COBEM 2023 27TH INTERNATIONAL CONGRESS OF MECHANICAL ENGENEERING

ABCM Brazilian Society of Mechanical Sciences and Engineering
27th ABCM International Congress of Mechanical Engineering
December 04-08, 2023, Florianópolis-SC, Brazil

COB-2023-1183
ON THE APPLICATION OF PHYSICS-INFORMED NEURAL NETWORKS IN THE MODELING OF ROLL WAVES

Bruno Fagherazzi Martins da Silva
Valdirene da Rosa Rocho
Guilherme Henrique Fiorot
Federal University of the Rio Grande of the Sul - UFRGS
bfaghe@gmail.com
valdirenerocho@gmail.com
guilherme.fiorot@ufrgs.br

Abstract. *Roll wave instability in free-surface flows is a long-studied phenomenon in fluid mechanics. Existing methods to predict roll wave properties are time-consuming, expensive, or inaccurate, prompting further exploration by the scientific community. This study evaluates the performance of Physics-Informed Neural Networks (PINNs) in modeling roll waves for laminar flows of Newtonian fluids. The objective was to determine if PINNs can successfully model roll wave behavior using a limited dataset and the governing differential equations. Seven PINNs were defined and trained using a subset of numerical results from a 2D transient flow simulation, together with the Shallow Water Equations. PINNs were used to predict wave interface heights and average streamwise velocities, which were compared to reference numerical data to assess accuracy. Results showed that PINNs accurately predicted roll wave properties such as frequency and wavelength. Wave heights and average velocities were also predicted with satisfactory accuracy, though the performance was poorer at wave peaks. Overall PINNs exhibited better performance in predicting flow height than velocity. Additionally, PINN performance decreased with higher Froude numbers, which can be attributed to underlying mathematical assumptions. This study demonstrates the potential of PINNs as mathematical tools for studying roll wave behavior, providing valuable insights and highlighting challenges in their application.*

Keywords: *Roll waves, Physics-informed neural networks, Free-surface flow, Scientific machine learning, Shallow water equations.*

1. INTRODUCTION

In the field of fluid mechanics, free-surface flows are a class of problems in which the fluid is not fully enclosed by solid boundaries. Most commonly, the surface of the fluids is exposed to air at atmospheric pressure. Many human-made or naturally occurring flows are of this kind, such as rivers, the ocean, gutters, canals, flumes, drainage ditches, and others (Fox *et al.*, 2020).

Disturbances in free surface flows can come from many sources, and often they cause a familiar type of phenomenon: a wave. Waves also happen in different forms, depending on the flow properties and the type of perturbation. Roll waves are a class of waves that form from disturbances of the interface of multi-layer flows. They occur on but are not limited to free-surface flows, on shallow-water regimes, and are the subject of this work. These waves arise due to the interplay between gravitational and viscous forces acting on the flow associated with the proper perturbations to the flow. Such perturbations may be caused by changes in the fluid bed, obstacles, and moving boundaries, among others.

The behavior of flow disturbances and, in this context, of roll waves is an important aspect to be considered by engineers that are involved in the design and construction of channels and structures. The prediction of dynamic stresses caused by the waves is an important part of the design of such structures. Roll waves can also play an important role in the design of fluid transport systems that may happen in a multilayer regime such as oil-gas transport in pipes in the Oil and Gas exploration industry, as the waves can influence the transport properties of the flow and vary the friction, as well as cause the mixing of said fluids (Balmforth and Mandre, 2004; Ng and Mei, 1994; Maciel *et al.*, 2013; Julien and Hartley, 1986; Aydin *et al.*, 2015; Vieira *et al.*, 2014).

Traditional methods for studying roll waves and fluid flows, in general, include well-known Computational Fluid Dynamics (CFD) models such as Finite Volumes Method, Finite Elements Method, and lattice-Boltzmann Method, among others for approximation of the solutions to the equations of motion (Ng and Mei, 1994; Rocho *et al.*, 2022; Peng *et al.*, 2011), analytical studies on the linear stability of the flow and propagation of instabilities (Maciel *et al.*, 2013; Ng and Mei, 1994; Balmforth and Mandre, 2004), as well as experimental setups (Fiorot *et al.*, 2015; Maciel *et al.*, 2017; Coussot, 1994).

The present study aims to study roll waves using a novel framework in the computational solution of Partial Differ-

ential Equations (PDEs) called Physics Informed Neural Networks (PINNs), first proposed in the seminal work of Raissi *et al.* (2019). This methodology has been used in many physics, mathematical, and engineering applications in the last couple of years, in subjects ranging from quantum mechanics, and linear elasticity to power systems and heat transfer, among other topics (Raissi *et al.*, 2019; Bararnia and Esmaeilpour, 2022; Haghighat *et al.*, 2020; Jin *et al.*, 2021; Mahmoudabadbozchelou and Jamali, 2021; Mahmoudabadbozchelou *et al.*, 2022). To the best of the authors' knowledge, it has never been used in the study of roll waves.

Neural Networks (NNs) are a class of machine learning algorithms that approximate some input dataset using a nonlinear computational graph, which has the relevant property of being universal function approximators according to (Raissi *et al.*, 2019; Hornik *et al.*, 1989). This computational graph has degrees of freedom called weights and biases that are tuned to best approximate the input data by minimizing a loss function – usually the Mean Squared Error (MSE) of the proposed approximation in relation to the input data, thus becoming an optimization problem (Weidman, 2019)).

PINNs differ from regular NNs as they introduce the residuals of the PDEs used to describe the phenomena at hand into the loss function, thus making it possible to not only fit the input data (such as experimental data and/or known boundary and initial conditions) but also minimize said residuals, thus enforcing that the solution also meets the physical requirements posed by the PDEs (Raissi *et al.*, 2019).

Furthermore, due to the function-like behavior of neural networks and the use of automatic differentiation by PINN algorithms, the PDEs and outputs can be evaluated at any desired point in the domain, eliminating the need for mesh generation to solve the PDEs, which is known to be a critical time-consuming, problem-dependent part of most CFD methods that rely on numerical solvers, widely known as a bottleneck in those applications (Cai *et al.*, 2021b).

Moreover, due to the structure of PINNs and how they approximate the solutions to the PDEs, there is no need for the problems to be well-posed and defined in the mathematical sense. That is, boundary and initial conditions are not required to be defined in the whole boundary, and data points can be incomplete and/or sparsely distributed, which is another significant difference compared to analytical and conventional CFD methods. One example of this is the use of two-dimensional and two-component (2D2C) velocity observations to reconstruct the full 3D flow field from Cai *et al.* (2021a).

The objective of the present work is to answer the question: "Can PINNs be used successfully to model the behavior of roll waves based on the differential equations that describe them and a limited dataset obtained from high-resolution numerical results? What are the challenges and highlights found in such an application?". The lack of literature regarding the use of PINNs in the study of roll waves alone provides a great opportunity for such development. However, the expressive increase in the number of studies and advancements in Machine Learning and Neural Networks technologies in recent years, aligned with the aforementioned benefits of PINNs compared to conventional CFD methods, make this research even more interesting.

2. MATHEMATICAL DESCRIPTION OF ROLL WAVES

The establishment of the equations starts from the conservation equations: continuity and generalized momentum conservation (represented by Cauchy's equations). Figure 1 exemplifies the geometry of a shallow water problem subdue to roll waves.

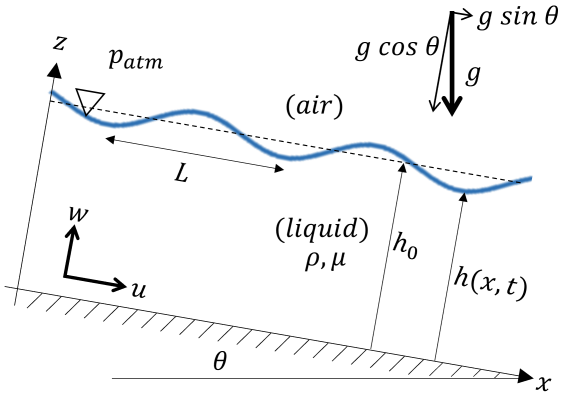

Figure 1. Schematics representation of the free-surface flow subdue to hydrodynamic instabilities.

Assumptions regarding the flow and the fluid rheological behavior must be defined. The roll waves phenomenon develops under shallow water conditions, where the characteristic free surface height h_0 is much smaller than the characteristic longitudinal dimension L. Flow is also assumed to be laminar, where viscous forces are dominant, considering a fluid to be Newtonian with dynamic viscosity μ.

Figure 1 graphically represents this problem, and also explicits another common hypothesis regarding the cross-section

of this flow: the width in y-coordinate is much larger than the vertical scale, thus allowing the assumption of a 2D flow, with $(x,\ z)$ being the 2D cartesian coordinates. In this case, the velocity field is $(u,\ w)$, the longitudinal (streamwise) and vertical velocities, respectively. The driven force is represented by g, the acceleration of gravity which is the only external force acting on the fluid. The free-surface height $h(x,t)$ is the instantaneous interface height at a given position along x and time-stamp t.

For this system, the Navier-Stokes equations are simplified and rewritten as shown in Eqs. (1), (2) and (3):

$$\frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} = 0, \tag{1}$$

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + w\frac{\partial u}{\partial z}\right) = -\frac{\partial p}{\partial x} + \rho g \sin\theta + \frac{\partial}{\partial z}\left(\mu\frac{\partial u}{\partial z}\right), \tag{2}$$

$$\frac{\partial p}{\partial z} = -\rho g \cos\theta. \tag{3}$$

Subsequently, the following boundary conditions are defined:

- at the channel bed, $z = 0$, impermeability condition (no-slip): $u(x, 0,\ t) = w(x, 0,\ t) = 0$;
- at the free surface, $z = h$, kinematic condition: $w(x, h,\ t) = \dfrac{\partial h}{\partial t} + u\dfrac{\partial h}{\partial x}$.

The simplified system of equations (Eqs. 1, 2 and 3) admits solution for plane Poiseuille-type flow. Integrating Eq. (3) vertically, that is, in relation to the depth of flow, gives the mathematical formulation for hydrostatic pressure:

$$p(z) = g\cos\theta(h - z). \tag{4}$$

Solving the conservation equations (mass and momentum) for the steady and uniform regime, one obtains the following classical solution:

$$u = \frac{h^2}{2}\left(\frac{\rho g \sin\theta}{\mu}\right)\left[1 - \left(1 - \frac{z}{h}\right)^2\right], \tag{5}$$

while the mean flow velocity ($\overline{u}$) is given by Eq. (6),

$$\overline{u} = \frac{h^2}{3}\left(\frac{\rho g \sin\theta}{\mu}\right). \tag{6}$$

For the steady and uniform case $h = h_0$ is the flow interface height, which is a constant. It is common in the literature on roll waves to assume that this solution applies also to transient flows as well (Maciel *et al.*, 2013; Ng and Mei, 1994; Julien and Hartley, 1986). Then for a time and spatial flow height $h = h(x,t)$, the velocity profile and the mean flow velocity also become dependent on (x,t) through Eq. (5) and (6), and are rewritten as $u(x,z,t)$ and $\overline{u}(x,t)$, respectively.

The aim now is to put into evidence the property $h(x,t)$. Then, Equations (1), (2), and 3 are vertically averaged by integrating them from 0 to h, that is, from the flow bed to the interface height between liquid and air, in a process known as von Kármán's momentum integral (Ng and Mei, 1994; Balmforth and Mandre, 2004; Maciel *et al.*, 2013). Lastly, a non-dimensionalization process is performed in the resulting equations, which allows for a series of simplifications, resulting in the following dimensionless equations of motion (in the following, all variables are dimensionless and the notion was readily changed to facilitate the understanding and avoid notation overload):

$$\frac{\partial h}{\partial t} + \frac{\partial(h\overline{u})}{\partial x} = 0, \tag{7}$$

$$h\left(\frac{\partial \overline{u}}{\partial t} + \alpha\overline{u}\frac{\partial \overline{u}}{\partial x}\right) + \frac{h}{\mathsf{F}^2}\frac{\partial h}{\partial x} + (1-\alpha)\overline{u}\frac{\partial h}{\partial t} = h - \frac{\overline{u}}{h}, \tag{8}$$

where α is momentum distribution coefficient (Ng and Mei, 1994; Maciel *et al.*, 2013) and for this special case $\alpha = 1.2$. The Froude number (F) of the flow is defined using u_0 and h_0, which are the spatial and temporal averages of $\overline{u}$ and h, respectively.

$$\mathsf{F} = \frac{u_0}{\sqrt{g h_0 \cos\theta}}. \tag{9}$$

This system of equations formed by Eq. (7) and (8) is simpler than the use of the complete equations of motion, due to the reduced number of variables and coordinates. However, it carries the set of assumptions used during its derivation. These assumptions can limit the application and their validity may be challenged, as will be seen in section 4.

3. PHYSICS INFORMED NEURAL NETWORKS (PINNs)

A Neural Network (NN) is a computational tool that has the objective of making predictions based on data it has never seen before, using the underlying patterns about that data learned during training with examples from the same dataset. From a black box point of view, they can be described simply as nonlinear mappings that convert inputs to outputs. On a more elaborate description, Neural Networks, and more specifically in this work, fully connected feedforward Deep Neural Networks (DNNs), are computational graphs composed of stacked layers of neurons, each of which computes a linear step and a nonlinear step on the data that enters it, with known linear coefficients (weights and biases) and nonlinear functions (activation functions). The inputs to each neuron are the outputs of the previous layer, except for the first layer whose inputs are the actual DNN inputs.

In this work, DNNs will be used for regression. This means that if a given phenomenon is fully described as a previously unknown (latent) function $\mathbf{y}^*(\mathbf{x})$, then a neural network will approximate it as

$$\mathbf{y}^*(\mathbf{x}) \approx \mathbf{y}(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{W}), \tag{10}$$

where $\mathbf{y}(\mathbf{x})$ is the approximation by the NN, $\mathbf{x}$ is the input vector, $\mathcal{N}$ represents the Neural Network computational graph, which is a function, and $\mathbf{W}$ is the tensor of weights and biases of all neurons in the network. Here the semicolon notation is used to separate variables such as $\mathbf{x}$ from parameters such as $\mathbf{W}$. Also, $\mathbf{x}$ and $\mathbf{y}$ are vector quantities as they can represent any number of scalar inputs and outputs, respectively.

A good prediction can only be achieved, however, if the weights and biases are tuned properly, which is done in a process called training. In order to tune these parameters, it is necessary to compare the output of the prediction by the network with the "correct" answers for those given inputs, which are called training examples. The comparison is made employing a loss function $\mathcal{L}$, which is usually some measure of the distance between the prediction and the correct reference result. Some examples of loss functions used in regression are the Mean Squared Error (MSE) and the Mean Absolute Error (MAE).

The training process hence becomes a problem of minimization of the loss function by adjusting the components of $\mathbf{W}$, for which well-established optimization algorithms can be used. Examples of such algorithms include gradient-based algorithms such as Stochastic Gradient Descent (SGD), Adam and Adagrad (Ruder, 2016), or even heuristic methods such as Particle Swarm Optimization (PSO) (Grimaldi *et al.*, 2004). The optimization algorithm of choice is run iteratively adjusting the weights and biases until a maximum number of iterations, or epochs in the NN jargon, are performed, or until the loss function reaches a desired threshold value.

Physics Informed Neural Networks (PINNs) are a class of NNs first defined in the work of Raissi *et al.* (2019) that leverage the information contained in the homogeneous partial differential equations (PDEs) that govern physical phenomena being modeled by the NNs. This is achieved by inserting the residual of the PDEs evaluated in a set of so-called collocation points in the physical domain, together with known data, into the loss function used in training. PINNs thus minimize the data error as well as the physical error, making sure that the results output by the NN not only are a good fit to the available data (which often includes boundary and initial conditions) but are also physically meaningful. Figure 2 shows a schematic of both a regular NN and a PINN.

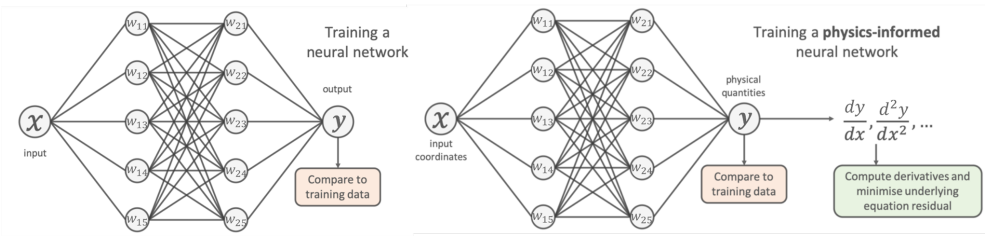

Figure 2. Schematic representation of a regular Neural Network (left) and a Physics Informed Neural Network (right). Networks contain two hidden layers, each with five neurons. Weights and biases are represented by vectors and activation functions are implicitly represented by connecting lines. Adapted from Moseley (2023).

As defined in Raissi *et al.* (2019), the PDEs can be generally expressed as:

$$\frac{\partial \mathbf{y}}{\partial t} + D[\mathbf{y}] = 0, \tag{11}$$

such that $\mathbf{y}(x,\ t)$ is the latent solution to the differential equation, and $D[\ \cdot\ ]$ is a generic nonlinear differential operator. And $f(x,\ t)$ is then defined as the left side of Eq. (11), that is, the residual of the PDE. Hence, the loss function $\mathcal{L}(\mathbf{W})$ to be minimized is given by:

$$\mathcal{L}(\mathbf{W}) = \mathcal{L}_{\text{data}}(\mathbf{W}) + \beta \mathcal{L}_{\text{physics}}(\mathbf{W}), \tag{12}$$

with

$$\mathcal{L}_{\text{data}}(\mathbf{W}) = \frac{1}{N_d} \sum_{j=1}^{m} \sum_{i=1}^{N_d} \left| y_j \left(x_d^i,\, t_d^i;\, \mathbf{W} \right) - y_j^i \right|^2, \tag{13}$$

$$\mathcal{L}_{\text{physics}}(\mathbf{W}) = \frac{1}{N_c} \sum_{i=1}^{N_c} \left| f \left(x_c^i,\, t_c^i;\, \mathbf{W} \right) \right|^2, \tag{14}$$

where $\mathbf{y} \in \mathbb{R}^m$ is the prediction made by the PINN with y_j representing each of its m scalar components. The set $\left\{ x_d^i,\, t_d^i,\, y^i \right\}_{i=0}^{N_d}$ represents the N_d training examples given to the network, which may consist of initial and boundary conditions, as well as data points anywhere in the domain. The $\left\{ x_c^i,\, t_c^i,\, y^i \right\}_{i=0}^{N_c}$ represents the set of N_c collocation points, defined as the points distributed in the domain where the $f(x,\, t)$ will be evaluated (Raissi *et al.*, 2019). And β is the relative weight of the physics component of the loss, relative to the data component. It has been shown that correct tuning of this quantity can lead to better convergence of the network (Jin *et al.*, 2021).

Since all activation functions, weights, and biases are known, after one proceeds to run forward through the network using $x,\, t$ to obtain a prediction $\mathbf{y}$, it is easy to run backward applying simple differentiation rules to obtain the derivative of the output with respect to the inputs, as long as of all the operations performed onto the inputs are tracked. This process is called Automatic Differentiation (AD), and it has been used for years in the Neural Networks field to calculate the gradient of the loss function with respect to the weights to perform minimization steps in gradient-based optimization algorithms (Raissi *et al.*, 2019; Weidman, 2019).

4. BENCHMARK DATA

In the present paper, CFD results obtained from Rocho *et al.* (2022) were used as a reference for the training of the PINNs. They were obtained using the Finite Volume Method to solve the 2D Navier-Stokes Equations for free-surface flow in a setup like the one described in Fiorot *et al.* (2015) and employed the Volume of Fluids technique to trace the interface between the test-fluid and air. The flow was solved into a steady and uniform state and then used as an initial condition for transient simulation which an oscillatory boundary condition was applied. The transient boundary condition was a sinusoidal velocity inlet with known frequency and amplitude. Results regarding the velocity and pressure fields of the flow, as well as the position of the liquid-air interface, were acquired. From this point onwards the terms "CFD", "numerical", and "reference" will be used interchangeably to refer to this dataset.

In Rocho *et al.* (2022) the test-fluid was glycerin with density $\rho = 1237$ kg/m^3 and dynamic viscosity of $\mu = 0.211527$ Pa/s. The chosen slope of the channel is $\theta = 8°$. The simulations were carried out for seven different Froude numbers, i.e., $\mathsf{F} \in \{0.83, 0.92, 0.98, 1.00, 1.12, 1.25, 1.36\}$.

With the provided data at hand, an exploratory analysis of the data was performed to identify patterns and gain insights into them, which is shown in Fig. 3.

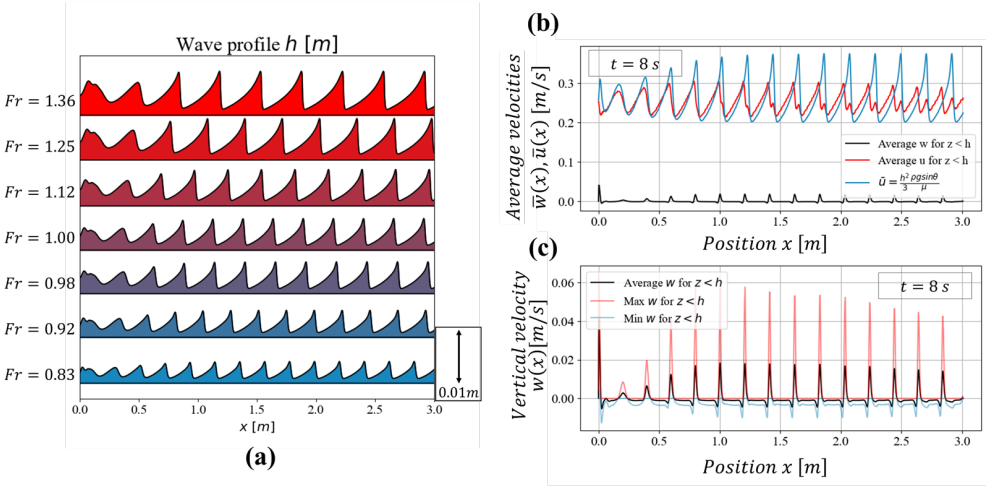

Figure 3. Exploratory analysis of the CFD data. (a) Wave profile at $t = 8$ s for all available Froude numbers. (b) Average vertical velocity, average numerical and theoretical longitudinal velocities for $\mathsf{F} = 0.83$ at $t = 8$ s. (c) Average, minimum, and maximum vertical velocities as a function of x for $\mathsf{F} = 0.83$ at $t = 8$ s.

Figure 3(a) showcases the fact that larger Froude numbers lead to larger amplitudes and wavelengths, and smaller frequencies. In Fig. 3(b) the red and black lines are calculated by vertically averaging the velocities obtained numerically

from the reference data, and the blue line represents the extrapolation of the plane-Poiseuille average longitudinal velocity presented in Eq. (6), i.e., the results expected from the theory.

Because of the assumption of shallow water flow, mentioned in Section 2. the vertical velocity and its gradients are assumed to be negligible. Figure 3(b) shows that the average vertical velocity is indeed roughly one order of magnitude smaller than the longitudinal velocity. Nonetheless, Fig. 3(c) displays that both average and maximum vertical velocities spike periodically, in the same positions where the wave peaks occur. It is hypothesized but not proved here that the gradients in such locations are responsible for the discrepancies between theoretical and numerical results of average longitudinal speed, as they might be redirecting energy and momentum from the streamwise flow.

For the training of the PINNs and validation of the results, it was chosen to use the average streamwise velocities obtained via Eq. (6), since the PDEs presented in Eq. (7) and Eq. (8) use this equation as an assumption during their derivation. The interface height, however, was chosen to be the one obtained numerically from the reference data, as they agree well with experiments performed by (Fiorot *et al.*, 2015).

Although these choices slightly distance the work from the numerical results used as a reference, as seen in Fig. 3(b), it stays consistent with the theory described in Section 2. which can still present several useful insights in the limiting cases, formation, and stability analysis of roll waves, such as the ones performed in the literature (Balmforth and Mandre, 2004; Maciel *et al.*, 2013; Ng and Mei, 1994; Rocho *et al.*, 2022).

For all Froude numbers the spatial domain in the x direction, which ranges from $x = 0$ to $x = 3$ m, was divided into 1,700 equal parts ($\Delta \approx 0.0017$ m). The time domain, ranging from $t = 8$ s to $t = 10$ s was divided into 1,000 steps ($\Delta t = 2 \times 10^{-3}$ s). The use of this time domain was limited for the time stamp the roll waves were stationary. However, a visual analysis of the data showed that the wave period for all Froude numbers simulated is slightly smaller than 400×10^{-3} s, excluding the need to use the full-time domain. The use of solutions in fewer time stamps optimizes the training of the networks as it reduces the domain and computational time. For the application of the PINNs to the CFD data for analysis 200 solutions at different time stamps were used, whereas during the architecture definition, a total of 160 were used, to increase processing speed.

5. PINN ARCHITECTURE DEFINITION

The four main elements that comprise the architecture of a Neural Network are the number of layers, the number of neurons in each layer, the nonlinear activation function, and the optimization algorithm. These elements are often referred to as hyperparameters because they directly affect the results of the predictions performed by the networks, but they are not used directly in the mathematical operations like the weights and biases, hence the prefix "hyper" (Benardos and Vosniakos, 2007; Yu and Zhu, 2020).

All PINNs built in this paper have two scalar inputs (x, t) and two scalar outputs $(h, \bar{u})$. Using a simple black box notation (which ignores the inner workings of the network), the PINNs trained in this work have the following functional behavior:

$$\begin{bmatrix} \overline{u} \\ h \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} x \\ t \end{bmatrix} ; \mathbf{W} \right). \tag{15}$$

As mentioned in Yu and Zhu (2020), the learning rate is also an important hyperparameter to be defined. During all simulations used in this work, a feature called ReduceLROnPlateau learning rate scheduler was used (Pytorch, 2023).

An important hyperparameter that is exclusive to PINNs is the number and location of collocation points, as they define where in the domain the differential equations will be evaluated. Some studies have been published regarding techniques to improve the efficiency of choice of the collocation points such as Jin *et al.* (2021); Nabian *et al.* (2021); Jagtap and Karniadakis (2021). In this work, the Residual Adaptive Refinement (RAR) methodology from Jin *et al.* (2021) was used due to its simplicity and ease of implementation. The optimization algorithm was chosen to be the Adam algorithm (Kingma and Ba, 2014). This decision was made based on the author's successful previous experience with it similar problems.

The number of layers, number of neurons per layer, and activation functions were defined using a structured testing methodology. A total of 24 trials were performed to determine these three hyperparameters, with one of them failing to converge (24 layers). The method used consisted of keeping all hyperparameters constant in a preset value and varying only the test subject. In these trials, the PINNs were trained for 40,000 epochs with 5,000 random training examples from the reference data with $\mathsf{F} = 0.83$, in the full space domain and in the first 160 time steps after $t = 8$ s. The value of β, from Eq. (12), was chosen to be unity based on trial-and-error runs before these tests.

All CFD data was provided in SI units, so they were properly scaled according to the procedures discussed in Section 2. After that, all variables were normalized to have a mean of zero and a standard deviation of one, which is a common practice in literature and shown to have a significant positive impact on the performance of training, as the network can deal with all quantities having the same order of magnitude (Weidman, 2019).

The loss function used in these trials as well as in the training that will be discussed in the next section takes the form

of Eq. (12), where the general Eq. (13) and Eq. (14) are now explicitly specified as follows, using Eq. (7) and Eq. (8):

$$\mathcal{L}_{\text{data}}\left(\mathbf{W}\right) = \frac{1}{N_d}\left(\sum_{i=1}^{N_d}\left|h\left(x_d^i,\, t_d^i;\, \mathbf{W}\right) - h^i\right|^2 + \sum_{i=1}^{N_d}\left|\overline{u}\left(x_d^i,\, t_d^i;\, \mathbf{W}\right) - \overline{u}^i\right|^2\right), \tag{16}$$

$$\mathcal{L}_{\text{physics}}\left(\mathbf{W}\right) = \frac{1}{N_c}\sum_{i=1}^{N_c}\left[\left(\frac{\partial h}{\partial t} + \frac{\partial(h\overline{u})}{\partial x}\right)_i^2 + \left(\frac{\partial h\overline{u}}{\partial t} + \alpha\frac{\partial \overline{u}^2 h}{\partial x} + \frac{h}{\mathsf{F}^2}\frac{\partial h}{\partial x} + (1-\alpha)\overline{u}\frac{\partial h}{\partial t} - h + \frac{\overline{u}}{h}\right)_i^2\right]. \tag{17}$$

After training, the results were compared to the complete CFD dataset in order to determine the data loss of Eq. (16). The residuals of Eq. (7) and Eq. (8) were calculated in a total of 13,000 random collocation points in the space-time domain, and the mean absolute residual was calculated. Also, the total time duration of training was recorded to have a superficial comparison of computational effort, as the hardware used was kept in similar operating conditions for all tests.

Then, the results of the testing for each of the three analyses regarding the hyperparameters are depicted in Fig. 4.

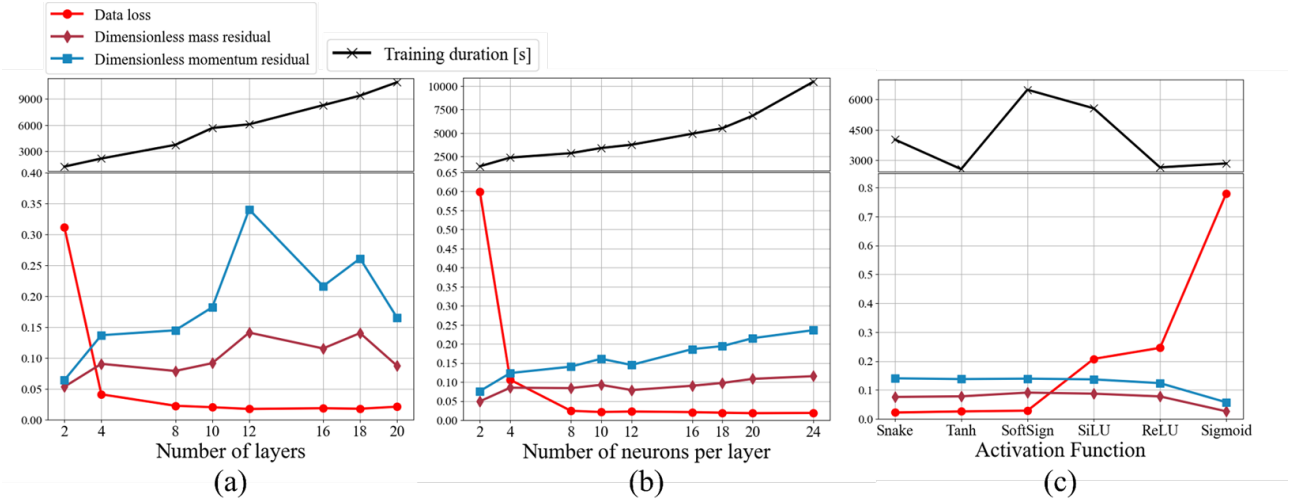

Figure 4. Results of the 23 successful trials performed to define the PINNs architecture. The metrics evaluated are the Data Loss, dimensionless mass residual, dimensionless momentum residual, and training duration. The hyperparameters tested were (a) Number of layers. (b) Number of neurons per layer. (c) Activation function.

Figure 4 (a) and (b) clearly show that the training time increases approximately linearly with the number of layers and with the number of neurons per layer, respectively. Similarly, the data loss decreases approximately exponentially with these quantities and reaches a plateau at around 8 layers and 12 neurons.

It is also visible that the PDE residuals increase with the number of neurons until they reach a more stable value. That may happen because the smaller number of neurons lack the nonlinear capacity to properly represent the data and tend to remain at the trivial solution in which $\bar{u}(x,t) \approx h(x,t) \approx 0$.

Finally, regarding Fig. 4(c), the snake function performed well on the periodic data, as predicted by Ziyin *et al.* (2020), but the hyperbolic tangent showed similar results with a computational time around 37% smaller. This is probably because the latter was built-in and optimized for use in the PyTorch framework, whereas the Snake function is user-defined. Based on these results, the choice was made on an architecture with 8 layers, with 12 neurons in each of them, and a hyperbolic tangent ($\tanh$) activation function.

6. IMPLEMENTATION OF THE PINNs

With the PINN architecture fully defined using the methodology presented in the last section, the networks were trained in the available CFD data, training a total of seven PINNs, one for each Froude number. Each PINN was trained on 1,000 random data points from the CFD dataset, for a total of 100,000 epochs. The value of β was now chosen to be 0.5 as it showed a better performance compared to other tested values.

After training, the PINNs were input the whole space and time domain to the PINN and calculated the Mean Squared Error (MSE) with respect to the reference CFD data, to evaluate the accuracy of the network. Also, the mean absolute mass and momentum residuals were evaluated over a 2000×2000 grid of collocation points in the space-time domain to assess the ability of the PINNs to conserve said quantities. Lastly, a comparison of these quantities for all our data sets was performed to understand how the Froude number affects the performance of the designed PINNs. The parameters used are compiled in Tab. 1, and the results are discussed in Section 4. Also, 4,000,000 uniformly distributed collocation points and 340,200 reference points were used.

Table 1. Information and parameters used in PINN implementation.

PARAMETERS	USED VALUES
Froude numbers (test cases)	0.83, 0.92, 0.98, 1.00, 1.12, 1.25, 1.36
Space domain range	0 m to 3 m, with $\Delta x = 0.0017$ m
Time domain range	8 s to 8.4 s, with $\Delta t = 0.02$ s
Activation function	Hyperbolic Tangent – $\tanh(\ )$
Number of layers	8
Number of neurons per layer	12
Initial number of collocation points	10,000 (randomly distributed)
Number of training examples	1,000 with RAR at every 5,000 epochs
Physics weight (β)	0.5
Number of epochs	100,000, without stop criterion
Optimization algorithm	Adam
Learning rate scheduler	ReduceLROnPlateau

7. RESULTS ANS DISCUSSION

In this section, the results obtained via the procedures detailed in Section 6will be discussed. Figures 5 and 6 display the wave interface height and free-surface velocity used for training (obtained numerically as discussed in Section 4), the values produced by the PINN, and the difference between them for a specific Froude number, $\mathrm{F} = 0.83$. Plots in Fig. 5 and 6 of the wave profile are shown in regard to x at a given instant ($t = 8.2$ s), and also in regard to the time at a given point in the domain ($x = 1.5$ m) to assess the PINN results more easily.

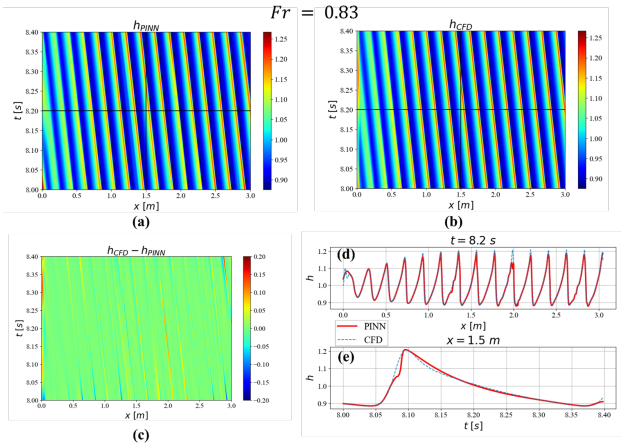

Figure 5. Dimensionless wave interface height for $\mathrm{F} = 0.83$. (a) Results predicted by the PINN. (b) Reference results obtained from Rocho *et al.* (2022). (c) Difference between CFD and PINN results. (d) Interface height at $t = 8.2$ s. (e) Interface height at $x = 1.5$ m. Plots (d) and (e) correspond to black lines in the heatmaps (a) and (b).

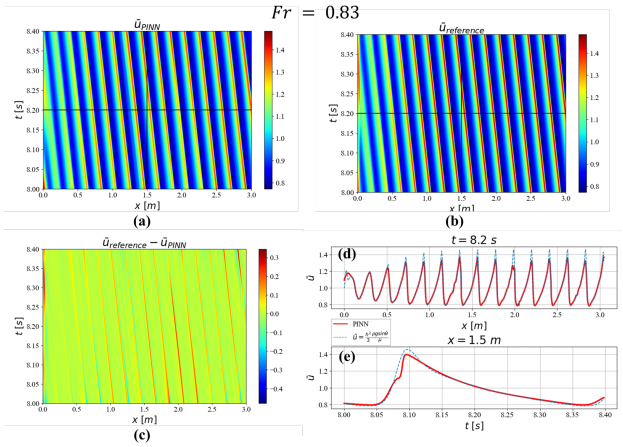

Figure 6. Dimensionless vertically averaged longitudinal velocity $\bar{u}$ for $\mathrm{F} = 0.83$. (a) Results predicted by the PINN. (b) Reference results obtained from Eq. (6). (c) Difference between reference and PINN results. (d) $\bar{u}$ at $t = 8.2$ s. (e) $\bar{u}$ at $x = 1.5$ m. Plots (d) and (e) correspond to black lines in the heatmaps (a) and (b).

Looking at Fig. 5 it is possible to see the similarity between the results predicted by the PINN and the reference data, using only 1,000 training examples out of the 340,200 available. Comparing Fig. 5(a) and Fig. 5(b) it is noticeable that the PINN well represented the oscillatory behavior of the waves both in space and time. Adding to that, an analysis of Fig. 5(d) shows that the frequency and wavelength of the predicted solution match closely with the reference data.

Figure 5(c) shows that the difference between the predicted and reference has an order of magnitude that ranges from zero to 0.2 in the whole domain. In most of the domain, the error is close to zero, however, it is noticeable that the PINNs performed poorly close to the peaks of the waves, underestimating the interface height by around 15% in the worst cases, as can be seen clearly in Fig. 5(d), at around $x = 2$ m. Figure 6 shows similar results to Fig. 5.

As with the wave height, the PINN was able to precisely predict the wavelength and frequency of $\bar{u}$ as can be seen by the close matching of the curves in Fig. 6(d) and Fig. 6(e). However, the PINN again underestimated the average velocity in the wave peaks, even more severely than it did for the height, as can be seen in Fig. 6(c), which shows larger errors.

The same analysis was performed for all Froude numbers mentioned in Section 4. Four metrics were chosen to evaluate the performance of the PINNs: The Mean Squared Error (MSE), the Mean Absolute Error (MAE), as well as the mass

and momentum residuals represented by the right-hand sides of Eqs. (7) and (8). The results of the error analysis are presented in Fig. 7.

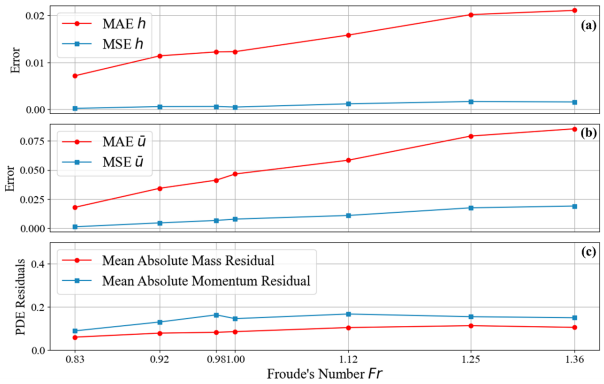

Figure 7. Performance assessment of the PINNs per Froude number via MAE, MSE for (a) wave interface height prediction and (b) average longitudinal velocity. Plot (c) shows the residuals of the roll wave differential equations.

Figure 7(a) and (b) show that both measures of error between the PINN predictions and the reference data tend to increase with the Froude number. The PDE residuals are shown not to vary much with the Froude number.

Also, the MSE was smaller than the MAE for both and also increased less with the Froude number. The smaller value of MSE is expected, since the differences are smaller than unity, their squares will also be smaller. The smaller slope of MSE with respect to the Froude number, on the other hand, can be explained by the fact that the MSE error was a metric being directly minimized during training, as shown by Eq. (16), and the MAE metric will be only indirectly minimized. As noticed in the in-depth analysis of the results for $\mathsf{F} = 0.83$, the PINNs performed significantly better predicting h than predicting $\bar{u}$, with the error for being 2 to 3 times larger than for h.

8. CONCLUSIONS

In the introduction of this work a two-fold question was posed regarding the application of PINNs in the study of roll waves: "Can PINNs be used successfully to model the behavior of roll waves based on the differential equations that describe them and a limited dataset obtained from high-resolution numerical results? What are the challenges and highlights found in such an application?".

After the work has been performed using the methodology expressed in Section 3. and in the face of the results presented in Section 4it is possible to return to the questions posed in the introduction of this work. Regarding the first part of the question, the answer is positive. As observed in Section 4. using a small subset of the available CFD data (<0,3% of the total) and the information contained in the PDEs that describe roll waves, it was possible to capture very accurately some of their key aspects such as wavelength and frequency, for all tested Froude numbers.

The height and average longitudinal velocity of the waves were also satisfactorily predicted with mean absolute errors below 10% but tended to increase with the Froude number. The residuals of the dimensionless PDEs remained consistently below 0.2 for all Froude numbers. To consider these results as accurate and/or sufficient, however, depends on the desired application.

As for the challenges encountered during this implementation, three main topics are noted. The first regards the PINN architecture definition, in general. Although many studies exist providing structured ways of defining hyperparameters for NNs and PINNs, as well as several methodologies to optimize training accuracy and speed are present in the literature, the process of building the right neural network is still problem dependent. This contrasts with some straightforward and well-defined methods of increasing efficiency in traditional CFD methods, such as mesh refining or the use of higher-order elements. This, however, can be a consequence of the relatively young age of the PINN methodology, compared to well-established methods.

The second and third topics are specific to the implementation of the framework to roll waves. PINNs did not perform well on higher Froude numbers, as well as on the peak of the waves. Despite the efforts of the author to improve the quality of the results in these situations, it wasn't possible during the development of this work. Many new technologies can already be found in the literature regarding PINNs, and the application of more advanced concepts and techniques might be able to deal with the limitations stated, and possible subjects for future works. Yet, the necessity of such techniques confirms these items are challenges inherent in this problem.

On the other hand, the quality of the results and the performance of the PINN in such conditions could be further related to the set of equations employed and the theory of roll waves being applied. Although well established, the assumptions made to reach the PDEs used in training might fail in some cases, such as were briefly touched on in Section 3.for the vertical velocity and its gradients. Hence it is possible that future works on the validation of the theory and a clearer

definition of the validity of the assumptions may be necessary. Overall, the present work brings light to applications of the PINN model such as the one here developed to roll waves problems, being those numerical, experimental, or even in-situ measurements, which could take advantage of results to further investigate specific scenarios not easily accessible, especially when predicting real events.

9. REFERENCES

Aydin, T.B., Torres, C.F., Karami, H., Pereyra, E. and Sarica, C., 2015. "On the characteristics of the roll waves in gas–liquid stratified-wavy flow: A two-dimensional perspective". *Experimental Thermal and Fluid Science*, Vol. 65, pp. 90–102.

Balmforth, N.J. and Mandre, S., 2004. "Dynamics of roll waves". *Journal of Fluid Mechanics*, Vol. 514, pp. 1–33.

Bararnia, H. and Esmaeilpour, M., 2022. "On the application of physics informed neural networks (pinn) to solve boundary layer thermal-fluid problems". *International Communications in Heat and Mass Transfer*, Vol. 132, p. 105890.

Benardos, P. and Vosniakos, G.C., 2007. "Optimizing feedforward artificial neural network architecture". *Engineering applications of artificial intelligence*, Vol. 20, No. 3, pp. 365–382.

Cai, S., Mao, Z., Wang, Z., Yin, M. and Karniadakis, G.E., 2021a. "Physics-informed neural networks (pinns) for fluid mechanics: A review". *Acta Mechanica Sinica*, Vol. 37, No. 12, pp. 1727–1738.

Cai, S., Wang, Z., Wang, S., Perdikaris, P. and Karniadakis, G.E., 2021b. "Physics-informed neural networks for heat transfer problems". *Journal of Heat Transfer*, Vol. 143, No. 6.

Coussot, P., 1994. "Steady, laminar, flow of concentrated mud suspensions in open channel". *Journal of Hydraulic Research*, Vol. 32, No. 4, pp. 535–559.

Fiorot, G.H., Maciel, G.F., Cunha, E.F. and Kitano, C., 2015. "Experimental setup for measuring roll waves on laminar open channel flows". *Flow Measurement and Instrumentation*, Vol. 41, pp. 149–157.

Fox, R.W., McDonald, A.T. and Mitchell, J.W., 2020. *Fox and McDonald's introduction to fluid mechanics*. John Wiley & Sons.

Grimaldi, E.A., Grimaccia, F., Mussetta, M. and Zich, R., 2004. "Pso as an effective learning algorithm for neural network applications". In *Proceedings. ICCEA 2004. 2004 3rd International Conference on Computational Electromagnetics and Its Applications, 2004.* IEEE, pp. 557–560.

Haghighat, E., Raissi, M., Moure, A., Gomez, H. and Juanes, R., 2020. "A deep learning framework for solution and discovery in solid mechanics". *arXiv preprint arXiv:2003.02751*.

Hornik, K., Stinchcombe, M. and White, H., 1989. "Multilayer feedforward networks are universal approximators". *Neural networks*, Vol. 2, No. 5, pp. 359–366.

Jagtap, A.D. and Karniadakis, G.E., 2021. "Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations." In *AAAI Spring Symposium: MLPS*. pp. 2002–2041.

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

Julien, P.Y. and Hartley, D.M., 1986. "Formation of roll waves in laminar sheet flow". *Journal of Hydraulic Research*, Vol. 24, No. 1, pp. 5–17.

Kingma, D.P. and Ba, J., 2014. "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980*.

Maciel, G.F., Ferreira, F.O., Cunha, E.F. and Fiorot, G.H., 2017. "Experimental apparatus for roll-wave measurements and comparison with a 1d mathematical model". *Journal of Hydraulic Engineering*, Vol. 143, No. 11, pp. 040170461–0401704610. doi:10.1061/(ASCE)HY.1943-7900.0001366. URL `https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29HY.1943-7900.0001366`.

Maciel, G.d.F., Ferreira, F.d.O. and Fiorot, G.H., 2013. "Control of instabilities in non-newtonian free surface fluid flows". *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, Vol. 35, No. 3, pp. 217–229.

Mahmoudabadbozchelou, M. and Jamali, S., 2021. "Rheology-informed neural networks (rhinns) for forward and inverse metamodelling of complex fluids". *Scientific reports*, Vol. 11, No. 1, pp. 1–13.

Mahmoudabadbozchelou, M., Karniadakis, G.E. and Jamali, S., 2022. "nn-pinns: Non-newtonian physics-informed neural networks for complex fluid modeling". *Soft Matter*, Vol. 18, No. 1, pp. 172–185.

Moseley, B., 2023. "So, what is a physics-informed neural network?" url:https://benmoseley.blog/my-research/so-what-is-a-physics-informed-neural-network/: :text=Physics%2Dinformed%20neural%20networks%3A%20A,Journal%20of%20Computational%20Physics.. Acesso em: 30 de mar. de 2023.

Nabian, M.A., Gladstone, R.J. and Meidani, H., 2021. "Efficient training of physics-informed neural networks via importance sampling". *Computer-Aided Civil and Infrastructure Engineering*, Vol. 36, No. 8, pp. 962–977.

Ng, C.O. and Mei, C.C., 1994. "Roll waves on a shallow layer of mud modelled as a power-law fluid". *Journal of Fluid Mechanics*, Vol. 263, pp. 151–184.

Peng, Y., Zhou, J. and Burrows, R., 2011. "Modeling free-surface flow in rectangular shallow basins by using lattice

boltzmann method". *Journal of Hydraulic Engineering*, Vol. 137, No. 12, pp. 1680–1685.

Pytorch, 2023. "Reducelronplateau". url:https://pytorch.org/docs/stable/generated/torch.optim.lr-scheduler.ReduceLROnPlateau.html. Acesso em: 30 de mar. de 2023.

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

Rocho, V.R., Fiorot, G.H. and Viçosa Möller, S., 2022. "Influence of the perturbation amplitude and the froude number on the establishment length of roll waves". *Journal of Engineering Mechanics*, Vol. 148, No. 10, p. 04022057.

Ruder, S., 2016. "An overview of gradient descent optimization algorithms". *arXiv preprint arXiv:1609.04747*.

Vieira, R.E., Kesana, N.R., Torres, C.F., McLaury, B.S., Shirazi, S.A., Schleicher, E. and Hampel, U., 2014. "Experimental investigation of horizontal gas–liquid stratified and annular flow using wire-mesh sensor". *Journal of Fluids Engineering*, Vol. 136, No. 12.

Weidman, S., 2019. *Deep Learning from Scratch: Building with Python from First Principles*. O'Reilly Media.

Yu, T. and Zhu, H., 2020. "Hyper-parameter optimization: A review of algorithms and applications". *arXiv preprint arXiv:2003.05689*.

Ziyin, L., Hartwig, T. and Ueda, M., 2020. "Neural networks fail to learn periodic functions and how to fix it". *Advances in Neural Information Processing Systems*, Vol. 33, pp. 1583–1594.

10. RESPONSIBILITY NOTICE

The authors are solely responsible for the printed material included in this paper.