ASCE

# Physics-Informed Neural Network Approach for Solving the One-Dimensional Unsteady Shallow-Water Equations in Riverine Systems

Zeda Yin, S.M.ASCE[1]; Jimeng Shi[2]; Linlong Bian, S.M.ASCE[3]; William H. Campbell[4]; Sumit R. Zanje, S.M.ASCE[5]; Beichao Hu[6]; and Arturo S. Leon, M.ASCE[7]

**Abstract:** In recent years, many researchers have used machine learning approaches to bridge the relationship between big data and physics in the practical engineering field. However, the widely used machine learning models are highly dependent on the quality and quantity of data. These long-term monitoring data usually are expensive to obtain in water system. This paper presents a novel neural network structure, the physics-informed neural network (PINN), which can implement the shallow-water equations (SWEs) directly so that the training stage is based fully on physical laws. Similar to numerical models, our PINN model requires the same data as the numerical method, e.g., boundary conditions, the digital elevation of the terrain, and so forth. Because the SWEs are solved directly in our framework, this framework can be understood as a data-free method. The PINN was tested using two case studies: a flow spike in a hypothetical trapezoidal channel, and a historical scenario of downstream Cypress Creek, Houston. The results indicated great agreement with the widely used numerical solver, HEC-RAS. **DOI: 10.1061/JHEND8.HYENG-13572.** 

**Author keywords:** Physics-informed neural networks (PINN); Shallow-water equations (SWEs); Flood predictions.



## Introduction

Shallow-water equations (SWEs) are the most important and only governing equations for open-channel flow in computational hydraulics (Brunner 2002; Zhou 1995; Stansby and Zhou 1998; Toro 1992; Palu and Julien 2020). In the last few decades, numerical solutions have been considered widely as the most effective approach for solving nonlinear partial differential equations (PDEs), including Navier–Stokes equations, SWEs, and so forth (Desrochers et al. 2020; Labuhn et al. 2020; Eivazi et al. 2022; Huang et al. 2022; Hu and McDaniel 2023). However, the numerical method needs to balance many aspects in real applications (e.g., order of scheme accuracy, numerical convergence, and Courant numbers). In recent years, many researchers have investigated new techniques to solve the nonlinear PDEs for conventional engineering problems. Machine learning (ML) is one of the rapidly rising and latest techniques that have been used widely in the engineering field. Deep learning (DL) models can bridge approximation relations between input and output variables by conducting multiple elementary operations constructed by artificial neural networks (ANNs). The loss function normally is constructed to describe the difference between ANN outputs and observational ground truth [mostly using mean squared error (MSE) or mean absolute error (MAE)]. By minimizing the loss function, ideally to zero, through various optimizations, an ANN model can predict results that are very close to the real solutions.

In hydrology and hydraulics, a number of previous studies achieved great success by using this end-to-end, black-box ML model (Tamiru and Dinka 2021; Zhao et al. 2019; Sadler et al. 2018; Adnan et al. 2021; Li and Brewer 2020; Bui et al. 2020; Kim and Kim 2020; Sriram 2021; Wu et al. 2020; Zahura et al. 2020; Zhang et al. 2020). Shi et al. (2023) evaluated the performance of five distinct deep learning models in predicting water stages at the Miami River. The training data set was obtained from the historical records, and the deep learning models demonstrated superior performance compared with the numerical solver, particularly when the training data were sufficient. Song et al. (2023) introduced a model based on point-to-point methodology for predicting flow patterns around bridge piers within a channel. This approach effectively handles the training data derived from unstructured meshes.

Conventional image-based networks such as convolutional neural networks (CNNs) used to have difficulties to deal with those image data from unstructured meshes. However, this end-to-end, black-box ML model has several drawbacks. Firstly, this type of ML model requires a large amount of data for training, and this amount is even higher for DL models. Obtaining such an amount of data is very difficult, and normally expensive. DL models often provide low-accuracy results if the training data set is not sufficient.

[1]Ph.D. Candidate, Dept. of Civil and Environmental Engineering, Florida International Univ., 10555 W Flagler St., EC 2400, Miami, FL 33174 (corresponding author). ORCID: https://orcid.org/0000-0001-9146-5097. Email: zyin005@fiu.edu

[2]Ph.D. Candidate, Knight Foundation School of Computing and Information Sciences, Florida International Univ., 11200 SW 8th St., CASE 352, Miami, FL 33199. Email: jshi008@fiu.edu

[3]Postdoc, Dept. of Civil and Environmental Engineering, Florida International Univ., 10555 W Flagler St., EC 2400, Miami, FL 33174. Email: lbian@fiu.edu

[4]Master's Student, Dept. of Civil and Environmental Engineering, Florida International Univ., 10555 W Flagler St., EC 2400, Miami, FL 33174. Email: wcamp003@fiu.edu

[5]Postdoc, Dept. of Civil and Environmental Engineering, Florida International Univ., 10555 W Flagler St., EC 2400, Miami, FL 33174. Email: szanj001@fiu.edu

[6]Ph.D. Candidate, Dept. of Mechanical and Material Engineering, Florida International Univ., Miami, FL 33174. Email: bhu007@fiu.edu

[7]Associate Professor, Dept. of Civil and Environmental Engineering, Florida International Univ., 10555 W Flagler St., EC 2400, Miami, FL 33174. ORCID: https://orcid.org/0000-0002-7117-7351. Email: arleon@fiu.edu

Note. This manuscript was submitted on November 7, 2022; approved on September 16, 2024; published online on November 5, 2024. Discussion period open until April 5, 2025; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Hydraulic Engineering***, © ASCE, ISSN 0733-9429.

Secondly, because this type of ML is a pure black-box model, the computation process is almost impossible to interpret and transfer to human knowledge. This situation often is described as data rich, knowledge poor (Iskhakov and Dinh 2020). Lastly, not all of these DL models are physics-based or partially physics-based. For non-physics-based models, the model output cannot reflect the changes in physical conditions, so the model output is completely unconstrained by physics, therefore, the model may provide an inaccurate or even nonsense output when the physical conditions change. For partially physics-based models, sufficient physics input variables are fed to ML models to obtain the outputs. For example, the density of the fluid, the acceleration of gravity (which sometimes can be ignored because it is constant in most cases), and the height of the fluid above the object can be used as input features to feed the ML model to obtain the hydrostatic pressure. Research shows that this method is physics-based and its output can reflect the change in physical conditions. However, machine learning can only partially fit the physical law constraint in most cases because it is impossible for the training data set to cover the entire possible domain. These partially physics-based ML models may perform well if the test data are within the range of the training domain, but they may provide a larger error if the test data are far from the training domain (Willard et al. 2020). This problem is even more obvious when the physics law is very complex.

To overcome this problem, a new class of deep learning models, called physics-informed neural networks (PINNs), have been developed in recent years (Cai et al. 2021b). Instead of minimizing mean squared error from training data, PINNs minimize the L1 or L2 norm of the loss function that is constructed by the governing equations (which primarily are PDEs). PINNs have achieved great success in many fields, including computational fluid dynamics (CFD) (Eivazi et al. 2022; Huang et al. 2022; Hu and McDaniel 2023; Iskhakov and Dinh 2020; Mao et al. 2020; Yang et al. 2019; Jin et al. 2021), heat transfer (Cai et al. 2021a; Bararnia and Esmaeilpour 2022), and so forth. Some researchers applied PINNs to solving SWEs as well. Bihlo and Popovych (2022) applied a PINN to solve the spheric atmospheric waves problem. De Wolff et al. (2021) discretized hyperbolic ocean modeling equations using the finite-element method and solved for ocean systems using a PINN. Hurler (2021) ignored the digital elevation terrain effect and solved the one-dimensional (1D) shallow-water equation on a very small scale.

Despite the rapid growth of the PINN approach in the last 3–4 years, its application to SWEs, as well as to the broader fields of hydrology and hydraulics, remains understudied. Although several studies have attempted to utilize PINNs to solve PDEs, most of them have concentrated on solving other PDEs such as the Navier–Stokes and Burgers equations. Only Hurler (2021) endeavored to apply PINNs to the SWEs. Unfortunately, Hurler focused on an ideal microscale scenario, and completely neglected the impact of terrain and bathymetry changes and friction, which is far from application level. Moreover, the commonly used flow rate form of SWEs has never been investigated in PINN-related work. In addition, many studies could not minimize the loss function to a sufficiently low level, so the PINN error is large, which makes it hard to use in real applications. Most importantly, rivers, canals, natural streams, and creeks are naturally open-channel flow, and their water stages and flow rates are heavily influenced by the terrain. According to authors' knowledge, there is no existing PINN framework that can solve these open-channel flow problems with the consideration of the terrain information and friction. This study not only applies the state-of-the-art PINN approach to the hydraulic field, but also provides a fundamental and highly extensible framework for future research to build upon.

To achieve the objective, this paper provides a detailed account of how to apply the PINN approach to solve open-channel problems in a riverine system. The PINN can work as a surrogate model for conventional numerical models (e.g., HEC-RAS). Our PINN model takes exactly the same input variables as HEC-RAS and provides the same output variables. The hydraulic parameters are calculated numerically using the terrain shape data extracted from the digital elevation model (DEM). The numerical calculation process is coupled into our code. To overcome the accuracy problem and training time problem, we improved the original PINN framework by adding weights to the loss function components and training neural networks using the multistage method, which is presented in detail in the "Methodology" section. To better demonstrate the approaches and results, this paper employed some case studies to explore the effect of different forms of SWEs and different study areas. After the construction, the PINN was tested using two different cases: a hypothetical scenario, and a historical scenario on downstream Cypress Creek, Houston.

## Methodology

### One-Dimensional Shallow-Water Equations

One-dimensional SWEs (the unidirectional form is called the Saint-Venant equations) can be derived from Navier–Stokes equations after a few assumptions. Because it assumes that pressure distribution is approximately hydrostatic pressure, the classic pressure–velocity coupling problem does not exist. Due to the benefits of these simplifications, One-dimensional SWEs are used widely in large-scale hydraulic systems as governing equations. One-dimensional SWEs consist of the mass conservation equation and the $x$-direction momentum equation. According to Chaudhry (2014), the SWEs for cross-sections of arbitrary shape can be written

$$\frac{\partial \boldsymbol{U}}{\partial t}+\frac{\partial \boldsymbol{F}}{\partial x}=\boldsymbol{S} \tag{1}$$

where the vector variable $\boldsymbol{U}$, the flux vector $\boldsymbol{F}$ and the source term vector $\boldsymbol{S}$ are given by

$$\boldsymbol{U}=\begin{bmatrix} A \\ Q \end{bmatrix}, \qquad \boldsymbol{F}=\begin{bmatrix} Q \\ \dfrac{Q^2}{A}+gI_1 \end{bmatrix}, \qquad \boldsymbol{S}=\begin{bmatrix} 0 \\ gA(S_0-S_f) \end{bmatrix} \tag{2}$$

where $A$ = cross-sectional wetted area; $Q$ = flow discharge in cross section; $gI_1$ = hydrostatic thrust; $S_f$ = friction slope; $S_0$ = elevation slope of terrain; $t$ = time; and $x = x$-direction coordinate. The hydrostatic thrust part can be expressed as (Franzini and Soares-Frazão 2016)

$$g\frac{\partial I_1}{\partial x}=gA\frac{\partial h}{\partial x} \tag{3}$$

where $h$ = free surface water elevation.

The friction slope in fluid dynamics cannot be calculated analytically. In North America, the most common method to obtain the friction slope is using Manning's equations. By rearranging Manning's equations, we can express the friction slope as Eq. (4). The elevation slope of the terrain can be calculated based on the change of riverbed slope based on the DEM and bathymetry data. The elevation slope of the riverbed can be expressed as Eq. (5)

$$S_f=\left(\frac{nv}{KR^{0.667}}\right)^2 \tag{4}$$

$$S_0 = -\frac{dz}{dx} \tag{5}$$

where $n$ denotes Manning's roughness coefficient, which is set to 0.013 for the hypothetical case in this paper to represent a concrete channel (Chow 1959); $v$ = flow velocity in the cross section; $K$ is a unit conversion factor (1 for SI units, and 1.486 for imperial units); $R$ = hydraulic radius; and $z$ = bed elevation value from a certain datum.

There is no analytical approach for the calculation of the hydraulic parameter because the cross-sectional terrain shape is mostly highly irregular and complex. The numerical approach to calculate these hydraulic parameters is discussed in the following section. However, these hydraulic parameters [wetted cross-sectional area ($A$), wetted perimeter ($P$), hydraulic radius ($R$), and top width of river ($B$)] are dependent only on the cross-sectional water depth and terrain shape. The terrain shape can be obtained using the DEM and bathymetry. Thus, if we use $\mathrm{DEM}_x$ to denote the raster digital elevation data at all cross sections, we can express these hydraulic parameters in the form of Eq. (6). Furthermore, by substituting these hydraulic parameters into 1D SWEs, we can express the entire SWEs as a form of Eq. (7)

$$[A, R, P, B] = f(\mathrm{DEM}_x, h) \tag{6}$$

$$[v, h] = f(x, t, \mathrm{DEM}_x) \tag{7}$$

By the nature of the neural networks, a PINN approximates the unknown function by stacking elementary operations using the input variables $X$, weights $W$, and bias $b$. As the input of the neural networks, two input variables $x$ and $t$ for solving SWE can be expressed as a two-dimensional matrix: $X = [x, t]$. The output of the neural networks also is a two-dimensional matrix, $Y$, which contains the predicted solution to SWEs (transport scalars $\hat{v}$ and $\hat{h}$). Thus, the entire calculation process can be expressed as a function of input variables, weights, and bias

$$[\hat{v}, \hat{h}] = NN_{W,b} = \Phi(WX + b) \tag{8}$$

where $\Phi$ is the activation function of the neural networks.

### PINN Framework

Most PINN frameworks have three steps for solving the problem: a forward step, loss function construction, and a backward step. The overall PINN framework diagram for 1D SWEs used in this paper is shown in Fig. 1.

In the forward step, a fully connected neural network is employed to predict the output, $\hat{v}$ and $\hat{h}$. The ANN used in this framework predicts $\hat{v}$ and $\hat{h}$ from input variables $x$ and $t$ exactly as any other ANN would. However, the key difference between a PINN and standard methods lies in the subsequent step. When the equation outputs, $\hat{v}$ and $\hat{h}$, are predicted, automatic differentiation can be used to obtain the partial derivative terms. The detailed process of automatic differentiation is presented in the "Automatic Differentiation" section. In addition, with the information of DEM data, the numerical methods are employed to obtain four hydraulic parameters: $A$, $R$, $P$, and $B$. Because all these elements are available, it is possible to construct the mass and momentum of SWEs as Eqs. (1) and (2). In this paper, the most frequently used unsteady flow boundary condition (BC) types are the flow hydrograph for upstream and the stage hydrograph for downstream. The boundary conditions are coupled into the framework, which is discussed in detail in the "Coupling Boundary Conditions" section.

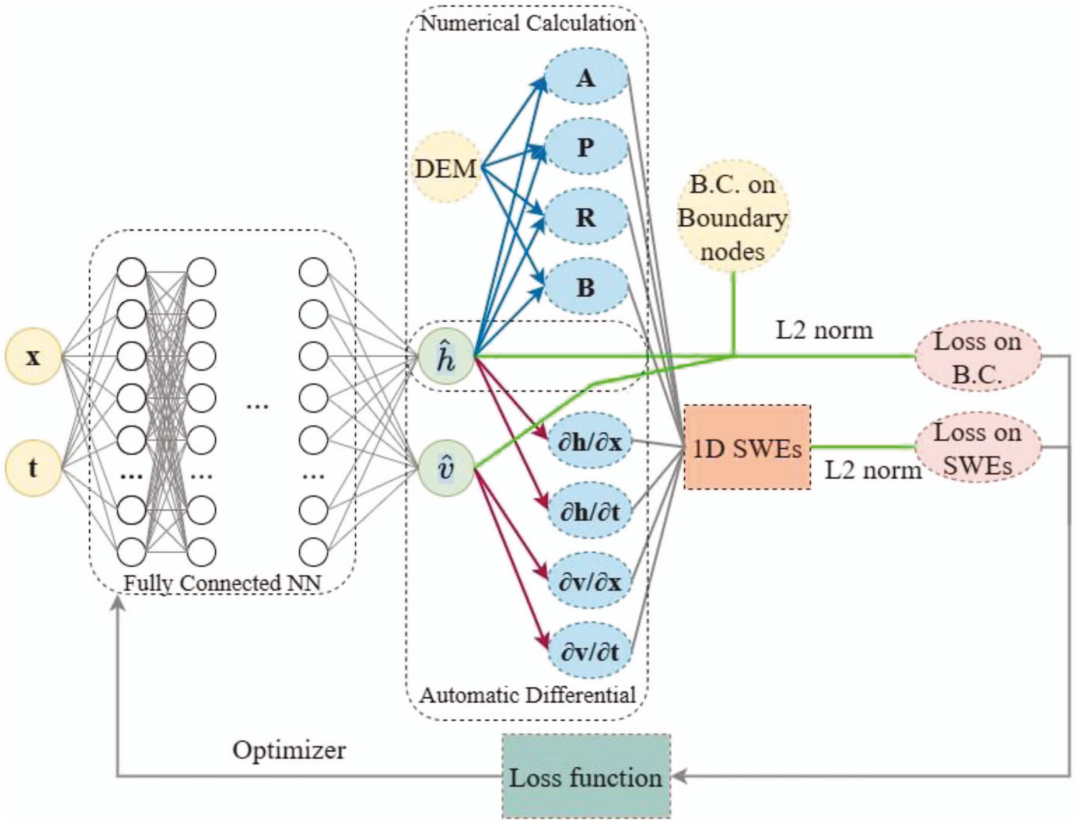


**Fig. 1.** (Color) Physics-informed neural network framework diagram for 1D shallow-water equations.

Before constructing the loss function, L2 norm operation is added on both the shallow-water equations and the boundary conditions to ensure that the loss value is always positive in this paper. The L2 norm operation can be expressed as

$$\|X\|^2 = \sqrt{\sum_{k=1}^{n} X_k^2} \tag{9}$$

where $X$ = difference between prediction value and target value; and $n$ = total number of sample points.

In this paper, the loss function is constructed by the summation of four components: loss from the mass equation, loss from the momentum equation, loss from the upstream boundary condition, and loss from the downstream boundary condition. When the loss function is constructed, the backward process can be conducted. In the backward process, the Adam optimizer (Kingma and Ba 2014) is employed to optimize the loss function. The final loss function value is sent back to the beginning of the framework to achieve a closed-loop optimization.

### Automatic Differentiation

The differentiation in the conventional numerical methods is calculated approximately using the discretization, and its accuracy is highly dependent on the order of the scheme. Furthermore, many problems can occur during this procedure, such as convergence, conservativeness, boundedness, and transportiveness. One of the benefits of neural networks is that the partial derivatives can be calculated easily using automatic differentiation. Thus, these problems do not exist in this framework. Automatic differentiation used in this framework can calculate the partial derivatives of neural network outputs evaluating their trace of composition. Because the computations in the neural networks consist of a finite set of elementary mathematical operations, the values in each elementary operation are known, and the derivatives can be calculated. Thus, the partial derivatives of the neural network outputs can be calculated by forward propagating the derivatives of each elementary mathematical operation (Baydin et al. 2018; Verma 2000). Automatic differentiation is used to compute the derivatives of constructed variables, which are then applied in the backward optimization process.

### Numerical Calculations on Hydraulic Parameters

The hydraulic parameters at each cross section are needed in order to construct the SWEs. These hydraulic parameters usually are difficult to obtain analytically because the shape of the cross sections usually is highly irregular and complex. To overcome this problem and generalize the framework, a numerical method based on DEM data is used in this work to obtain the hydraulic parameters $A$, $R$, $P$, and $B$. From the DEM data, we can obtain a finite number of sample points depending on the DEM resolution. Each sample point contains two values, [$a$, $b$], where $a$ is the cross-sectional distance from the start point, and $b$ is the elevation of these sample points. To simplify the numerical calculation, the $x$-axis is transformed from the original position to the predicted water depth by subtracting the entire elevation value from the predicted water depth. The transformation for numerical calculations is presented in Fig. 2.

After the transformation, all sample points with positive elevation values can be ignored. The remaining sample points, from $N_k$ to $N_m$ (Fig. 2), are used for the numerical calculation. By applying the trapezoidal rule, the cross-sectional wetted area can be expressed as Eq. (10). The wetted perimeter can be written as the summation of the truncated length of all neighboring points [Eq. (11)]. The hydraulic radius [Eq. (12)], is obtained simply by dividing the cross-sectional wetted area by the wetted parameters. Because the water free-surface always is parallel to the $x$-axis, the top width can be calculated by subtracting the first point from the last point [Eq. (13)]

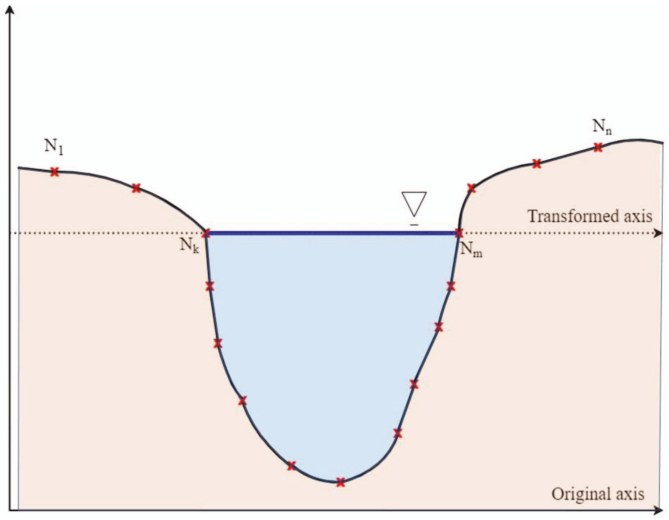


**Fig. 2.** (Color) Transformation for numerical method to calculate hydraulic parameters.

$$A = -\sum_{i=k}^{m-1} \frac{(b_{i+1} + b_i) * (a_{i+1} - a_i)}{2} \tag{10}$$

$$P = \sum_{i=k}^{m-1} \sqrt{(a_{i+1} - a_i)^2 + (b_{i+1} - b_i)^2} \tag{11}$$

$$R = \frac{A}{P} \tag{12}$$

$$B = a_m - a_k \tag{13}$$

### Coupling Boundary Conditions

Two major approaches can be used in coupling boundaries into the framework: hard-written, and soft-optimization. The hard-written approach can help obtain rapid convergence during the optimization stage because the framework only needs to optimize the loss functions based on two sets of equations: loss from the mass and momentum equations. However, this hard-written approach generates restrictions on the computation domain, which significantly influences the extrapolations. Therefore, to better generalize the framework, the soft-optimization approach is used in this paper. The framework of the soft coupling approach is shown in Fig. 3.

During one iteration of optimization, the predicted free-surface water elevation, $\hat{h}$, and velocity, $\hat{v}$, are treated separately based on their origins. For internal nodes, predicted free-surface water elevation, $\hat{h}$, and velocity, $\hat{v}$, are used to construct the SWEs, and then used to calculate the loss from SWEs. For boundary nodes, upstream predicted velocity, $\widehat{v_{bc}}$, and downstream predicted free-surface water elevation, $\widehat{h_{bc}}$, are used for the L2 norm operation with the given boundary conditions rather than constructing the SWEs. By minimizing the mean squared error of two boundary nodes, the boundary condition is coupled softly into the framework.

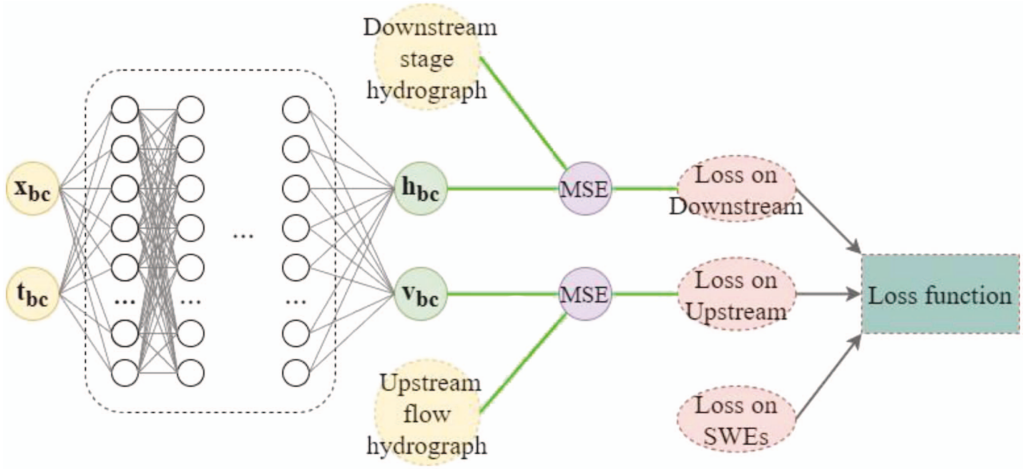


**Fig. 3.** (Color) Soft-optimization approach for coupling boundary conditions.

### *Improvements in PINN Framework to Solve Large-Scale Problems*

Dealing with large numbers always is an issue in deep learning. All previous PINN frameworks were tested only at the small scale—most tested domains ranged from −1 to 1. However, in real hydraulic applications, the computation domain of the variables usually has a large scale, ranging from $10^3$ to $10^5$ m. In addition, the input hydrograph often contains large numbers, especially when using flow rates as input. These situations may result in lack of convergence, dead neurons, and gradient vanishing problems. To overcome these problems, the linear activation function exponential linear unit (ELU) is used in this paper, and two improvements are added to the PINN framework: adding weights, and using multistage training.

#### Adding Weights to Loss Function Components

According to the methodology of this framework, the loss function for this framework consists of four components: loss value of upstream boundary conditions, loss value of downstream boundary conditions, loss value of SWEs mass equation, and loss value of the SWEs momentum equation. Ideally, these components should be summed together to construct the loss function. However, this will cause convergence and accuracy problems during the training phase because some terms may be significantly greater than others. For example, the loss value of downstream boundary conditions is dependent on the magnitude of water elevation, $h$, whereas the loss value of the SWEs mass equation is dependent on $\partial h/\partial t$ and $\partial v/\partial x$. However, for all common cases, the values of $\partial h/\partial t$, which usually are $10^{-3} - 10^{-8}$, are significantly less than the value of $h$ (which usually is $10^0 - 10^2$). In addition, the magnitude of the upstream boundary (flow rate, $Q$) also will be significantly greater than the magnitude of the downstream boundary (water elevation, $h$) if the flow rate–based PINN in the section "Flow Rate–Based 1D Shallow-Water Equations" is used. This different order of magnitude problem will make the neural network biased on certain components, which will heavily influence the output accuracy. Normal ML and DL models solve this problem by using normalization approach; however, the normalization approach technically cannot be added to this framework due to its nature.

To overcome this problem, we introduce and couple a weights estimation method into this framework (Fig. 4). The idea of this method is simple. We can manually multiply each loss function component by four different weights to ensure that the orders of the components are close before summing them together. The order of output variables $u$ and $h$ can be estimated using the known boundary conditions. Furthermore, $\Delta x$ and $\Delta t$ are dependent only on how the cross section is divided and on the chosen timestep; therefore, the order of the partial differential elements in the SWEs also can be estimated. The weight value is determined by the inverse order of the estimated values of the four loss terms. This estimation method is simple and intuitive, and it can be conducted easily in Python.

There is no need to distribute the final weights of the four components evenly, and it is undoable because the $v$ and $h$ are unknown before the neural network is trained. The concept of this method is to solve the large order of magnitude differences problem, and the small order difference on each component of order difference (within $10^4$) will not influence the output at all after testing.

#### Multistage Training Strategy

SWEs have infinite sets of solutions if they are not constrained by boundary conditions. Finding the final solution of SWEs is not time efficient in PINNs when trying to optimize four components at the same time, because the linear combination of SWEs always has major changes when the boundary condition is changed during the optimization. Ideally, it is better to optimize boundaries first, then optimize two SWEs components. However, this is almost impossible in a single neural network framework. To accelerate the optimization during the training stage, we use a multistage training strategy in this framework.

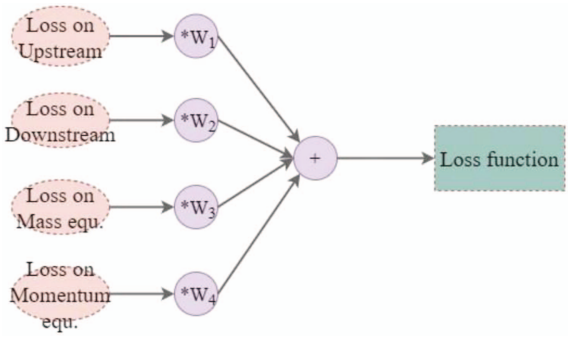


**Fig. 4.** (Color) Demonstration of weights estimation method to overcome the significant order of magnitude difference problem.

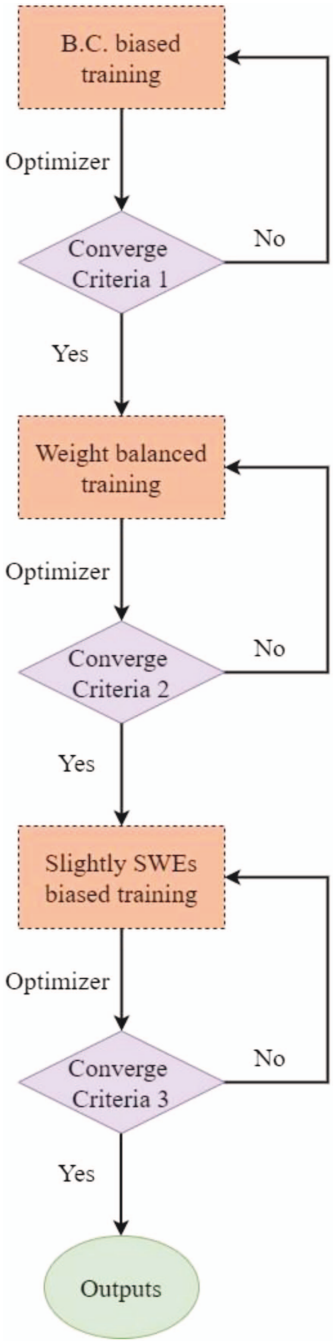


**Fig. 5.** (Color) Diagram of multistage training strategy.

Because we added weights for each loss function component in the last section, it is feasible to control the optimizer to emphasize the desired components. The diagram of the multistage training strategy is shown in Fig. 5. There are three stages for the entire training process, and each stage has unique weights, $W_1$–$W_4$, and learning rate, $lr$. Using superscripts to denote the stage number, there are a total of 12 weights ($W_1^1, W_2^1, \ldots, W_3^4, W_4^4$) and 3 learning rates ($lr^1, lr^2, lr^3$).

During the first training stage, the optimizer is biased on BC by enlarging $W_1^1$ and $W_2^1$. This BC-biased training can help neural networks rapidly converge to boundaries. After the loss value meets the first convergence criterion, weights $W_1^2$ and $W_2^2$ need to be decreased to balanced values, as stated in the last section, and it is helpful to decrease $lr^2$ as well to stabilize the convergence. Optionally, $W_3^3$ and $W_4^3$ can be increased slightly to achieve more-accurate results in the third stage. However, the third stage is not mandatory, and it could be compromised if pursuing higher time efficiency.

### Loss Function Expression

The loss function of a PINN is designed to minimize both the data mismatch at the boundary and the violation of physical laws. The PDE residual term ensures that the neural network's predictions satisfy the underlying physical laws, which, in our case, are the shallow-water equations. Based on Eqs. (1)–(4), the PDE residual term is expressed as Eq. (14). Because our upstream boundary condition is the flow rate or flow velocity, and the downstream boundary condition in our study is the water stage, the boundary loss terms are expressed as Eqs. (15) and (16). The total loss function, which combines these terms, is expressed as Eq. (17)

$$\mathcal{L}_{physics} = \frac{1}{N_{physics}} \sum_{i=1}^{N_{physics}} \left| \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} - \boldsymbol{S} \right|^2 \tag{14}$$

$$\mathcal{L}_{up\_bc} = \frac{1}{N_{up\_bc}} \sum_{i=1}^{N_{up\_bc}} |\widehat{v_{bc}} - v_{bc}|^2 \tag{15}$$

$$\mathcal{L}_{down\_bc} = \frac{1}{N_{down\_bc}} \sum_{i=1}^{N_{down\_bc}} |\widehat{h_{bc}} - h_{bc}|^2 \tag{16}$$

$$\mathcal{L}_{total} = W_1 \times \mathcal{L}_{up\_bc} + W_2 \times \mathcal{L}_{down\_bc} + \begin{bmatrix} W_3 \\ W_4 \end{bmatrix} \times \mathcal{L}_{physics} \tag{17}$$

where $N_{physics}$ represents all collocation points; $N_{up\_bc}$ represents all upstream boundary points; $N_{down\_bc}$ represents all downstream boundary points; and $W_1$–$W_4$ = weights that balance the contributions of each term, as mentioned in the previous section.

### Hypermeters for This Study

In this study, a multilayer perceptron with 8 hidden layers, each containing 80 hidden units, is used as the neural network architecture. The rectified linear unit (ReLU) activation function is applied to each layer, because neither the input nor the output variables can be normalized in the PINN framework. The learning rates $lr^1$, $lr^2$, and $lr^3$ used in this study were $1 \times 10^{-4}$, $1 \times 10^{-5}$, and $1 \times 10^{-6}$, respectively. The weight decay was set to $1 \times 10^{-16}$. A total of 9,000 epochs were used for all test cases. The collocation points correspond to actual cross sections, with one interpolation point between each neighboring pair of cross sections.

## Results and Discussions

Two case studies are presented in this section to demonstrate the performance of our PINN framework. The first case is a hypothetical terrain with a flow and water stage spike on the boundary conditions. The second case is a real stormwater event, from May 24–29, 2014, that occurred on downstream Cypress Greek, Houston.

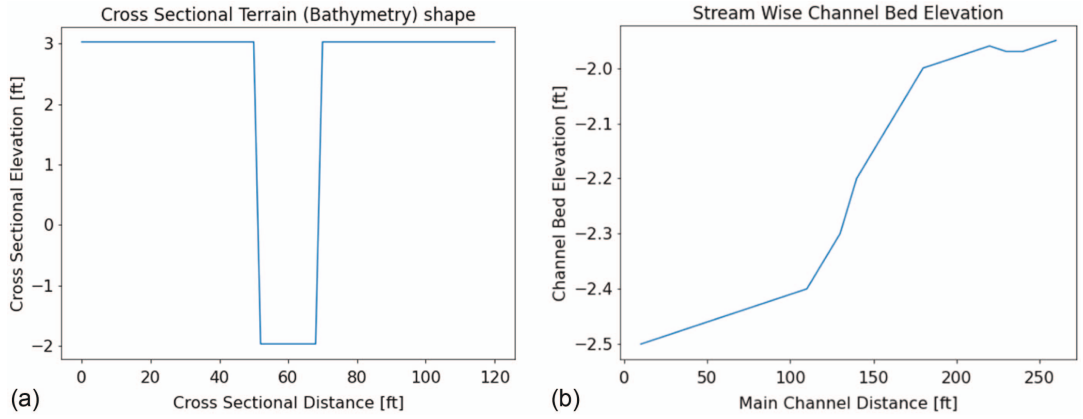


**Fig. 6.** (Color) Hypothetical terrain details for the hypothetical spike case: (a) cross-sectional shape; and (b) streamwise bed elevation.

### *Hypothetical Sudden Flow and Water Stage Spike in a Uniform Trapezoidal Channel*

The first case is a pure hypothetical test. To simplify the problem and make it more artificially channel-like, we created a channel with uniform trapezoidal cross sections [Fig. 6(a)]. To make the case more realistic, the streamwise channel bed elevation was nonlinear [Fig. 6(b)]. As the first case to test the model, the main channel intentionally was 79.25 m (260 ft) long in total. This short channel distance along with the uniform 3.05 m (10 ft) between each cross section was used to determine if the PINN framework can simulate the wave propagation.

#### Velocity-Based 1D Shallow-Water Equations

Neither term in Eq. (1), $\partial A/\partial t$ and $\partial(Av)/\partial x$, can be computed directly as partial differentials because $A$ lacks an explicit expression in neural networks. This issue also arises in numerical methods. Similar to the approach used in numerical methods, we assume that the change in flow area, $\Delta A$, for a small change in water elevation, $\Delta h$, can be approximated as $B\Delta h$. Substituting this into the continuity equation obtains

$$\frac{\partial h}{\partial t} + \frac{A}{B}\frac{\partial v}{\partial x} + u\frac{\partial h}{\partial x} = 0 \tag{18}$$

To simulate the sudden flow and water stage spike, we created upstream velocities and downstream water stages that changed rapidly in 10 min. The details of both boundary conditions are shown in Figs. 7(a and b).

To make the comparison more representative, we selected three cross sections, the 20th, 15th, and10th cross sections downstream, to visualize the results. As mentioned, there were a total of 26 cross sections in our hypothetical cases; therefore, the three selected cross sections were located in the middle of the stream. The PINN framework and HEC-RAS output for the velocity-based shallow-water equation are shown in Fig. 8. Both velocity and water stage profiles had good agreement. The mean absolute error for all cross-sectional velocities and water stages were 0.002743 m/s (0.009 ft/s) and 0.001219 m/s (0.004 ft), respectively.

#### Flow Rate–Based 1D Shallow-Water Equations

Flow rate–based shallow-water equations are used more frequently by engineers and researchers in North America. Most software, including HEC-RAS and SWMM, use flow rate–based shallow-water equations as their default. The flow rate–based 1D shallow-water equations can be expressed as

$$B\frac{\partial h}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{19}$$

$$\frac{\partial Q}{\partial t} + \frac{\partial\left(\frac{Q^2}{A}\right)}{\partial x} + gA\frac{\partial h}{\partial x} + gA(S_f - S_0) = 0 \tag{20}$$

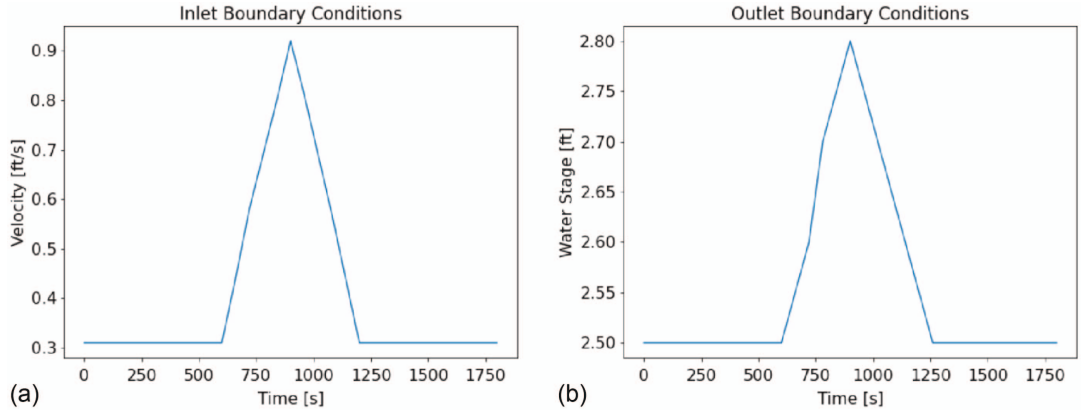


**Fig. 7.** (Color) Velocity based boundary conditions for the velocity based test case: (a) inlet velocity hydrograph; and (b) outlet water stage hydrograph.

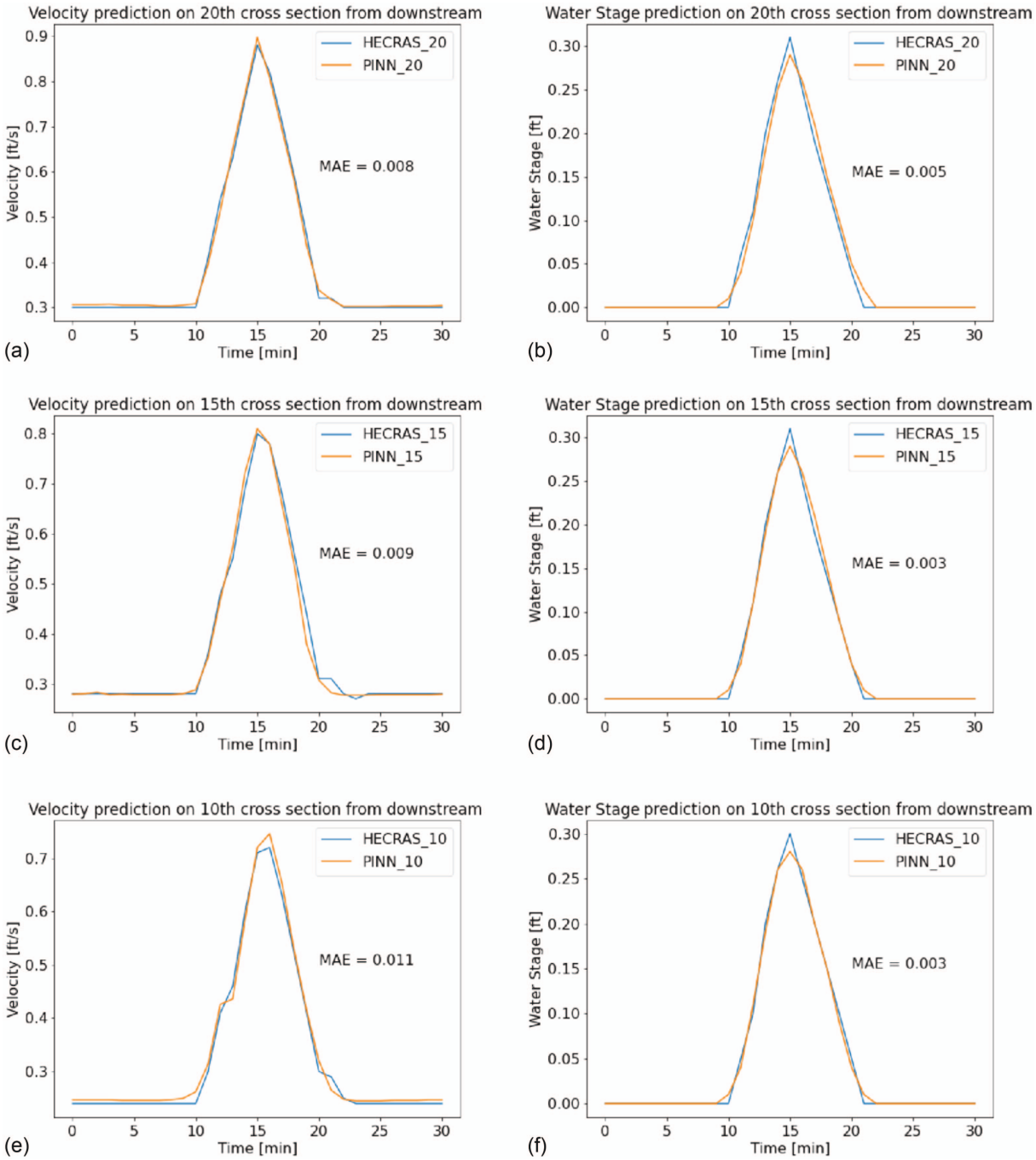


**Fig. 8.** (Color) PINN and HEC-RAS output for the velocity-based test case: (a, c, and e) velocity profile; and (b, d, and f) water stage profile.

The boundary conditions that were used to test the framework were exactly same as those in the last section, but using flow rate as the input parameter. Similarly, we chose the same three cross sections to visualize the results. The PINN framework and HEC-RAS output for velocity based shallow-water equation is shown in Fig. 9. Not surprisingly, the PINN can solve the flow rate–based 1D shallow-water equations with good accuracy. The mean absolute error for the PINN-predicted water stage was very close to that of the velocity-based PINN, i.e., 0.0012 m (0.004 ft). The mean absolute error was much higher for flow rate than for velocity because flow rate is at a much larger scale than velocity. In addition, different orders of magnitude for two variables (flow rate and water stage) without normalization could be another reason that flow rate prediction was slightly worse.

Both velocity-based and flow rate–based PINNs worked well in our hypothetical test case. However, in the real application case, the order of magnitude was much higher than in this small test case.

### *Downstream Cypress Creek*

Cypress Creek, located in Houston, flows from Snake Creek to the west fork of the San Jacinto River, eventually flowing to the Gulf of Mexico. The watershed has a drainage area of 691.53 $km^2$ (267 $mi^2$). Cypress Creek experiences two or three recurrent floods per year on average (Tang et al. 2020), and it also had devastating floods during Hurricane Harvey in August 2017. The downstream region of Cypress Creek is a mainly urban area, northern Houston, which makes the flood prediction valuable. The geographical

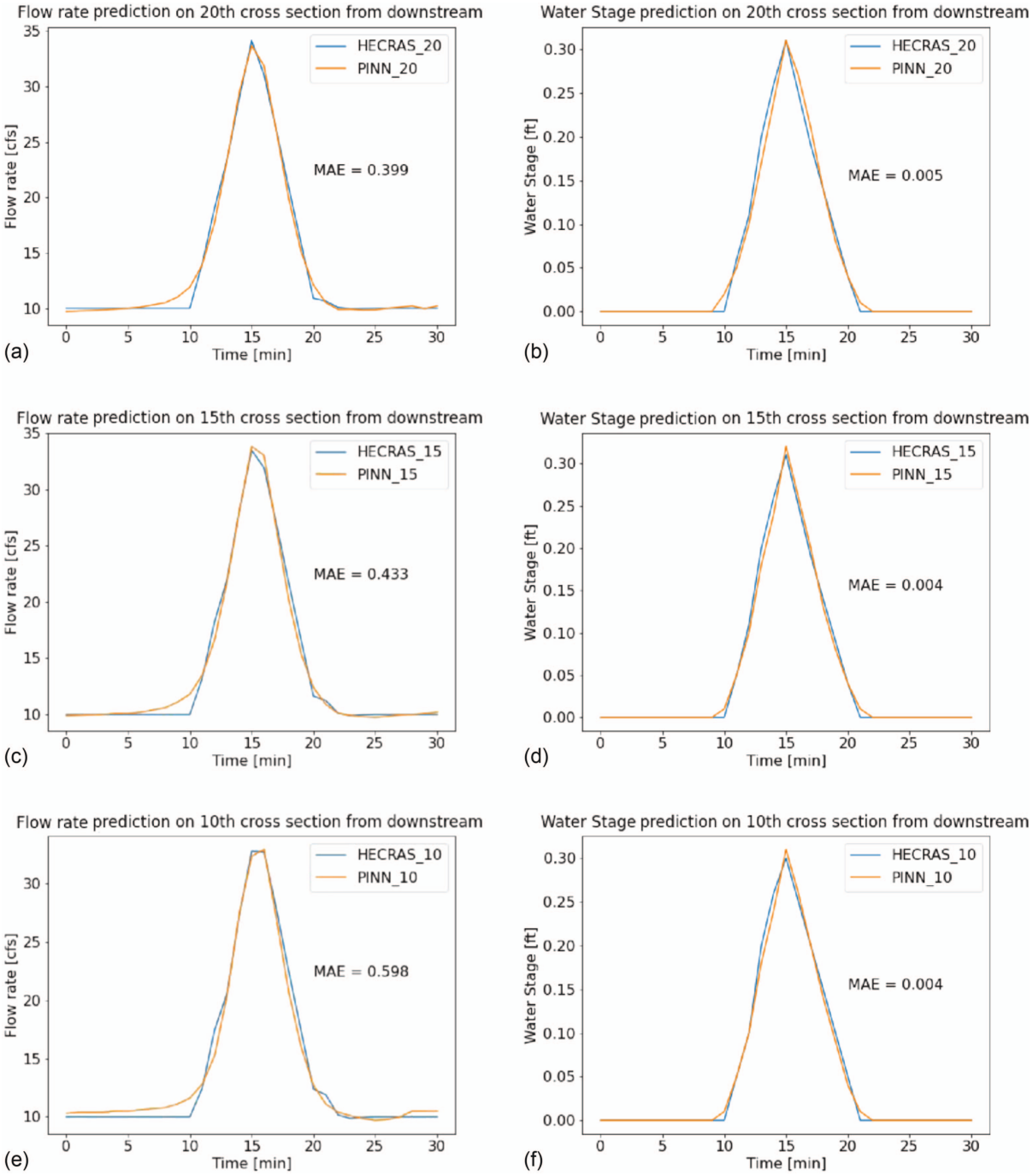


**Fig. 9.** (Color) PINN and HEC-RAS output for the flow rate–based test case: (a, c, and e) flow rate profile; and (b, d, and f) water stage profile.

location of the downstream region of Cypress Creek is shown in Fig. 10. The test rainfall event occurred on May 24–29, 2014, which was a nearly 3-day rainfall event.

### PINN Output

The HEC-RAS modeling of the scenario downstream referred to Leon et al. (2021). The PINN model employs the exact same set of parameters for the entire riverine system as the HEC-RAS model used by Tang et al. (2020) and Leon et al. (2021). The original model couples Hydrologic Engineering Center's Hydrologic Modeling System (HEC-HMS), so part of calculated runoff can discharge from the stream at each cross section by lateral inflow. The original model had a Nash–Sutcliffe efficiency ranging from 0.86 to 0.93, and $R^2$ values ranging from 0.87 to 0.92 across four validation events. To verify the proposed PINN framework, we simplified the system by removing these lateral flows. To simplify the model further, we also reduced the total number of cross sections. The reference HEC-RAS model has 70 cross sections with a cross section of about 121.92 m (400 ft). The terrain, bathymetry, cross-section layout, and boundary conditions were identical in the PINN and the HEC-RAS model in this paper. The PINN framework and HEC-RAS output for the downstream Cypress Creek test case is shown in Fig. 11. The MAE of the flow-rate prediction consistently ranged from 0.0833 to 0.10083 $m^3/s$ (2.943–3.561 cfs), which is very small considering that the actual flow rate scale is 0–11.326 $m^3/s$ (0–400 cfs). The MAE of the water stage prediction

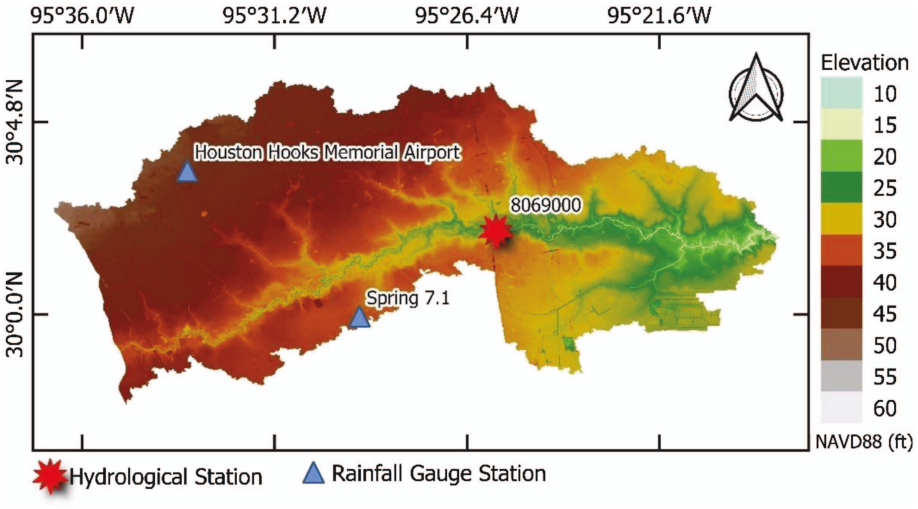


**Fig. 10.** (Color) Geographical location of downstream Cypress Creek.

ranged from 0.0152 to 0.079 m (0.050–0.259 ft). The PINN output trend slightly underestimated both flow rate and water stage at the test locations.

### Response Surface

Based on the methodology of the PINN framework, the optimizer iterates the neurons to fit the algebraic expression of specific 1D shallow-water equations. Thus, the optimization response surface is crucial in checking the difficulty of finding global minima for the optimizer. In our problem, the optimization variables had high dimensionality; therefore, it was unfeasible to plot the response surface directly. To reduce the dimensionality and best visualize the response surface, we selected an interior cross section and traced its loss function values in every iteration. One-dimensional shallow-water equations have only one set of real number solutions when both boundary conditions are certain; therefore, global minima theoretically must exist in our selected cross section. Because the optimizer can only have a certain loss function value for each iteration, the trisurface response surface was the only option for visualization. The three-dimensional (3D) response surface of the selected interior cross section is shown in Fig. 12, in which the $z$-axis is the loss function value during the training phase. In Fig. 12 it is obvious that global minima exist, even though many local minima appear to exist during the optimization stage. This also proves that it is theoretically feasible for the PINN framework to obtain the solution of the 1D shallow-water equations with small residuals.

### Location-Wise Extrapolation

Although the entire framework of the PINN looks like an optimization method to determine the solution of 1D shallow-water equations, it essentially is a neural network. Thus, it is sensible that the PINN can extrapolate the results. Because two input variables—spatial parameter $x$, and temporal parameter $t$—were used as input variables, we can extrapolate the results in both location and time directions.

In the PINN, there is no location limit for extrapolation, which means that we can extrapolate the result both inside and outside the original computational domain. Inside computational domain, we still can use the HEC-RAS XS interpolation function to generate new cross sections as a reference. In the HEC-RAS XS interpolation function, the model will interpolate the terrain information from the upper and lower cross sections, instead of taking additional terrain information from the DEM. Similarly, no additional terrain information is needed in the PINN extrapolation process because the weights and biases are trained already. To compare and visualize the results, we used HEC-RAS to interpolate two additional cross sections: 4,906.97 and 8,159.49 m (16,099 and 26,770 ft) from the end of the simulated river. The extrapolation inside the computational domain in the location-wise direction is shown in Fig. 13. The location-wise extrapolation inside the computational domain is still accurate. The MAE for the extrapolation inside the computational domain is even smaller than the MAE results in the "PINN Output" section.

One of the important features of this PINN framework is that it can extrapolate even outside the computational domain. In this case, there is no HEC-RAS reference that can be provided because it is functionally unfeasible. To verify if the extrapolation outside the domain is sensible, we tested two new locations: 457.2 m (1,500 ft) above the original upstream location, and 457.2 m (1,500 ft) below the original downstream location. As a comparison, we plotted the flow rate and water stage profile on the locations of the original upstream and downstream boundaries (Fig. 14). There was no significant difference in the results between boundary locations and 457.2 m (1,500 ft) from the boundary locations. This makes sense because 457.2 m (1,500 ft) is not long enough to make a difference.

### Time-Wise Extrapolation

Because a temporal parameter, $t$, also is one of the input variables in the PINN, the PINN also can extrapolate in the temporal direction. To verify the accuracy of temporal direction extrapolation, we used the historical data recorded by Water Station 8069000 (Fig. 10) as a reference. To avoid overextrapolation, we extrapolated only the next 30 h to visualize the result, which is sufficiently long after the end of storm event. The PINN extrapolation and water station–recorded data are shown in Fig. 15. The mean absolute error of the extrapolation results was similar to that of the non-extrapolation prediction, and the trend fit the historical curve very well.

## *Comparison of PINN Method and Other Well-Developed Methods*

The PINN method for solving 1D unsteady SWEs has several advantages and disadvantages compared with other well-developed traditional methods.

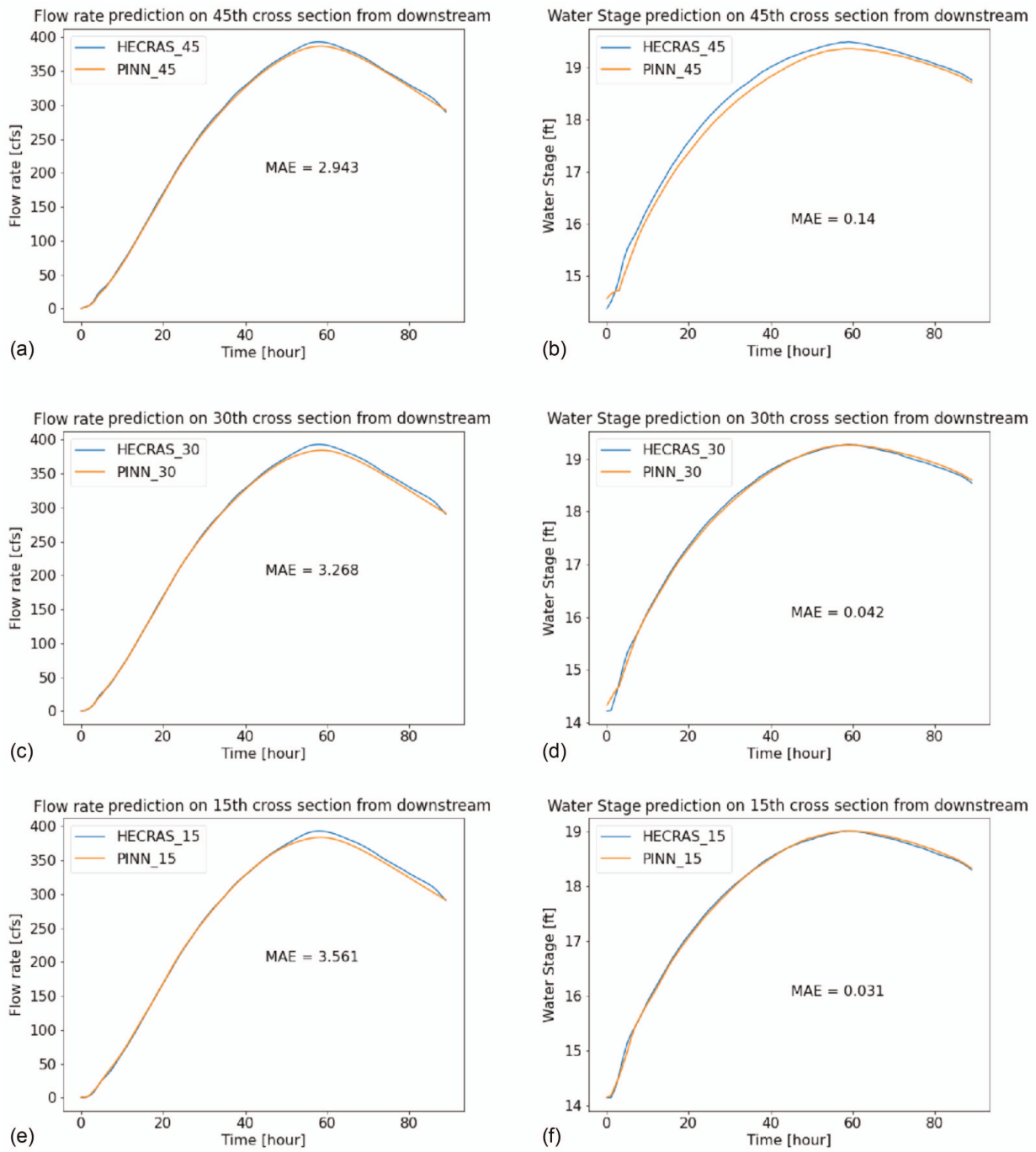


**Fig. 11.** (Color) PINN and HEC-RAS output for the downstream Cypress Creek test case: (a, c, and e) flow rate profile; and (b, d, and f) water stage profile.

Compared with numerical models (e.g. HEC-RAS), the PINN method has the following advantages:

1. Its accuracy does not compromise the choice and order of scheme accuracy, numerical convergence, or Courant numbers.
2. The PINN method has the ability to forecast the water stage and flow rate at locations beyond the computational domain. In HEC-RAS, the computational domain must be extended if a location of interest falls outside the current domain, which can be challenging if the next upstream or downstream station is not readily available or is located far away. In such scenarios, the PINN method can accurately predict the values at locations slightly outside the computational domain with reasonable accuracy.
3. It has the capability to perform timewise extrapolation, enabling it to predict future timesteps and perform data imputation on the current time range. Numerical models have no capability for prediction. Prediction requires additional support of predicted rainfall data and hydrological models, which is far beyond the scope of this discussion. In terms of data imputation, the common methods to fill missing data in a few timesteps use linear regression or simple statistical models. However, neural networks have repeatedly demonstrated superior accuracy compared with these

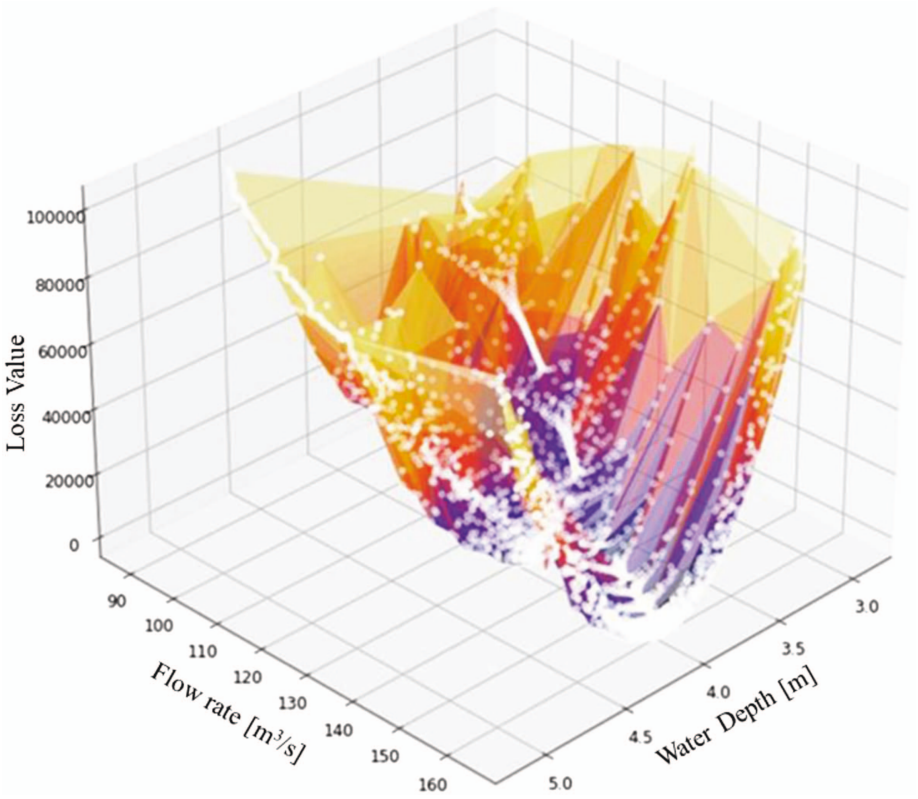


**Fig. 12.** (Color) Response surface of during training.

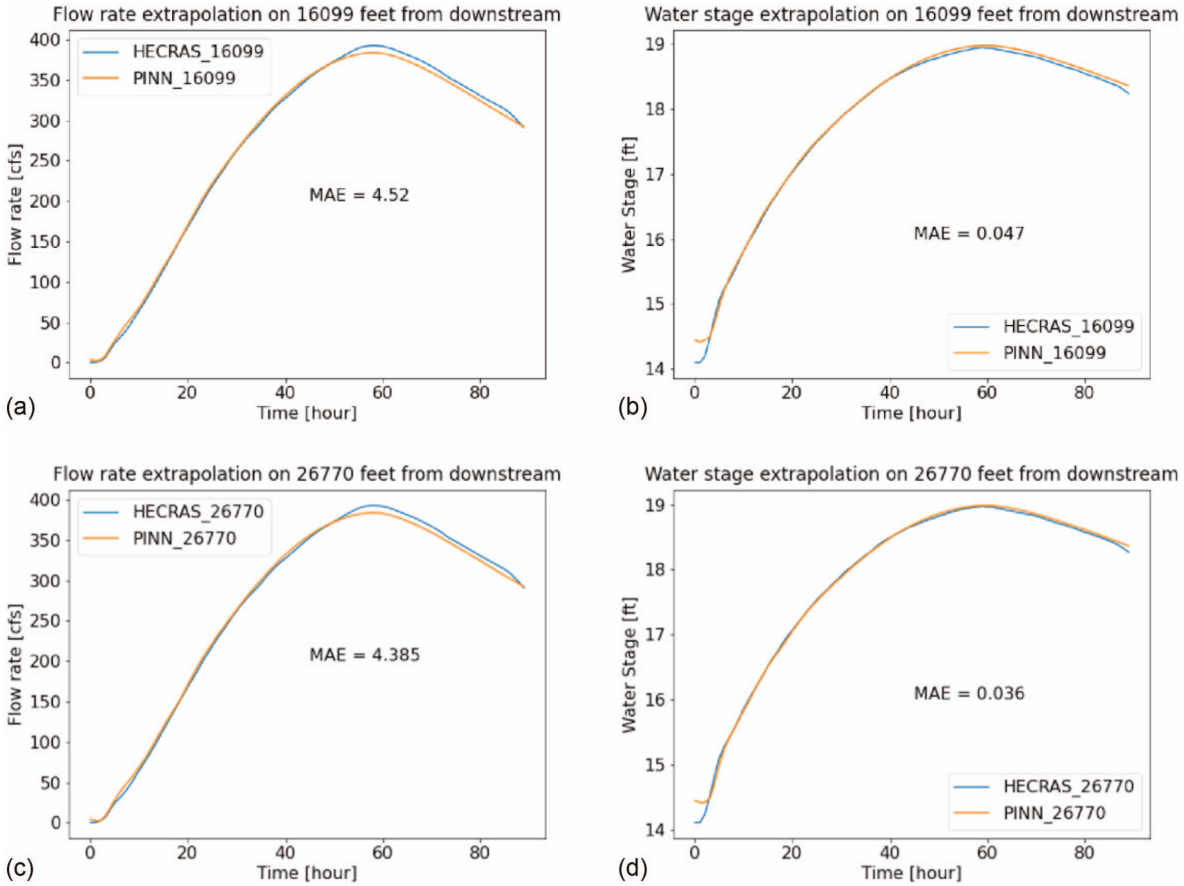


**Fig. 13.** (Color) PINN and HEC-RAS inside computation domain extrapolation: (a and c) flow rate profile; and (b and d) water stage profile.

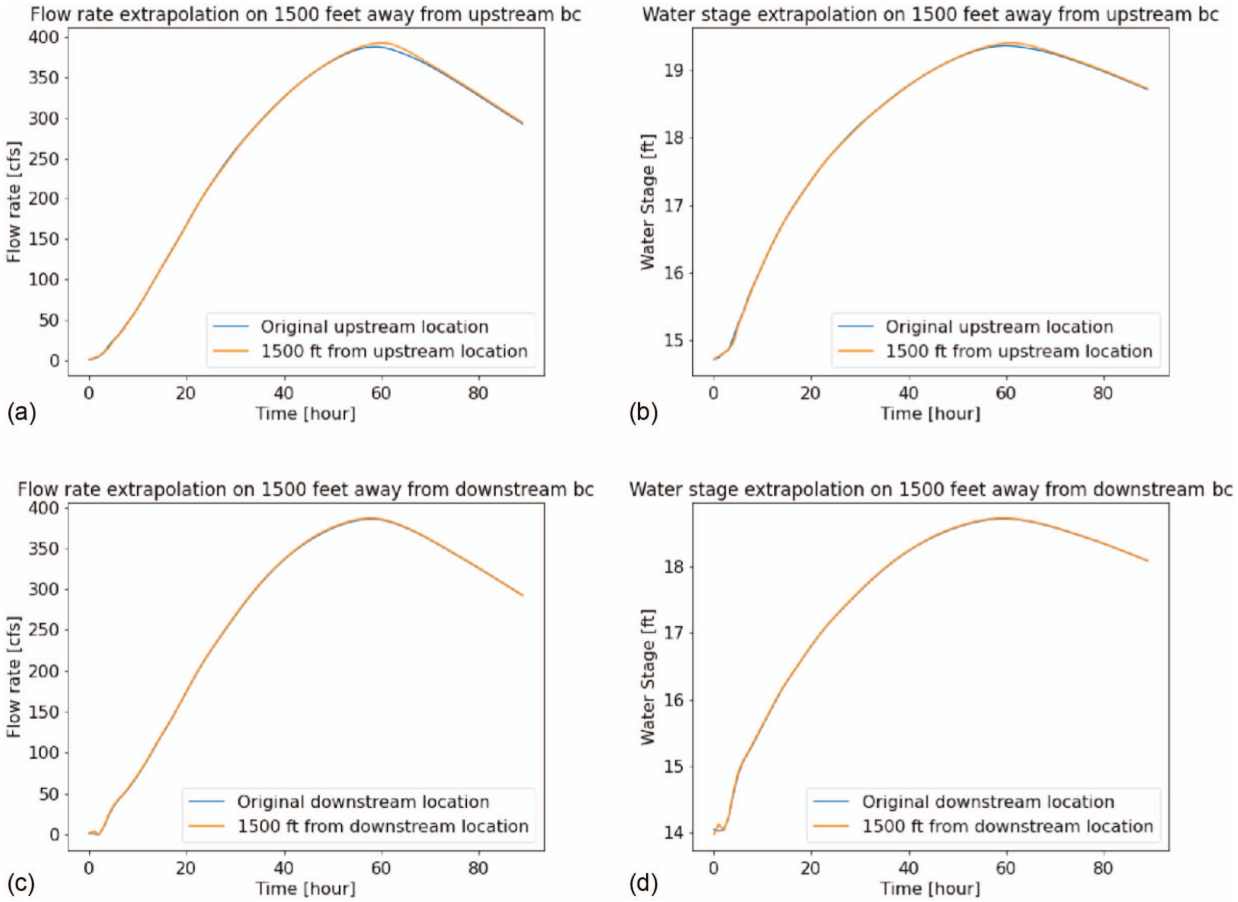


**Fig. 14.** (Color) PINN and HEC-RAS outside computation domain extrapolation: (a and c) flow rate profile; and (b and d) water stage profile.

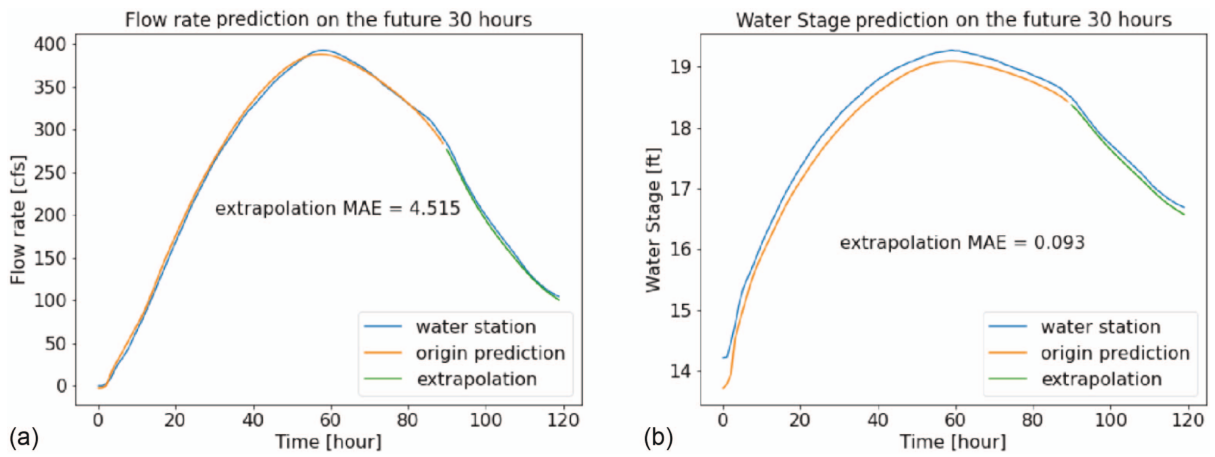


**Fig. 15.** (Color) PINN temporal direction extrapolation and Water Station 8069000 recorded: (a) flow rate profile; and (b) water stage profile.

simple statistical models (Jerez et al 2010; Saad et al. 2020; Boursalie et al. 2022).

The PINN method also has two major drawbacks compared with numerical methods. The first major drawback is training time. The training and extrapolation cost of the PINN and HEC-RAS are presented in Table 1. Multiple studies have indicated that the training of a physics-informed neural network is a slower process than solving the same partial differential equations using numerical methods (Raissi et al. 2019; Markidis 2021; Cuomo et al. 2022).

**Table 1.** Training and extrapolation cost

| Model | Training and simulation time | Extrapolation time (s) |
|---|---|---|
| Hypothetical case | | |
| HEC-RAS | 1 min | — |
| PINN | 14 h | <0.1 |
| Cypress Creek case | | |
| HEC-RAS | 3 min | — |
| PINN | 29 h | <0.1 |

The present paper's use of the PINN approach is no exception. The second major drawback is that the PINN for solving 1D unsteady SWEs still is not well-developed compared with numerical models. To address complex riverine systems involving various hydraulic structures (such as weirs, dams, reservoirs, and spillway gates), our current implementation of the PINN model requires additional theoretical or empirical equations to be incorporated. Despite its potential, the PINN still requires further development compared with well-established numerical methods.

Compared with traditional machine and deep learning methods, the PINN approach offers several advantages

1. As mentioned previously, traditional machine and deep learning models greatly rely on large quantities of high-quality training data, which may be difficult or impossible to obtain for many riverine systems. In contrast, the PINN approach, which is a data-free approach, does not have this issue.
2. The traditional machine and deep learning methods are not constrained by any physical law in mathematics because they only regress the existing training data. The PINN model incorporates one or more mathematical expression of physical laws into its framework. This feature enhances the reliability of the model, particularly for rare or extreme conditions.

One disadvantage of the PINN approach is that it typically requires more epochs to achieve convergence compared with traditional deep learning methods. Although this usually does not mean that the PINN is slower, because it usually has far fewer training data than traditional deep learning methods, it does means that the PINN models are more sensitive to hyperparameter settings.

### *Limitations of This Proposed Method*

There are two major limitations of our proposed method. The first major problem is training time. The PINN method typically requires a longer training time than other standard data-driven deep learning models. This extended timeframe is attributed to the optimizer encountering greater challenges in identifying the optimal solution. The training period typically spans 1.5–18 h depending on the problem complexity and GPU capability (Hu and McDaniel 2023; Li et al. 2024). However, our proposed method has a slower training speed than these PINN cases. This is due to the integration of a numerical method for calculating hydraulic parameters in our methodology, and GPUs generally perform poorly when handling numerical computations. Our method required 14 h for the hypothetical case and 29 h for the Cypress Creek case using an NVIDIA RTX 3070 (Santa Clara, California). Increasing the training speed could be an important direction for future work.

Another major limitation is generalization. Some may assume that a PINN could naturally resolve the generalization issue because it directly fits the equation rather than a specific data set. However, this assumption does not hold true for PDE problems. PDEs typically have infinite solutions when boundary and initial conditions are undetermined, and have a unique solution only when these conditions are well-defined. Thus, in our study, the PINN can derive only the approximate form of SWEs under specific boundary conditions. If different boundary conditions are applied, the model needs to retrain, like most other deep learning models. A potential solution to this challenge involves incorporating the boundary conditions as input variables, thereby expanding the approximation form of SWEs to a more abstract estimation. This more-abstract estimation can be used to regress the SWEs in a certain range of boundary conditions. An example applied to the steady case was presented by Yin et al. (2023).

## Conclusion

This paper presents and tested a physics-informed neural network for solving the 1D unsteady shallow-water equations. Due to its unique features, it could be used in many water resource engineering aspects, including rectifying numerical scheme errors, estimating water stage in unmeasured bathymetry cross-sections, and deducing external boundary conditions, among others. In this paper, our PINN framework was tested using a hypothetical scenario and a historical flooding scenario of the downstream of Cypress Creek, Houston. The PINN framework accurately predicted the results in both test cases with a very small mean absolute error. The key conclusions are as follows:

- Our PINN framework can solve both velocity-based and flow rate–based shallow-water equations with minor different compared with numerical methods (HEC-RAS). For the hypothetical test case, the mean absolute errors ranged from 0.0024 to 0.0034 m/s (0.008–0.011 ft/s) for velocity prediction, from 0.0113 to 0.0169 $m^3/s$ (0.399–0.598 cfs) for flow rate, and from 0.00091 to 0.00152 m (0.003–0.005 ft) for water stage.
- For the downstream Cypress Creek test case, the PINN worked well even when there was a huge incoming flow. The mean absolute error for flow rate and water stage prediction ranged from 0.0833 to 0.1033 m3/s (2.943–3.651 cfs) and from 0.0152 to 0.0789 m (0.05–0.259 ft), respectively. This also proves that the PINN has the capability to provide an accurate prediction of both flow rate and water stage under a potential flooding scenario.
- The response surface of the PINN shows the existence of global minima, which proves that it is theoretically feasible for the PINN to obtain the solution of the 1D shallow-water equations with small residuals.
- The PINN can perform location-wise and timewise extrapolation without constraints. The location-wise extrapolation results were compared with those of the HEC-RAS XS interpolation function. The timewise extrapolation results were compared with water stage data recorded by water station. Both extrapolations had excellent agreement with reference data.

## Data Availability Statement

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

## Acknowledgments

The authors gratefully acknowledge the financial support from the National Science Foundation under Grant CBET 2203292. The authors are grateful to the anonymous reviewers for their constructive comments, which helped to significantly improve the quality of the manuscript.

## Notation

*The following symbols are used in this paper:*

$A$ = wetted cross-sectional area;
$B$ = top width of river;
$h$ = free-surface water elevation;
$\hat{\boldsymbol{h}}$ = predicted free-surface water elevation;
$P$ = wetted perimeter;

$Q$ = cross-sectional flow rate;
$R$ = hydraulic radius;
$v$ = cross-sectional velocity; and
$\hat{v}$ = predicted cross-sectional velocity.

## References


Adnan, R. M., A. Petroselli, S. Heddam, C. A. G. Santos, and O. Kisi. 2021. "Short term rainfall-runoff modelling using several machine learning methods and a conceptual event-based model." *Stochastic Environ. Res. Risk Assess.* 35 (3): 597–616. https://doi.org/10.1007/s00477-020-01910-0.

Bararnia, H., and M. Esmaeilpour. 2022. "On the application of physics informed neural networks (PINN) to solve boundary layer thermal-fluid problems." *Int. Commun. Heat Mass Transfer* 132 (Mar): 105890. https://doi.org/10.1016/j.icheatmasstransfer.2022.105890.

Baydin, A. G., B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. 2018. "Automatic differentiation in machine learning: A survey." *J. Mach. Learn. Res.* 18 (153): 1–43.

Bihlo, A., and R. O. Popovych. 2022. "Physics-informed neural networks for the shallow-water equations on the sphere." *J. Comput. Phys.* 456 (May): 111024. https://doi.org/10.1016/j.jcp.2022.111024.

Boursalie, O., R. Samavi, and T. E. Doyle. 2022. "Evaluation metrics for deep learning imputation models." In *AI for disease surveillance and pandemic intelligence: Intelligent disease detection in action*, 309–322. Cham, Switzerland: Springer.

Brunner, G. W. 2002. "HEC-RAS (river analysis system)." In *North American water and environment congress & destructive water*, 3782–3787. Reston, VA: ASCE.

Bui, D. T., N. D. Hoang, F. Martínez-Álvarez, P. T. T. Ngo, P. V. Hoa, T. D. Pham, P. Samui, and R. Costache. 2020. "A novel deep learning neural network approach for predicting flash flood susceptibility: A case study at a high frequency tropical storm area." *Sci. Total Environ.* 701 (Jan): 134413. https://doi.org/10.1016/j.scitotenv.2019.134413.

Cai, S., Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis. 2021a. "Physics-informed neural networks (PINNs) for fluid mechanics: A review." *Acta Mech. Sin.* 37 (12): 1727–1738. https://doi.org/10.1007/s10409-021-01148-1.

Cai, S., Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis. 2021b. "Physics-informed neural networks for heat transfer problems." *J. Heat Transfer* 143 (6): 060801. https://doi.org/10.1115/1.4050542.

Chaudhry, M. H. 2014. Vol. 415 of *Applied hydraulic transients*. New York: Springer.

Chow, V. T. 1959. *Open-channel hydraulics*. New York: McGraw-Hill.

Cuomo, S., V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. 2022. "Scientific machine learning through physics–informed neural networks: Where we are and what's next." *J. Sci. Comput.* 92 (3): 88. https://doi.org/10.1007/s10915-022-01939-z.

Desrochers, N. M., M. Trudel, D. L. Peters, G. Siles, and R. Leconte. 2020. "Hydraulic model calibration using water levels derived from time series high-resolution SAR images." *J. Hydraul. Eng.* 146 (4): 05020001. https://doi.org/10.1061/(ASCE)HY.1943-7900.0001687.

de Wolff, T., H. Carrillo, L. Martí, and N. Sanchez-Pi. 2021. "Assessing physics informed neural networks in ocean modelling and climate change applications." In *Proc., AI: Modeling Oceans and Climate Change Workshop at ICLR 2021*. Appleton, WI: International Conference on Learning Representations.

Eivazi, H., M. Tahani, P. Schlatter, and R. Vinuesa. 2022. "Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations." *Phys. Fluids* 34 (7): 075117. https://doi.org/10.1063/5.0095270.

Franzini, F., and S. Soares-Frazão. 2016. "Efficiency and accuracy of Lateralized HLL, HLLS and Augmented Roe's scheme with energy balance for river flows in irregular channels." *Appl. Math. Modell.* 40 (17–18): 7427–7446. https://doi.org/10.1016/j.apm.2016.02.007.

Hu, B., and D. McDaniel. 2023. "Applying physics-informed neural networks to solve Navier–Stokes equations for laminar flow around a particle." *Math. Comput. Appl.* 28 (5): 102. https://doi.org/10.3390/mca28050102.

Huang, Y., Z. Zhang, and X. Zhang. 2022. "A direct-forcing immersed boundary method for incompressible flows based on physics-informed neural network." *Fluids* 7 (2): 56. https://doi.org/10.3390/fluids7020056.

Hurler, M. 2021. "Learning free-surface flow with physics-informed neural networks." Master's thesis, Dept. of Scientific Computing, Institute of Parallel and Distributed Systems, Univ. of Stuttgart.

Iskhakov, A. S., and N. T. Dinh. 2020. "Physics-integrated machine learning: Embedding a neural network in the Navier-Stokes equations. Part I." Preprint, submitted August 24, 2020. http://arxiv.org/abs/2008.10509.

Jerez, J. M., I. Molina, P. J. García-Laencina, E. Alba, N. Ribelles, M. Martín, and L. Franco. 2010. "Missing data imputation using statistical and machine learning methods in a real breast cancer problem." *Artif. Intell. Med.* 50 (2): 105–115. https://doi.org/10.1016/j.artmed.2010.05.002.

Jin, X., S. Cai, H. Li, and G. E. Karniadakis. 2021. "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations." *J. Comput. Phys.* 426 (Feb): 109951. https://doi.org/10.1016/j.jcp.2020.109951.

Kim, H. I., and B. H. Kim. 2020. "Flood hazard rating prediction for urban areas using random forest and LSTM." *KSCE J. Civ. Eng.* 24 (12): 3884–3896. https://doi.org/10.1007/s12205-020-0951-z.

Kingma, D. P., and J. Ba. 2014. "Adam: A method for stochastic optimization." Preprint, submitted December 22, 2014. http://arxiv.org/abs/1412.6980.

Labuhn, K., A. D. Gronewold, T. Calappi, A. MacNeil, C. Brown, and E. J. Anderson. 2020. "Towards an operational flow forecasting system for the Upper Niagara River." *J. Hydraul. Eng.* 146 (9): 05020006. https://doi.org/10.1061/(ASCE)HY.1943-7900.0001781.

Leon, A. S., L. Bian, and Y. Tang. 2021. "Comparison of the genetic algorithm and pattern search methods for forecasting optimal flow releases in a multi-storage system for flood control." *Environ. Modell. Software* 145 (Nov): 105198. https://doi.org/10.1016/j.envsoft.2021.105198.

Li, J., and S. Brewer. 2020. "A performance comparison of unsupervised machine learning algorithms for clustering water depth datasets at urban drainage systems." Preprint, submitted April 30, 2020. https://doi.org/10.31223/osf.io/ycw3v.

Li, Y., P. Ni, L. Sun, and Y. Xia. 2024. "Finite element model-informed deep learning for equivalent force estimation and full-field response calculation." *Mech. Syst. Signal Process.* 206 (Jan): 110892. https://doi.org/10.1016/j.ymssp.2023.110892.

Mao, Z., A. D. Jagtap, and G. E. Karniadakis. 2020. "Physics-informed neural networks for high-speed flows." *Comput. Methods Appl. Mech. Eng.* 360 (Mar): 112789. https://doi.org/10.1016/j.cma.2019.112789.

Markidis, S. 2021. "The old and the new: Can physics-informed deep-learning replace traditional linear solvers?" *Front. Big Data* 4 (Nov): 669097. https://doi.org/10.3389/fdata.2021.669097.

Palu, M. C., and P. Y. Julien. 2020. "Test and improvement of 1D routing algorithms for dam-break floods." *J. Hydraul. Eng.* 146 (6): 04020043. https://doi.org/10.1061/(ASCE)HY.1943-7900.0001755.

Raissi, M., P. Perdikaris, and G. E. Karniadakis. 2019. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations." *J. Comput. Phys.* 378 (Feb): 686–707. https://doi.org/10.1016/j.jcp.2018.10.045.

Saad, M., M. Chaudhary, F. Karray, and V. Gaudet. 2020. "Machine learning based approaches for imputation in time series data and their impact on forecasting." In *Proc., IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, 2621–2627. New York: IEEE.

Sadler, J. M., J. L. Goodall, M. M. Morsy, and K. Spencer. 2018. "Modeling urban coastal flood severity from crowd-sourced flood reports using Poisson regression and Random Forest." *J. Hydrol.* 559 (Apr): 43–55. https://doi.org/10.1016/j.jhydrol.2018.01.044.

Shi, J., Z. Yin, R. Myana, K. Ishtiaq, A. John, J. Obeysekera, A. Leon, and G. Narasimhan. 2023. "Deep learning models for water stage predictions in South Florida." Preprint, submitted June 28, 2023. http://arxiv.org/abs/2306.15907.

Song, Y., C. Shen, and X. Liu. 2023. "A surrogate model for shallow water equations solvers with deep learning." *J. Hydraul. Eng.* 149 (11): 04023045. https://doi.org/10.1061/JHEND8.HYENG-13190.

Sriram, R. 2021. "Utilizing random forest machine learning models to determine water table flood levels through volunteered geospatial information." Preprint, submitted April 27, 2021. https://doi.org/10.31223/X5QS4C.

Stansby, P. K., and J. G. Zhou. 1998. "Shallow-water flow solver with non-hydrostatic pressure: 2D vertical plane problems." *Int. J. Numer. Methods Fluids* 28 (3): 541–563. https://doi.org/10.1002/(SICI)1097-0363(19980915)28:3<541::AID-FLD738>3.0.CO;2-0.

Tamiru, H., and M. O. Dinka. 2021. "Application of ANN and HEC-RAS model for flood inundation mapping in lower Baro Akobo River Basin, Ethiopia." *J. Hydrol.: Reg. Stud.* 36 (Aug): 100855. https://doi.org/10.1016/j.ejrh.2021.100855.

Tang, Y., A. S. Leon, and M. L. Kavvas. 2020. "Impact of size and location of wetlands on watershed-scale flood control." *Water Resour. Manage.* 34 (5): 1693–1707. https://doi.org/10.1007/s11269-020-02518-3.

Toro, E. F. 1992. "Riemann problems and the WAF method for solving the two-dimensional shallow water equations." *Philos. Trans. R. Soc. London, Ser. A* 338 (1649): 43–68. https://doi.org/10.1098/rsta.1992.0002.

Verma, A. 2000. *An introduction to automatic differentiation*, 804–807. Bangalore, India: Current Science Association.

Willard, J., X. Jia, S. Xu, M. Steinbach, and V. Kumar. 2020. "Integrating physics-based modeling with machine learning: A survey." Accessed March 14, 2022. https://arxiv.org/abs/2003.04919.

Wu, Z., Y. Zhou, H. Wang, and Z. Jiang. 2020. "Depth prediction of urban flood under different rainfall return periods based on deep learning and data warehouse." *Sci. Total Environ.* 716 (May): 137077. https://doi.org/10.1016/j.scitotenv.2020.137077.

Yang, X. I. A., S. Zafar, J. X. Wang, and H. Xiao. 2019. "Predictive large-eddy-simulation wall modeling via physics-informed neural networks." *Phys. Rev. Fluids* 4 (3): 034602. https://doi.org/10.1103/PhysRevFluids.4.034602.

Yin, Z., L. Bian, B. Hu, J. Shi, and A. S. Leon. 2023. "Physic-informed neural network approach coupled with boundary conditions for solving 1D steady shallow water equations for riverine system." In *Proc., World Environmental and Water Resources Congress 2023*, 280–288. Reston, VA: ASCE, Environmental and Water Resource Institute.

Zahura, F. T., J. L. Goodall, J. M. Sadler, Y. Shen, M. M. Morsy, and M. Behl. 2020. "Training machine learning surrogate models from a high-fidelity physics-based model: Application for real-time street-scale flood prediction in an urban coastal community." *Water Resour. Res.* 56 (10): e2019WR027038. https://doi.org/10.1029/2019WR027038.

Zhang, R., R. Zen, J. Xing, D. M. S. Arsa, A. Saha, and S. Bressan. 2020. "Hydrological process surrogate modelling and simulation with neural networks." In *Advances in knowledge discovery and data mining: 24th Pacific-Asia Conf., PAKDD 2020*, 449. Singapore: Springer.

Zhao, G., B. Pang, Z. Xu, D. Peng, and L. Xu. 2019. "Assessment of urban flood susceptibility using semi-supervised machine learning model." *Sci. Total Environ.* 659 (Apr): 940–949. https://doi.org/10.1016/j.scitotenv.2018.12.217.

Zhou, J. G. 1995. "Velocity-depth coupling in shallow-water flows." *J. Hydraul. Eng.* 121 (10): 717–724. https://doi.org/10.1061/(ASCE)0733-9429(1995)121:10(717).