{N_I} [u_q - \bar{u}_q]^2, \tag{3.6b}$$

$$MSE_B = \frac{1}{N_B} \sum_{r=1}^{N_B} [u_r - \bar{u}_r]^2, \tag{3.6c}$$

$$MSE_d = \frac{1}{N_d} \sum_{s=1}^{N_d} [u_s - \bar{u}_s]^2. \tag{3.6d}$$

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

## 3.2. Experiments for the performance evaluation of the PINN

In this section, the PINNs with different architectures are used to solve the IVPs and the BVP (or the IBVP), and the examination of the convergence of their solutions is included. The PINN for modelling the Duffing oscillator has six hidden layers, and each has 64 nodes with batch normalization and dropout (dropout rate = 0.5). Initialization is achieved by a variance scaling (VS) initializer and activations are swish with $\beta = 1.0$. For the Burger's equation, it has five hidden layers with 128 nodes per layer. Batch normalization is applied to every single hidden layer. A combination of a normal Xavier initializer and tanh activations is used. Figure 2 and tables 1–2 illustrate the evidence of the use of the NURS to the sub-domains across the entire domains in training the PINNs. The entire domain of the Duffing oscillator is divided into three sub-domains (see Figure 2(a) and Figure 2(b)), while the one of the Burger's equation has four sub-domains (see Figure 2(c) and 2(d)). It functions well on imposing samples in a more concentrated manner neighboring the left end or both ones of the entire domains depending on the nature of the problems. Regarding the Duffing oscillator, the IVP uses only two samples to the $MSE_I$. Each of (I), (II), and (III) has 84.99%, 10%, and 5% of 7,001 samples, respectively, for the $MSE_f$. The BVP uses each two samples at t=0 and t=9 for the $MSE_B$, and each sub-domain uses 42.83%, 14.28%, and 42.83% of 7,004 ones, respectively, for the $MSE_f$. In the case of the Burger's equation, the $MSE_I$ uses 10,000 samples for the IVP. Each of (I), (II), (III), and (IV) has 38.82%, 1.18%, 0.59%, and 0.59% of 17,000 samples, respectively, for the $MSE_f$. Whereas the $MSE_I$ only uses 1,000 samples for the IBVP, and it must be much less in comparison to the 10,000 samples of the IVP case. Each of the sub-domains has 81.20%, 1.25%, 1.25%, and 3.75% of 8,005 samples, respectively, for the $MSE_f$. Each of the intersections (90, -4), (90, -2), (90, 0), (90, 2), and (90, 4) imposed along the right end boundary has a single sample for the $MSE_B$. It turns out that the IBVP needs much fewer samples than 17,000 ones of the IVP for the highly accurate solution. The Duffing oscillator is tested using 1,000 uniformly distributed samples throughout the domain, and the Burger's equation is sampled specifically along the dashed lines shown in (c) or (d) of Figure 2.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

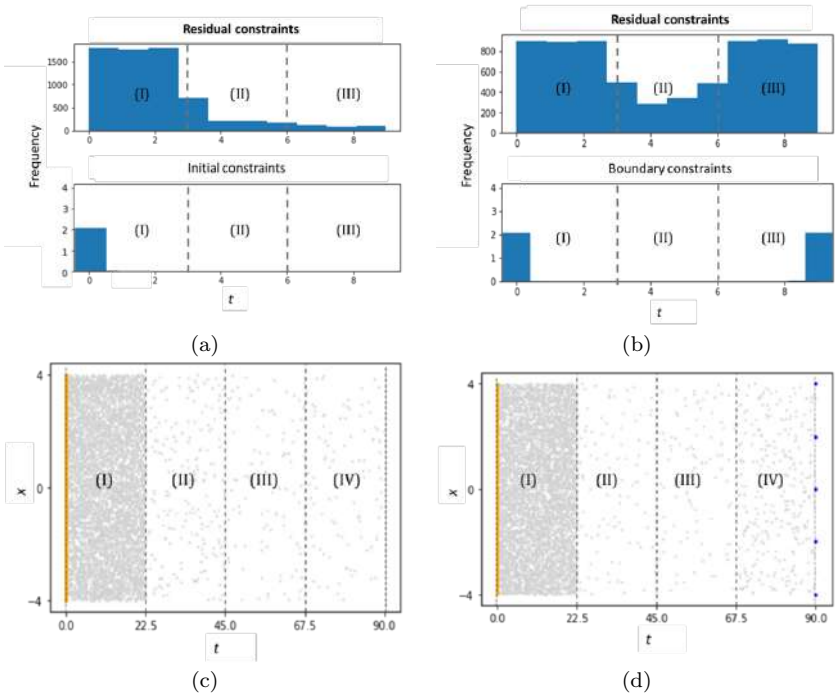



**Figure 2.** Distributions of samples across the sub-domains. (a) and (b) are, respectively, for the IVP and BVP of the Duffing oscillator; (c) and (d) are, respectively, for the IVP and IBVP of the Burger's equation. In Figure (c) and (d), orange and gray indicate samples, respectively, for $MSE_I$ and $MSE_f$, and blue displays samples for $MSE_B$.

**Table 1.** Number of samples for the Duffing oscillator

| Problem type | $MSE_I$ | $MSE_f$ | | | | $MSE_B$ | Total |
|---|---|---|---|---|---|---|---|
| | | (I) | (II) | (III) | Sub-total | | |
| IVP | 2 | 5,950 | 700 | 350 | 7,000 | – | 7,001 |
| BVP | – | 3,000 | 1,000 | 3,000 | 7,000 | 4 | 7,004 |

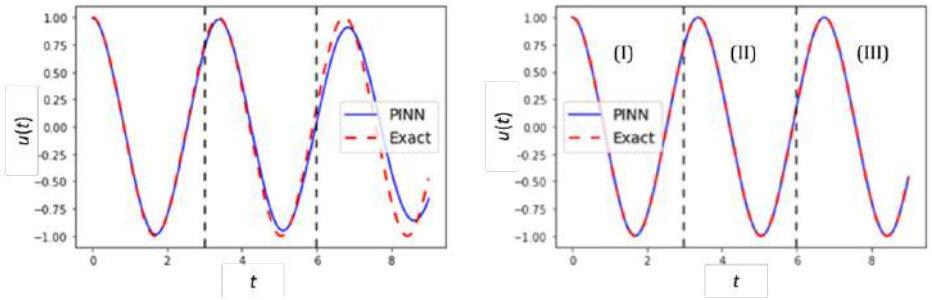
**Table 2.** Number of samples for the Burger's equation

| Problem type | $MSE_I$ | $MSE_f$ | | | | | $MSE_B$ | Total |
|---|---|---|---|---|---|---|---|---|
| | | (I) | (II) | (III) | (IV) | Sub-total | | |
| IVP | 10,000 | 6,600 | 200 | 100 | 100 | 7,000 | – | 17,000 |
| BVP | 1,000 | 6,500 | 100 | 100 | 300 | 7,000 | 5 | 8,005 |

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

To discuss the solution accuracy of the MPINNs as well as the PINNs, we quantify the global truncation errors using the root mean square error (RMSE) presented below:

$$RMSE = \sqrt{\frac{1}{NT}\sum_{n=1}^{NT}(u_n - \bar{u}_n)^2}. \tag{3.7}$$




(a) (b)

**Figure 3.** Convergence of the solutions to the Duffing oscillator. (a) Comparison of the solution by the PINN and the exact one to the IVP. (b) Comparison of the solution by the PINN and the exact one to the BVP.

**Table 3.** Comparison of the RMSEs of the Duffing oscillator

| Problem | RMSE($m$) | | | |
|---|---|---|---|---|
| type | (I) | (II) | (III) | Overall |
| IVP | $1.673 \times 10^{-2}$ | $5.967 \times 10^{-2}$ | $1.697 \times 10^{-1}$ | $1.043 \times 10^{-1}$ |
| BVP | $7.443 \times 10^{-4}$ | $1.307 \times 10^{-3}$ | $6.192 \times 10^{-4}$ | $9.391 \times 10^{-4}$ |

Figure 3 and Table 3 indicate that the RMSEs of the IVP increase with $t$, and (III) is the critical sub-domain. Unlike the IVP, the largest RSME occurs in (II) for the BVP, but it is relatively very small. The overall RMSE of the IVP increase by 11,006.4 %, and the local RMSE in (III) rises by 27,306.3 % in comparison to the BVP. As shown in Figure 4 and Table 4, the sub-domains at $t = 45$ and $t = 67.5$ are critical for the IVP of the Burger's equation. The IBVP finds the critical sub-domain in the middle, but it is also relatively small. The overall RMSE of the IVP increases by 659.69 %, and the local RMSE at $t = 45$ rises by 475.87 % compared to the IBVP. When it comes to the IVPs, it is obvious from the findings that the PINNs tend to increase the local RMSEs over $t$. This trend is more pronounced in the case of the Duffing oscillator. In the case of the Burger's equation, the local RMSEs start to increase rapidly after $t = 45$ and decrease somewhat around $t = 90$, but the decrement is not significant. This phenomenon appears to be mainly caused by a reduction in the influence of the initial constraints. In contrast, the boundary constraints play an imperative role in the high-quality convergence of the solutions for the BVP and the IBVP, especially in the critical sub-domains. One of the driving forces behind this study is the IVP issue. In section 4, we propose an alternative strategy to address it for the examples.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

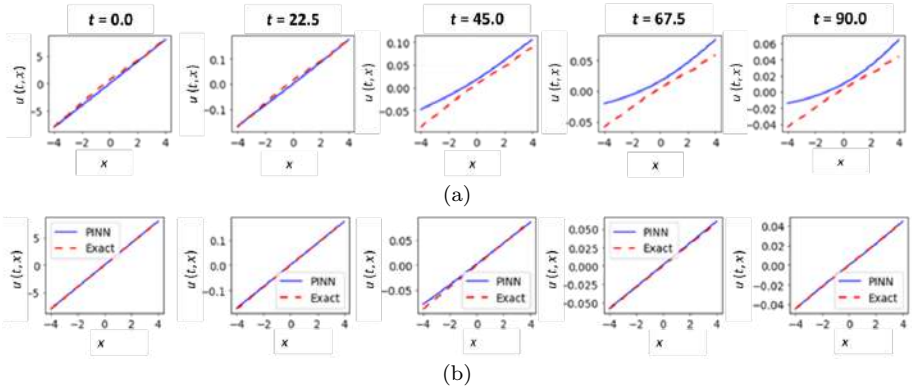



**Figure 4.** Convergence of the solutions to the Burger's equation. (a) Comparison of the solution by the PINN and the exact one to the IVP. (b) Comparison of the solution by the PINN and the exact one to the IBVP.

**Table 4.** Comparison of the RMSEs of the Burger's equation

| Problem | RMSE$(m/s)$ | | | | | |
|---|---|---|---|---|---|---|
| type | $t = 0$ | $t = 22.5$ | $t = 45$ | $t = 67.5$ | $t = 90$ | Overall |
| IVP | $1.057 \times 10^{-3}$ | $2.198 \times 10^{-3}$ | $2.277 \times 10^{-2}$ | $2.247 \times 10^{-2}$ | $1.616 \times 10^{-2}$ | $1.293 \times 10^{-2}$ |
| IBVP | $7.837 \times 10^{-4}$ | $2.101 \times 10^{-3}$ | $3.954 \times 10^{-3}$ | $1.271 \times 10^{-3}$ | $3.973 \times 10^{-4}$ | $1.702 \times 10^{-3}$ |

# 4. MPINN

In machine learning, there are several strategies for better adaptation of parameters to new inputs by learning from prior models. In this regard, various meta-learning techniques have been introduced in the literature [34]. This section discusses the MPINN based on the MAML that fine-reinitializes $\boldsymbol{\theta}$ using bi-level optimization. Comparative evaluations on the convergence of the solutions to the IVPs by the PINNs and MPINNs are implemented.

## 4.1. Meta-learning based on the MAML

An ANN is a typical nonlinear system, and its solution is highly dependent on its initial parameter values, as is well known. A meta-learning process is designed to improve the adaptability of the initial values of $\boldsymbol{\theta}$ that leads to better solution convergence. To achieve this, bi-level optimization is required. The outer learner, which reinitializes a meta-model for each inner learner in every single meta-iteration, enables each inner one to learn more adaptable features to a new task than a single learner. As a result, bi-level optimization can improve model accuracy as $\boldsymbol{\theta}$ alters its direction and magnitude of increments in a different update manner (see Figure 5), but to improve it, $\boldsymbol{\theta}$ can also be updated in a similar manner, as discussed in section 5 for the Burger's equation. Figure 5 summarizes (a) of Figure 6. Figure 5 and 6 display that first, training on $\boldsymbol{D}_{su}^{(i)}$ updates $\boldsymbol{\theta}$ to $\boldsymbol{\theta}^{(i)}$ by adapting $\boldsymbol{\theta}$ to $\boldsymbol{x}_{su}^{(i)}$. $\boldsymbol{\theta}^{(i)}$ takes a next forward step and achieves $\boldsymbol{\theta}^{(i),'}$ when $\sum_{i=1}^{NQ} L^{(i)}(\boldsymbol{\theta}^{(i),'}, \boldsymbol{D}_{qu}^{(i)})$ minimizes. Therefore, $\boldsymbol{\theta}$ reaches $\boldsymbol{\theta}^{(i),'}$ through $\boldsymbol{\theta}^{(i)}$. $\boldsymbol{\theta}^*$ is more adjustable to a new task when

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

$\boldsymbol{D}_{su}^{(i)}$ and $\boldsymbol{D}_{qu}^{(i)}$ are more similar in characteristics to $\boldsymbol{D}_{te}$.

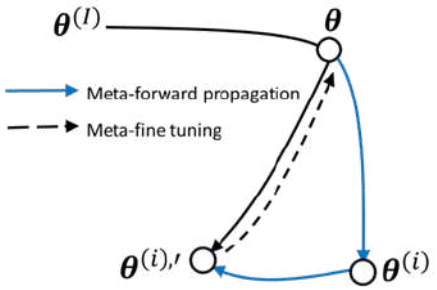



**Figure 5.** The key concept of the MAML for a single new task.

In the meta-training, the BFGS optimizer is also used. The first step begins with both $\boldsymbol{\theta}^{(I)}$ and $\boldsymbol{H}^{(I)}$, and a procedure with the steps of 1 - 6 for each of $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$ is repeated until one of the criteria stops the optimizer. The BFGS keeps $\boldsymbol{H}_k$ positive definite for updating formulations:

$$\text{Step 1: find } \boldsymbol{d}_k \text{ by solving } \boldsymbol{H}_k\boldsymbol{d}_k = -\nabla L(\boldsymbol{\theta}_k, \boldsymbol{D}_{tr}). \tag{4.1a}$$

$$\text{Step 2: perform a line search in } \boldsymbol{d}_k \text{ for computing } \alpha_k = \alpha \text{ by minimize } L(\boldsymbol{\theta}_k + \alpha\boldsymbol{d}_k, \boldsymbol{D}_{tr}). \tag{4.1b}$$

$$\text{Step 3: set } \boldsymbol{s}_k = \alpha_k\boldsymbol{d}_k \text{ and update } \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{s}_k. \tag{4.1c}$$

$$\text{Step 4: compute } \boldsymbol{g}_k = \nabla L(\boldsymbol{\theta}_{k+1}, \boldsymbol{D}_{tr}) - \nabla L(\boldsymbol{\theta}_k, \boldsymbol{D}_{tr}). \tag{4.1d}$$

$$\text{Step 5: update } \boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{g}_k\boldsymbol{g}_k{}^T}{\boldsymbol{g}_k{}^T\boldsymbol{s}_k} - \frac{\boldsymbol{H}_k\boldsymbol{s}_k\boldsymbol{s}_k{}^T\boldsymbol{H}_k{}^T}{\boldsymbol{s}_k{}^T\boldsymbol{H}_k\boldsymbol{s}_k}. \tag{4.1e}$$

$$\text{Step 6: return to step 1.}$$

In (a) of Figure 6, $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$ is deployable and $NS$ is nine for both examples, and $NQ$ becomes automatically equal to $NS$. $\boldsymbol{\theta}$ is updated to $\boldsymbol{\theta}^{(i)}$ by each inner learner when the update of $\boldsymbol{\theta}_k$ in Eq.(4.1c) on $\boldsymbol{D}^{(i)}_{su}$ is completed for $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$. Each inner learner follows the learning rule of the PINN. The outer learner updates $\boldsymbol{\theta}$ to $\boldsymbol{\theta}^{(i),\prime}$ ($\boldsymbol{\theta}^{(i),\prime}$ means $\boldsymbol{\theta}^*$ in the test phase) in substance through $\boldsymbol{\theta}^{(i)}$. It is also achieved by Eq.(4.1c) when Eq.(4.2) instead of $L(\boldsymbol{\theta}_k, \boldsymbol{D}_{tr})$ is applied to Eq.(4.1a). Eq.(4.2) indicates the total of individual query losses.

$$\sum_{i=1}^{NQ} L^{(i)}(\boldsymbol{\theta}^{(i),\prime}_{mk}, \boldsymbol{D}^{(i)}_{qu}). \tag{4.2}$$

The optimizer stops the outer learner by (C3) when each $iter^m$ reaches 17 and 40, respectively, for the Duffing oscillator and the Burger's equation. The transfer of the parameters (see (b) of Figure 6) significantly drops the number of the iterative pre-meta-tunings. For the Duffing oscillator, $f^m_{\boldsymbol{\theta}^{(1)}}(\boldsymbol{x}^{(1)}_{su})$ undergoes 1,077 iterations for the training, but $f^m_{\boldsymbol{\theta}^{(6)}}(\boldsymbol{x}^{(6)}_{su})$ needs five iterations only, and the rest necessitate a significantly smaller number of iterations too. For the Burger's equation, $f^m_{\boldsymbol{\theta}^{(1)}}(\boldsymbol{x}^{(1)}_{su})$

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

undergoes 140 iterations, while each of $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$ for i = 2, 4, 5, 8, and 9 requires only one. An automatic evaluation of $f^m_{\boldsymbol{\theta}^*}(\boldsymbol{x}_{te})$ on $\boldsymbol{D}_{te}$ follows the meta-training for computing the RMSE of the MPINNs.

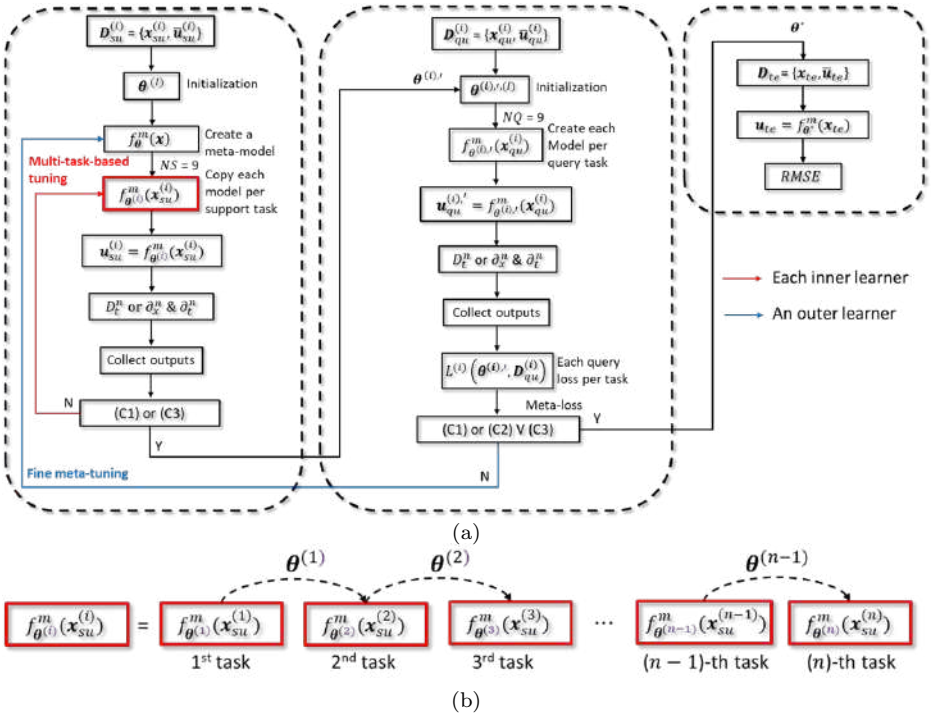



**Figure 6.** Schematic MPINN algorithm with the dual learners based on the MAML. (a) The meta-tuning and meta-test phases for the MPINN. (b) The application of the transfer learning to $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$.

## 4.2. Experiments for the comparative performance evaluation of the PINNs and the MPINNs

In this section, the experimental results of the IVPs by the PINNs and the MPINNs are discussed. Figure 7 and Table 5 show that the MPINN decreases the overall RMSE and the local RMSE in (III) of the Duffing oscillator problem, respectively, by 98.84 % and 98.83 % relative to the PINN. The former reduces the overall RMSE and the local RMSE at t = 45, respectively, by 79.81% and 85.89 % for the Burger's equation, as shown in Figure 8 and Table 6, relative to the latter. The meta-training samples both examples in the same manner shown in (a) and (c) of Figure 2.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

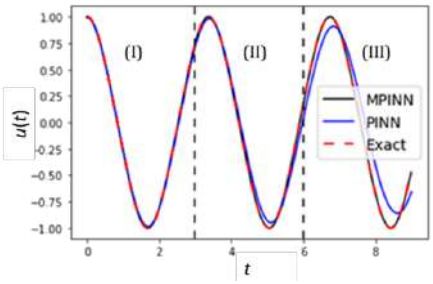



**Figure 7.** Comparison of the convergence of the solutions to the IVP of the Duffing oscillator.

**Table 5.** Comparison of the RMSEs of the IVP of the Duffing oscillator

| Solver | RMSE | | | |
|---|---|---|---|---|
| | (I) | (II) | (III) | Overall |
| PINN | $1.673 \times 10^{-2}$ | $5.967 \times 10^{-2}$ | $1.697 \times 10^{-1}$ | $1.043 \times 10^{-1}$ |
| MPINN | $1.854 \times 10^{-4}$ | $7.842 \times 10^{-4}$ | $1.979 \times 10^{-3}$ | $1.212 \times 10^{-3}$ |

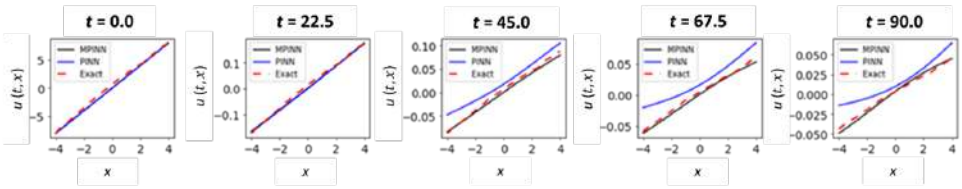



**Figure 8.** Comparison of the convergence of the solutions to the IVP of the Burger's equation.

**Table 6.** Comparison of the RMSEs of the IVP of the Burger's equation

| Solver | RMSE | | | | | |
|---|---|---|---|---|---|---|
| | $t = 0$ | $t = 22.5$ | $t = 45$ | $t = 67.5$ | $t = 90$ | Overall |
| PINN | $1.057 \times 10^{-3}$ | $2.198 \times 10^{-3}$ | $2.277 \times 10^{-2}$ | $2.247 \times 10^{-2}$ | $1.616 \times 10^{-2}$ | $1.293 \times 10^{-2}$ |
| MPINN | $5.937 \times 10^{-4}$ | $3.651 \times 10^{-3}$ | $3.213 \times 10^{-3}$ | $1.993 \times 10^{-3}$ | $3.605 \times 10^{-3}$ | $2.611 \times 10^{-3}$ |

Based on the experimental data, we conclude that the MPINNs with reinitialization capability effectively solves the IVPs presented in the examples, whereas the PINNs with no fine adjustment function struggle to solve them.

# 5. Adaptability and correlation analyses

To showcase the adaptability of initial parameters, we first examine the evolution of the initial values of $\boldsymbol{\theta}^{[l]}$ of $f_{\boldsymbol{\theta}}^{m}(\boldsymbol{x})$ before (either the VS or the normal Xavier is implemented for pre-meta-training) and after the fine reinitialization by the meta-training and then sort out hidden layers that play a key role in the reinitialization

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

and adaptation (see Figure 9 and 10). From the data, there is no doubt that $\boldsymbol{\theta}^{(0)}$ initiates it for both IVP problems.

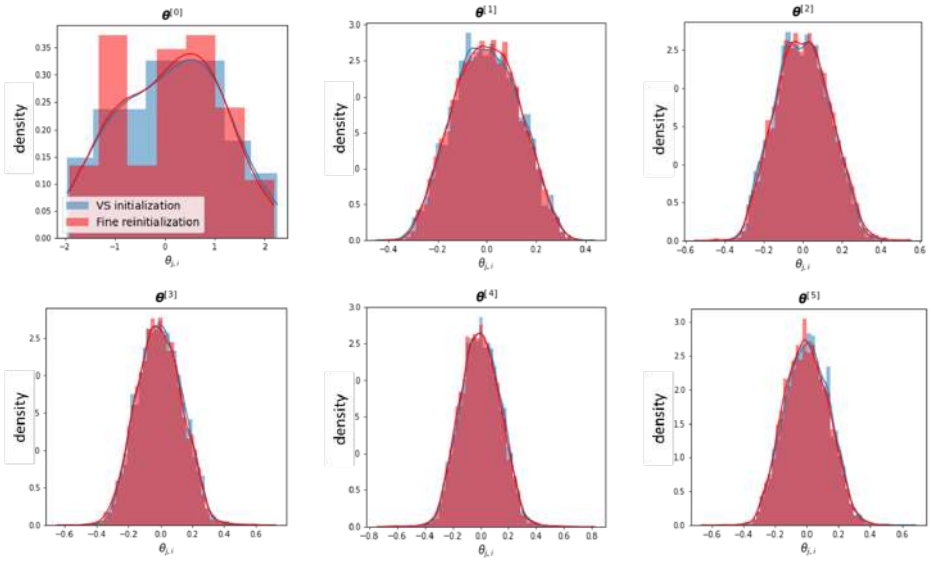



**Figure 9.** Evolution of the distribution of the initial values of $\boldsymbol{\theta}^{[l]}$ of the Duffing oscillator.

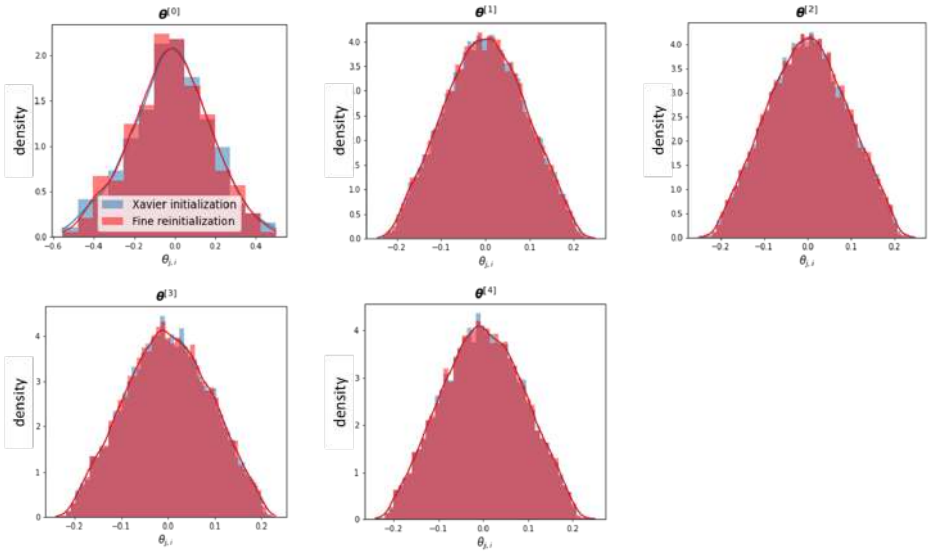



**Figure 10.** Evolution of the distribution of the initial values of $\boldsymbol{\theta}^{[l]}$ of the Burger's equation.

This section investigates how the initial values of $\boldsymbol{\theta}^{[0]}$s are related to the outputs of the examples. Additionally, as a supplementary to Figure 5, changes in the initial values of $\boldsymbol{\theta}^{[0]}$s before and after the application of the meta-learning are highlighted. As for the issues, correlation analyses are conducted using Kendall's

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

rank-order correlation coefficient. To make heatmaps for the analyses more readable, in addition to different colors, a square factor with different sizes is added to them. The size of a square in a correlation matrix generally corresponds to the strength of the correlation, with a larger size indicating a stronger correlation. The color of the square indicates the sign of the correlation, with blue and red typically representing positive and negative correlations, respectively. The depth of the color, either darker blue or darker red, represents the strength of the correlation with the square size.

Figure 11 displays the sampling of input coordinates for $u(t, x)$ that is used for correlation analysis between $\theta_{j,i}^{[0]}$ and $u(t, x)$. 256 samples are collected at random. In Figures 12-15, an x-axis or y-axis is graduated from left to right or top to bottom, respectively, and the graduations are labeled with $\theta_{j,i}^{[0]}$ values where necessary. To label them, the Duffing oscillator begins with $i = 1$, then selects $j = 1, 2, \ldots, 64$. The Burger's equation begins with $i = 1$, then selects $j = 1, 2, \ldots, 128$, and this process repeats for $i = 2$. For the first example, Figure 12.(a) and 12.(b) show the heatmaps of the correlation between $\boldsymbol{\theta}^{[0]}$ initialized, respectively, by the VS (before the use of the meta-learning) and the re-initializer (it means the whole meta-learning process) and the output, $u(t)$. The comparative analysis of both heatmaps demonstrates that they become less correlated during the meta-learning. The PINN with the two-point initial condition is sensitive to the initial value of $\boldsymbol{\theta}^{[0]}$, and the mitigation of the sensitivity, as inferred from the lower correlation between the initial value of $\boldsymbol{\theta}^{[0]}$ and $u(t)$, improves the model accuracy. This suggests that the meta-learning effectively finds the optimal initial parameters, which leads to the improved solution accuracy in the presence of the less sensitivity to $u(t)$, as the elements of $\boldsymbol{\theta}^{[0]}$ are updated individually. It can be explained by comparing Figure 14.(a) with 14.(b) which depict the correlations between $\boldsymbol{\theta}^{[0]}$s before and after the fine reinitialization. In the case of the Burger's equation (see Figure 13.(a) and 13.(b)), unlike the Duffing oscillator, the initial value of $\boldsymbol{\theta}^{[0]}$ and the output, $u(t, x)$ become more correlated during the meta-learning. This increased sensitivity leads to the improved model accuracy, as the model captures more complex relationships between $x$ and $u(t, x)$. It seems that the increase in the model accuracy due to the application of the meta-learning comes at the cost of the decreased robustness. The reason for this appears to be the integration of relatively intricate geometric domain information into the initial condition with a large number of samples used for discretization, making it difficult to decrease the sensitivity. However, it is imperative to be cautious, as this could lead to a less robust model that is vulnerable to the risk of overfitting and noise or inaccuracies in new input data, before finding an acceptable trade-off between robustness and model accuracy.

Figure 14 and 15 show the heatmaps of the correlation between $\boldsymbol{\theta}^{[0]}$s initialized, respectively, by the VS and the Xavier initializers (before the use of the meta-learning) and the re-initializer for both examples. The Duffing oscillator (see Figure 14.(a) and 14.(b)) demonstrates that the correlation becomes less during the meta-learning. From the data, it is clear that the re-initializer updates the elements of $\boldsymbol{\theta}^{[0]}$ independently in different directions and with different increments compared to the VS used in the pre-meta-learning. The observation supports Figure 5 well. For the case of the Burger's equation, in contrast to the former, $\boldsymbol{\theta}^{[0]}$s become more correlated as shown in Figure 15.(a) and 15.(b) which suggests that the elements of $\boldsymbol{\theta}^{[0]}$ are updated in an interdependent manner in the application of the re-initializer.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

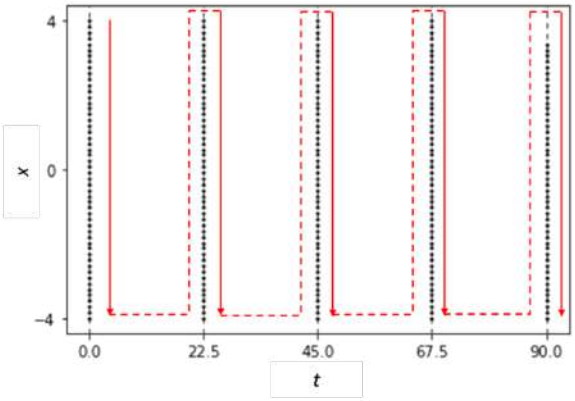



**Figure 11.** 256 samples for labeling the y-axis of Figure 13.(a) and 13.(b) with $(t, x)$ top to bottom: first select $t = 0$, then select x between 4 and -4, and repeat this for $t$ on the dashed lines (see the direction of arrows).

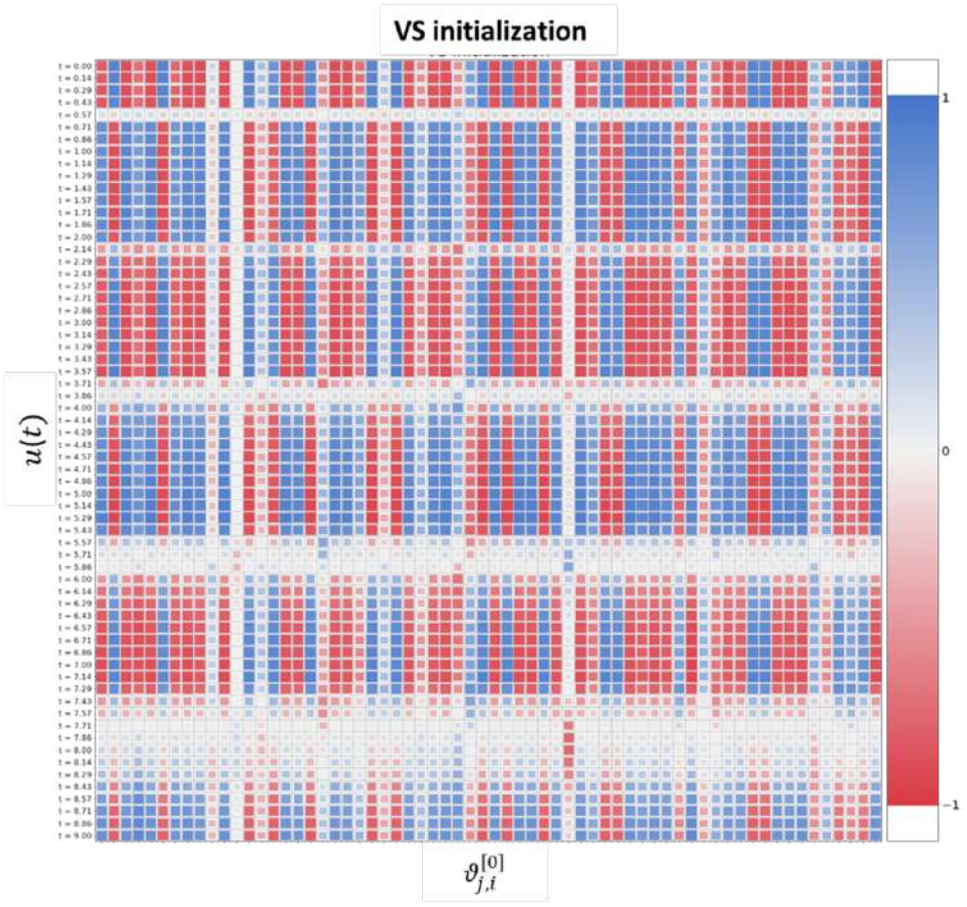



**Figure 12. (a)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$ initialized in the last meta-iteration and $u(t)$ for the Duffing oscillator.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

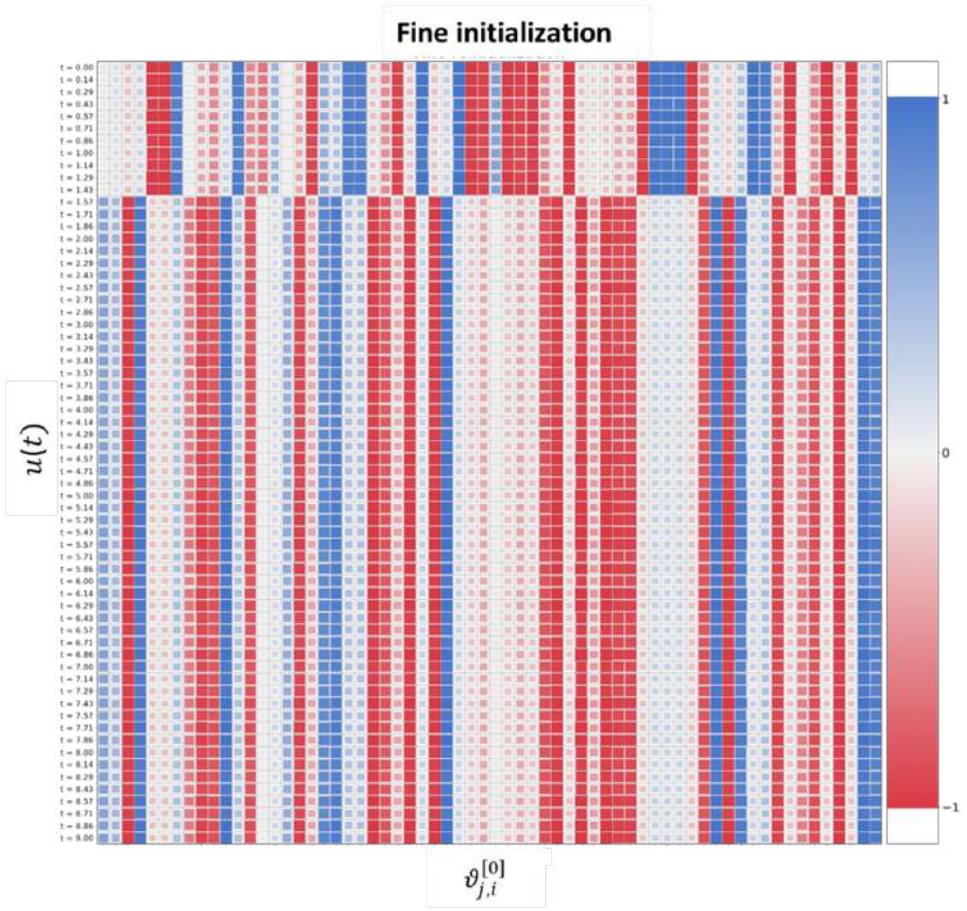



**Figure 12. (b)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$ initialized in the last meta-iteration and $u(t)$ for the Duffing oscillator.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

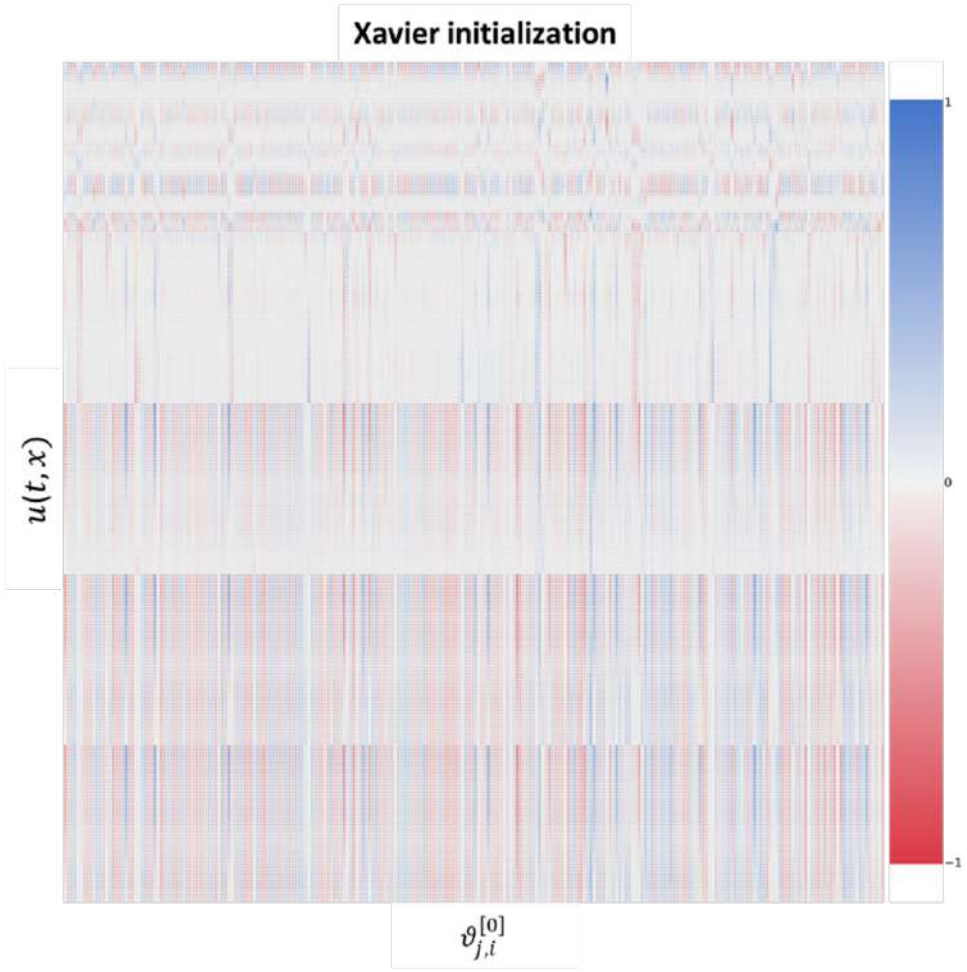



**Figure 13. (a)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$ initialized by the Xavier and $u(t,x)$ for the Burger's equation.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

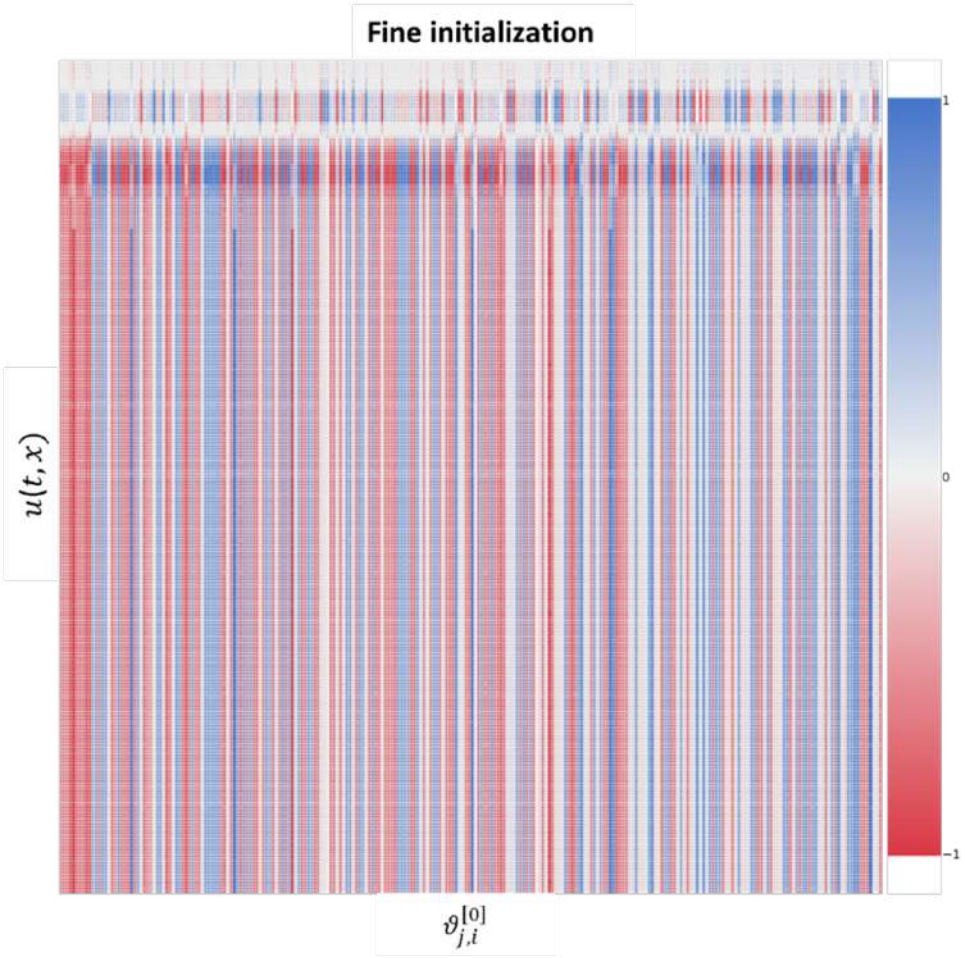



**Figure 13. (b)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$ initialized in the last meta-iteration and $u(t,x)$ for the Burger's equation.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

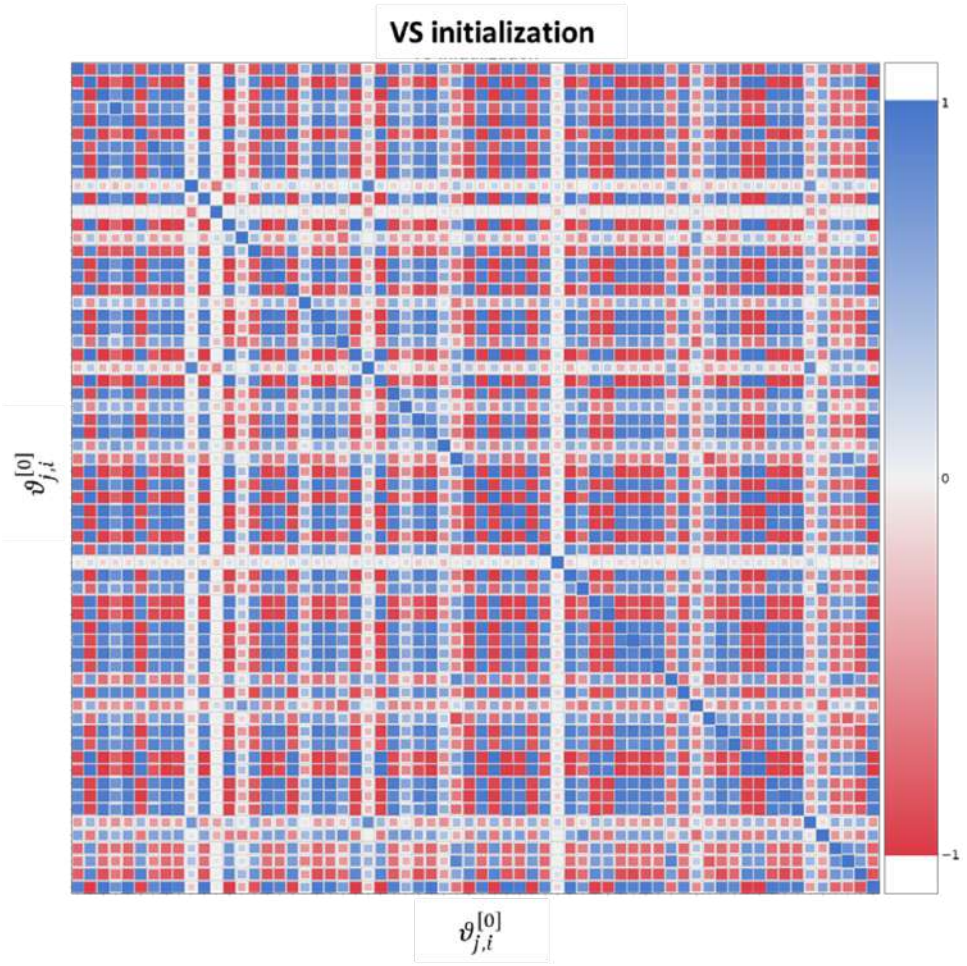



**Figure 14. (a)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$s initialized by the VS for the Duffing oscillator.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

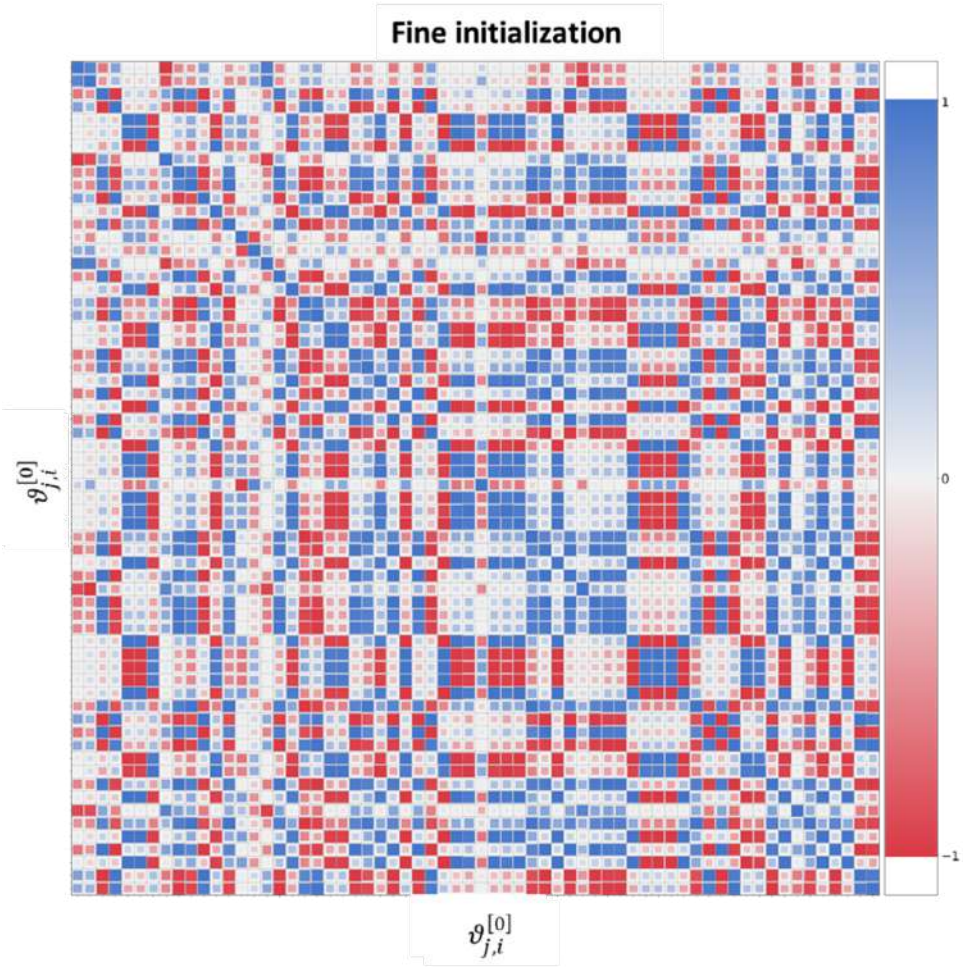



**Figure 14. (b)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$s initialized in the last meta-iteration for the Duffing oscillator.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

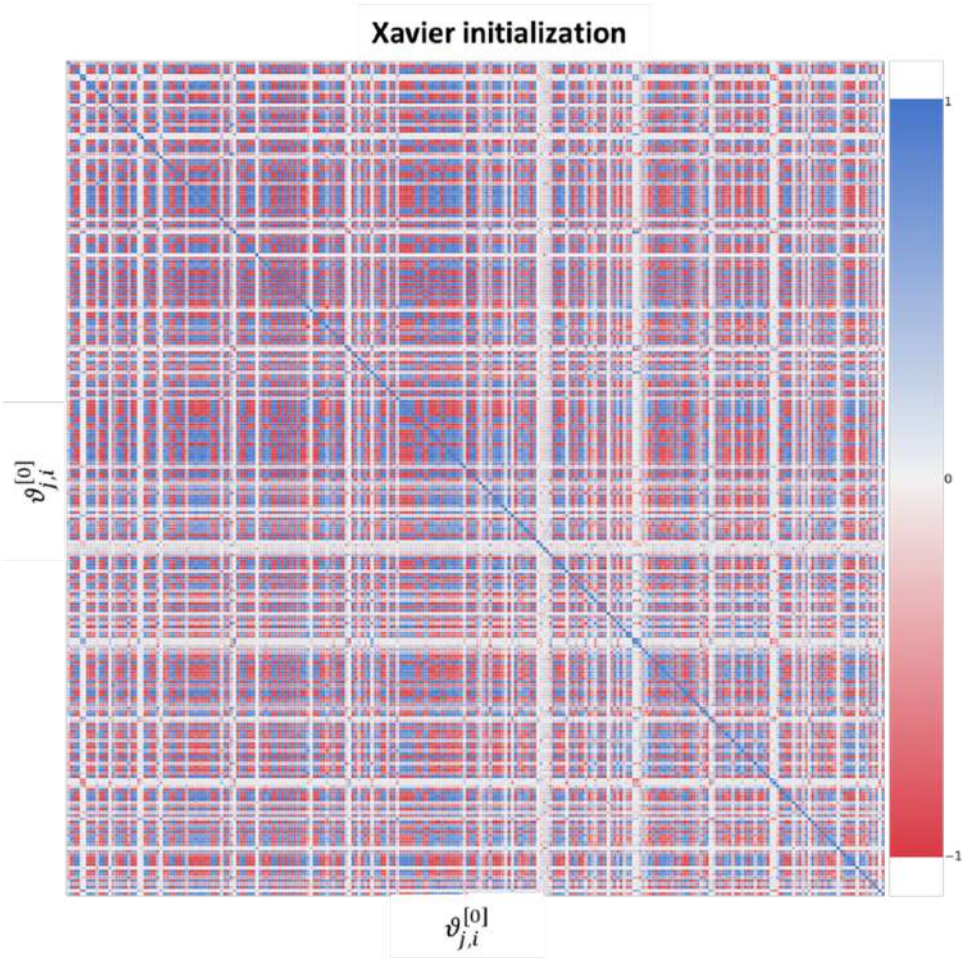



**Figure 15. (a)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$s initialized by the Xavier for the Burger's equation.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

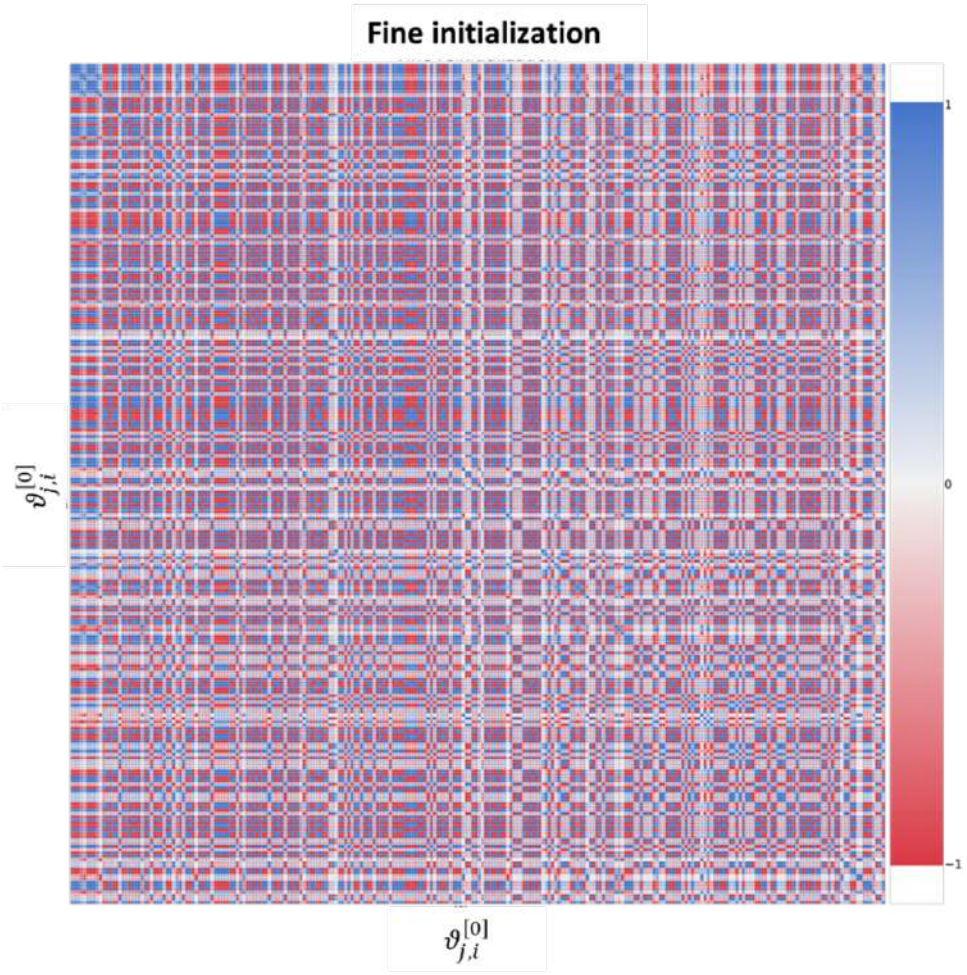



**Figure 15. (b)** Correlation analysis between $\boldsymbol{\theta}^{[0]}$s initialized in the last meta-iteration for the Burger's equation.

# 6. Conclusion

IVPs, which have significant importance in science and engineering and are associated with various real-world challenges, need for their accurate numerical solution as an essential task. In this study, we deal with the examples that are very simple but reflect the characteristics of IVPs well. The case studies are devised to encompass both scenarios: one where the output repeats itself or other similar patterns (the Duffing oscillator) and another where there is substantial variation in the gradient of the output with $t$ (the Burger's equation). Through the scenarios, in addition to the precise numerical solutions of the IVPs, it is possible to observe a distinct dissimilarity in the relation between the solutions with high accuracy and the model's sensitivity to them when the fine reinitialization is implemented.

The PINNs with a single learner struggle to find accurate solutions to the IVPs. As evidence of this, the experimental data demonstrate the large RMSEs. It implies

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

that the use of PINNs in science and engineering is limited because, unlike BVP and IBVP, IVPs may necessitate fine reinitialization. In response to the limitation, we propose the MPINN with the dual learners based on the MAML. The collaboration of the multiple number of inner learners and the outer learner overcomes the limitation of the PINNs with no fine re-initializer, as it enables the optimal reinitialization of $\boldsymbol{\theta}$ and gives the solutions with higher precision. In this process, $\boldsymbol{\theta}^{[0]}$ initiates the fine reinitialization for both IVPs. The findings show how closely the solution convergence of the IVPs is coupled to the initial values of $\boldsymbol{\theta}^{[0]}$s.

For the Duffing oscillator, the meta-learning decreases the sensitivity of the initial value of $\boldsymbol{\theta}^{[0]}$ to $u(t)$ by updating its components in the different manner, leading to the highly accurate solution. On the contrary, in the case of the Burger's equation, updating the components of $\boldsymbol{\theta}^{[0]}$ in the interdependent manner increases the sensitivity of the meta-model to $u(t, x)$. The findings from the correlation analyses hint that the application of the meta-learning process alone for solving the PDE may be insufficient to optimize both issues at the same time, despite the improvement in the model accuracy. Applying joint optimization to the MPINN, which may reset the correlation between the initial parameters and the output, could provide a double-purpose solution. Regarding savings in computation cost, the use of the transfer learning between $f^m_{\boldsymbol{\theta}^{(i)}}(\boldsymbol{x}^{(i)}_{su})$ contributes to the reduction of the number of iterations required for the pre-meta-training of each inner learner.

# 7. Future works

A lot of progress has been made in self-tuning neural networks through joint optimization involving neural network architecture search and hyperparameters. It could be imperative to apply the optimization method to the MPINN for achieving both higher model accuracy and robustness. To realize it, our upcoming study will develop a hypergradient descent-based MPINN. The optimal trade-off between higher model accuracy and lower sensitivity using an event-triggered mechanism to activate a hypergradient descent process could be achieved.

# Acknowledgements

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was supported by King Mongkut's Institute of Technology Ladkrabang (KMITL) of Thailand (grant number: 2566-02-01-007).

# References

[1] R. Zhang, Y. Liu and H. Sun, *Physics-informed multi-lstm networks for metamodeling of nonlinear structures*, Computer Methods in Applied Mechanics and Engineering, 2020, 369, 113226.

[2] S. Dong and Z. Li, *Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations*, Computer Methods in Applied Mechanics and Engineering, 2021, 387, 114129.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

[3] X. Meng, Z. Li, D. Zhang and G. E. Karniadakis, *PPINN: Parareal physics-informed neural network for time-dependent pdes*, Computer Methods in Applied Mechanics and Engineering, 2020, 370, 113250.

[4] R. Zhang, Y. Liu and H. Sun, *Physics-guided convolutional neural network (phycnn) for data-driven seismic response modeling*, 2019.

[5] M. De Florio, E. Schiassi and R. Furfaro, *Physics-informed neural networks and functional interpolation for stiff chemical kinetics*, Chaos: An Interdisciplinary Journal of Nonlinear Science, 2022, 32(6), 063107.

[6] J. Sirignano and K. Spiliopoulos, *Dgm: A deep learning algorithm for solving partial differential equations*, Journal of Computational Physics, 2018, 375, 1339–1364.

[7] 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, 2019, 378, 686–707.

[8] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis and P. Perdikaris, *Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data*, Journal of Computational Physics, 2019, 394, 56–81.

[9] L. Sun, H. Gao, S. Pan and J.-X. Wang, *Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data*, Computer Methods in Applied Mechanics and Engineering, 2020, 361, 112732.

[10] A. D. Jagtap, E. Kharazmi and G. E. Karniadakis, *Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems*, Computer Methods in Applied Mechanics and Engineering, 2020, 365, 113028.

[11] L. Sun and J.-X. Wang, *Physics-constrained bayesian neural network for fluid flow reconstruction with sparse and noisy data*, Theoretical and Applied Mechanics Letters, 2020, 10(3), 161–169.

[12] N. Geneva and N. Zabaras, *Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks*, Journal of Computational Physics, 2020, 403, 109056.

[13] M. Baymani, A. Kerayechian and S. Effati, *Artificial neural networks approach for solving stokes problem*, Applied Mathematics, 2010, 1 No. 4, 288–292.

[14] H. Gao, L. Sun and J.-X. Wang, *Phygeonet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state pdes on irregular domain*, Journal of Computational Physics, 2021, 428, 110079.

[15] Y. Ma, X. Xu, S. Yan and Z. Ren, *A preliminary study on the resolution of electro-thermal multi-physics coupling problem using physics-informed neural network (PINN)*, Algorithms, 2022, 15(2).

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

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

[17] S. A. Niaki, E. Haghighat, T. Campbell et al., *Physics-informed neural network for modelling the thermochemical curing process of composite-tool systems during manufacture*, Computer Methods in Applied Mechanics and Engineering, 2021, 384, 113959.

[18] Q. Zhu, Z. Liu and J. Yan, *Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks*, Computational Mechanics, 2021, 67, 619–635.

[19] L. Lu, X. Meng, Z. Mao and G. E. Karniadakis, *Deepxde: A deep learning library for solving differential equations*, SIAM Review, 2021, 63(1), 208–228.

[20] H. Lee and I. S. Kang, *Neural algorithm for solving differential equations*, Journal of Computational Physics, 1990, 91(1), 110–131.

[21] A.J. Meade and A. Fernandez, *The numerical solution of linear ordinary differential equations by feedforward neural networks*, Mathematical and Computer Modelling, 1994, 19(12), 1–25.

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

[23] H. Robinson, S. Pawar, A. Rasheed and O. San, *Physics guided neural networks for modelling of non-linear dynamics*, Neural Networks, 2022, 154, 333–345.

[24] Y. Peng, P. A. Flach, C. Soares and P. Brazdil, *Improved dataset characterisation for meta-learning*, in *Discovery Science* (Edited by S. Lange, K. Satoh and C. H. Smith), Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, 141–152.

[25] Y. Bengio, S. Bengio and J. Cloutier, *Learning a synaptic learning rule*, in *IJCNN-91-Seattle International Joint Conference on Neural Networks*, ii, 1991, 969 vol.2–.

[26] S. Bengio, Y. Bengio, J. Cloutier and J. Gecsei, *On the Optimization of a Synaptic Learning Rule*, Routledge, London, 2014, 265–287.

[27] S. Hochreiter, A. S. Younger and P. R. Conwell, *Learning to learn using gradient descent*, in *Artificial Neural Networks – ICANN 2001* (Edited by G. Dorffner, H. Bischof and K. Hornik), Springer Berlin Heidelberg, Berlin, Heidelberg, 2001, 87–94.

[28] A. Younger, S. Hochreiter and P. Conwell, *Meta-learning with backpropagation*, in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222)*, 3, 2001, 2001–2006 vol.3.

[29] C. Finn, P. Abbeel and S. Levine, *Model-agnostic meta-learning for fast adaptation of deep networks*, in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, JMLR.org, 2017, 1126–1135.

[30] X. Liu, X. Zhang, W. Peng et al., *A novel meta-learning initialization method for physics-informed neural networks*, Neural Computing and Applications, 2022, 34, 14511–14534.

[31] F. d. A. Belbute Peres, Y.-f. Chen and F. Sha, *HyperPINN: Learning parameterized differential equations with physics-informed hypernetworks*, in *The Symbiosis of Deep Learning and Differential Equations*, 2021.

[32] A. F. Psaros, K. Kawaguchi and G. E. Karniadakis, *Meta-learning pinn loss functions*, Journal of Computational Physics, 2022, 458, 111121.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS

[33] M. Penwarden, S. Zhe, A. Narayan and R. M. Kirby, *A metalearning approach for physics-informed neural networks (PINNs): Application to parameterized pdes*, Journal of Computational Physics, 2023, 477, 111912.

[34] M. Huisman, J. N. van Rijn and A. Plaat, *A survey of deep meta-learning*, Artificial Intelligence Review, 2021, 54, 4483–4541.

DOI https://doi.org/10.12150/jnma.2024.485 | Generated on 2024-12-19 03:53:36
OPEN ACCESS