

Article

Parametrization of Subgrid Scales in Long-Term Simulations of the Shallow-Water Equations Using Machine Learning and Convex Limiting

Md Amran Hossan Mojamder , Zhihang Xu , Min Wang  and Ilya Timofeyev * 

Department of Mathematics, University of Houston, Houston, TX 77204, USA; mmojamde@cougar.net.uh.edu (M.A.H.M.); zxu29@central.uh.edu (Z.X.); mwang55@central.uh.edu (M.W.)

* Correspondence: itimofey@cougar.net.uh.edu

Abstract

We present a method for parametrizing sub-grid processes in the shallow water equations. We define coarse variables and local spatial averages and use a feed-forward neural network to learn sub-grid fluxes. Our method results in a local parametrization that uses a four-point computational stencil, which has several advantages over globally coupled parametrizations. We demonstrate numerically that our method improves energy balance in long-term turbulent simulations and also accurately reproduces individual solutions. The long-term simulations refer to numerical studies where a fluid flow is simulated over a duration long enough to reach a statistical steady state. The neural network parametrization can be easily combined with flux limiting to reduce oscillations near shocks. More importantly, our method provides reliable parametrizations, even in dynamical regimes that are not included in the training data.

Keywords: shallow water equations; sub-grid parametrization; neural networks; monolithic convex limiting

MSC: 65M99

1. Introduction

Simulations of atmosphere–ocean models involve solving numerically complex partial differential equations (PDEs) that span a wide range of temporal and spatial scales. However, due to limited computational resources, climate models may not be able to fully resolve all important physical processes (e.g., [1]). To manage this complexity, coarse models have been developed that use parameterizations to approximate the effects of unresolved subgrid-scale processes. These parameterizations can introduce considerable uncertainty and bias into climate predictions (e.g., [2–6]).

A promising approach is to use machine learning to learn improved parameterizations by training on outputs from high-resolution simulations that capture the subgrid dynamics more accurately. It has been demonstrated across many applications that machine learning models can learn complex relationships by minimizing prediction errors over training datasets. Among machine learning techniques, Neural Networks (NNs) have become particularly popular in climate modeling. They have been applied to various problems such as solving the forced Burgers' equation [7,8], ocean modeling [9–11], cloud representation [12], residual heating and moistening [13], and convection [14,15]. Beyond large-scale climate applications, NNs have also shown strong performance in hydraulic engineering problems.



Received: 30 January 2026

Revised: 7 March 2026

Accepted: 9 March 2026

Published: 12 March 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

For example, [16] compared several machine learning models for predicting scour depth in meandering channels and demonstrated that data-driven approaches outperform traditional empirical formulas in capturing complex nonlinear hydraulic responses. NNs are advantageous due to their *universal approximation properties* [17,18], which enable them to represent complex nonlinear mappings with high accuracy. One representative example is the Physics-Informed Neural Network (PINN) framework [19–22], a neural network-based approach that has been widely recognized for its broad applicability to solving PDE problems arising in numerous areas [23–26]. Moreover, once trained, neural networks enable fast and efficient inference, making them suitable for repeated evaluation within large-scale climate and geophysical simulations. Such implementations can further benefit from modern GPU and CPU architectures, particularly when using reduced-precision arithmetic [27].

Several issues arise when applying neural networks in the context of partial differential equations. For physics-informed models, while they offer flexibility and often reduce the need for large training datasets, neural network solvers may still face certain optimization challenges [28,29], particularly for stiff, multiscale, or convection-dominated problems [30–32]. Moreover, for most data-driven neural network approaches, it is typically necessary to perform fully resolved simulations to generate training data. This may limit their applicability over wide parameter ranges, since high-fidelity simulations can be computationally expensive. Thus, to make neural networks computationally efficient and practical, it is important to develop machine learning approaches that can be easily generalized to parameters outside of the training regime. For atmosphere/ocean applications, external forcing is an important parameter, and parametrizations should respond adequately to a wide range of forcings outside of the training regime. In the context of fluid dynamics, it is important to develop parametrizations that are local in physical space. Local parametrizations are computationally efficient and can be trained and modified independently for different spatial locations. In addition, local parametrizations can be analyzed and trained using localized PDE models instead of large-scale models describing the global behavior of the ocean or atmosphere. For instance, it is challenging to develop parametrizations for turbulence near a wall (e.g., [33–35]), and local machine learning parametrizations can be developed and fine-tuned using relatively inexpensive fully-resolved simulations. In addition, local parametrizations typically require smaller training and testing datasets, and fully resolved simulations can be performed on a smaller computational domain. It is also important to develop machine learning methods that produce physically admissible numerical solutions (e.g., positive water height). It is well-understood that standard NNs may not obey physical conservation laws such as mass or energy conservation, and this can lead to non-physical numerical solutions. In this paper, this is mitigated by combining the NN parametrization with flux limiters to ensure physically valid numerical solutions of the coarse reduced model. Flux limiters have been widely used in the context of classical numerical methods to alleviate computational problems in hyperbolic conservation laws due to large gradients.

The Shallow Water Equations (SWEs) are a system of hyperbolic partial differential equations for the horizontal velocity and the water height. These equations describe the flow of a constant-density fluid layer where the horizontal length scale is much larger than the depth. In this paper, we use Neural Networks to learn nonlinear fluxes in the coarse discretization of the SWEs. In particular, we utilize a four-point stencil as input to a feed-forward neural network and reconstruct non-linear fluxes in the coarse discretization of the SWEs. The NN parametrization can be effectively interpreted as a higher-order flux approximation. This approximation is less diffusive compared to the standard Lax-Friedrichs discretization on a coarse mesh. Moreover, the NN parametrization is local since it requires

only four neighboring points to reconstruct the flux. We also demonstrate that our approach allows the application of the NN parametrization to a range of parameters outside of the training regime. In particular, we demonstrate that the NN parametrization generalizes well in regimes with increased external forcing up to 40%. In addition, we show that NN parametrization generalizes to simulations with topography and Manning's friction.

The main novelty of our approach is that we use neural networks to model fluxes and not the solution itself. Thus, we believe that the neural network generalizes better for dynamical regimes not included in the training data, including simulations with topography. In addition, we also compare both spectra in long-term stationary simulations and individual solutions, since analysis of energy spectra is essential for many ocean applications. Another advantage of our machine learning approach is that it can be easily combined with traditional numerical methods for hyperbolic conservation laws to ensure the admissibility of numerical solutions. In particular, we combine our NN parametrization with a Monolithic Convex Limiting (MCL) strategy (see, e.g., [36,37]) to ensure that the numerical solution is within the admissible set. We demonstrate numerically that the MCL strategy does not affect the energy spectra in long-term turbulent simulations. On the other hand, the MCL improved the behavior of individual solutions near shocks.

The rest of the paper is organized as follows. In Section 2, we introduce the 1D shallow water equations and their numerical discretization. In Section 3 we introduce coarse variables and subgrid fluxes (Section 3.1), discuss network architecture and training (Section 3.2), and introduce the MCL strategy (Section 3.3). Unlike [7], which relies on conditional Generative Adversarial Networks, we employ a feed-forward neural network to construct the subgrid parametrization, resulting in a simpler training procedure and improved computational efficiency. We present our numerical results in Section 4 and summarize our findings in Section 5.

2. Problem Formulation

2.1. Shallow Water Equation

In this paper, we consider one-dimensional Shallow Water Equations (SWEs) [38]. These equations are suitable for modeling fluid flows where the horizontal length scale is significantly larger than the vertical fluid depth. These equations can be written using a conservative form

$$\begin{bmatrix} h \\ q \end{bmatrix}_t + \begin{bmatrix} q \\ q^2/h + \frac{1}{2}gh^2 \end{bmatrix}_x = \begin{bmatrix} 0 \\ \rho(x, t) \end{bmatrix}, \quad (1)$$

where g is gravitational acceleration, $h(x, t)$ is the fluid depth, $q = hv$ is the discharge, $v(x, t)$ is the horizontal fluid velocity, and $\rho(x, t)$ is a large-scale stochastic forcing. It is also possible to include viscosity in these equations to model the internal friction of the fluid. The first equation represents the conservation of mass, and the second equation is the conservation of momentum when $\rho = 0$. These equations are supplemented with periodic boundary conditions

$$h(0, t) = h(L, t), \quad v(0, t) = v(L, t).$$

The initial conditions at time $t = 0$ are given by: $h(x, 0) = h_0(x)$ and $v(x, 0) = v_0(x)$.

Equation in (1) can be written as a conservation law

$$u_t + (f(u))_x = \rho,$$

where

$$u = \begin{bmatrix} h \\ q \end{bmatrix}, \quad f(u) = \begin{bmatrix} q \\ q^2/h + \frac{1}{2}gh^2 \end{bmatrix}, \quad \text{and} \quad \rho = \begin{bmatrix} 0 \\ \rho(x, t) \end{bmatrix}.$$

The *stochastic forcing* term $\rho(x, t)$, which models large-scale random forcing at the surface of the fluid (e.g., the wind stress), is given by:

$$\rho(x, t) = A \sum_{k \in K} \left[\alpha_k(t) \cos\left(2\pi k \frac{x}{L}\right) + \beta_k(t) \sin\left(2\pi k \frac{x}{L}\right) \right], \quad (2)$$

where A is the amplitude, K is the set of forced Fourier wavenumbers, and α_k and β_k are coefficients evolving according to the AR(1) (autoregressive model of order 1) process

$$\alpha_k(t + \Delta t) = \psi \alpha_k(t) + \sigma \epsilon_{k,1}(t), \quad \beta_k(t + \Delta t) = \psi \beta_k(t) + \sigma \epsilon_{k,2}(t), \quad (3)$$

where $0 < \psi = 1 - \gamma \Delta t$ with $\gamma > 0$ and $\sigma > 0$ are AR(1) parameters, and $\epsilon_{k,i}(t)$, $i = 1, 2$ are i.i.d. Normal $N(0, \Delta t)$ random variables. Coefficients α_k and β_k are time-correlated Normal $N(0, \sigma^2 / (1 - \psi^2))$ random variables. Equation (3) can be viewed as a temporal discretization of the Ornstein–Uhlenbeck process. In this paper, we use $K = \{1, 2, 3\}$, i.e., the first three wavenumbers are forced.

2.2. Space-Time Discretization

In space, system (1) is discretized on a uniform fine-mesh with $\Delta x = L/N_f$ and cells $C_i = [x_{i-1/2}, x_{i+1/2}]$, $i = 0, \dots, N_f - 1$. The endpoints of each interval are defined as $x_{i-1/2} = i\Delta x$ and $x_{i+1/2} = (i+1)\Delta x$. The midpoints of each interval C_i are defined as $x_i = (i + 1/2)\Delta x$. We assume that Δx is small enough and the fine-mesh discretization resolves all physical processes of interest. Thus, we use a standard Local Lax–Friedrichs (LLF) (e.g., [39]) space discretization where Equation (1) is discretized as

$$\frac{d}{dt} u_i = - \frac{F_{i+1/2} - F_{i-1/2}}{\Delta x} + \rho_i(t), \quad i = 0, \dots, N_f - 1, \quad (4)$$

where $u_i \equiv [h_i, q_i]^T$ is the average over the cell C_i . The forcing $\rho_i(t) = [0, \rho_i(t)]^T$, and we assume that the forcing is slow-varying in space so that $\int_{C_i} \rho(x, t) dx \approx \rho(x_i, t)$. Thus, we use $\rho_i(t) := \rho(x_i, t)$. The LLF flux is vector-valued and is given by

$$F_{i+1/2}(u_i, u_{i+1}) = \frac{f(u_i) + f(u_{i+1})}{2} - \frac{\lambda_{i+1/2}}{2} (u_{i+1} - u_i) \equiv \mathcal{F}(u_{i+1}, u_i), \quad (5)$$

where scalar

$$\lambda_{i+1/2} = \max\left(|v_i| + \sqrt{gh_i}, |v_{i+1}| + \sqrt{gh_{i+1}}\right) \quad (6)$$

represents the upper bound for the local wave speed. The second term in Equation (5) represents additional viscosity and ensures the stability of the space discretization. In particular, it is known that the LLF scheme does not produce spurious oscillations near shocks. The scheme in (5) is symmetric, i.e., there is a function $\mathcal{F}$ such that right and left fluxes can be expressed as $F_{i+1/2} = \mathcal{F}(u_i, u_{i+1})$ and $F_{i-1/2} = \mathcal{F}(u_{i-1}, u_i)$.

The LLF scheme can be rewritten using bar states; this is useful in the context of Monolithic Convex Limiting. The bar states represent the spatially averaged exact solution of the Riemann problem for the shallow water equations. These states serve as the main building block of the Monolithic Convex Limiting algorithm [36,40] discussed in Section 3.3. If we define the intermediate bar states as

$$\bar{u}_{i+1/2} = \frac{u_{i+1} + u_i}{2} - \frac{1}{2\lambda_{i+1/2}} (f(u_{i+1}) - f(u_i)),$$

then the LLF scheme can be written as

$$\frac{d}{dt}u_i = \frac{1}{\Delta x}[\lambda_{i-1/2}(\bar{u}_{i-1/2} - u_i) + \lambda_{i+1/2}(\bar{u}_{i+1/2} - u_i)] + \rho_i(t),$$

for $i = 0, 1, \dots, N_f - 1$. An important property of the bar states is that they satisfy $\bar{u}_{i+1/2} \in \mathcal{A}$ if $\mathcal{A}$ is an invariant set of the shallow water equations and $u_i, u_{i+1} \in \mathcal{A}$.

In this work, we use Heun's time-stepping method. For simplicity, we denote the right-hand side of the semi-discrete system (4) as $g(u(x, t))$, and consider the initial value problem

$$u_t = g(u(t)), \quad u(0) = u^0,$$

where $u(t) := [u_1(t), \dots, u_{N_f}(t)]^\top$. We denote $u^k = u(t_k)$ as the numerical solution at discrete time t_k , with a fixed time step $\Delta t = t_{k+1} - t_k$. Heun's method computes u^{k+1} from u^k in two steps:

$$(\text{Predictor step}): \quad \tilde{u}^{k+1} = u^k + g(u^k)\Delta t,$$

$$(\text{Corrector step}): \quad u^{k+1} = u^k + \frac{1}{2}(g(u^k) + g(\tilde{u}^{k+1}))\Delta t.$$

This method provides a second-order accurate time integration while maintaining an explicit scheme. Heun's method belongs to the class of strong stability preserving (SSP) methods [41–43] with an appropriate condition on the time-step, Δt . In particular, Heun's method maintains stability when used with spatial discretizations that are stable under forward Euler time-discretization; the strong stability preserving coefficient for Heun's method is $c = 1$. Therefore, in most practical applications, the CFL condition is sufficient to guarantee stability. Moreover, if the CFL condition is satisfied, then it is also possible to show that $u_i^{k+1} \in \mathcal{A}$ (set of admissible solutions) since it can be expressed as a convex combination of admissible values $u_i^k, \bar{u}_{i-1/2}, \bar{u}_{i+1/2} \in \mathcal{A}$. Thus, the resulting space-time scheme is also positivity preserving for the water height. Time-splitting is used to add the stochastic forcing, $\rho_i(t)$, after the second Heun step. The LLF discretization (5) is first-order in space, and the Heun time-stepping is second-order accurate in time.

3. Methodology

Performing fully resolved simulations of the SWEs requires a very fine mesh, which can be computationally expensive. This issue becomes particularly important when studying the long-term behavior of climate systems on a large scale. Therefore, it is desirable to obtain a coarse-mesh discretization of the SWE in many situations, including climate studies. Here, we address this issue in the context of stationary long-term simulations. In particular, we develop a coarse model that improves the turbulent cascade and energy transfer between different Fourier modes. To this end, we use machine learning to develop a sub-grid flux parametrization in a model for spatial averages.

3.1. Coarse Mesh and Subgrid Fluxes

Previously, we considered the discretization of the SWE of a fine-mesh with $\Delta x = L/N_f$. Next, we define a coarse mesh

$$\bar{C}_I := \bigcup_{j=nI}^{n(I+1)-1} C_j = [x_{nI-1/2}, x_{n(I+1)-1/2}], \quad I = 0, \dots, N_c - 1$$

with the number of coarse cells $N_c = N_f/n$ and the corresponding mesh size $\Delta X = n\Delta x$. We also define coarse variables by spatial averaging

$$U_I(t) = \frac{1}{n} \sum_{j=nI}^{n(I+1)-1} u_j(t), \quad I = 0, \dots, N_c - 1, \quad (7)$$

which implies $U_I \equiv [H_I, Q_I]^\top$ and $Q_I = V_I H_I$ where H_I , Q_I , and V_I are the water height, discharge, and velocity on a coarse mesh. Averaging Equation (4) and taking into account the flux structure (5) and the telescoping sum, we obtain equations for U_I

$$\begin{aligned} \frac{d}{dt} U_I &= -\frac{F_{n(I+1)-1/2} - F_{nI-1/2}}{\Delta X} + \rho_I^U \\ &= -\frac{\tilde{F}_{n(I+1)-1/2} - \tilde{F}_{nI-1/2}}{\Delta X} - \frac{F_{n(I+1)-1/2}^{visc} - F_{nI-1/2}^{visc}}{\Delta X} + \rho_I^U, \end{aligned} \quad (8)$$

where the fluxes $F_{n(I+1)-1/2}$ and $F_{nI-1/2}$ at the two interfaces of the coarse cell $\bar{C}_I$ are each decomposed into two parts: a nonlinear part of the fine-mesh flux, denoted by $\tilde{F}_{n(I+1)-1/2}$ and $\tilde{F}_{nI-1/2}$, and a viscous flux, denoted by $F_{n(I+1)-1/2}^{visc}$ and $F_{nI-1/2}^{visc}$, respectively. Considering Equation (5), the nonlinear part of the flux is given by $\tilde{F}_{i-1/2} = (f(u_{i-1}) + f(u_i))/2$ and the viscous flux is given by $F_{i-1/2}^{visc} = \lambda_{i-1/2}(u_i - u_{i-1})/2$. The equation above is exact but not closed since $F_{nI-1/2} = \mathcal{F}(u_{nI-1}, u_{nI})$ and $F_{n(I+1)-1/2} = \mathcal{F}(u_{n(I+1)-1}, u_{n(I+1)})$ depend on fine-mesh variables. The forcing $\rho_I^U := [0, \rho_I^U]^\top$ is the average of the stochastic forcing over the coarse cell $\bar{C}_I$, i.e.,

$$\rho_I^U = \frac{1}{n} \sum_{j=nI}^{n(I+1)-1} \rho_j(t), \quad I = 0, \dots, N_c - 1.$$

In this paper, our goal is to develop closures for the nonlinear and viscous parts of the right-hand side. We assume that the forcing is slowly varying in space, so that the averaged forcing does not deviate significantly from the fine-grid forcing terms in the interval $\bar{C}_I$, i.e.,

$$\rho_I^U \approx \rho(x_j, t) \quad \text{for } j = nI, \dots, n(I+1) - 1.$$

Therefore, we omit the superscript U in the forcing term for the rest of the paper, and we use $\rho_I(t) := \rho(X_I, t)$ with $X_I = (I + 1/2)\Delta X$ in coarse simulations.

Equation (7) defines the resolved variables in coarse simulations, and sub-grid variables can be defined as fluctuations

$$\tilde{u}_i = u_i - U_{I(i)}.$$

Equation (8) can be rewritten using coarse variables U_I and fluctuations $\tilde{u}_i$, and the goal of sub-grid modeling is to find an appropriate closure that eliminates fluctuations $\tilde{u}_i$ from the equations for coarse variables.

We can rewrite Equation (8) as

$$\frac{d}{dt} U_I = -\frac{F_{I+1/2}^c - F_{I-1/2}^c}{\Delta X} + \rho_I,$$

where the “true” coarse fluxes are given by $F_{I+1/2}^c = \tilde{F}_{I+1/2}^c + F_{I+1/2}^{c,visc}$ with

$$\tilde{F}_{I+1/2}^c = \tilde{F}_{n(I+1)-1/2} \quad \text{and} \quad F_{I+1/2}^{c,visc} = F_{n(I+1)-1/2}^{visc},$$

and similarly for $F_{I-1/2}^c$.

The goal of sub-grid modeling is to find functions $G_{I+1/2}(\vec{U})$ and $G_{I-1/2}(\vec{U})$ (depend only on coarse variables) that approximate fluxes $F_{I+1/2}^c$ and $F_{I-1/2}^c$. Here, we treat the nonlinear and viscous parts of the subgrid flux separately and develop different subgrid models. In particular, we use a nonlinear function (neural network) to estimate the nonlinear flux $\tilde{F}_{I+1/2}^c$ and model the viscous flux $F_{I+1/2}^{c,visc}$ using the “stabilizing” LLF term. The resulting sub-grid model is given by

$$\tilde{F}_{I+1/2}^c \approx G_{I+1/2}(\vec{U}) \equiv \mathcal{G}(U_{I-1}, U_I, U_{I+1}, U_{I+2}), \quad (9)$$

$$F_{I+1/2}^{c,visc} \approx G_{I+1/2}^{visc} \equiv \frac{\Lambda_{I+1/2}}{2}(U_{I+1} - U_I), \quad (10)$$

with

$$\Lambda_{I+1/2} = \max(|V_I| + \sqrt{gH_I}, |V_{I+1}| + \sqrt{gH_{I+1}}).$$

With this approximation, the reduced model takes the following form

$$\frac{d}{dt}U_I = -\frac{G_{I+1/2} - G_{I-1/2}}{\Delta X} - \frac{G_{I+1/2}^{visc} - G_{I-1/2}^{visc}}{\Delta X} + \rho_I. \quad (11)$$

In [44], we demonstrated that linear regression can be used to justify Equation (10). At the same time, linear regression indicates that the nonlinear flux cannot be well approximated in the usual way, i.e., $\tilde{F}_{I+1/2}^c \not\approx \mathcal{F}(U_I, U_{I+1})$, especially near shocks.

To develop an accurate approximation of $\tilde{F}_{I+1/2}^c$, we utilize a neural network to estimate $\mathcal{G}$ such that $G_{I+1/2} = \mathcal{G}(U_{I-1}, U_I, U_{I+1}, U_{I+2})$ and $G_{I-1/2} = \mathcal{G}(U_{I-2}, U_{I-1}, U_I, U_{I+1})$. Thus, we develop a symmetric four-point stencil approximation for the nonlinear part of the flux $\tilde{F}_{I+1/2}^c$. A straightforward approach would be to consider a two-point stencil for $\mathcal{G}$, i.e., $G_{I+1/2} = \mathcal{G}(U_I, U_{I+1})$. However, relying solely on a two-point stencil for computing the subgrid fluxes resulted in a suboptimal performance of the coarse model. Thus, we utilized a four-point stencil, as discussed earlier. Similar findings were reported in [7,45–47], where a two-point stencil was found insufficient for developing an accurate sub-grid approximation. In particular, an analytical mode-reduction approach for stochastic multiscale systems was utilized in [45–47] and provided a rigorous justification for using the four-point stencil. This limitation likely arises because using only two points does not provide adequate information for estimating the local curvature of the solution. By increasing the stencil size to include additional neighboring resolved modes, the sub-grid model can better determine whether a sub-grid flux computation is required within a steep gradient or a smoother region.

3.2. Network Architecture, Dataset, and Training

To approximate the subgrid flux function $\mathcal{G}$ mentioned above, we employ a *feedforward neural network* (FNN). We consider a set of vector-valued functions $\tilde{\mathcal{G}}_1, \dots, \tilde{\mathcal{G}}_p$, and the target function $\mathcal{G}$ can be expressed as the composition

$$\mathcal{G} = \tilde{\mathcal{G}}_p \circ \dots \circ \tilde{\mathcal{G}}_1.$$

Each function $\tilde{\mathcal{G}}_i$ is referred to as the *i*-th *layer* of the network. In this context, $\tilde{\mathcal{G}}_1$ is the *input layer* and $\tilde{\mathcal{G}}_p$ is the *output layer*, and each function in between is referred to as a *hidden layer*.

The input layer accepts an 8-dimensional input, and the output layer computes two fluxes (for H and Q). Thus, our subgrid approximation is local, and the sub-grid flux approximation can be trained and evaluated independently for different sub-domains, if necessary. We use 3 hidden layers with 128 neurons each. To examine the sensitivity of the proposed closure model to architectural choices, we conduct additional experiments

by varying the network width, depth, and activation function. For network width, reducing the number of neurons to 64 per hidden layer resulted in a clear degradation in predictive accuracy, indicating insufficient expressive capacity to capture the nonlinear flux corrections. Increasing the width to 256 neurons yielded comparable or slightly improved performance, but with increased computational cost and no substantial long-term stability benefit. For network depth, using only two hidden layers led to noticeable instability in long-time simulations, suggesting that sufficient depth is required to represent the closure dynamics robustly. Increasing the depth beyond three layers did not produce significant accuracy improvements and introduced additional training complexity. We use the GELU (Gaussian Error Linear Unit) [48] activation function to address limitations associated with other activation functions. Unlike ReLU or leaky ReLU, GELU introduces regularization by smoothly blending the linear and nonlinear behavior. The GELU activation is defined as

$$f(z) = z \cdot \Phi(z),$$

where $\Phi(z)$ is the cumulative distribution function (CDF) of the standard normal distribution. This formulation allows the activation to approximate the identity for large positive inputs while smoothly suppressing negative inputs. GELU has been shown to enhance performance in deep neural networks by providing smooth, differentiable activation with input-dependent gating that improves gradient flow during training (e.g., [49]). We further compared activation functions and observed that ReLU-based architectures exhibited reduced stability and limited smoothness in the learned corrections. In contrast, GELU provided smoother nonlinear mappings and improved gradient flow, leading to more stable long-term predictions. Overall, these experiments indicate that while moderate architectural variations are feasible, the chosen configuration (three hidden layers with 128 neurons and GELU activation) provides a balanced trade-off between expressive capacity, stability, and computational efficiency. Figure 1 schematically illustrates the network architecture, with hidden neurons omitted for simplicity.

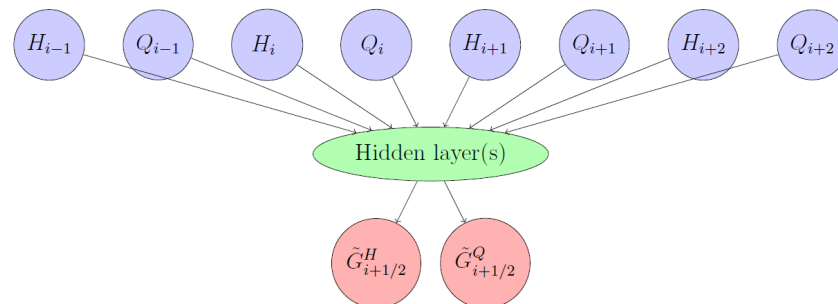


Figure 1. Visual representation of the neural network architecture used to approximate the sub-grid flux. The input layer consists of four resolved modes, each with two values (height and discharge). The hidden layers process this input, and the output layer provides the estimated subgrid flux components.

Training Dataset. We generate the training dataset by simulating the fine-mesh discretization of the SWE in (4) with $N_f = 1024$ and computing the coarse variables U_I and "true" fluxes $\tilde{F}_{I+1/2}^c$. Since we consider a spatially homogeneous problem, it is sufficient to compute the training data only for $I = 0$. We use the Heun time-stepping with $\Delta t = 0.01$ and simulate each trajectory for the total time $T = 400$. We sample data with the time-step $\Delta t^{sample} = 0.2$. Thus, each trajectory generates 2000 snapshots. It is important to note

that we do not include data for the intermediate stage of Heun's method. We generate 100 trajectories with different initial conditions generated randomly as follows

$$h_0(x) = H_0 + A_h \left(\sin\left(\frac{2\pi x}{L} + \phi_h^{(1)}\right) + \sin\left(\frac{4\pi x}{L} + \phi_h^{(2)}\right) \right), \quad (12)$$

$$v_0(x) = V_0, \quad (13)$$

where $x \in [0, L]$ with $L = 100$. The height average $H_0 = 2$ is constant for all initial conditions. The remaining parameters are randomly generated following the distributions

$$V_0 = \text{Unif}[1, 2], \quad A_h = \text{Unif}[0.1, 0.6], \quad \phi_h^{(1)}, \phi_h^{(2)} = \text{Unif}[0, 2\pi] \quad (14)$$

where $\text{Unif}[a, b]$ represents the continuous uniform distribution on $[a, b]$. We use $n = 8$ averaging points.

Smoothness Indicator. Smoothness Indicators (see, e.g., [50] and references therein) have been used in WENO schemes to measure local smoothness of the solution. The smoothness indicator plays a crucial role in high-resolution numerical schemes, particularly in weighted, essentially non-oscillatory methods. It measures the smoothness of a function $u(x)$ in specific regions of the computational domain, ensuring that the scheme adapts its weights to minimize oscillations near discontinuities while maintaining high-order accuracy in smooth regions.

The smoothness indicator β is defined as

$$\beta_i \equiv \beta(u_{i-1}, u_i, u_{i+1}) = \frac{13}{12}(u_{i-1} - 2u_i + u_{i+1})^2 + \frac{1}{4}(u_{i-1} - u_{i+1})^2. \quad (15)$$

This formula consists of two terms: the first term, $\frac{13}{12}(u_{i-1} - 2u_i + u_{i+1})^2$, represents the second-order derivative approximation, which captures the curvature of the function. The second term, $\frac{1}{4}(u_{i-1} - u_{i+1})^2$, represents the square of the first-order derivative, which ensures stability and reduces numerical oscillations. An example of the smoothness indicator, β_I , for a particular snapshot of the coarse solution $H_I(t)$ with $I = 0, \dots, N_c - 1$ is presented in Figure 2.

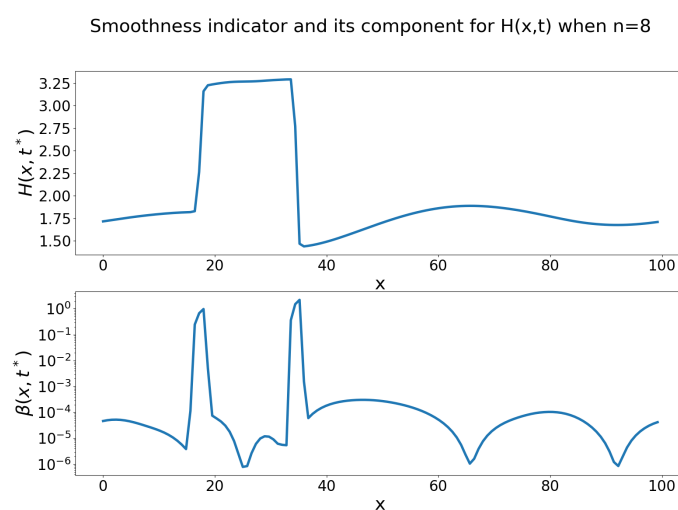


Figure 2. **Top:** a typical profile of the coarse solution $H_I(t)$; **bottom:** the corresponding smoothness indicator β_I in (15).

Figure 2 demonstrates that the smoothness indicator β accurately captures near-shock regions. The top part of Figure 2 also indicates that the majority of data $H_I(t)$,

$I = 0, \dots, N_c - 1$ corresponds to smooth regions since there is a relatively small number of shocks and they are highly localized in space. To improve the robustness of the learning process, we filter the training and testing data based on the smoothness indicator β . In the full dataset, the distribution of β is highly skewed toward small values, corresponding primarily to smooth flow regions. Specifically, the raw dataset has a mean of 1.35×10^{-2} , standard deviation of 1.05×10^{-1} , and spans a wide range from 8.88×10^{-12} to 7. Although the maximum value is large due to rare shock events, the 75th percentile is only 4.97×10^{-4} , indicating that most of the data corresponds to smooth states. Approximately 25% of the samples are associated with near-shock regions, identified by $\beta > 4 \times 10^{-4}$.

It is known that neural networks approximate continuous functions well [17,18], but perform poorly for discontinuous data. Therefore, training directly on this imbalanced dataset may bias the model toward predominantly smooth states while simultaneously exposing it to extreme shock outliers, which can degrade NN stability. To mitigate this issue, we compute the 0.6 and 0.8 quantiles of β and only retain samples that lie within this interval. This quantile-based filtering concentrates the learning process on moderately active transitional regions, excluding both extremely smooth states and extreme discontinuities. As a result, the neural network is trained on flow regimes where nonlinear effects are significant yet numerically controlled.

The initial unfiltered dataset consists of 3,200,000 samples. After applying the 0.6–0.8 quantile filtering, the dataset is reduced to 640,000 samples. The filtered β distribution becomes tightly concentrated, with a mean of 4.02×10^{-4} , standard deviation of 1.15×10^{-4} , and range of $[2.40 \times 10^{-4}, 6.51 \times 10^{-4}]$. This substantial reduction in variance confirms that the retained samples correspond to controlled high-gradient regions without extreme shock spikes, thereby improving numerical stability and enhancing predictive robustness. Finally, we randomly partition the filtered dataset into training and validation subsets: 80% of the samples are used for training, while the remaining 20% are reserved for validation.

Network Training. The training process for the neural network $\tilde{\mathcal{G}}$ follows a structured approach to ensure optimal learning and generalization. Training is performed using the Adam optimizer with a learning rate of 0.001, which facilitates efficient convergence.

The loss function employed is the *focal loss* [51], which encourages the model to focus more on difficult samples (e.g., data near shock regions). The classical focal loss was originally proposed for binary classification tasks to address class imbalance. Let $y \in \{\pm 1\}$ denote the ground-truth class and p be the model's estimated probability for the class with label $y = 1$, define

$$p_t = \begin{cases} p, & \text{if } y = 1, \\ 1 - p, & \text{otherwise,} \end{cases}$$

so that p_t represents the predicted probability assigned to the true class. The α -balanced variant of focal loss is then defined as

$$\text{FL}_{p_t} = -\alpha_t(1 - p_t)^\gamma \log p_t,$$

where $\gamma \geq 0$ is the focusing parameter that reduces the impact of well-classified samples and α_t is a class-dependent scaling factor. For regression tasks, class probabilities are unavailable. Instead, we consider the residual $r_j = \hat{y}_j - y_j$, where $\hat{y}_j = G(x_j; \omega, b)$ is the neural network's prediction for the input x_j , and y_j is the corresponding target value. We adopt a focal-style weighting mechanism and define the loss function as

$$C(\omega, b) := \frac{\alpha}{N} \sum_{j=1}^N \left(1 - e^{-r_j^2}\right)^\gamma r_j^2.$$

In this formulation, the term $\left(1 - e^{-r_j^2}\right)^\gamma$ acts as an adaptive weight. For large residuals, the exponential term is negligible, and the loss reduces to a scaled MSE, thereby emphasizing harder examples. In the present problem, shock regions correspond to rare, spatially localized, high-gradient events. The focal weighting reduces the dominance of smooth regions during training and encourages improved accuracy in the vicinity of shocks. In our experiments, we set $\alpha = 1$ and $\gamma = 2$.

To mitigate overfitting, early stopping with a patience of 100 epochs was employed, with a maximum training budget of 2000 epochs. The numerical simulations utilized a single compute node equipped with 4 CPU cores and 2 NVIDIA L40S GPUs, each with 46 GB of memory. Multi-GPU execution was enabled for training. Training was performed using PyTorch v2.4.1, and convergence was typically achieved within the allocated epoch budget. Training of each neural network model required approximately 2–4 h of wall-clock time.

3.3. Monolithic Convex Limiting

Using a neural network to estimate subgrid fluxes can lead to issues such as under- or overshooting, which may cause negative fluid heights and/or non-physical spurious oscillations near shocks. To prevent this, we apply the *monolithic convex limiting* (MCL) strategy (see, e.g., [36,40]). The MCL method allows controlling the subgrid fluxes produced by the neural network, ensuring that the results stay within an admissible set and, in particular, that the fluid height remains positive. First, we describe the general MCL methodology for the 1D SWE equations, and then we discuss how it is applied to our case.

3.3.1. General MCL Methodology for the 1D SWE

Here we present the general setup for the MCL methodology. The numerical scheme is presented for the SWE discretized on a general uniform mesh x_i . Consider a discretization of the SWE of the form

$$\frac{d}{dt}u_i = -\frac{(F_{i+1/2} + \tilde{G}_{i+1/2}^*) - (F_{i-1/2} + \tilde{G}_{i-1/2}^*)}{\Delta x}, \quad (16)$$

where $F_{i\pm 1/2}$ are the LLF fluxes given by Equation (5) and $\tilde{G}_{i\pm 1/2}^*$ are constrained approximations for a higher-order flux correction $\tilde{G}_{i\pm 1/2}$. Next, we write (16) in the equivalent bar state form (see [36,40] for details)

$$\frac{d}{dt}u_i = \frac{1}{\Delta x} \left[\lambda_{i-1/2}(\bar{u}_{i-1/2}^{*,+} - u_i) + \lambda_{i+1/2}(\bar{u}_{i+1/2}^{*,+} - u_i) \right], \quad (17)$$

where $\lambda_{i+1/2}$ is the maximum wave speed given by (6). The low-order LLF part is given by

$$\bar{u}_{i+1/2} := \frac{u_{i+1} + u_i}{2} - \frac{1}{2\lambda_{i+1/2}}(f(u_{i+1}) - f(u_i))$$

and

$$\bar{u}_{i+1/2}^{*,\pm} = \bar{u}_{i+1/2} \pm \frac{\tilde{G}_{i+1/2}^*}{\lambda_{i+1/2}}.$$

In the SWE context, the admissible set of solutions consists of all states $u = [h, hv]^\top$ such that $h \geq 0$. The bar state $\bar{u}_{i+1/2}$ represents an averaged exact solution of the Riemann problem with the initial states u_i and u_{i+1} [52,53]. Therefore, it can be shown that $\bar{u}_{i+1/2}$ is in the admissible set if $h_i, h_{i+1} \geq 0$. After applying the time-stepping scheme, Equation (17) indicates that $u_i(t + \Delta t)$ can be expressed as a convex combination of $\bar{u}_{i-1/2}^{*,+}$, $\bar{u}_{i+1/2}^{*,+}$, and u_i . Thus, if these three states are admissible, then $u_i(t + \Delta t)$ is also in the admissible set. The

MCL approach for the SWE [37] ensures that $\bar{u}_{i\pm 1/2}^{*,\mp}$ is also admissible whenever $\bar{u}_{i\pm 1/2}$ is admissible, with details given in [36,40]. We will discuss how to construct $\tilde{G}_{i+1/2}^*$ below.

Given that $\bar{u}_{i\pm 1/2}^{*,\mp}$ is admissible, each forward stage of Heun's method advances the initial condition as follows:

$$u_i^{\text{MCL}} = u_i + \frac{\Delta t}{\Delta x} \left[\lambda_{i-1/2} (\bar{u}_{i-1/2}^{*,+} - u_i) + \lambda_{i+1/2} (\bar{u}_{i+1/2}^{*,+} - u_i) \right].$$

If the time step satisfies the CFL condition $\Delta t / \Delta x [\lambda_{i+1/2} + \lambda_{i-1/2}] \leq 1$, then u_i^{MCL} is a convex combination of three admissible states. Therefore, for time-steps satisfying the CFL conditions, u_i^{MCL} is within the admissible set.

To construct $\tilde{G}_{i+1/2}^*$, The MCL algorithm for the SWE system [37,40] imposes local bounds on h and v . Consider components of the vector-valued flux $\tilde{G}_{i+1/2} = [\tilde{G}_{i+1/2}^h, \tilde{G}_{i+1/2}^q]^\top$ and the vector-valued solution $\bar{u}_{i+1/2} = [\bar{h}_{i+1/2}, \bar{h}_{i+1/2} \bar{v}_{i+1/2}]^\top$, where $\bar{v}_{i+1/2} := \bar{q}_{i+1/2} / \bar{h}_{i+1/2}$. The local bounds are defined as

$$\begin{aligned} h_i^{\min} &:= \min(\bar{h}_{i-1/2}, \bar{h}_{i+1/2}), & h_i^{\max} &:= \max(\bar{h}_{i-1/2}, \bar{h}_{i+1/2}), \\ v_i^{\min} &:= \min(\bar{v}_{i-1/2}, \bar{v}_{i+1/2}), & v_i^{\max} &:= \max(\bar{v}_{i-1/2}, \bar{v}_{i+1/2}) \end{aligned}$$

and the local maximum principle can be formulated [37] as

$$\begin{aligned} h_i^{\min} &\leq \bar{h}_{i+1/2}^{*,+} \leq h_i^{\max}, & h_{i+1}^{\min} &\leq \bar{h}_{i+1/2}^{*,+} \leq h_{i+1}^{\max}, \\ \bar{h}_{i+1/2}^{*,+} v_i^{\min} &\leq \bar{q}_{i+1/2}^{*,+} \leq \bar{h}_{i+1/2}^{*,+} v_i^{\max}, & \bar{h}_{i+1/2}^{*,+} v_{i+1}^{\min} &\leq \bar{q}_{i+1/2}^{*,+} \leq \bar{h}_{i+1/2}^{*,+} v_{i+1}^{\max}. \end{aligned} \quad (18)$$

Since $\bar{h}_{i+1/2}^{*,\pm} = \bar{h}_{i+1/2} \pm \tilde{G}_{i+1/2}^{h,*} / \lambda_{i+1/2}$, the water height constraints can be rewritten as

$$\begin{aligned} \lambda_{i+1/2} (\bar{h}_{i+1/2} - h_i^{\max}) &\leq \tilde{G}_{i+1/2}^{h,*} \leq \lambda_{i+1/2} (\bar{h}_{i+1/2} - h_i^{\min}), \\ \lambda_{i+1/2} (h_{i+1}^{\min} - \bar{h}_{i+1/2}) &\leq \tilde{G}_{i+1/2}^{h,*} \leq \lambda_{i+1/2} (h_{i+1}^{\max} - \bar{h}_{i+1/2}). \end{aligned}$$

Therefore, it follows that the bound-preserving approximation to $\tilde{G}_{i+1/2}^h$ is

$$\tilde{G}_{i+1/2}^{h,*} = \begin{cases} \min(\tilde{G}_{i+1/2}^h, \lambda_{i+1/2} \min(\bar{h}_{i+1/2} - h_i^{\min}, h_{i+1}^{\max} - \bar{h}_{i+1/2})) & \text{if } \tilde{G}_{i+1/2}^h \geq 0, \\ \max(\tilde{G}_{i+1/2}^h, \lambda_{i+1/2} \max(\bar{h}_{i+1/2} - h_i^{\max}, h_{i+1}^{\min} - \bar{h}_{i+1/2})) & \text{otherwise.} \end{cases} \quad (19)$$

The constraint for the discharge is defined using $\Delta \tilde{G}_{i+1/2}^{q,*} = \tilde{G}_{i+1/2}^{q,*} - \tilde{G}_{i+1/2}^{h,*} \bar{v}_{i+1/2}$. The limited flux for the discharge $\tilde{G}_{i+1/2}^{q,*}$ is computed from the limited difference $\Delta \tilde{G}_{i+1/2}^{q,*}$ given by (see [37])

$$\begin{aligned} \lambda_{i+1/2} \bar{h}_{i+1/2}^{*,+} (\bar{v}_{i+1/2} - v_i^{\max}) &\leq \Delta \tilde{G}_{i+1/2}^{q,*} \leq \lambda_{i+1/2} \bar{h}_{i+1/2}^{*,+} (\bar{v}_{i+1/2} - v_i^{\min}), \\ \lambda_{i+1/2} \bar{h}_{i+1/2}^{*,+} (v_{i+1}^{\min} - \bar{v}_{i+1/2}) &\leq \Delta \tilde{G}_{i+1/2}^{q,*} \leq \lambda_{i+1/2} \bar{h}_{i+1/2}^{*,+} (v_{i+1}^{\max} - \bar{v}_{i+1/2}) \end{aligned}$$

Conditions above can be rewritten as

$$\Delta \tilde{G}_{i+1/2}^{q,*} = \begin{cases} \min(\Delta \tilde{G}_{i+1/2}^q, \Lambda_{i+1/2} \min(\bar{h}_{i+1/2}^{*,+} (\bar{v}_{i+1/2} - v_i^{\min}), \bar{h}_{i+1/2}^{*,+} (v_{i+1}^{\max} - \bar{v}_{i+1/2}))) & \text{if } \Delta \tilde{G}_{i+1/2}^q \geq 0, \\ \max(\Delta \tilde{G}_{i+1/2}^q, \Lambda_{i+1/2} \max(\bar{h}_{i+1/2}^{*,+} (\bar{v}_{i+1/2} - v_i^{\max}), \bar{h}_{i+1/2}^{*,+} (v_{i+1}^{\min} - \bar{v}_{i+1/2}))) & \text{otherwise.} \end{cases} \quad (20)$$

Conditions (18) guarantee positivity preservation and admissibility of numerical solutions to the 1D SWE.

3.3.2. Application of MCL in the Context of Machine Learning

Here we discuss the application of the MCL algorithm on the coarse grid. Here, the coarse index I should be substituted into all MCL formulas presented in Section 3.3.1. To apply the MCL formalism to our machine learning model, we rewrite the model in (4) as

$$\frac{d}{dt}U_I = -\frac{\bar{G}_{I+1/2} - \bar{G}_{I-1/2}}{\Delta X} - \frac{\tilde{G}_{I+1/2}^* - \tilde{G}_{I-1/2}^*}{\Delta X} - \frac{G_{I+1/2}^{visc} - G_{I-1/2}^{visc}}{\Delta X} + \rho_I, \quad (21)$$

where $\bar{G}_{I+1/2}$ is the nonlinear part of the LLF flux, and $\tilde{G}_{I+1/2}^*$ is the limited difference between the neural network and $\bar{G}_{I+1/2}$, i.e.

$$\bar{G}_{I+1/2} = \frac{f(U_{I+1}) + f(U_I)}{2}, \quad \tilde{G}_{I+1/2}^* = MCL(G_{I+1/2} - \bar{G}_{I+1/2}),$$

where $G_{I+1/2}$ is the neural network model given by (9). In the context of MCL formalism in (16) and the reduced model in (11), $u_I \equiv U_I$, $F_{I+1/2} = \bar{G}_{I+1/2} + G_{I+1/2}^{visc}$, and $\tilde{G}_{I+1/2}^*$ is computed using (19) and (20). In this context, $\tilde{G}_{I+1/2}^*$ can be interpreted as the limited difference between low-order and high-order neural network approximations of nonlinear terms. Therefore, the application of the MCL formalism is rather straightforward—first, the bar states are computed using the LLF scheme, and then the limiter is applied to the difference between the fluxes generated by the neural network and the nonlinear part of the LLF flux. Time-splitting is used to add the stochastic forcing after the second Heun step.

4. Numerical Results

In this section, we describe our numerical results for the reduced model with and without the MCL. By analogy with turbulence modeling, we refer to the simulations of the full model as DNS. We verified numerically that simulations of the full model with $N_f = 1024$ are adequate for reproducing the solutions of the SWE. In particular, the numerical diffusion is small and has a minimal effect. Therefore, we use $N_f = 1024$ to generate training and testing datasets and benchmark the performance of our reduced models.

To train our neural network model, we consider the averaging window $n = 8$, which results in the number of coarse degrees of freedom $N_c = 128$. After training, we also use the same neural network (without re-training) to estimate coarse fluxes in simulations with $N_c = 64, 256$, and 512 coarse variables. This allows us to test how well our machine learning model generalizes with respect to changes in the resolution of coarse equations. To distinguish between reduced models with different spatial resolution, the number of coarse degrees of freedom is appended to the abbreviation of the corresponding reduced model. For instance, NN reduced models (11) *without the limiter* are referred to as NN-64, NN-128, NN-256, and NN-512. NN reduced models (21) *with the limiter* are referred to as NN-MCL-128 and NN-MCL-256. We also compare our reduced models with simulations of the straightforward LLF discretization of the SWE in (4) and denote those as LLF-64, LLF-128, LLF-256, and LLF-512.

The length of the domain is $L = 100$ m, the final time is $T = 400$ s, and the gravitational acceleration is $g = 9.812$ m/s². We use periodic boundary conditions, and initial conditions are generated according to (12), (13). The Heun method is used to perform the time-stepping for the deterministic part, and Euler time-stepping is utilized to add the stochastic forcing after the second Heun step. The computational time step is $\Delta t = 0.01$. We performed a preliminary investigation and determined that this time step satisfied the CFL condition in all simulations, i.e., $\Delta t < \max_{i,t}(\lambda_{i+1/2})\Delta x$ and $\Delta t < \max_{I,t}(\Lambda_{I+1/2})\Delta X$. For the forcing in (2), (3) we use $A = 0.1$, $\psi = 1 - \Delta t$, and $\sigma = 1.41\sqrt{\Delta t}$, so that $\alpha_k(t), \beta_k(t) \sim N(0, 2/(2 - \Delta t))$ with decaying covariance $\mathbb{E}[\alpha_k(t + m\Delta t)\alpha_k(t)] \sim \psi^m$. We use the same computational time-step $\Delta t = 0.01$ to update $\alpha_k(t)$ and $\beta_k(t)$ after the time-stepping for the SWE.

We generate long stationary time series of dependent variables and compute energy spectra in Fourier space. For any periodic function $u(x, t)$, the Fourier expansion is given by $u(x, t) = \sum_k \hat{u}(t) e^{i2\pi kx/L}$, and the energy spectra is defined as

$$e_k = \frac{1}{T} \int_0^T |\hat{u}_k(t)|^2 dt.$$

We use $T = 400$ and sample data every $\Delta t^{sample} = 0.2$.

Figure 3 depicts the energy spectra of the water height $h(x, t)$ in the simulations of the full model and reduced models with resolutions $N_c = 64, 128, 256$, and 512. Note that the NN is trained in the regime $n = 8$ ($N_c = 128$). At the lowest resolution $N_c = 64$, the NN reduced model exhibits the largest discrepancies compared with the fully resolved model. As the resolution increases ($N_c = 128, 256, 512$), the NN reduced model aligns more closely with the true spectrum. Figure 3 demonstrates that the reduced machine learning models performs much better than the LLF discretization with the same spatial resolution. In particular, the NN reduced model reproduces the inertial range of the full model up to wavenumber $k \approx 13$ and $k \approx 37$ for $N_c = 128$ and $N_c = 256$, respectively. This demonstrates that the NN reduced model improves the representation of energy transfer across scales. However, the NN-128 model is more diffusive compared to the NN-256 model. Representation of the inertial range in simulations with other values of N_c is also improved by the NN reduced model. Energy spectra for the discharge, q , follow a similar trend; some of the results for q are presented later in this paper. These results indicate that the NN reduced model results in a better approximation of fine-scale structures, ensuring a more accurate representation of energy transfer across scales. Thus, the neural network considerably improves the missing subgrid dynamics in coarse simulations.

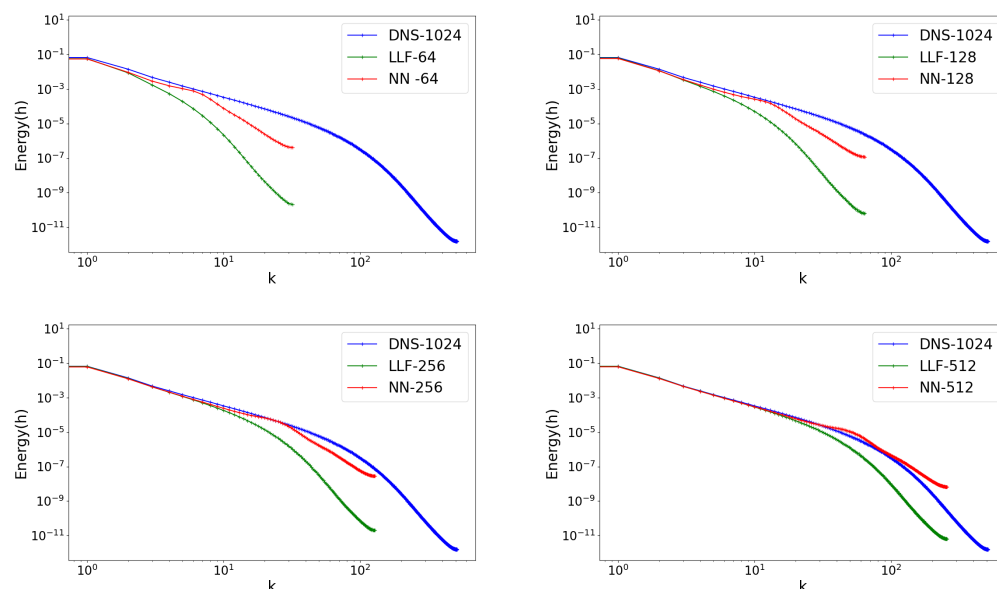


Figure 3. Energy spectra for the water height $h(x, t)$ in long stationary simulations of the full model and reduced models with different resolutions. Note that the reduced neural network model is trained in the $N_c = 128$ regime and is used for other resolutions without retraining. Blue line—DNS-1024 (direct numerical simulation with 1024 grid points), red Line—reduced neural network model with resolution N_c , green Line—local Lax–Friedrichs solver with resolution N_c . Simulations with $N_c = 64, 128, 256, 512$ —top left, top right, bottom left, bottom right, respectively.

The NN reduced model conserves the total mass $\int h(x, t) dx$ up to a very high precision, less than 0.1% relative error. This is due to the construction of the reduced model in

flux form. The kinetic and potential energies (given by $\frac{1}{2} \int uq dx$ and $\frac{1}{2} g \int h^2 dx$, respectively) fluctuate over time in response to the random forcing $\rho(x, t)$ applied to the equation for the discharge. The power spectra indicate that the reduced model is slightly more diffusive compared with the fully resolved model, resulting in slightly lower total energy. Indeed, both potential and kinetic energies are slightly smaller in the simulations of the reduced model compared with the fully resolved model if both models are simulated with the identical forcing; this is especially visible at the energy peaks (in time). The largest (peak) relative errors for the potential and kinetic energies are approximately 3–5% and 10–12%, respectively [44]. The MCL does not significantly affect the mass conservation and energy balance.

4.1. Simulations with Larger Forcing

Next, we investigate how well the NN reduced model performs outside the training regime. To this end, we consider larger forcing magnitudes of $A = 0.12$ and 0.14 , corresponding to 20% and 40% increases in the forcing magnitude. Since forcing is only applied to the discharge, q , an increase in forcing magnitude has a stronger effect on q than on h . Table 1 summarizes the total spectral energy $E = \frac{1}{2} \sum_k e_k$ for h and q . As expected, the increase in the energy for h is not significant, but energy for q increases by 27.5% and 54.6%, respectively, due to the nonlinear nature of the equation for the discharge. The NN reduced model responds correctly to the increase in forcing and, overall, accurately reproduces the energy increase in q . The discrepancy between the total energy in the DNS and the NN reduced model is larger for q than for h , most likely due to the nonlinear fluxes in the equation for the discharge. Similar to the approach in [47], to fix the overall energy budget, it is possible to introduce a single (possibly trainable) parameter to multiply the NN parametrization in the equation for the discharge, which is a straightforward modification of the NN parametrization approach presented here, and we do not discuss it in the present paper.

Table 1. Comparison of total spectral energy for h and q for different resolutions, $N_c = 128$ and $N_c = 256$ in the full model (E_{DNS}) and NN reduced model (E_{NN}). The spectral energy is defined as the sum of spectra $E = \frac{1}{2} \sum_k e_k$ for h and q , respectively. The initial conditions for the water depth are initialized with $H_0 = 2$. Therefore, the spectral energy for h includes the mean energy; the fluctuating energy can be computed as $E^h - H_0^2$.

N_c	$A = 0.1$				$A = 0.12$				$A = 0.14$			
	E_{DNS}^h	E_{NN}^h	E_{DNS}^q	E_{NN}^q	E_{DNS}^h	E_{NN}^h	E_{DNS}^q	E_{NN}^q	E_{DNS}^h	E_{NN}^h	E_{DNS}^q	E_{NN}^q
128	4.093	4.078	1.858	1.564	4.12	4.10	2.37	2.02	4.148	4.128	2.873	2.496
256	4.093	4.083	1.858	1.677	4.12	4.11	2.37	2.17	4.148	4.136	2.873	2.655

Figure 4 depicts energy spectra for h and q in simulations with $N_c = 128$ and $A = 0.1$, 0.12 , and 0.14 . Energy spectra computed from the NN reduced model represent a considerable improvement over coarse LLF discretization, indicating that the neural network effectively reconstructs the subgrid contributions. The NN reduced model maintains fidelity for a wide range of forcing amplitudes, demonstrating robustness in capturing the small-scale energy dynamics. These results suggest that the NN model can generalize well across varying amplitudes and recover fine-scale structures without retraining.

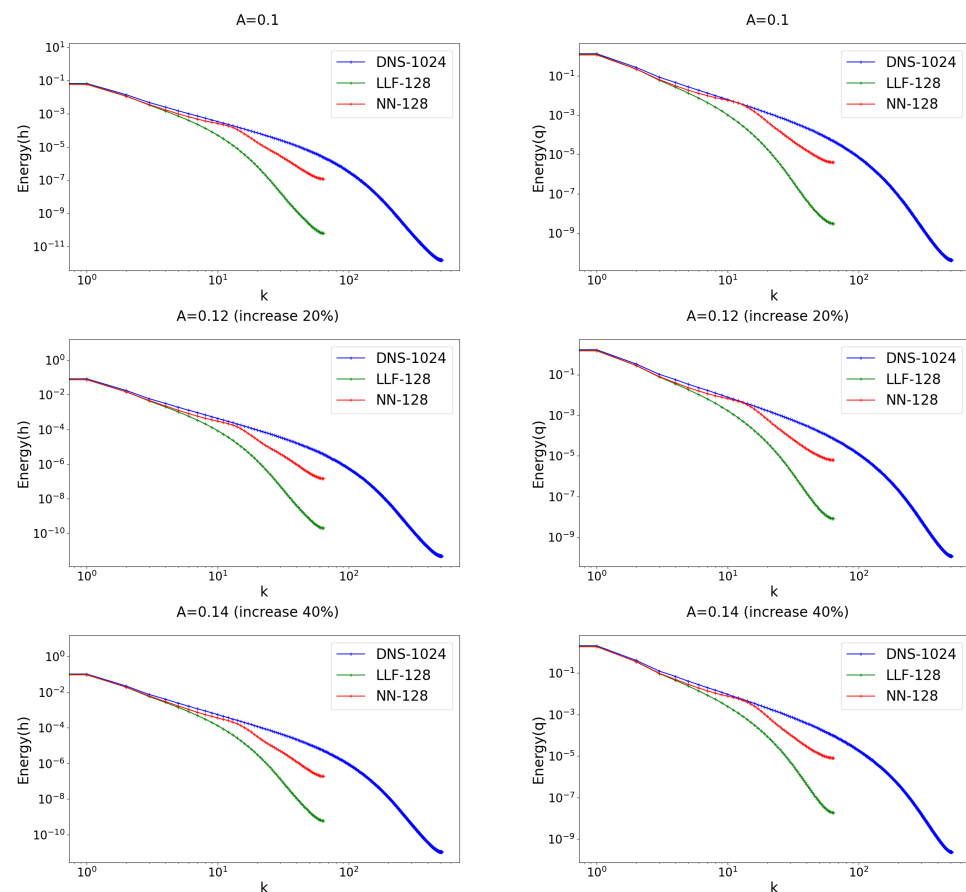


Figure 4. Energy spectra for h (left) and q (right) in simulations with $N_c = 128$ and forcing amplitudes $A = 0.1$ (top row), 0.12 (middle row), and 0.14 (bottom row). Note that the neural network parametrization is originally trained on $A = 0.1$ and $N_c = 128$; it is used without retraining in other regimes.

To further assess the performance of the NN reduced model, we compare individual solutions computed from the DNS and NN-256 with the same initial condition and forcing realization. We plot snapshots for $h(x, t)$ at times $t = 198, 209$, and 334 in Figure 5. These snapshots correspond to stages where the flow exhibits strong nonlinear behavior, including the steepening of hydraulic jumps and sharp transitions in the free surface. Solutions of the NN reduced model exhibit strong spurious oscillations near shocks. This indicates that the neural network does not reproduce fluxes at discontinuities with the same accuracy as in smooth regions. Since it is known that neural networks perform well when approximating continuous functions [17,18], it is not surprising that the quality of the NN approximation for fluxes diminishes near shocks because the fluxes are also discontinuous at that point. In addition, filtering of the training data (Section 3.2) can also contribute to the discrepancies of subgrid flux estimation by the NN at the shock, since the filtered dataset does not contain data with very high gradients. To alleviate these oscillations, we apply the MCL limiter to the nonlinear fluxes generated by the neural network, as discussed in Section 3.3. The left and right parts of Figure 5 depict snapshots without and with the MCL, respectively. We also performed simulations of the NN reduced model with $N_c = 128$ (see [44] for details). The overall performance of the NN reduced model with $N_c = 128$ is similar to the results for NN-256, except that NN-128 is more diffusive near shocks and does reproduce sharp gradients as well as NN-256, as is evident from the plot of the spectrum in Figure 3. The energy spectra for both h and q are not affected by the MCL and are therefore not shown for brevity.

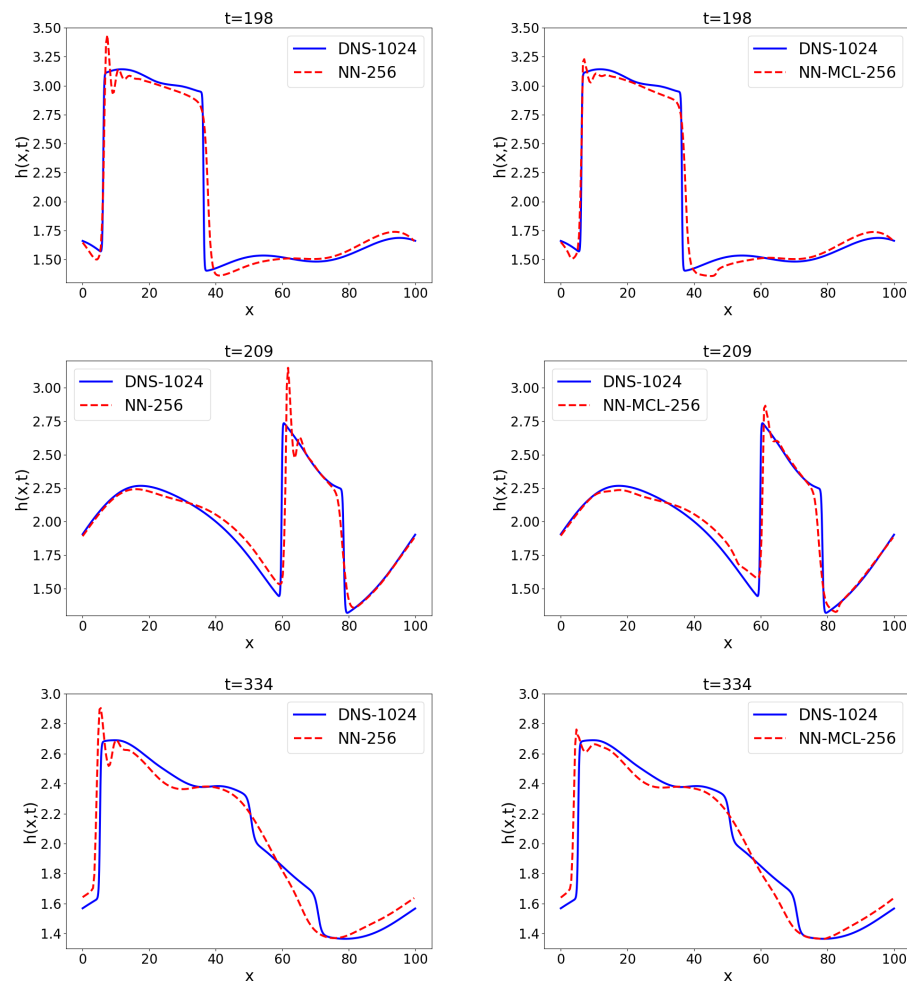


Figure 5. Comparison between snapshots $h(x, t)$ in simulations obtained from the reduced neural network model with 256 grid points (NN-256, red) and the reference direct numerical simulation with 1024 grid points (DNS-1024, blue) in the **left column**, and the neural network model with monotonicity-constrained learning at resolution 256 (NN-MCL-256, red) and DNS-1024 (blue) in the **right column**, at times $t = 198, 209, 334$. The NN-256 model shows very good agreement with the DNS reference, successfully capturing the hydraulic jump and the nonlinear wave propagation. The monotonicity-constrained learning (MCL) strategy is essential in reducing oscillations near shocks in simulations of the neural network reduced model.

4.2. Simulations with Topography and Manning Friction

To illustrate the applicability of our NN model outside of the training regime, we also perform simulations of SWE with bottom topography and Manning friction. This is a challenging test since we consider changes in the geometry of the computational domain, but do not retrain the NN parametrization.

We add the terms $[0, S_{topo} + S_{fric}]^T$ to the right-hand side of the SWE Equation (1) and perform fully resolved and coarse simulations with the NN parametrization trained in the original regime, as outlined in Section 3.2. The topography and friction terms are given by

$$S_{topo} = -g h \partial_x b(x), \quad S_{fric} = -g m^2 \frac{q |q|}{h^{7/3}},$$

where $b(x)$ is a known bottom topography and m is the Manning's roughness coefficient. The inclusion of these source terms follows the standard formulation of shallow water flow over variable topography with friction [54]. We used the numerical scheme in [55]; the scheme is non-negativity preserving and well-balanced (see also [56] for a discussion on this

issue). The Manning friction is a nonlinear term that damps the momentum, thus reducing the velocity of the flow. The Manning friction can also be expressed as $S_{fric} = -gm^2u|u|/h$; thus, this damping effect is particularly relevant for high-velocity flows. Since Manning friction does not depend on the gradient of the solution, it does not affect the shock structure for low to medium-velocity flows. The main consequence of including the Manning friction is the reduction in the shock magnitude. Spurious oscillations near the shock might be reduced slightly with Manning friction, but are still quite pronounced and require the application of the MCL strategy. We consider the Gaussian topography $b(x) = A_b \exp[-B(x - L/2)^2]$ with the amplitude $A_b = 0.3$ and steepness $B = 0.1$. The Manning's roughness coefficient is $m = 0.05$; additional examples are presented in [44]. The initial condition consists of a spatially varying free surface $\eta(x, 0)$ superimposed on the bottom topography, with the discharge $q(x, 0)$ chosen so that the flow is initially subcritical, i.e.,

$$\eta_0(x) = H_0 + A_h \left(\sin\left(\frac{2\pi x}{L} + \phi_h^{(1)}\right) + \sin\left(\frac{4\pi x}{L} + \phi_h^{(2)}\right) \right),$$

$$h_0(x) = \eta_0(x) - b(x), \quad v_0(x) = V_0,$$

where $x \in [0, L]$ with $L = 100$. The height average $H_0 = 2$ is constant for all initial conditions. The remaining parameters are randomly generated following the distributions in (14).

As a reference solution, we use a numerical solution computed with the LLF scheme on a fine grid with $N_x = 1024$. We consider coarse-grid simulations with $N_c = 128$ and $N_c = 256$. Figure 6 depicts snapshots of the free surface elevation $\eta(x, t) = h(x, t) + b(x)$ in simulations with $N_c = 256$. Coarse simulations are similar, but slightly more diffusive, as discussed previously for simulations without topography. Snapshots for $N_c = 128$ are not depicted for brevity (details can be found in [44]). These results support our conclusions drawn previously. In particular, the NN reduced model reproduces the main features of the reference solution, including the deformation of the free surface above the topography and the sharp gradients induced by the interaction between topography and friction. The NN model without a limiter can exhibit small overshoots near steep fronts, whereas the NN-MCL model noticeably reduces these oscillations while preserving the overall amplitude and phase of the solution. Moreover, similar to our previous findings, the NN-MCL model does not significantly affect the spectra of long-term simulations. Simulations with topography and friction show that the NN reduced model applies to a wide range of parameters outside of the training regime. We would like to point out that any machine learning model geared towards reproducing solutions directly (e.g., PINN and its variants) would most likely fail the topography test since in those models, it is very difficult to take geometry changes into account. Changes in computational geometry result in large differences in the solution itself, and our NN coarse model is able to track those changes very well (c.f. Figures 5 and 6).

Our numerical results demonstrate that the combination of NN subgrid parametrization and convex limiting yields a stable and accurate reduced-order model even in the presence of nontrivial source terms such as topography and Manning friction. Moreover, since our machine learning formalism is designed to learn fluxes in the SWE, its applicability is much broader than that of machine learning methods that directly learn the PDE solution.

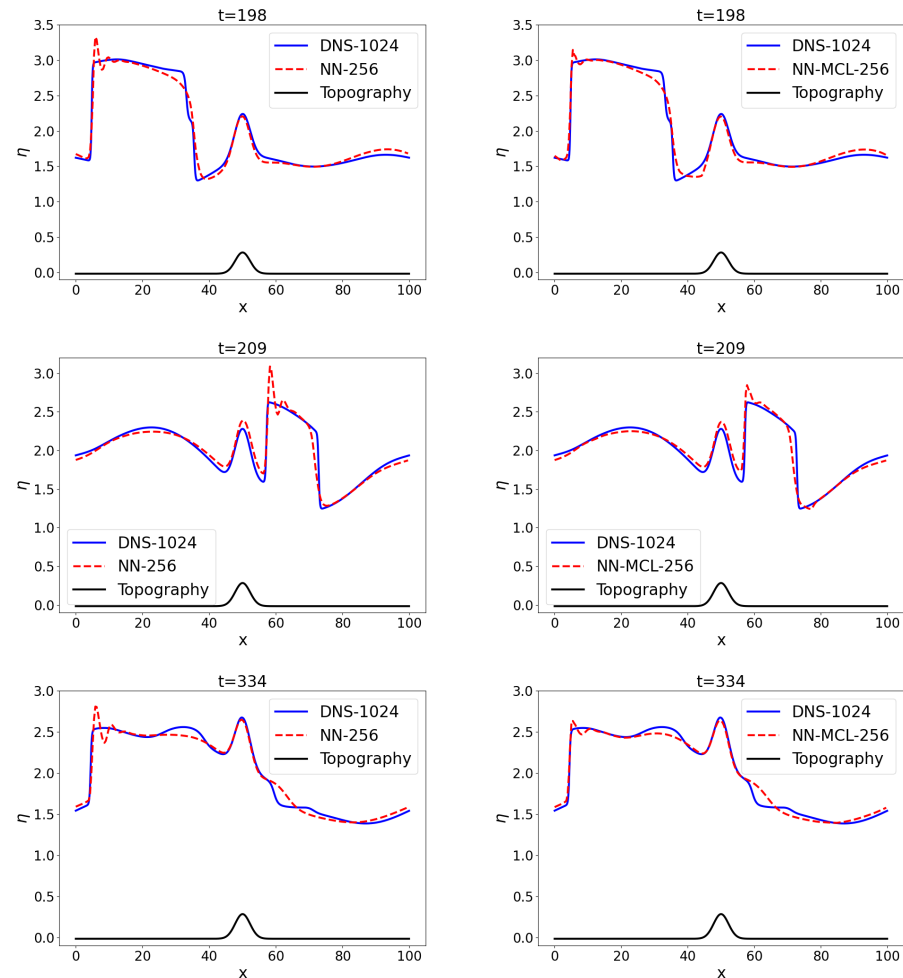


Figure 6. SWE simulations with topography and Manning friction. Comparison between snapshots of the free surface $\eta(x, t)$ in simulations of the reference direct numerical simulation with 256 grid points (NN-256, red) and DNS-1024 (blue) (**left column**) and the neural network model with monotonicity-constrained learning at resolution 256 (NN-MCL-256, red) and DNS-1024 (blue) (**right column**) at times $t = 198, 209, 334$. The monotonicity-constrained learning (MCL) strategy is essential in reducing oscillations near shocks in simulations of the NN reduced model. Note that the NN parametrization was trained on data generated with $S_{topo} = S_{fric} = 0$.

5. Conclusions

In this paper, we developed a reduced-order model for the shallow water equations by using a feed-forward neural network to approximate subgrid fluxes. Applying machine learning techniques to learn subgrid fluxes is a novel aspect of our work. As a consequence, the developed approach has several advantages. In particular, the subgrid flux parametrization is local, requiring only a four-point computational stencil to compute the subgrid flux at a specific coarse location. Moreover, our machine learning approach can be easily combined with flux limiting for hyperbolic problems. Flux limiting ensures that numerical solutions of the coarse equations remain within a physically admissible set. Another novel aspect of our work is using the focal loss cost function [51] in the context of neural networks applied to partial differential equations. Focal loss emphasizes samples that are particularly challenging for training, such as shock regions for hyperbolic problems.

We performed several types of tests: (i) comparison of energy spectra in long-term turbulent simulations, (ii) comparison of individual solutions, (iii) simulations with a larger forcing magnitude, and (iv) simulations with bottom topography and Manning's friction. Our results demonstrate that our Neural Network (NN) parametrization improves energy

exchange across scales. The NN reduced model is slightly more diffusive than the fully resolved simulations, which is a direct consequence of the coarse resolution of the reduced model and the need to dissipate energy at small scales. However, the NN reduced model represents an improvement compared with the bare truncation. In particular, our reduced model has a better representation of the inertial range compared with the bare truncation at the same resolution. An averaging window $n = 8$ (NN-128 reduced model vs the fully resolved model (DNS-1024)) is a realistic choice in most ocean applications. For instance, the ocean resolution of standard Global Circulation Models is 1° or 100 km [57]. The high-resolution eddy-permitting global ocean models have resolutions of approximately 10–25 km [58,59]. Our results indicate that our approach can yield improvements in the development of computationally efficient coarse models of ocean dynamics.

Test (ii) evaluated the performance of our reduced model by comparing individual solutions at full and coarse resolutions. We ensured that the same forcing realization was used in both cases. Our findings show that the NN reduced model is quite accurate and reproduces individual solutions very well. Some spurious oscillations may develop near shocks, but the monolithic convex limiting significantly reduces these oscillations while not affecting the energy spectra.

Tests (iii) and (iv) demonstrate that our reduced model generalizes well to parameters outside of the training regime. These tests were performed using an NN trained in a regime with a fixed forcing magnitude, a flat bottom, and no friction. In test (iii), the NN coarse model responds well to changes in the forcing magnitude and tracks changes in the discharge energy. Test (iv) is particularly challenging since it involves a different geometry of the computational domain. Our NN reduced model reproduces the energy spectra and individual solutions in test (iv) with the same accuracy as in the training regime. This demonstrates the versatility of our approach, which is a direct consequence of the fact that we use NNs to model subgrid fluxes and not the solution itself.

Overall, we demonstrated the applicability and versatility of using NNs to model subgrid fluxes in computational fluid dynamics. Our modeling approach preserves the structure of the equations, such as flux formulation. This leads to several advantages compared to learning the solution directly. First, our method is computationally efficient, since NN subgrid fluxes at different spatial nodes can be computed in parallel. While this is not significant for the 1D case, this can become important for 2D layered simulations. Second, our method can be easily combined with traditional flux limiting schemes to ensure admissibility of solutions, thus using the powerful existing numerical analysis machinery. Third, since we train an NN to reproduce subgrid fluxes and not solutions, our NN reduced equation applies without retraining to a wide range of parameters outside of the training regimes.

Our approach can be extended to 2D multi-layered shallow-water simulations in a relatively straightforward manner. In particular, neural networks modeling fluxes in two different directions can be trained independently for each layer, and the overall training dataset can be split into several parts for training each individual network. The main challenges for extending our work to 2D are (i) training and testing data generation is considerably more computationally expensive, (ii) data storage can also be an issue, especially for layered shallow-water equations, and (iii) the implementation issue since the MCL strategy in 2D is more complex. However, we do not foresee any conceptual issues when extending our approach to 2D shallow water problems.

Most climate change ocean applications utilize primitive equations, where the vertical structure becomes more important. Our method does not model the vertical transport or mixing (e.g., the diapycnal parametrization) in the primitive equations. We believe that after discretizing primitive equations in the vertical direction, our approach can be used

in each horizontal layer; however, this requires a separate investigation, since primitive equations are often discretized in isopycnal coordinates (density-based layers).

Our results raise several important questions for future research. One direction is to explore whether additional physical properties, such as entropy inequalities, can be incorporated into the model. Also, flux limiter constraints can be potentially included in the cost function as well. Since our method is suitable for conservation laws, it could be extended to other systems. One system of particular interest is multi-layer shallow-water dynamics in 2D, which is closely related to primitive equations commonly used to simulate ocean dynamics. In this application, it is particularly interesting to investigate the effect of the Coriolis force, which is absent in 1D. Another interesting case is the shallow water equations with wetting and drying.

Author Contributions: Conceptualization, I.T.; Methodology, Z.X., M.W. and I.T.; Software, M.A.H.M., Z.X. and I.T.; Validation, M.A.H.M. and Z.X.; Investigation, M.A.H.M., Z.X. and M.W.; Data curation, M.A.H.M. and Z.X.; Writing—original draft, M.A.H.M., Z.X., M.W. and I.T.; Writing—review & editing, Z.X., M.W. and I.T.; Visualization, M.A.H.M.; Supervision, Z.X., M.W. and I.T.; Project administration, I.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Acknowledgments: The authors would like to thank D. Kuzmin for helpful discussions about the MCL strategy.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Schneider, T.; Teixeira, J.; Bretherton, C.S.; Brient, F.; Pressel, K.G.; Schär, C.; Siebesma, A.P. Climate Goals and Computing the Future of Clouds. *Nat. Clim. Chang.* **2017**, *7*, 3–5. [\[CrossRef\]](#)
2. Wilcox, E.M.; Donner, L.J. The Frequency of Extreme Rain Events in Satellite Rain-Rate Estimates and an Atmospheric General Circulation Model. *J. Clim.* **2007**, *20*, 53–69. [\[CrossRef\]](#)
3. Bechtold, P.; Köhler, M.; Jung, T.; Doblas-Reyes, F.; Leutbecher, M.; Rodwell, M.J.; Vitart, F.; Balsamo, G. Advances in Simulating Atmospheric Variability with the ECMWF Model: From Synoptic to Decadal Time-Scales. *Q. J. R. Meteorol. Soc.* **2008**, *134*, 1337–1351. [\[CrossRef\]](#)
4. Farneti, R.; Gent, P.R. The Effects of the Eddy-Induced Advection Coefficient in a Coarse-Resolution Coupled Climate Model. *Ocean Model.* **2011**, *39*, 135–145. [\[CrossRef\]](#)
5. Stevens, B.; Bony, S. What Are Climate Models Missing? *Science* **2013**, *340*, 1053–1054. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Griffies, S.M.; Winton, M.; Anderson, W.G.; Benson, R.; Delworth, T.L.; Dufour, C.O.; Dunne, J.P.; Goddard, P.; Morrison, A.K.; Rosati, A.; et al. Impacts on Ocean Heat from Transient Mesoscale Eddies in a Hierarchy of Climate Models. *J. Clim.* **2015**, *28*, 952–977. [\[CrossRef\]](#)
7. Alcala, J.; Timofeyev, I. Subgrid-scale Parametrization of Unresolved Scales in Forced Burgers Equation Using Generative Adversarial Networks (GAN). *Theor. Comput. Fluid Dyn.* **2021**, *35*, 875–894. [\[CrossRef\]](#)
8. Subel, A.; Chattopadhyay, A.; Guan, Y.; Hassanzadeh, P. Data-Driven Subgrid-Scale Modeling of Forced Burgers Turbulence Using Deep Learning with Generalization to Higher Reynolds Numbers via Transfer Learning. *Phys. Fluids* **2021**, *33*, 031702. [\[CrossRef\]](#)
9. Bolton, T.; Zanna, L. Applications of Deep Learning to Ocean Data Inference and Subgrid Parameterization. *J. Adv. Model. Earth Syst.* **2019**, *11*, 376–399. [\[CrossRef\]](#)
10. Zanna, L.; Bolton, T. Data-Driven Equation Discovery of Ocean Mesoscale Closures. *Geophys. Res. Lett.* **2020**, *47*, e2020GL088376. [\[CrossRef\]](#)
11. Guillaumin, A.P.; Zanna, L. Stochastic-Deep Learning Parameterization of Ocean Momentum Forcing. *J. Adv. Model. Earth Syst.* **2021**, *13*, e2021MS002534. [\[CrossRef\]](#)

12. Rasp, S.; Pritchard, M.S.; Gentine, P. Deep Learning to Represent Subgrid Processes in Climate Models. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 9684–9689. [\[CrossRef\]](#)
13. Brenowitz, N.D.; Bretherton, C.S. Spatially Extended Tests of a Neural Network Parametrization Trained by Coarse-Graining. *J. Adv. Model. Earth Syst.* **2019**, *11*, 2728–2744. [\[CrossRef\]](#)
14. Krasnopolsky, V.M.; Fox-Rabinovitz, M.S.; Belochitski, A.A. Using Ensemble of Neural Networks to Learn Stochastic Convection Parameterizations for Climate and Numerical Weather Prediction Models from Data Simulated by a Cloud Resolving Model. *Adv. Artif. Neural Syst.* **2013**, *2013*, 485913. [\[CrossRef\]](#)
15. Yuval, J.; O’Gorman, P.A.; Hill, C.N. Use of Neural Networks for Stable, Accurate and Physically Consistent Parameterization of Subgrid Atmospheric Processes with Good Performance at Reduced Precision. *Geophys. Res. Lett.* **2021**, *48*, e2020GL091363. [\[CrossRef\]](#)
16. Akbar, Z.; Murtaza, N.; Pasha, G.A.; Iqbal, S.; Ghumman, A.R.; Abbas, F.M. Predicting scour depth in a meandering channel with spur dike: A comparative analysis of machine learning techniques. *Phys. Fluids* **2025**, *37*, 045158. [\[CrossRef\]](#)
17. Cybenko, G. Approximation by Superpositions of a Sigmoidal Function. *Math. Control. Signals Syst.* **1989**, *2*, 303–314. [\[CrossRef\]](#)
18. Hornik, K.; Stinchcombe, M.; White, H. Multilayer Feedforward Networks Are Universal Approximators. *Neural Netw.* **1989**, *2*, 359–366. [\[CrossRef\]](#)
19. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [\[CrossRef\]](#)
20. Cuomo, S.; Di Cola, V.S.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific machine learning through physics-informed neural networks: Where we are and what’s next. *J. Sci. Comput.* **2022**, *92*, 88. [\[CrossRef\]](#)
21. Cai, S.; Mao, Z.; Wang, Z.; Yin, M.; Karniadakis, G.E. Physics-informed neural networks (PINNs) for fluid mechanics: A review. *Acta Mech. Sin.* **2021**, *37*, 1727–1738. [\[CrossRef\]](#)
22. Wang, S.; Teng, Y.; Perdikaris, P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM J. Sci. Comput.* **2021**, *43*, A3055–A3081. [\[CrossRef\]](#)
23. Jin, X.; Cai, S.; Li, H.; Karniadakis, G.E. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *J. Comput. Phys.* **2021**, *426*, 109951. [\[CrossRef\]](#)
24. Hao, T.T.; Yan, W.J.; Chen, J.B.; Yuen, K.V. Variational Bayesian Data Assimilation with Time-Varying Multi-Physics-Informed Neural Network for Solving Dimension-Reduced Probability Density Evolution Equation. *Reliab. Eng. Syst. Saf.* **2026**, *271*, 112216. [\[CrossRef\]](#)
25. Hu, A.V.; Kabala, Z.J. Predicting and reconstructing aerosol–cloud–precipitation interactions with physics-informed neural networks. *Atmosphere* **2023**, *14*, 1798. [\[CrossRef\]](#)
26. Thawon, I.; Achariyaviriya, W.; Katongtung, T.; Wanison, R.; Syahputra, W.N.H.; Mona, Y.; Tippayawong, N.; Suttakul, P. Physics-informed neural networks in the energy sector: Progress, trends, and future directions. *Energy Rep.* **2026**, *15*, 109013. [\[CrossRef\]](#)
27. Vanhoucke, V.; Senior, A.; Mao, M.Z. Improving the Speed of Neural Networks on CPUs. In Proceedings of the Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011, Lake Tahoe, NV, USA, 12–17 December 2011.
28. Basir, S.; Senocak, I. Critical investigation of failure modes in physics-informed neural networks. In Proceedings of the AiAA SCITECH 2022 Forum, San Diego, CA, USA, 3–7 January 2022; p. 2353.
29. Rathore, P.; Lei, W.; Frangella, Z.; Lu, L.; Udell, M. Challenges in training pinns: A loss landscape perspective. *arXiv* **2024**, arXiv:2402.01868. [\[CrossRef\]](#)
30. Krishnapriyan, A.; Gholami, A.; Zhe, S.; Kirby, R.; Mahoney, M.W. Characterizing possible failure modes in physics-informed neural networks. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 26548–26560.
31. Xu, Z.; Wang, M.; Wang, Z. Weak TransNet: A Petrov-Galerkin based neural network method for solving elliptic PDEs. *arXiv* **2025**, arXiv:2506.14812.
32. Liu, Z.; Cai, W.; Xu, Z.Q.J. Multi-scale deep neural network (MscaledDNN) for solving Poisson-Boltzmann equation in complex domains. *arXiv* **2020**, arXiv:2007.11207. [\[CrossRef\]](#)
33. Germano, M.; Piomelli, U.; Moin, P.; Cabot, W.H. A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids A Fluid Dyn.* **1991**, *3*, 1760–1765. [\[CrossRef\]](#)
34. Lilly, D.K. A proposed modification of the Germano subgrid-scale closure method. *Phys. Fluids A Fluid Dyn.* **1992**, *4*, 633–635. [\[CrossRef\]](#)
35. Hoque, M.A.; Mallik, M.S.I.; Hossain, M.S.; Gope, R.C.; Uddin, M.A. Large eddy simulation of a turbulent channel flow using dynamic Smagorinsky subgrid scale model and differential equation wall model. *Int. J. Thermofluids* **2024**, *22*, 100676. [\[CrossRef\]](#)
36. Kuzmin, D. Monolithic Convex Limiting for Continuous Finite Element Discretizations of Hyperbolic Conservation Laws. *Comput. Methods Appl. Mech. Eng.* **2020**, *361*, 112804. [\[CrossRef\]](#)
37. Hajduk, H. Algebraically Constrained Finite Element Methods for Hyperbolic Problems with Applications in Geophysics and Gas Dynamics. Ph.D. Thesis, Technische Universität, Dortmund, Germany, 2022.

38. de Saint-Venant, A.J.C.B. Théorie du mouvement non permanent des eaux, avec application aux crues des rivières et à l'introduction de marées dans leurs lits. *Comptes Rendus De L'Académie Des Sci.* **1871**, *73*, 154–174.
39. LeVeque, R.J. *Finite Volume Methods for Hyperbolic Problems*; Cambridge University Press: Cambridge, UK, 2002; Volume 31.
40. Hajduk, H.; Kuzmin, D. Bound-preserving and entropy-stable algebraic flux correction schemes for the shallow water equations with topography. *arXiv* **2022**, arXiv:2207.07261.
41. Shu, C.W.; Osher, S. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J. Comput. Phys.* **1988**, *77*, 439–471. [\[CrossRef\]](#)
42. Gottlieb, S.; Shu, C.W.; Tadmor, E. Strong stability-preserving high-order time discretization methods. *SIAM Rev.* **2001**, *43*, 89–112. [\[CrossRef\]](#)
43. Gottlieb, S.; Ketcheson, D.; Shu, C.W. *Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations*; World Scientific: Singapore, 2011.
44. Mojamder, M.A.H. Machine Learning Parametrization of Sub-Grid Fluxes in 1D Shallow Water Equations, in Preparation. Ph.D. Thesis, University of Houston, Houston, TX, USA, 2026.
45. Dolaptchiev, S.I.; Timofeyev, I.; Achatz, U. Subgrid-scale closure for the inviscid Burgers-Hopf equation. *Comm. Math. Sci.* **2013**, *11*, 757–777.
46. Dolaptchiev, S.I.; Achatz, U.; Timofeyev, I. Stochastic closure for local averages in the finite-difference discretization of the forced Burgers equation. *Theor. Comput. Fluid Dyn.* **2013**, *27*, 297–317. [\[CrossRef\]](#)
47. Zacharuk, M.; Dolaptchiev, S.I.; Achatz, U.; Timofeyev, I. Stochastic subgrid-scale parametrization for one-dimensional shallow-water dynamics using stochastic mode reduction. *Q. J. R. Meteorol. Soc.* **2018**, *144*, 1975–1990. [\[CrossRef\]](#)
48. Hendrycks, D.; Gimpel, K. Gaussian Error Linear Units (GELUs). *arXiv* **2023**, arXiv:1606.08415v5.
49. Lee, M. Mathematical Analysis and Performance Evaluation of the GELU Activation Function in Deep Learning. *J. Math.* **2023**, *2023*, 4229924. [\[CrossRef\]](#)
50. Zhang, Y.T.; Shu, C.W. ENO and WENO schemes. In *Handbook of Numerical Analysis*; Elsevier: Amsterdam, The Netherlands, 2016; Volume 17, pp. 103–122.
51. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
52. Guermond, J.L.; Popov, B. Invariant domains and first-order continuous finite element approximation for hyperbolic systems. *SIAM J. Numer. Anal.* **2016**, *54*, 2466–2489. [\[CrossRef\]](#)
53. Harten, A.; Lax, P.D.; van Leer, B. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Rev.* **1983**, *25*, 35–61. [\[CrossRef\]](#)
54. Bermudez, A.; Vazquez, M.E. Upwind methods for hyperbolic conservation laws with source terms. *Comput. Fluids* **1994**, *23*, 1049–1071. [\[CrossRef\]](#)
55. Michel-Dansac, V.; Berthon, C.; Clain, S.; Foucher, F. A well-balanced scheme for the shallow-water equations with topography or Manning friction. *J. Comput. Phys.* **2017**, *335*, 115–154. [\[CrossRef\]](#)
56. Li, M.; Guyenne, P.; Li, F.; Xu, L. Positivity-Preserving Well-Balanced Central Discontinuous Galerkin Method for the Nonlinear Shallow Water Equations. *J. Sci. Comput.* **2017**, *71*, 994–1034. [\[CrossRef\]](#)
57. Eyring, V.; Bony, S.; Meehl, G.A.; Senior, C.A.; Stevens, B.; Stouffer, R.J.; Taylor, K.E. Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization. *Geosci. Model Dev.* **2016**, *9*, 1937–1958. [\[CrossRef\]](#)
58. Haarsma, R.J.; Roberts, M.J.; Vidale, P.L.; Senior, C.A.; Bellucci, A.; Bao, Q.; Chang, P.; Corti, S.; Fučkar, N.S.; Guemas, V.; et al. High Resolution Model Intercomparison Project (HighResMIP v1.0) for CMIP6. *Geosci. Model Dev.* **2016**, *9*, 4185–4208. [\[CrossRef\]](#)
59. Roberts, M.J.; Reed, K.A.; Bao, Q.; Barsugli, J.J.; Camargo, S.J.; Caron, L.P.; Chang, P.; Chen, C.T.; Christensen, H.M.; Danabasoglu, G.; et al. High-Resolution Model Intercomparison Project phase 2 (HighResMIP2) towards CMIP7. *Geosci. Model Dev.* **2025**, *18*, 1307–1332. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.