Received July 26, 2022; | Revised ; | Accepted

DOI: xxx/xxxx

RESEARCH PAPER

On Water Wave Dynamics Using Physics Informed Neural Networks

Waasif Nadeem[1] | Volker John[2,3] | Zahra Lakdawala[1]

[1]Lahore University of Management Sciences, Department of Mathematics, School of Science and Engineering, Lahore, Pakistan
[2]Weierstrass Institute for Applied Analysis and Stochastics, Leibniz Institute in Forschungsverbund Berlin e. V. (WIAS), Berlin, Germany
[3]Freie Universität of Berlin, Department of Mathematics and Computer Science, Berlin, Germany

Correspondence
Zahra Lakdawala. Email: zahralakda@gmail.com

Summary

A vanilla feed forward neural network consists of neurons and layers and the mapping between input and output is approximated using a non linear function. The network is conventionally trained using data. In this work, we set up a neural network such that the spatio-temporal solution of time-dependent wave propogation models is learnt. This is done by providing the physical model, such as the 1D wave and shallow water wave equations and its associated boundary and initial conditions, as rules for learning the network. We invesitage the feasibility of data-driven and model-driven network predictions against numerical solutions. We further investigate the accuracy of the trained network through different parameter configurations and look closely into the multi-objective loss function that is constructed by including the residual error of the physical equations and the associated initial and boundary conditions. We present the results of numerical solutions against solutions obtained from data and model-driven neural network using data and physics informed rules for learning. Lastly, we construct a hybrid data and physics driven network and show that this significantly improves the accuracy of the physics-driven network.

KEYWORDS:
Physics informed neural networks, wave equation, data trained network, model-based learning, neural network

1 | INTRODUCTION

Partial differential equations (PDEs) are used to mathematically formulate the physical problems and hence, provide us with their solution. The physics of various real life applications comping from physics, biology, electrostatics, finance and other disciplines can be expressed in a complex system of PDEs. Our focus is to investigate the use of neural networks for predicting solutions for wave propogation models (acoustic wave equation and shallow water equation). In most real-scale models, it is either imposiible or infeasible to find an analytical solution for PDEs so we rely heavily on numerical schemes to find solutions. Numerical methods, however, can be very costly in terms of time and space and are proven to be quite inefficient for constantly changing environment. Efficient numerical computation has become increasingly important in this field. The solution is usually sought using one of the classical numerical methods such as Finite Difference, Finite Volume, Finite Element, etc.

In the past decade, we have seen machine learning methods such as Gaussian Regression Method, Neural Networks and Deep Learning approaches achieve great results specially in the field of image processing and natural language processing. It has become a widely popular field of research to train machine learning algorithms with physical rules . It has been shown that if one trains a neural networks with appropriate physics, the engine can be used to predict solutions for complex systems of partial

differential equations. A physics informed neural network (PINN) learns the solution to a PDE using the information regarding initial and boundary conditions of the equation in addition to the solution itself. The theory of neural networks state that it can learn any function but as the functions get complicated the number of layers needed to learn that function approaches infinity, which is not something we can apply in practice. PINNs learns the physics and the associated with much less layers than what is required for a conventional network. In addition, it is shown that PINN can also be used to predict solutions of PDEs with slight change in their initial conditions. The issue of efficiency, convergence and accuracy of solutions obtained from using PINNs is explored deeply . Regarding this aspect, only a few studies illustrate how the changes in the neural network configurations affect the solution , . Even fewer report on how the optimization functional can be modified for various physics to get better convergence and accuracy.

This work focuses towards employing neural networks and deep learning frameworks to as an alternative method for finding solutions of such partial differential equations. We consider the 1D wave equation and 1D Shallow Water Equation (SWE) along with its associated initial and boundary conditions as physical rules to train a neural network. We employ three ways of training the neural networks i) using data from numerical solutions; ii) using only the model/physics as given by the physical equation and its associated initial and boundary conditions; iii) using a hybrid approach that combines both the data and physical model to train the network.

This paper is organised as follows: we first provide a short introduction to the 1D wave and shallow water models considered as the physical models for training of PINNs. This is followed by a description of a vanilla feed forward neural networks. We constructa multi-objective loss function for optimizing the hyperparameters of the network. We further discuss a worked out a worked out example for how the loss function is modified to incorporate the physical model at hand. Section 4 shows the solution and evolution plots for data and physics trained neural networks and compares them against reference(numerical) solutions. We draw out a detailed discussion of the results in Section 5 summarizing our work and findings.

2 | THE CONSIDERED INITIAL-BOUNDARY VALUE PROBLEMS

An n-th order partial differential equations, along with its initial and boundary conditions, can be written in a parameterized form:

$$\begin{aligned}
\mathcal{PDE}\left(u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \ldots, \nu\right) &= 0 \text{ for } x \in \Omega, t \in (0, T], \\
\mathcal{IC}\left(u, \frac{\partial u}{\partial t}, \frac{\partial^2 u}{\partial t^2}, \ldots\right) &= 0 \text{ for } t = 0, \\
\mathcal{BC}\left(u, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}, \ldots\right) &= 0 \text{ for } x \in \Gamma, t \in (0, T],
\end{aligned}$$

where Ω and Γ denote the spatial domain and its boundary, respectively. Here, $\mathcal{PDE}$ describes the differential operators and $\nu = (\nu_1, \nu_2, \ldots)$ denotes the parameters of the partial differential equation. The solution of the initial-boundary value problem with initial conditions $\mathcal{IC}$ and boundary conditions $\mathcal{BC}$ is denoted by $u(x, t)$.

We focus on two models that describe wave dynamics. Waves are very important in the field of fluid dynamics, acoustics, and many other physical problems as they are an effective way to transmit information (sound, fluid, etc.) across several spatio-temporal scales. Our focus is mainly on hydrological applications, however the results can be informative for anyone studying wave dynamics. We consider in our investigations the 1D acoustic wave and shallow water equation.

2.1 | Wave equation

A wave equation, modeling the propagation of a wave at a fixed speed is defined as:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad \text{where } x \in [x_0, x_{\text{end}}], t \in (0, t_{\text{end}}]. \tag{1}$$

Here, $u(x, t)$ is the displacement in the second space dimension (y-direction) and c is the velocity of wave. The equation is equipped the initial and boundary conditions

$$u_0 = u(x, 0) = u_{\text{in}}(x), \quad \left.\frac{\partial u(x, t)}{\partial t}\right|_{t=0} = u'_{\text{in}}(x), \quad u(x_0, t) = u(x_{\text{end}}, t). \tag{2}$$

2.2 | Shallow water equation

Shallow water equations (SWE) are a set of equations that are derived from physical conservation laws for mass and momentum to describe fluid flow problems. They can be derived by depth averaging the Navier–Stokes equations. They are used in predicting cyclones, storm surges, flows around structures etc..

For simplicity, the 1D SWE is used in our study of deep learning techniques. This equation is derived from the 2D equations of mass and momentum conservation based on assumption of incompressibility of water, hydrostatic pressure distribution, and a sufficiently small channel slope. The conservative form of 2D SWE takes the following form

$$\frac{\partial}{\partial t}Q + \frac{\partial}{\partial t}F(Q) + \frac{\partial}{\partial t}G(Q) = 0,$$

where u and v are the depth-averaged water velocity in x and y directions, h is the water height with respect to a bottom profiled domain as a zero line, and $Q = (h, hu, hv)^T$, $F(Q) = (hu, hu^2 + \frac{1}{2}gh^2, huv)^T$, $G(Q) = (hu, huv, hu^2 + \frac{1}{2}gh^2)^T$. This equation has to be equipped with appropriate initial and boundary conditions.

The 1D SWE, after having differentiated the terms using the product rule, is written as

$$\begin{aligned} \frac{\partial h}{\partial t} + u\frac{\partial h}{\partial x} + h\frac{\partial u}{\partial x} &= 0 \text{ with } x \in [x_0, x_{\text{end}}], t \in (0, t_{\text{end}}], \\ h\frac{\partial u}{\partial t} + u\frac{\partial h}{\partial t} + u^2\frac{\partial h}{\partial x} + 2uh\frac{\partial u}{\partial x} + g\frac{\partial h}{\partial x} &= 0 \text{ with } x \in [x_0, x_{\text{end}}], t \in (0, t_{\text{end}}], \end{aligned} \tag{3}$$

where h is the water depth and u is depth-averaged velocity. Hence, in contrast to the wave equation, the solution of the 1D SWE is vector-valued. The initial and boundary conditions are given by

$$u_0 = u(x, 0), \quad h_0 = h(x, 0), \quad u(x_0, t) = u_t^{x_0}, \quad u(x_{\text{end}}, t) = u_t^{x_{\text{end}}}. \tag{4}$$

What means u_t? In the former version there was also an initial condition for $\partial_t u$ and $\partial_t h$, but there is no second order time derivative.

3 | DATA DRIVEN AND PHYSICS INFORMED NEURAL NETWORKS

3.1 | A data-driven neural network

should we use the abbreviation DDNN for data-driven neural network? Neural networks can generally be described by three components, namely neurons, layers and a global architecture. Each neuron of a layer is connected to each neuron of the next layer and is associated with a weight, a bias term, and an activation function σ. If the weight w_{jk}^l connects the k-th neuron in the $(l-1)$th layer to the j-th neuron in the l-th layer, the relationship for the output u_j^l can be written as

$$u_j^l = \sigma\left(\sum_k w_{jk}^l u_k^{l-1} + b_j^l\right) = \sigma\left(z_j^l\right).$$

The choice of the activation function will be one aspect in our studies.

The feed forward network uses the principle of back propogation to address how a variation in weight and biases impact the output error. The sample signal is applied by an activation function before it is passed to the next layer. For example, a sample input x with a sigmoid activation function sees a transformation of $f(x) = \frac{1}{1+e^{-x}}$ when it meets the next layer.

A neural network further needs a routine to update its weights and biases in the direction of minimizing a loss function defined on the output. This routine is generally a (stochastic) gradient descent algorithm or a quasi Newton method. In a supervised regression problem, the output loss is often the mean squared error (MSE) computed using the difference of the actual output value and the desired target value $y(x, t)$:

$$\mathcal{L}_{\text{data}} = \frac{1}{N}\sum_{x,t}\left\|y(x,t) - \hat{u}^L(x,t)\right\|_2^2, \tag{5}$$

where L is the number of layers and N is the number of uniformly sampled collocation points defined as the data set $\{x_i, t_i\}, i \in \{1, \ldots, N\{$. The output $\hat{u} = f(x, t, \theta)$ is expressed as a function of the input (x, t) and network parameters θ (weights and biases). The training process aims to find θ, given the input (x, t), by solving the following optimization problem

$$\text{argmin}_\theta \mathcal{L}\left(\theta \mid (x, t, y)\right),$$

where the function y maps (x,t) to measurements/true solution at those coordinates. The training is complete once the network has found parameters such that a desired accuracy of the loss function is reached. The function that is learnt is a function that is based just on the input data at (x,t).

3.2 | Physics informed training of the neural network

If one is interested in finding the solution of an initial-boundary value problem, then it seems to be a quite natural idea to incorporate this problem into the loss function, i.e., the physics instead of only data. The problem statement for predicting fluid flows models the physics in form of partial differential equations that is defined on a domain, along with some boundary conditions on the boundaries and some initial conditions. To incorporate such physics into the training step of the neural network, the parameterized form of the problem is considered. Let us denote the neural network by $U(x,t,\theta)$ and the function to be learnt by $\hat{u}(x,t)$, where learning is based on minimizing a multi-objective loss functional

$$\mathcal{L}(\theta,\nu)=\sum_{i=1}^{k}\mathcal{L}_i(\theta,\nu).$$

In PINNs, the residuals of the partial differential equation, initial, and boundary conditions are included, where we used the L^2-norm $\|\cdot\|_0$ (mean squared error / MSE) on uniformly sampled collocation points prior to training,

The PINNs with space and time coordinates as described in[?] consist of a multiple dense layers, with weights and biases as described in Section 3.1, along with a gradient layer. The derivatives of the output of the vanilla neural network, u, are used for calculating the strong residuals of the partial differential equation. The norm of these residuals, together with norms of residuals for initial and boundary conditions, are part of the loss functions. The loss function, which uses only the physics-based information is defined by

$$\mathcal{L}(\theta,\nu)=\mathcal{L}_{\text{PDE}}+\underbrace{\mathcal{L}_{\Gamma_1}+\ldots+\mathcal{L}_{\Gamma_n}}_{=\mathcal{L}_{\text{IC}}}+\underbrace{\mathcal{L}_{\Theta_1}+\ldots+\mathcal{L}_{\Theta_m}}_{=\mathcal{L}_{\text{BC}}}+\mathcal{L}_{\text{data}},\tag{6}$$

with Is it correct that $\mathcal{L}_{\text{data}}$ belongs to this functional? Shouldn't it appear only in the hybrid NN?

$$\mathcal{L}_{\text{PDE}}=\frac{1}{|\tilde{\Omega}|}\sum_{x,t\in\tilde{\Omega}}\|\mathcal{E}\|_0^2,\quad \mathcal{L}_{\text{IC}}=\frac{1}{|\tilde{\Theta}|}\sum_{t\in\Theta_i}\|\mathcal{I}\|_0^2,\quad \mathcal{L}_{\text{BC}}=\frac{1}{|\tilde{\Gamma}|}\sum_{t\in\Gamma_i}\|\mathcal{B}\|_0^2.$$

There are a number of undefined symbols included. Do we need them? The training process solves the following optimization problem

$$\text{argmin}_\theta\mathcal{L}(\theta,\nu\mid(x,t,y)).$$

PINNs use the gradient layer and take the whole data driven network as an input, so that the result of the gradient layer provides derivatives of the approximation to the solution of the partial differential equation predicted by the network.

Example 1 (Wave equation)**.** The following loss function shall enforce the network to compute an approximation $\hat{u}(x,t)=U(x,t,\theta)$ of the solution $u(x,t)$ of the initial-boundary value problem for the wave equation (1), (2):

$$\mathcal{L}=\underbrace{\frac{1}{|\Omega|}\sum_{(x,t)\in\Omega}\left\|\frac{\partial^2U(x,t,\theta)}{\partial t^2}-c^2\frac{\partial^2U(x,t,\theta)}{\partial x^2}\right\|_0^2}_{\mathcal{L}_{\text{PDE}}}+\underbrace{\frac{1}{|\tau_1|}\sum_{(x,t)\in\tau_1}\left\|U(x_0,t,\theta)\right\|_0^2}_{\mathcal{L}_{\tau_1}}+\underbrace{\frac{1}{|\tau_2|}\sum_{(t,x)\in\tau_2}\|u(x_{\text{end}},t,\theta)\|_0^2}_{\mathcal{L}_{\tau_2}}$$
$$+\underbrace{\frac{1}{|\gamma_1|}\sum_{x\in\gamma_1}\|U(x,0,\theta)-u_{\text{in}}(x)\|_0^2}_{\mathcal{L}_{\gamma_1}}+\underbrace{\frac{1}{|\gamma_2|}\sum_{x\in\gamma_2}\left\|\frac{\partial U(x,0,\theta)}{\partial t}-u'_{\text{in}}(x)\right\|_0^2}_{\mathcal{L}_{\gamma_2}}.$$

This is not clear.

- Do we take the L^2 norm in the space time domain? Then we do not need to sum anything.
- Do we take the L^2 norm only in space? Then we have to sum over the time steps.
- Or do we use an l^2 norm for the vector? Then we need to sum. A number of symbols have to be introduced.

Such notations like $(x,t)\in\Omega$ are wrong.

3.3 | Neural networks and optimization of hyperparameters

The effects of using different architectures of the neural networks with respect to the number of layers and nodes per layer were investigated in preliminary studies. We found that the impact of these parameters on the aspect we are most interested in, namely the predictions for unseen situations, was only weak. For the sake of brevity, we will present results only for a network with 6 deep (dense) layers containing $[128, 64, 32, 32, 64, 128]$ nodes, respectively. This is the ame configuration that was used in[?]. you said that there is some other paper which uses the same network

Our preliminary studies also showed that the choice of the activation function might considerably influence the quality of the predictions from the neural networks. In Section 4, results obtained with different activation functions will be presented: $\tanh(x)$, at least two more activation functions

The library TENSORFLOW[?] was used for constructing the neural networks and for solving the optimization problems. The optimization problems were solved with the limited-memory BFGS (L-BFGS) alogrithm[?]. This popular method is a quasi Newton method, which approximates the Hessian on the basis of a prescribed maximal number of previous iterates. number of vectors used in the numerical simulations, initial value for optimization

4 | RESULTS

Numerical solutions for the considered initial-boundary value problems were computed with a finite difference method in space and temporal discretization. For the wave equation, the Lax–Wendroff scheme[??] was utilized. These solutions were used to train the data-driven neural network.

We have investigated several aspects of the numerical simulations. First, the convergence of the optimization process was studied, where we explored which terms of the multi-objective loss functional were easily or hard to minimize. Then, of course, the quality of the solution predicted by the networks was studied. On the one hand, we considered exactly the interval were the data came from. And on the other hand, we exploreed what happens if a somewhat larger time interval is considered, i.e., the predictions of the network were applied to an unseen situation. We studied also the impact of using different activation functions. For the sake of brevity, only selected and representative results will be presented.

4.1 | Wave equation

We consider equation (1) for $x \in [0, 1]$, $t \in [0, 1]$, and $c = 1$. A solution of this equation is given by

$$u(x, t) = \frac{1}{2}\sin(\pi x)\cos(\pi t) + \frac{1}{3}\sin(3\pi x)\sin(3\pi t).$$

This function is for the boundary conditions a $u(0, t) = u(1, t) = 0$ and the initial conditions

$$u_0 = u(x, 0) = \frac{1}{2}\sin(\pi x), \quad \frac{\partial u}{\partial t}(x, 0) = \pi \sin(3\pi x)$$

the solution of the initial-boundary value problem (1) – (2).

The finite difference solution was obtained with $\Delta t = \Delta x = 0.0025$, i.e., with 400 discrete steps both in time and space. The training of the data-driven network was performed only in the time interval $[0, 0.8]$. The numerical solution at every timestep was used. Likewise, the PINNs was also trained for $t \in (0, 0.8)$. The application of the trained networks to an unseen situation consists in predicting the solution in $[0.8, 1]$.

First, results for the activation function $\tanh(x)$ will be presented and discussed. As already explained in Section 3, the formulation of the loss functional is different for both networks. Figure 1 illustrates how the each of the residual term in (**??**) is minimized in the multi-objective optimization. Moreover, we see that the estimation of hyperparameters requires less iterations using the data-driven network. However, the performance for the PINN is quite also satisfactory given no pre-computed solutions are required to train the network. the pictures need more explanations. I do not understand that the overall loss is smaller than the largest individual loss.

Figure 2 (left) shows the comparison of the results of the data- and physics-driven neural networks to the finite difference solution. In the right-hand side pictures the error maps for both networks are depicted, where the (abolute value of the) error in each node in space and time was computed. Whereas the errors in $t \in [0, 0.8]$ are of the same size for both networks, at most of the order 0.075, it can be seen that the PINNs performed better than the data-trained network for $t \in [0.8, 1]$. Before, the sentence was the other direction. I changed in it accordance with the labels of the pictures.

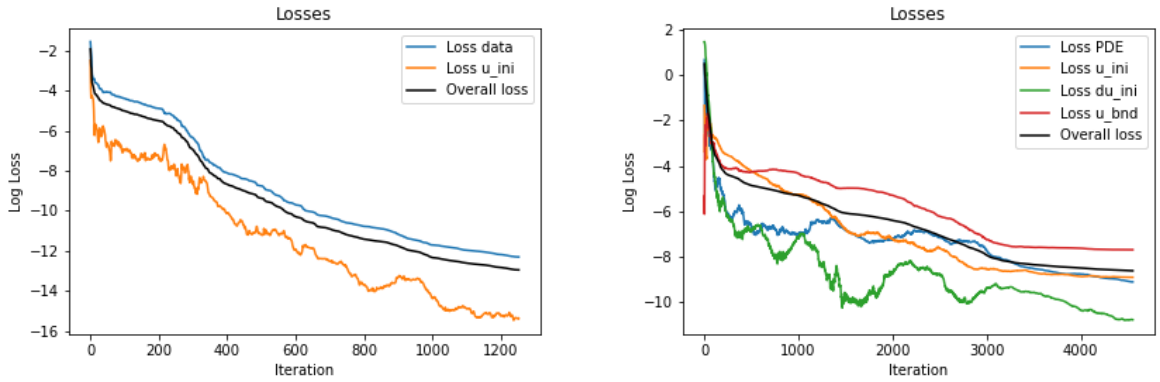

FIGURE 1 Wave equation. Activation function $\tanh(x)$. Convergence of the optimization process for minimizing the loss functionals and their individual terms. Note the different scaling of the abscissa.

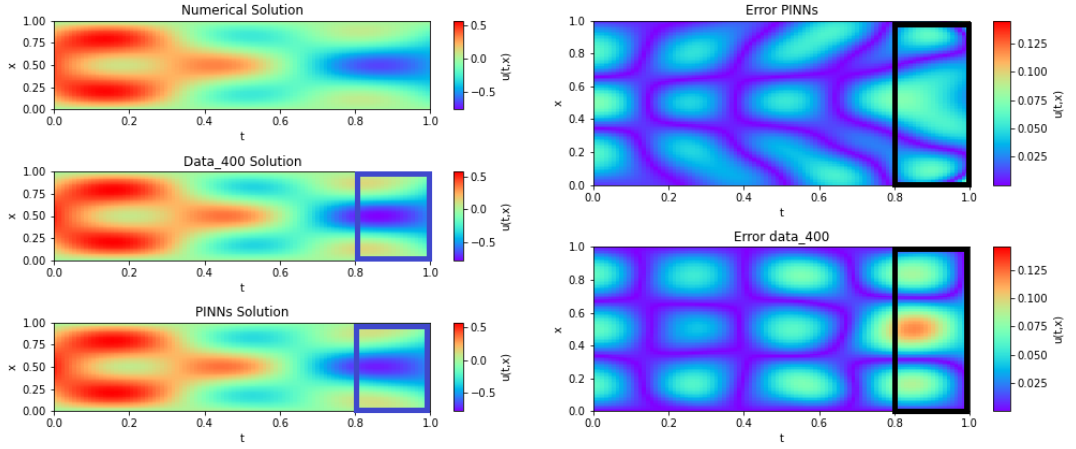

FIGURE 2 Wave equation. Activation function $\tanh(x)$. Left: Comparison of solution $u(x,t)$ obtained from data and physics trained network against reference solution. Right: The error map plotted for solutions obtained from data and physics trained network.
headlines of the pictures: 'finite difference solution', 'data-driven NN', 'PINN'
on right: data-driven picture above PINN picture (same sequence as on left)
'400' is irritating, since training for data used only 320 (321) finite difference solutions

Figure 3 presents more information on the prediction of the wave velocity. In addition to using all time steps in $[0, 0.8]$ for learning the data-driven network, also the case of using only every 4th time step. One can observe that in $[0, 0.8]$ the predictions from all networks are quite similar. However, there are differences for the unseen situation. The data-driven network lead to predictions with a much larger error in the middle of the interval for $t = 0.879$. And for longer time horizons, $t > 1.2$, the PINN prediction is significantly more reliable. To be honest, I do not see the new information of the left picture.

4.2 | Shallow water equation

The initial boundary value problem (3) – (4) for the shallow water equations was solved in $(x, t) \in [0, 1] \times [0, 0.2]$. The velocity is considered to be at equilibrium at $t = 0$, i.e. $u(x, 0) = 0$. Two different initial profiles for $h(x, 0)$ will be considered, leading to

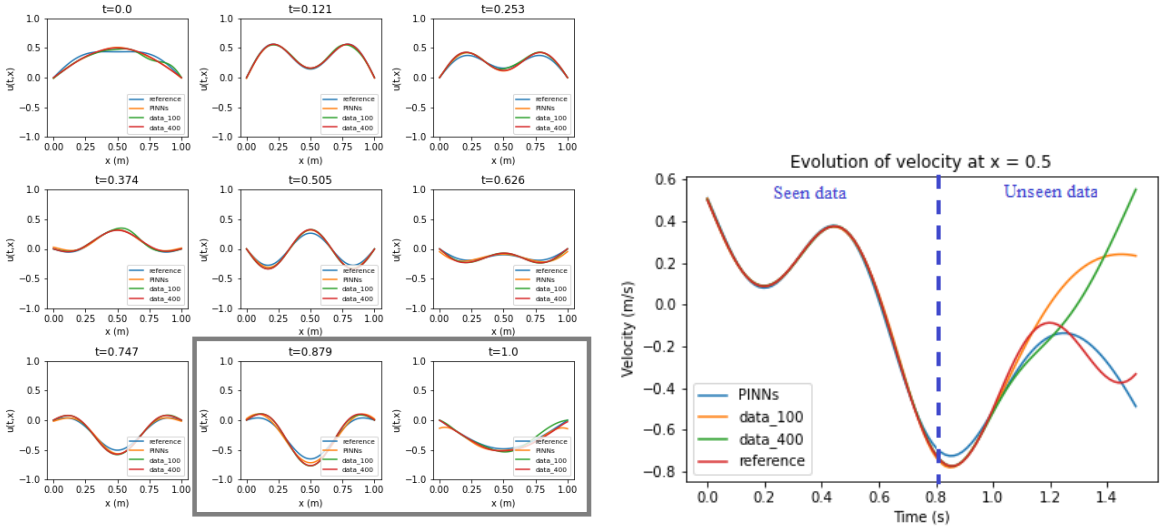

FIGURE 3 Wave equation. Activation function $\tanh(x)$. Comparison of the finite difference solution against solutions obtained from the data- and physics-trained networks. Left: $u(x,t)$ plotted at some time instances t. Right: Temporal evolution of the error at $x = 0.5$.

once more, '100' and '400' are disturbing, it should be '81' and '321'.

the same things should have in both pictures the same colors.

For $t \approx 0.85$, the yellow and green curce should be much more away from the red curve in the right picture, because the error is large.

solutions with different features. provide information about compuation of the solution to compare with: discretizations in time and space, finenss

Besides the data-driven neural network and the PINN, we will study for this example also a hybrid network, where the loss functional is the following linear combination of the data part (5) and the physics part (6) give concrete linear combination All networks will be trained with the numerical solution from $[0, 0.15]$ and the unsee situation is the solution in $(0.15, 0.2]$.

4.2.1 | Initial profile $h(x,0)$ leading to a smooth solution

The initial profile of the height considered in this example has the form $h(x,0) = \frac{1}{2}\sin(\pi x)$. The numerical solution of (3) – (4) can be seen in the upper part of Figure 5 .

Comparisons of the results obtained with the data-driven neural network and the PINN are presented in this picture as well and the corresponding errors in Figure 5 . For the seen interval $[0, 0.15]$, the results with the data driven neural network are clearly more accurate. This situation changes somewhat for the unseen interval, where the velocity error for the data driven network becomes large at the left spatial boundary and $t \to 0.2$. But altogether, one can conclude that both neural networks give satisfactory results for $t \in [0, 0.2]$

However, extending the unseen time interval a little bit further, the prediction from the data-driven neural network becomes unreliable, compare Figure 7 .

shall we present results for the hybrid network here, or just say that they are not much different, because the results for the other networks are not much different as well?

4.2.2 | Initial profile $h(x,0)$ leading to a solution with shocks

We solved the shallow water equation for a slightly more realistic dam flow example where the initial profile of $h(x,0) = 2+sin(2\pi x)$ results in a more complex wave propogation. The velocity is considered to be at equilibrium at $t = 0$, i.e. $u(x,0) = 0$. The solutions are compared against the reference/numerical solutions in figure 8 . Both data and physics driven network learn the solution within a 10^{-2} accuracy as can be seen in figure 9 .

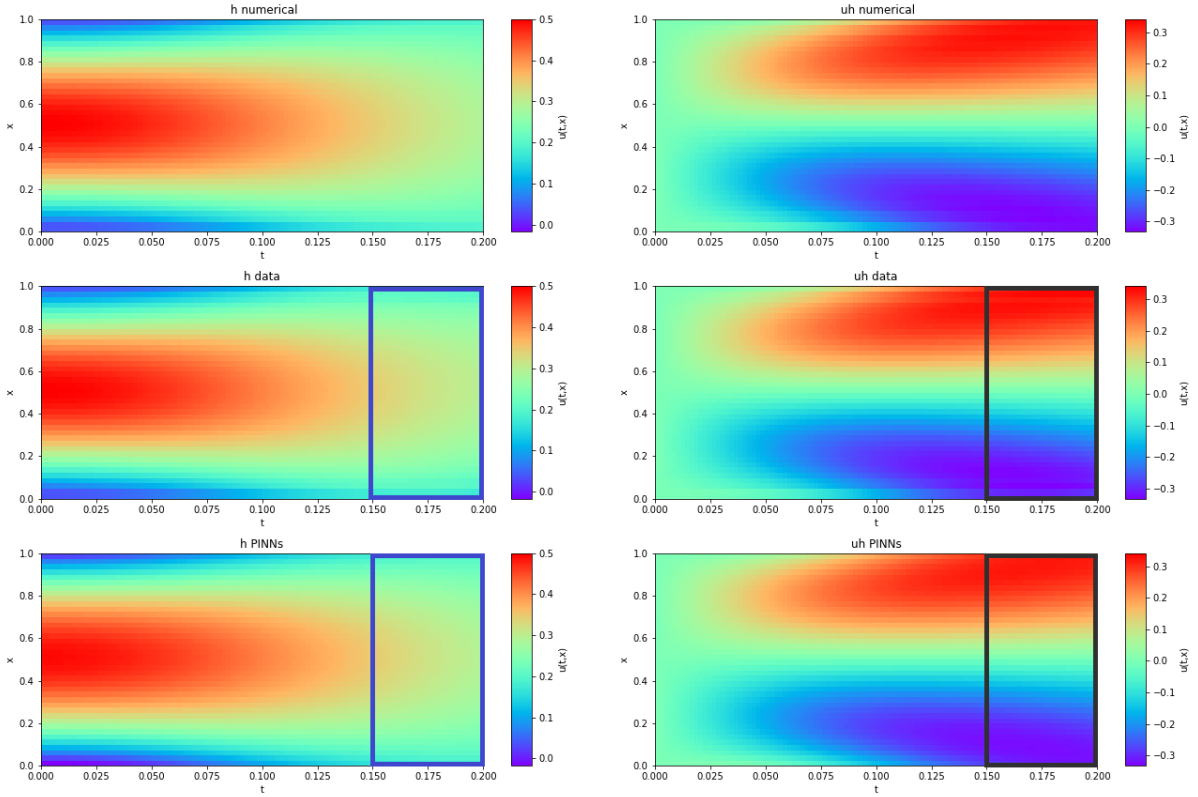

FIGURE 4 Shallow water equations, smooth solution. ctivation function tanh(x). Solution of the initial boundary value problem (top), solution obtained with the data-driven neural network (middle), solution obtained with the PINN (bottom).

For feed forward predictions of the solution for the time that's not incorporated in the training, the hybrid approach works better than PINNs. It can be seen in figure 10 that when looked on a micro scale, data-driven network overestimates the predicted velocities. Overall, learning for both networks is quite satisfactory for the case of gradual propogation of the wave.

5 | CONCLUSION

Cant colnclude that PINNS works best for SWE. changing layer and node configuration hardly makes a difference in training the networks. In the case of smooth solutions of wave models, PINNs predicts velocity(density) and/or height rather accurately. However for more complex/coupled systems, data-driven networks seem to work better. However data-fed networks need exhaustive datasets (solutions of model equations). In the absence of exhaustive data for learning of complex solutions, a hybrid training is recommended.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

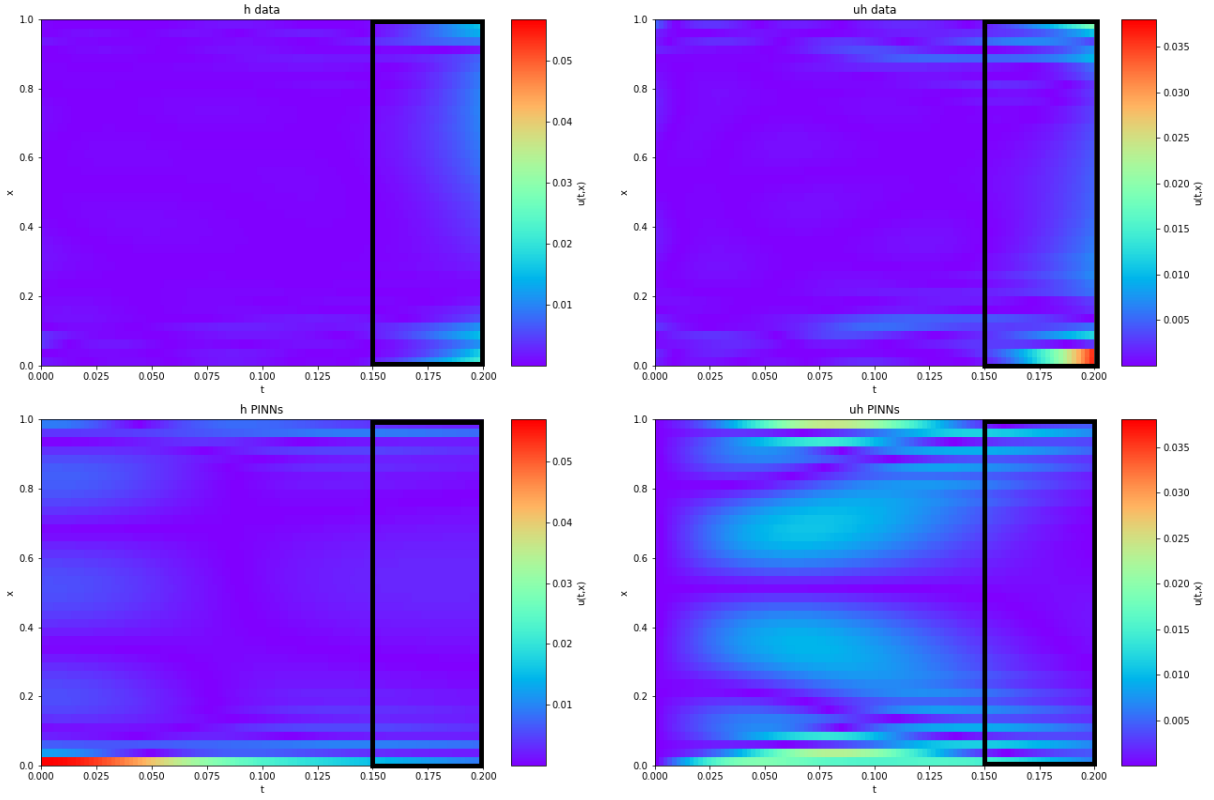

FIGURE 5 Error maps of $h(x,t)$ and $u(x,t)$ for the data driven neural network (left) and the PINN (right).

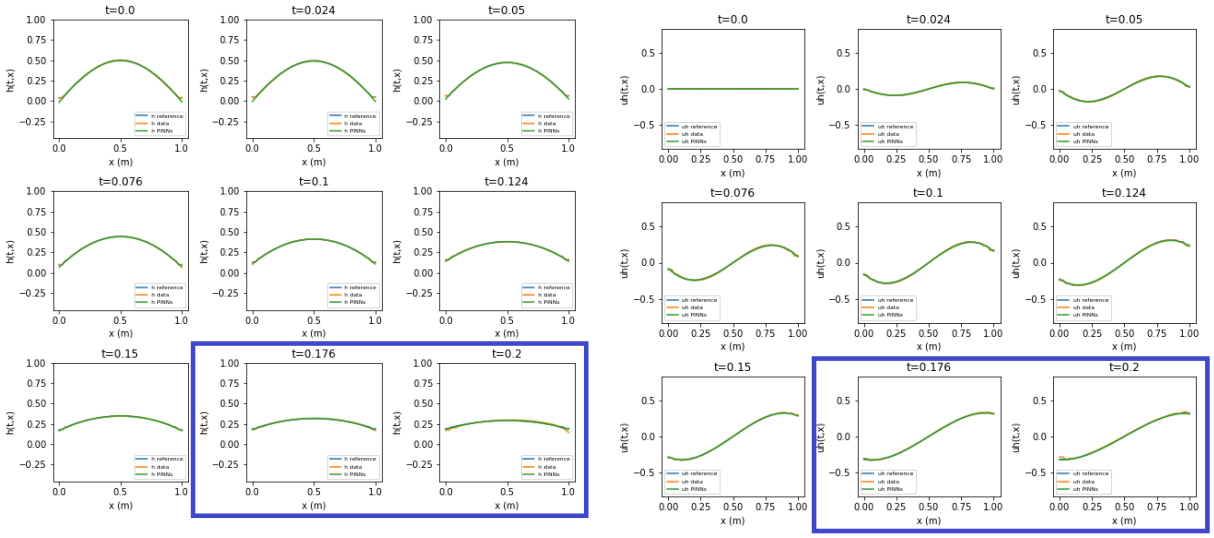

FIGURE 6 Shallow water equations, smooth solution. ctivation function tanh(x). Solutions $h(x,t)$ and $u(x,t)$ plotted at different t. Do we need this picture? What are the new information?

AUTHOR CONTRIBUTIONS

ZL developed the code base for physics informed deep learning neural networks and put the article together. WN trained, and evaluated the models. VJ reviewed and refined the results. All authors contributed to the article and approved the submitted version.

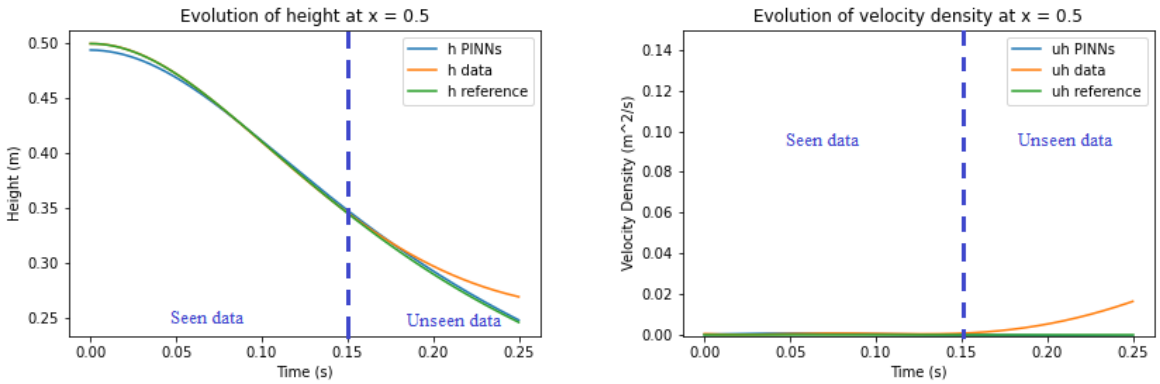

FIGURE 7 Shallow water equations, smooth solution. ctivation function $\tanh(x)$. Solutions $h(x = 0.5, t)$ and $u(x = 0.5, t)$ forward-in-time predictions.

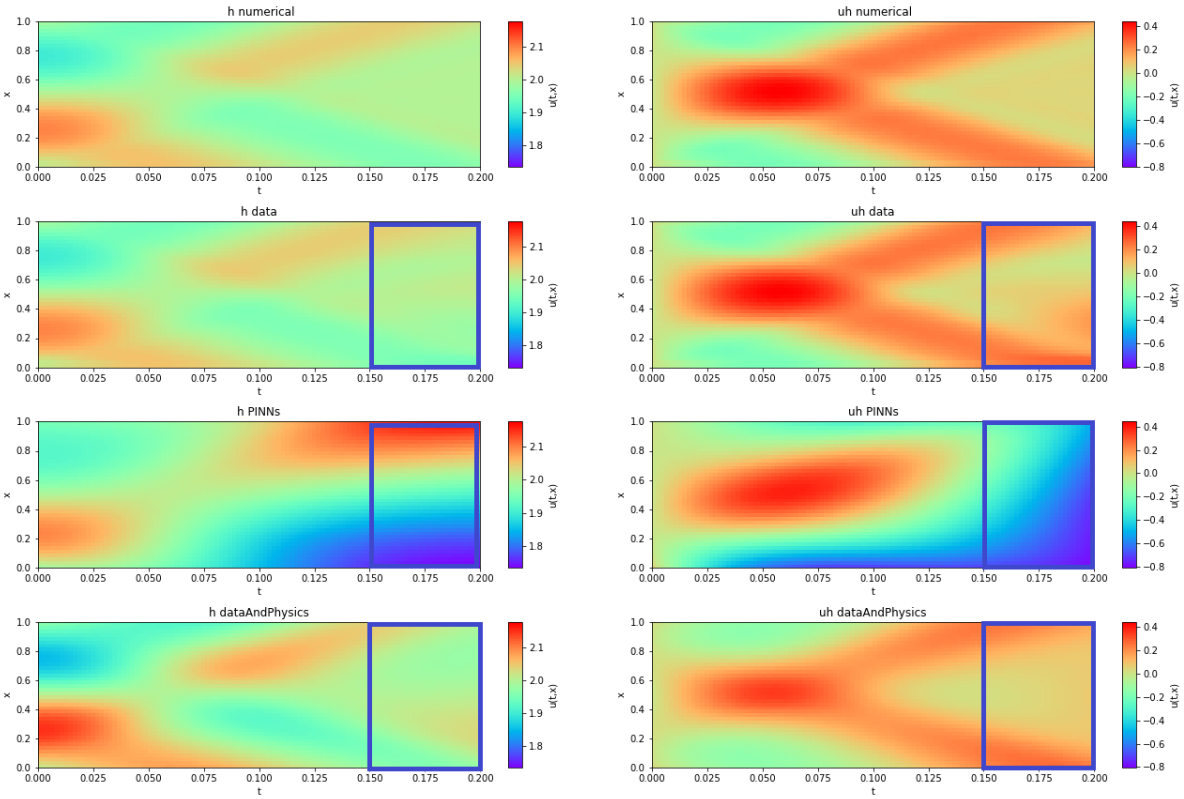

FIGURE 8 Case 2.2 Comparison of the $h(x, t)$ *and* $u(x, t)$ obtained from numerical, data and model-trained network

FUNDING

We acknowledge support from the DAAD through grant awards. We thank DAAD for their support.

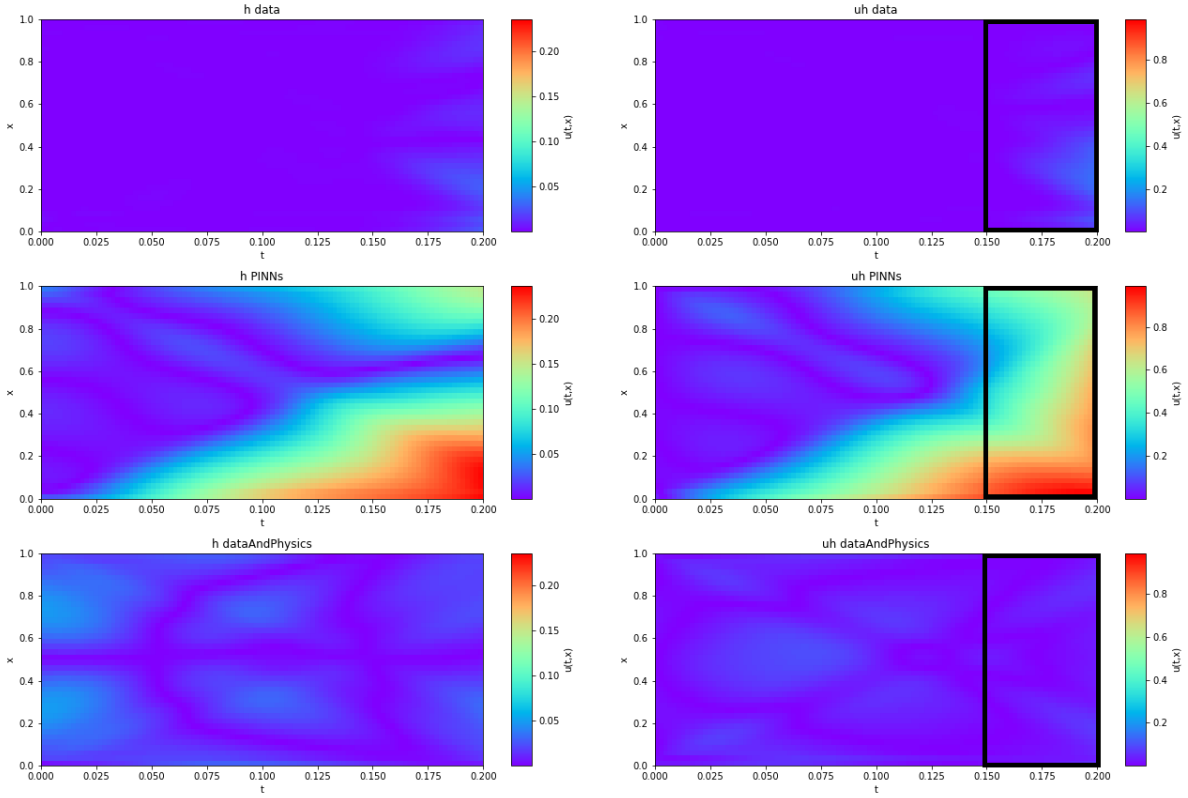

FIGURE 9 Case 2.2: The error maps of $h(x,t)$ and $u(x,t)$ is plotted as a color map.

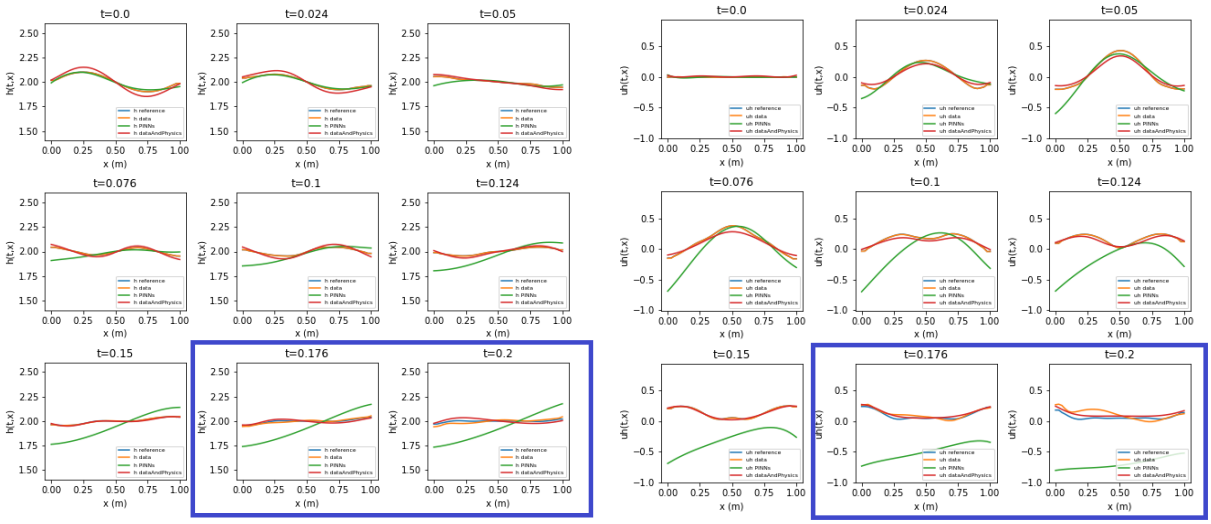

FIGURE 10 Case 2.2: Solutions $h(x,t)$ and $u(x,t)$ plotted at different t.

ACKNOWLEDGMENTS

We thank Prof. Muhammad Abubakr and Prof. Karsten Berns for helpful discussions at the early stage of this work.

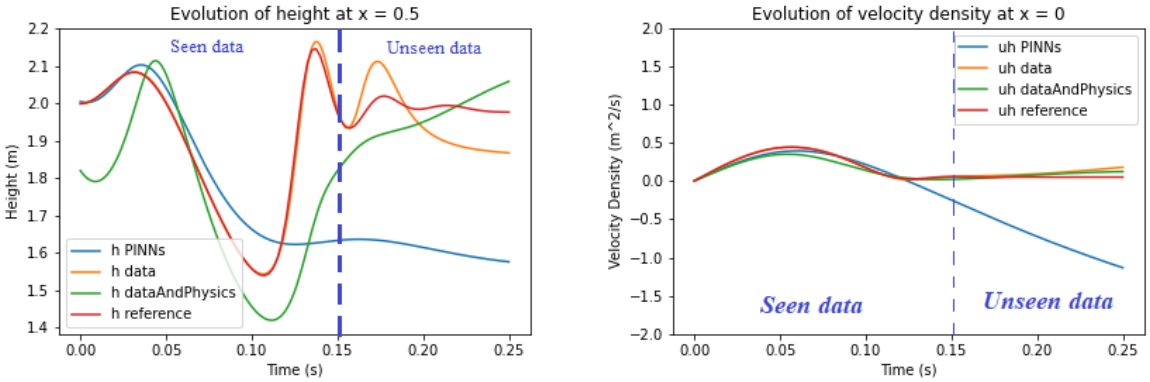

FIGURE 11 Case 2.2: Solutions $h(x = 0.5, t)$ and $u(x = 0.5, t)$ forward-in-time predictions.

SUPPLEMENTAL DATA

Supplementary Material should be uploaded separately on submission, if there are Supplementary Figures, please include the caption in the same file as the figure. LaTeX Supplementary Material templates can be found in the Frontiers LaTeX folder.

DATA AVAILABILITY STATEMENT

The datasets [GENERATED/ANALYZED] for this study can be found in the [NAME OF REPOSITORY] [LINK].