Water Resources Research

METHOD

10.1029/2024WR037555

Special Collection:
Advancing Interpretable AI/ML Methods for Deeper Insights and Mechanistic Understanding in Earth Sciences: Beyond Predictive Capabilities

Bridging Hydrological Ensemble Simulation and Learning Using Deep Neural Operators

Alexander Y. Sun[1], Peishi Jiang[2], Pin Shuai[3], and Xingyuan Chen[2]

[1]Bureau of Economic Geology, Jackson School of Geosciences, The University of Texas at Austin, Austin, TX, USA, [2]Pacific Northwest National Lab, Richland, WA, USA, [3]Civil and Environmental Engineering, Utah Water Research Laboratory, Utah State University, Logan, UT, USA

Key Points:

- A neural operator approach based on DeepONet is proposed to leverage existing large hydrological ensembles for deep learning
- DeepONet models are used to learn mappings between parameter space and model outputs, and between forcing and streamflow observations
- Demonstration over a real case study suggests that our DeepONet workflow is a promising tool for bridging common hydrological modeling tasks

Supporting Information:

Supporting Information may be found in the online version of this article.

Correspondence to:

A. Y. Sun and X. Chen,
alexsunda@gmail.com;
Xingyuan.Chen@pnnl.gov

Citation:

Sun, A. Y., Jiang, P., Shuai, P., & Chen, X. (2024). Bridging hydrological ensemble simulation and learning using deep neural operators. *Water Resources Research*, *60*, e2024WR037555. https://doi.org/10.1029/2024WR037555

Received 19 MAR 2024
Accepted 21 SEP 2024

Author Contributions:

Conceptualization: Alexander Y. Sun, Peishi Jiang, Pin Shuai, Xingyuan Chen
Data curation: Peishi Jiang, Pin Shuai
Formal analysis: Alexander Y. Sun, Peishi Jiang
Funding acquisition: Xingyuan Chen

Abstract Ensemble-based simulation and learning (ESnL) has long been used in hydrology for parameter inference, but computational demands of process-based ESnL can be quite high. To address this issue, we propose a deep neural operator learning approach. Neural operators are generic machine learning algorithms that can learn functional mappings between infinite-dimensional spaces, providing a highly flexible tool for scientific machine learning. Our approach is built upon DeepONet, a specific deep neural operator, and is designed to address several common problems in hydrology, namely, model parameter estimation, prediction at ungaged locations, and uncertainty quantification. Here we demonstrate the effectiveness of our DeepONet-based workflow using an existing large model ensemble created for an eastern U.S. watershed that is instrumented with 10 streamflow gages. Results suggest DeepONet achieves high efficiency in learning an ML surrogate model from the model ensemble, with the modified Kling-Gupta Efficiency exceeding 0.9 on holdout test sets. Parameter inference, carried out using the trained DeepONet surrogate model and genetic algorithm, also yields robust results. Additionally, we formulate and train a separate DeepONet model for physics-informed, seq-to-seq streamflow forecasting, which further reduces biases in the pre-trained DeepONet surrogate model. While this study focuses primarily on a single watershed, our approach is general and may be extended to enable learning from model ensembles across multiple basins or models. Thus, this research represents a significant contribution to the application of hybrid machine learning in hydrology.

Plain Language Summary Many common tasks in hydrology, such as data assimilation, parameter estimation, uncertainty quantification, require running a process-based model over many realizations. However, the computational burden of running models, especially when distributed models are involved, can be prohibitive. Leveraging existing ensembles for similar tasks presents a desirable solution to mitigate this challenge. In this study, we propose and demonstrate an integrated deep learning approach to seamlessly connect common forward and inverse modeling tasks in ensemble-based hydrological learning. Our methods utilize deep neural operator learning to effectively approximate functional mappings between model input and output spaces. The proposed neural operator based workflow comprises three key stages that are model-agnostic: learning a surrogate model from an existing ensemble, using the learned surrogate model for parameter estimation, and incorporating additional meteorological forcing data to further reduce model biases. The efficacy of our deep learning framework is demonstrated through a real case study.

1. Introduction

Ensemble-based simulation and learning (ESnL) may be loosely defined as methods and algorithms for sampling and learning from a system's parameter and state space to support the needs of downstream applications, such as surrogate modeling, optimization, uncertainty quantification (UQ), and machine learning (ML). ESnL has broad applications in Earth system modeling, serving as an essential tool in climate projection, weather and hydrological forecasting, and groundwater modeling (Castelletti et al., 2023; Deser et al., 2020; Dong et al., 2020; Giuliani et al., 2021; Houtekamer et al., 1996). Several notable categories of ESnL applications may be identified in the literature.

Ensemble-based data assimilation was introduced in early 2000s to improve the accuracy of model forecasts by sequentially updating model states using new observations (Anderson, 2001; Evensen, 2003; Whitaker & Hamill, 2002). Around the same time, ensemble Kalman filters and smoothers were developed to provide joint estimates of model states and parameters (Chen et al., 2013; Liu & Gupta, 2007; Moradkhani et al., 2005; Vrugt &

Project administration: Alexander Y. Sun, Xingyuan Chen
Supervision: Xingyuan Chen
Validation: Alexander Y. Sun, Peishi Jiang
Writing – review & editing: Alexander Y. Sun, Peishi Jiang, Pin Shuai, Xingyuan Chen

Robinson, 2007). In these applications, ensembles are formed by randomly perturbing the initial/boundary conditions and/or by sampling the prior distribution of model parameters.

Ensemble-based learning is a well-established technique that combines multiple models to improve the overall prediction accuracy. Since 1990s, a wide array of traditional ensemble learning methods have been introduced, such as bootstrap aggregation (bagging), boosting, and stacking (Dong et al., 2020; Sagi & Rokach, 2018). In these applications, the ensembles are created by using a group of the so-called base learners, formed by training with different ML algorithms or by training the same ML model through random initialization of weights. Ensembles may also be formed using different conceptual models or models calibrated with different objective functions. In hydrometeorology, for example, multi-model ensembles are commonly used in conjunction with ensemble learning methods (e.g., Bayesian model averaging) to improve forecast skills (Duan et al., 2007; Tebaldi & Knutti, 2007; Zounemat-Kermani et al., 2021).

In recent years, the rapid evolution of deep learning algorithms and generative artificial intelligence (genAI) has ushered in a new paradigm for ESnL. Specifically, deep learning architectures have demonstrated versatile skills in learning cross-domain function mappings, making the traditionally separate tasks under ESnL (e.g., forward modeling, inversion, UQ, and optimization) increasingly connected to each other (Willard et al., 2022; Yang et al., 2023). Deep learning models typically require a large amount of data to train. To deal with this issue, a large number of physics-informed ML methods have been introduced (Karniadakis et al., 2021). Physics-based simulation plays two highly related roles in ESnL, namely, generating synthetic training samples (labels), and infusing physics principles in the learning process to constrain outputs. Psaros et al. (2023) classified the use cases in physics-informed ML into four categories: (a) forward modeling of model states with a deterministic model structure and input parameters, but variable boundary/initial conditions; (b) joint estimation of model states and uncertain parameters with a deterministic model structure; (c) joint estimation of model states and uncertain parameters under unknown model structures; and (d) neural operator learning. While the first three categories are well known to the hydrological community and have long been studied under the classic inversion and UQ theoretic frameworks (Clark et al., 2008; Gupta et al., 1998; Liu & Gupta, 2007; N.-Z. Sun & Sun, 2015; Vrugt & Robinson, 2007), neural operator learning is relatively new, providing a powerful approach for bridging several common tasks in hydrological ESnL.

Simply speaking, neural operator learning aims to approximate a mapping between two function spaces (i.e., spaces in which points are functions) by using a neural network (Kovachki et al., 2023). Function spaces are infinite dimensional. A standard approach to learning functional mappings is first discretizing the function spaces and then learning mappings on a finite-dimensional grid. Such methods are generally discretization dependent and do not generalize well to new meshes resulting from, for example, change of grid resolutions and/or boundary conditions (Kovachki et al., 2023). In contrast, neural operators are designed to learn mappings between infinite-dimensional function spaces directly and, thus, are discretization invariant. At this time, two major neural operators originating from the scientific ML literature are Deep Operator Net (DeepONet) (Lu et al., 2021) and Fourier Neural Operator (FNO) (Li et al., 2020). Both DeepONet and FNO were initially proposed for learning solution operators of partial differential equations (PDEs). The key difference between the two lies in the parameterization of operators. FNO uses Fourier transform to represent the operators in the frequency domain, and then learns a mapping from the input space to the Fourier coefficient space (Li et al., 2020). FNO assumes uniform grids. On the other hand, DeepONet uses neural networks to directly learn the operators, thus can be applied to general problem settings without relying on specific structures imposed by Fourier transforms (Lu et al., 2021). So far, applications of deep neural operators are relatively limited in hydrology, with most of the existing studies focusing on climate and subsurface modeling (Jiang, Yang, et al., 2023; Pathak et al., 2022; A. Y. Sun et al., 2023; Taccari et al., 2024; Wen et al., 2022; Yan et al., 2022).

In this study, we sought to integrate DeepONet with ESnL to address some common problems in hydrology, namely, parameter estimation, prediction at ungaged locations, and UQ. The main contribution of this work lies in the development of a neural-operator-based, model-agnostic approach for bridging forward simulation, inverse mapping, UQ, as well as time series forecasting in hydrology. We demonstrated our DeepONet-based workflow over the upper Neversink River watershed, an eastern U.S. catchment that has been extensively studied over time by different U.S. water agencies (Baldigo & Lawrence, 2000; Burns et al., 2008; Harpold et al., 2010). In the following, we first introduce the study site, then describe the ensemble simulation and formulation of DeepONet methodologies, followed by results, discussion, and conclusions.

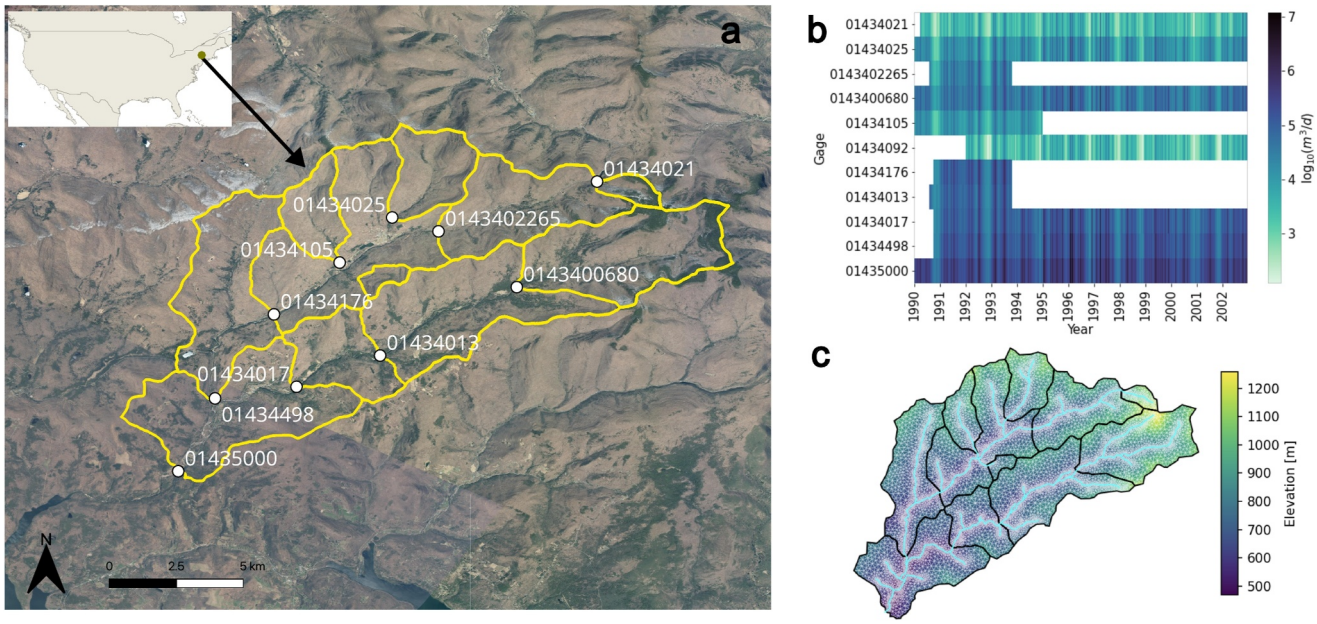

Figure 1. (a) The upper Neversink River watershed and locations of 10 USGS streamflow gages used in this study, (b) Length and magnitude (log scale) of the streamflow data records available at each of the 10 gages, and (c) Unstructured mesh used in the Advanced Terrestrial Simulator (ATS), where the dark solid lines correspond to subbasin boundaries.

2. Data

2.1. Study Site

The upper Neversink River Watershed (UNRW) is a headwater basin located in the Catskill Mountains of southeastern New York, draining an area of 173 km^2 (Burns et al., 2008). It comprises two main river branches, the East and West Branch of the Neversink River, which join upstream of the Neversink reservoir at the southwest corner of the watershed (Figure 1a). The watershed has steep hillslopes, with elevations ranging from 1,274 m at its headwaters in the summit of Slide Mountain to 408 m at the basin outlet near the Neversink reservoir. The area is underlain by flat-lying sedimentary bedrock comprising primarily of coarse sandstone and conglomerate with interbedded shale and siltstone; surficial geological features include till, kame terraces, outwash sand and gravel, alluvium, and exposed rock; soils of the region range from 0.1 to 1.5 m thick and are well drained (Burns et al., 2008; Harpold et al., 2010).

Located in the headwaters region of the Delaware River Basin, UNRW is typical of watersheds in northeastern U. S., having a warm-summer, humid continental climate. Vegetation mainly comprises of northern hardwood forest. Annual precipitation at the UNRW ranges from about 1,750 mm at its headwaters to 1,450 mm at the basin outlet, based on the 30-year data from Parameter-elevation Relationships on Independent Slopes Model (PRISM) (Daly et al., 1997). Peak flows, driven by snowmelt runoff, typically occur in late spring and early summer (April–June). Historical streamflow data used in this study were taken from 10 U.S. Geological Survey (USGS) gages instrumented at UNRW, with varying record lengths during our study period (Figure 1b).

2.2. Ensemble Simulation

A distributed hydrological model was developed for UNRW using the Advanced Terrestrial Simulator (ATS, v1.4), an open-source, coupled surface-subsurface simulator for modeling flow and reactive transport on unstructured meshes (Coon et al., 2019; Jiang et al., 2024). ATS solves a 2D diffusion wave approximation of the Saint-Venant equation for overland flow, while for subsurface flow it solves a 3D Richards equation for variably saturated porous media. The surface and subsurface domains are coupled by imposing pressure and flux continuity (Coon et al., 2020).

An unstructured mesh was created for UNRW using a Python-based Watershed Workflow tool developed specifically for process-based physical models (Coon & Shuai, 2022). The surface mesh consists of 18,555 triangular cells, with cell sizes varying gradually from ~100 m near the rivers to ~320 m away from the rivers (Figure 1c). The subsurface domain was discretized vertically into 20 terrain-following layers, with soil layers at the top and a fractured bedrock layer at the bottom, leading to a total of 389,655 cells in the UNRW model mesh. To help create the mesh and fill in cell properties, the Watershed Workflow tool automatically extracted properties from the public-domain data sources, including land cover types from the National Land Cover Database (Homer et al., 2012), and soil types and properties from the National Resources Conservation Service's (NRCS) Soil Survey Geographic Database (SSURGO) (NRCS, 2024). Permeabilities of the fractured bedrock and five soil types were calibrated as part of the inversion process (see below).

ATS implements the Priestley-Taylor equation (Priestley & Taylor, 1972) to model multiple evapotranspiration (ET) processes, including snow evaporation, canopy evaporation and transpiration, and land surface evaporation. The Priestley-Taylor equation is a simplified version of the Penman equation (Allen et al., 2006) that requires less measurements,

$$\lambda E_a = \alpha \frac{\Delta}{\Delta + \gamma}(R_n - G), \tag{1}$$

where E_a is actual ET, λ is latent heat of vapourization, α is a dryness coefficient, R_n is net radiation at the surface, G is sensible heat flux into the ground, Δ is slope of the saturation-vapor-pressure curve, and γ is the psychometric constant (Flint & Childs, 1991). Priestley-Taylor equation is applicable when ET is limited by soil water supply; the Priestley-Taylor coefficient (α) controls the portion of net radiation that is available to drive the ET process and is determined experimentally to be 1.26 for freely evaporating surfaces (Priestley & Taylor, 1972). For specific sites, α needs to be calibrated to reflect surface vegetation and local meteorological conditions.

We used the meteorological forcing data, including daily minimum and maximum temperature, precipitation, vapor pressure, and shortwave radiation, from Daymet (Thornton et al., 2024), which is provided at 1-km resolution over the continental North America from 1980 to the present. Note Daymet's snow water equivalent (SWE) data was not used in this study because SWE is simulated by ATS. In a recent study, Shuai et al. (2022) compared several meteorological forcing data sets, including Daymet, PRISM, and the North American Land Data Assimilation System (NLDAS) (Xia et al., 2012), for driving ATS simulations in a snow-dominated watershed in the Upper Colorado River Basin. Their findings indicated that Daymet gave the best performance for streamflow simulation among the data sets considered.

The UNRW model was spun up by first running the model for 1,000 years with a constant rainfall to reach the steady state, and then with cyclic yearly averaged Daymet forcing for 10 years. The model state from the second spinup was used to initialize the ensemble simulations. A total of 13 model parameters were selected for inversion based on sensitivity analyses (Jiang et al., 2024), following an information theoretic framework documented in Jiang, Shuai, et al. (2023). These selected parameters are related to various water and energy processes, including (a) snowmelt rate (sm-rate) and the temperature determining the onset of snowmelt (sm-diff); (b) Priestley-Taylor coefficients (α) related to canopy evaporation (PT-canopy), ground evaporation (PT-ground), snow evaporation (PT-snow), and transpiration (PT-trans); (c) Manning's coefficient (manning_n) affecting the surface runoff; and (d) fractured bedrock permeability (perm-bedrock) and five permeability values related to different soil types (perm-S1, perm-S2, perm-S3, perm-S4, and perm-S5) that affect subsurface runoff. A full list of parameters is provided in Table A2 of Appendix A. For ensemble generation, all parameters are assumed to be spatially uniform, leading to a 13-dimensional parameter space. Realizations of the parameters were generated using the quasi-random Sobol' sequence, which provides more uniform sampling of high-dimensional spaces than random sequences do (Sobol', 1967). A total of 600 parameter realizations were generated and simulated using ATS, out of which 596 runs completed for the study period 1991–2000, forming the ensemble used in our DeepONet demonstration. The ATS model inputs (i.e., Daymet forcing) and outputs were extracted at each gage location from each ensemble member.

It took about 20 hr to complete each model run using 128 cores on the Perlmutter supercomputer that is available through the U.S. National Energy Research Scientific Computing Center (NERSC, 2024). All ensemble runs were

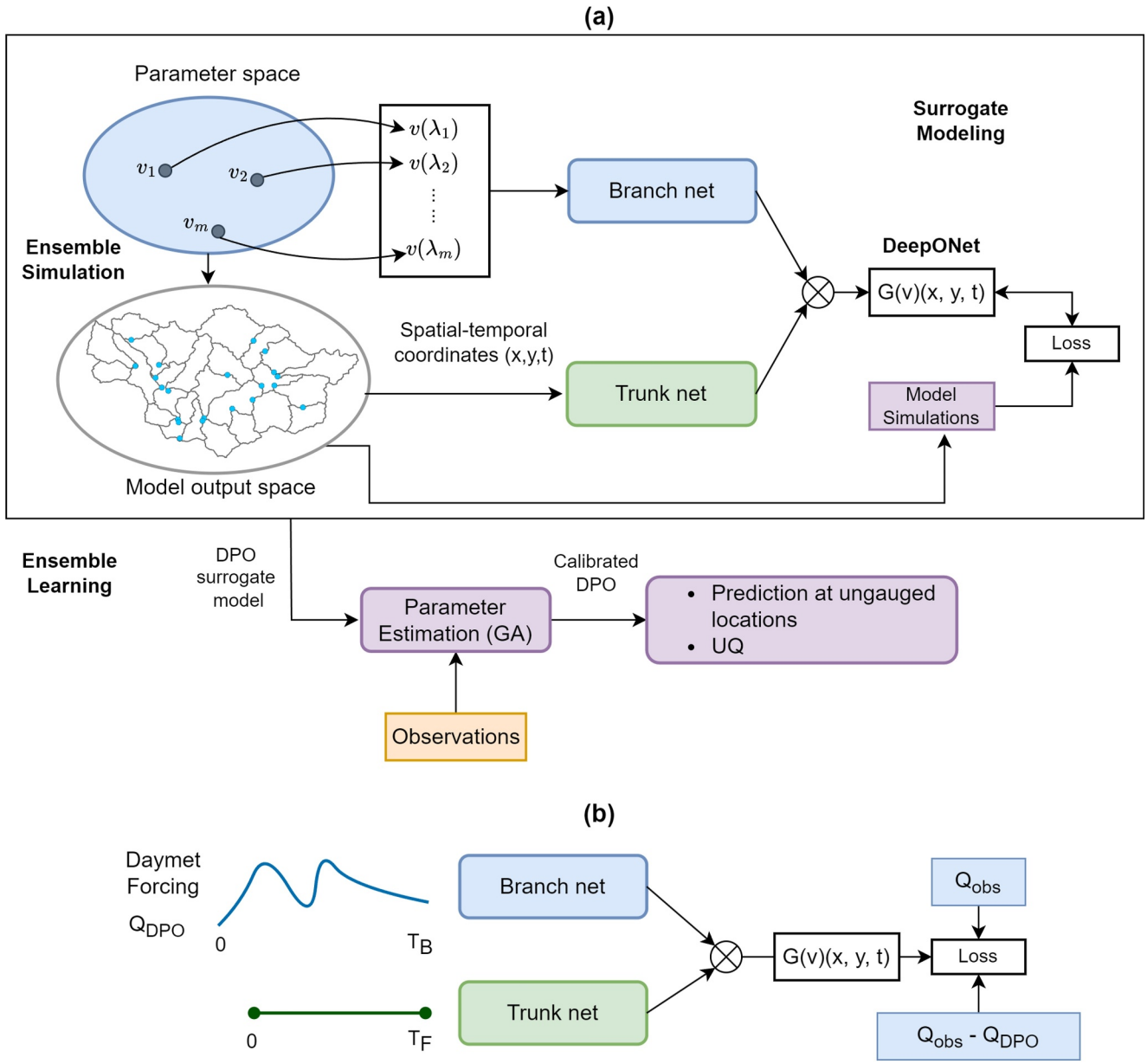

Figure 2. A hydrological ensemble simulation and learning (ESnL) workflow facilitated using DeepONet: (a) model architecture and workflow used in surrogate modeling and parameter estimation, where the inputs/outputs of each stage are labeled; (b) model architecture used in sequential learning. The subscript DPO stands for DeepONet, and other symbols are defined in the text.

carried out in parallel. The 1000-year spinup run typically completed within 30 min. More details on the ensemble creation process can be found in Jiang et al. (2024).

3. Methodology

3.1. Overview of the Workflow

An overview of our DeepONet-based ESnL workflow is provided in Figure 2a. The workflow consists of three main stages. During the forward modeling stage, an ATS ensemble is generated by randomly sampling the parameter space. A DeepONet model is then trained to learn an operator mapping from the parameter space to model output space, using only the ATS ensemble as training/test data. Once trained, the surrogate model can be

used to query any point in the parameter space. In the second stage, the pre-trained surrogate model is used to identify the optimal model parameters by minimizing the difference between DeepONet model outputs and gage observations. Although this stage resembles the traditional hydrological model calibration process, an important distinction lies in DeepONet's capability to learn functional mappings. Finally, the calibrated DeepONet surrogate model is used for downstream applications, such as prediction and UQ. Toward that end, we introduce a new DeepONet architecture designed for time series forecasting (Figure 2b).

The ensemble generation process has already been described under Section 2.2. We now focus on the remaining steps of the workflow in the following subsections.

3.2. DeepONet for Inverse Mapping

DeepONet is introduced to learn solution operators of nonlinear dynamical systems. Specifically, let $\mathcal{G}$ be an operator that maps from the input space $\mathcal{V}$ to the output space $\mathcal{U}$ of a system (Lu et al., 2021):

$$\mathcal{G}: \mathcal{V} \ni v \to u \in \mathcal{U}, \tag{2}$$

where $v = v(\lambda)$ and $u = u(\xi)$ are functions in the input and output space, respectively, and λ and ξ are coordinates. The goal of DeepONet is to learn an approximation of $\mathcal{G}(v)$ using a paired data set consisting of values of the input function $v(\lambda)$ sampled at m locations, $[v(\lambda_1), v(\lambda_2), \ldots, v(\lambda_m)]$, and the corresponding outputs, $[u_1(\xi), u_2(\xi), \ldots, u_m(\xi)]$. To this end, DeepONet makes use of two neural network models. The "branch" net takes samples from the input space as inputs, while the "trunk" net takes coordinates of the output space as inputs. Branch and trunk nets may be implemented using any type of artificial neural networks, such as the multilayer perceptron network (MLP), convolutional neural network (CNN), and residual net (ResNet) (He et al., 2016).

In DeepONet, the operator $\mathcal{G}(v)$ is approximated by a linear combination of nonlinear basis functions by using the branch net outputs as coefficients and trunk net outputs as basis functions (Lu et al., 2022). This is done by taking the inner product of outputs from branch and trunk nets,

$$\mathcal{G}_\theta(v)(\xi) = \sum_{l=1}^{p} b_l(v) tr_l(\xi) + b_0, \tag{3}$$

where $\{b_l\}_{l=1}^p$ and $\{tr_l\}_{l=1}^p$ are outputs of the branch net and trunk net; p is the number of neurons in the output layers; b_0 is an additive bias term added to improve performance; and θ are learnable parameters of the neural networks, which are obtained by minimizing the following loss function between target values and DeepONet outputs,

$$\mathcal{L}(\theta) = \frac{1}{m} \sum_{i=1}^{m} w_i \| u_i(\xi) - \mathcal{G}_\theta(v_i)(\xi) \|_2, \tag{4}$$

where $\{w_i\}_{i=1}^m$ are weights assigned to each input point and $\| \cdot \|_2$ is 2-norm. In this work, all input points were assigned equal weights.

Using the generalized universal approximation theorem for operators, it can be shown that Equation 3 may approximate any continuous nonlinear operator with an arbitrary level of accuracy (Lu et al., 2021). After training, the DeepONet neural operator $\mathcal{G}_\theta(v)(\xi)$ is continuous in the output functional space and, thus, can be used to calculate the output function value at any point ξ. This capability to "query" output values at new points not seen during training distinguishes neural operators from their finite-dimensional counterparts (Kovachki et al., 2023).

By design, DeepONet covers a wide range of applications in physical sciences. The input space may encompass multiple functions, for example, boundary/initial conditions, forcing terms, or parameters, which can be potentially high dimensional. Coordinates of the output space are normally the spatial and temporal coordinates of data points.

For parameter inference, DeepONet is trained to learn a mapping between the parameter space of a distributed hydrological model and the corresponding output space, which can be either simulated or observed streamflows at various locations and times, $Q(x,y,t)$,

$$\mathcal{G}_\theta^P : v \rightarrow Q(x,y,t), \tag{5}$$

where the superscript P denotes the parameter space, and x, y, and t represent the longitude, latitude, and time of either streamflow observations or ATS model outputs in our case. After training, the DeepONet $\mathcal{G}_\theta^P$ is used as a surrogate of the ATS. In the original DeepONet paper (Lu et al., 2021), inputs to the branch net are often referred to as "sensors" to emphasize that they are samples of the model state space. For the inversion mapping problem considered here, it is more intuitive to consider inputs to the branch net as "trial" solutions of the inverse problem.

After training $\mathcal{G}_\theta^P$, the genetic algorithm (GA), an evolutionary global optimization algorithm, is used to search the parameter space for the optimal model parameters. Simply speaking, GA is a heuristic optimization algorithm that simulates the process of natural selection by evolving a set of potential solutions (population) toward an optimal solution (Holland, 1992). During the evolution process, the algorithm iteratively refines the population over generations, favoring individuals with better fitness (selection) and encouraging the exploration of different solution spaces by randomly perturbing (mutation) and combining solutions (crossover) in each new generation.

In this work, we used a Python GA library, DEAP (Fortin et al., 2012), with a generation size of 100 and a population size of 300. The probabilities of crossover and mutation were set to 0.46 and 0.005, respectively. The loss function used is the normalized root mean square error (NRMSE) defined as

$$\texttt{NRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\widehat{Q}_i - Q_{obs,i}\right)^2}}{\left(Q_{obs}^{\max} - Q_{obs}^{\min}\right)}, \tag{6}$$

where $\widehat{Q}_i$ and $Q_{obs,i}$, $i = 1,\ldots,N$, are predicted and observed streamflow values used for calibration, and the denominator defines the range of the observed streamflow. Our experience suggests that using NRMSE instead of the commonly used mean square error (MSE) improves GA calibration. In the following text, we denote the calibrated DeepONet as `DeepONet-P` to distinguish it from the pre-trained surrogate model $\mathcal{G}_\theta^P$.

3.3. DeepONet for Time Series Forecasting

The DeepONet surrogate model described in the previous section may be used to query a spatio-temporal domain for any set of input model parameters. However, it assumes that t varies in a fixed domain $[0,T]$ and thus cannot be applied directly to forecasting, which involves a continuously expanding temporal domain (i.e., an extrapolation problem). To apply the DeepONet in the latter context, we adopted ideas from seq2seq learning (Sutskever et al., 2014) and designed a new DeepONet model, denoted as `DeepONet-F`, to learn the mapping between functions in two consecutive time periods, T_B and T_F, standing for lookback and forecast periods (Figure 2b),

$$\mathcal{G}_\theta^F : v_{T_B} \rightarrow u_{T_F}. \tag{7}$$

We trained `DeepONet-F` to learn the mapping between meteorological forcing and streamflow, simulating the potential use of DeepONet in an operational setting. The other motivation was to investigate whether model biases in a calibrated `DeepONet-P` model can be further reduced through the hybrid ML. Thus, streamflow simulated by `DeepONet-P` is used as a "physics-informed" predictor, as have been done in many physics-informed ML studies (e.g., Humphrey et al., 2016). To this end, the input-output training pairs are prepared in a moving-window fashion. At any time, inputs to the branch net include the Daymet meteorological forcing variables used in ATS, the static features (i.e., spatial coordinates of gages), and outputs from the calibrated ATS model, all corresponding to the antecedent period T_B.

Inputs to the trunk net are temporal coordinate of the forecast period $[0,T_F]$. The corresponding streamflow observations in $[0,T_F]$ are used as training targets. Alternatively and if conditions allow, we can also train a DeepONet to predict the mismatch between observations and ATS model predictions (Figure 2b). For the purpose

of hindcasting, gap-filling, or UQ, which is assumed in our demonstration study below, we use the `DeepONet−P` to generate the simulated streamflow as predictors, so that no additional ATS runs are necessary.

3.4. Implementation

We used Nvidia's open-source deep learning library, Modulus (v22.09), to train $\mathcal{G}_\theta^P$ (Nvidia, 2024). The branch net was configured using MLP, while the trunk net was configured using the Fourier neural network. The latter is a special type of fully connected neural networks that encodes the input features (i.e., latitude, longitude, and temporal coordinates in this case) into the spectral space to help reduce biases during training (Nvidia, 2024). We found the best MLP network structures through a grid search on the main hyperparameters, including the number of hidden layers and the number of hidden neurons in each layer of the branch and trunk nets. In the final configuration, we used four hidden layers, each consisting of 256 hidden neurons, in both the branch net and trunk net. The nonlinear activation function used is Sigmoid Linear Units or SiLU (Apicella et al., 2021). The total number of trainable parameters is 544k. For training, we used the Adam optimizer (Kingma & Ba, 2014) with an exponential-decay learning rate scheduler (Li & Arora, 2019). The initial learning rate was set to 1×10^{-3}, decaying at a rate of 0.85 after every 5,000 steps. The maximum number of training steps used is 80,000. Training generally took about 0.5 hr wallclock time on an Nvidia RTX3090 GPU. Parameter estimation using GA and $\mathcal{G}_\theta^P$ took less than 2 min on the same machine.

The `DeepONet−F` was implemented based on the open-source Python code, `DeepONet−Grid−UQ` (Moya et al., 2023). In this case, both the branch and trunk nets were configured using MLPs, each having three hidden layers, with 256 hidden neurons in each layer. The outputs from both networks have 100 neurons, which are used to calculate the dot product (Equation 3). Different from the original `DeepONet−Grid−UQ`, here we adopted a simple dropout strategy to allow for quantification of uncertainty in the network (Gal & Ghahramani, 2016). This was done by inserting a dropout layer in the MLPs, which randomly zero out the output of some neurons during training based on a user-specified probability. The dropout probability used is 0.25. Training of the `DeepONet−F` took less than 1 minute of wallclock time.

Test metrics used include NRMSE and the modified Kling-Gupta Efficiency (KGE) defined in Appendix C. For seq-to-seq learning, we also used the Continuous Ranked Probability Skill Score (CRPSS), a probabilistic scoring measure, to measure the performance of `DeepONet−F` ensemble forecasts (see Appendix C).

4. Results

4.1. DeepONet Surrogate Modeling

Only the ATS ensemble was used in training and testing the DeepONet surrogate model (Figure 2a). The 596 ATS ensemble members were randomly split into training (80%) and test (20%) sets. A summary of temporal periods used in various cases are summarized in Table 1. Other training details have been provided under Section 3.4.

The trained surrogate model $\mathcal{G}_\theta^P$ achieves a satisfactory performance at most gages, with the median KGE greater than 0.9 and the median NRMSE less than 0.03 on the test ensemble set (Figure 3). In particular, $\mathcal{G}_\theta^P$ does a better job in approximating ATS outputs for gages located on the mainstem of the upper Neversink River. The $\mathcal{G}_\theta^P$ approximation for gages located on smaller river branches is slightly worse, exhibiting a larger spread. In particular, the performance at Gage 01434021, the most remote gage located in the UNRW headwaters region, is the worst among all gages.

The performance of the surrogate model is consistent with previous deep learning surrogate modeling studies, and may be explained from either an information-theoretic perspective (Jiang et al., 2024; Jiang, Shuai, et al., 2023) or network-theoretic perspective (A. Y. Sun et al., 2022). Specifically, it is well known that information flow along river networks is directional, with downstream gages tending to be more informed about basin-wide processes (Alfonso et al., 2013; Yang & Burn, 1994). In this case, the downstream gages, especially the two gages at or downstream from the confluence of East and West Neversink River Branches (i.e., 01434498 and 01435000) reflect the influence of the entire watershed. By design, $\mathcal{G}_\theta^P$ is not given any explicit information of the watershed characteristics. Instead, information is implicitly passed down from the ATS ensemble used for training.

Table 1
Data and Time Splits Used in Training and Testing the DeepONets (Q_{obs}, Observed Streamflow; $Q_{\texttt{DeepONet-P}}$, Streamflow Simulated Using `DeepONet − P`*)*

Task	Period	Data used
$\mathcal{G}_\theta^P$ pre-training and testing	10/01/1991–09/30/2000	ATS ensemble
`DeepONet − P` calibration	10/01/1993–09/30/2000	Q_{obs}
`DeepONet − P` testing	10/01/1991–09/30/1993	Q_{obs}
`DeepONet − F` training	10/01/1993–09/30/2000	Q_{obs}, $Q_{\texttt{DeepONet-P}}$
`DeepONet − F` testing	10/01/1991–09/30/1993	Q_{obs}, $Q_{\texttt{DeepONet-P}}$

The ensemble size should be sufficiently large for the surrogate model to achieve good performance, while the optimal ensemble size depends on the dimensionality of the parameter space. We conducted an ablation study to investigate the effect of ensemble size on $\mathcal{G}_\theta^P$ training, in which we trained $\mathcal{G}_\theta^P$ models using subsets of the original training ensemble, ranging from 100 to 400 members. Results (Figure S1 in Supporting Information S1) show that the surrogate model performance (measured in median KGE) is suboptimal when the ensemble size is less than 200, but stabilizes when the ensemble size is greater than 300. While further improvements may be achieved, the training ensemble size used in our surrogate modeling is reasonable for the problem at hand.

4.2. Parameter Estimation

For model calibration using GA, we chose a training period in which six out of 10 USGS gages have relatively continuous records (see Figure 1b). We designed five different sets of experiments to test the sensitivity of model calibration to the quantity and location of observation data (Table 2). The UNRW basin outlet gage (01435000) was kept in all experiments as the baseline information source. For reference, the KGE values between the median of the ATS ensemble and respective gage observations are also shown (1st row in Table 2).

Results show model calibration is relatively robust with respect to the number of gages used (Table 2). Exp 1, which utilizes all training records from all six calibration gages, is not the best in terms of KGE. Exp 3, using mainly the East Branch gages, gives a better overall performance (best performances on Gage 4, 5, and 8) than Exp 2 that uses only the West Branch gages (best performance on Gage 7 only). This is because two of the three West Branch gages used in Exp 2 are located away from the mainstem, thus carrying less information. On the other hand, all East Branch gages are located on the mainstem. Interestingly, Exp 4, which only uses the two most downstream gages, gives overall better results (achieving the best performance on four gages, and second best on another four gages) than Exp 1 that uses all information. Finally, Exp 5, which uses only the outlet gage, gives results that are comparable to Exp 1. These findings are encouraging, suggesting the robust capability of `DeepONet−P` in interpolating to ungaged locations in a gaged watershed. It also suggests that information from extra gages is location dependent and may not translate directly into better calibration. The UNRW represents a

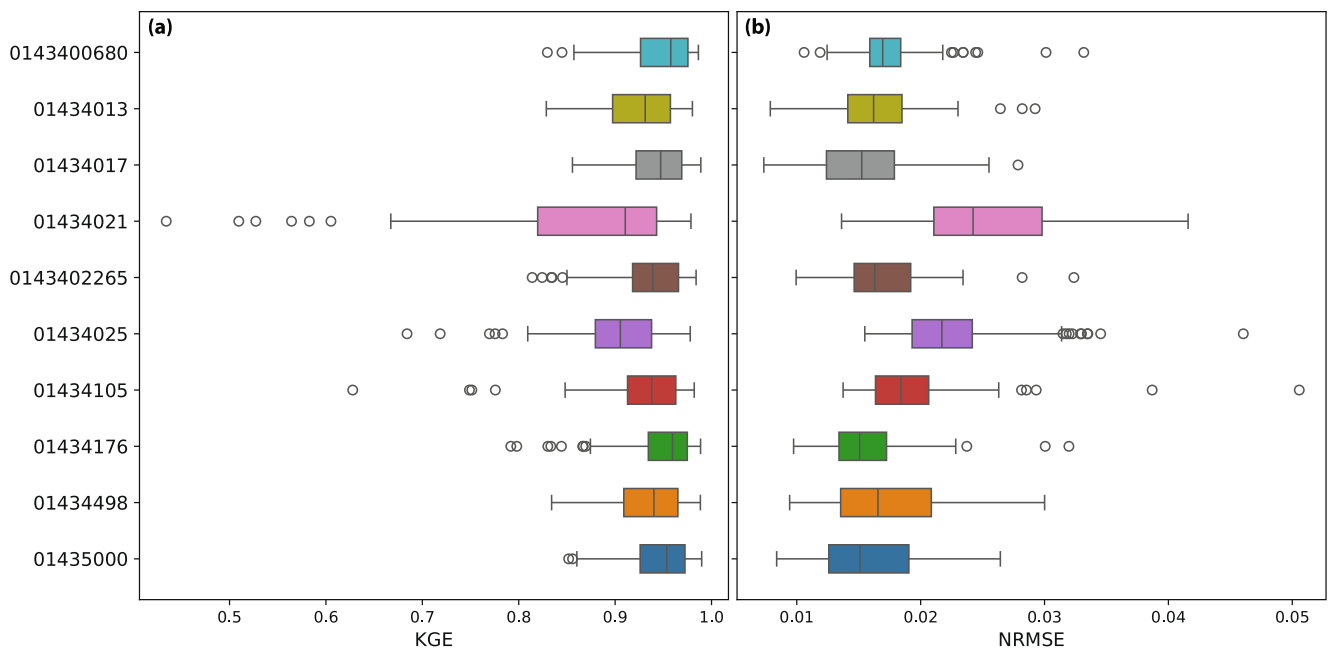

Figure 3. Boxplots of the $\mathcal{G}_\theta^P$ performance metrics on test ensemble sets: (a) modified Kling-Gupta efficiency and (b) normalized root mean square error.

Table 2
Kling-Gupta Efficiency (KGE) Values Obtained for All Stations in the Five Parameter Estimation Experiments Using GA

Exp.	Calib. Gages	1	2	3	4	5	6	7	8	9	10
ATS		−0.185	−0.111	−0.068	−7.856	−3.855	−1.228	−40.236	0.004	0.035	−1.801
1	1, 2, 5, 7, 8, 10	0.750	*0.727*	*0.760*	0.199	0.705	**0.761**	*0.483*	0.792	0.811	**0.788**
2	1, 2, 5, 7	0.740	0.710	0.739	0.166	0.680	0.748	**0.494**	0.760	0.779	0.761
3	1, 8, 10	0.739	0.721	0.756	**0.285**	**0.728**	0.746	0.465	**0.809**	*0.814*	0.773
4	1, 2	**0.753**	**0.732**	**0.765**	*0.256*	*0.726*	0.748	0.454	*0.801*	**0.815**	*0.780*
5	1	*0.751*	*0.727*	*0.760*	0.207	0.715	*0.751*	0.479	0.785	0.804	0.779

Note. Gages are labeled using numbers to save space, 1:01435000; 2:01434498; 3:01434176; 4:01434105; 5:01434025; 6:0143402265; 7:01434021; 8:01434017; 9:01434013; 10:0143400680. Gages used for calibration in each experiment are shown under the column Calib Gages. The best and second best KGE's for each gage are labeled in boldface and italic, respectively.

case of relatively densely instrumented watershed, but the actual number of gages required for a good model calibration may be reduced, as suggested by results presented herein.

Parameter values estimated from the five calibration experiments are compared in Figure 4 for the 13 parameters listed in Table A2. Results suggest that the snowmelt rate (sm-rate), the Priestley-Taylor coefficient α for some land cover types (PT-canopy, PT-ground and PT-snow), and several soil permeabilities (perm-S2, -S4, and -S5) show the most variations across different experiments. On the other hand, Manning's coefficient (manning_n), fractured bedrock log-permeability (perm_bedrock), and perm-S1 show limited variations. In a parallel work, it is shown through information theoretic analysis that the fractured bedrock permeability (perm_bedrock) and top soil permeability (perm-S1) are the two most sensitive and identifiable parameters of all the parameters (Jiang et al., 2024), confirming our observations.

The degree of variations reflects the sensitivity of streamflow to different physical processes and mechanisms modeled within ATS. Calibration results also depend on the parameter structure assumed for the UNRW, namely, uniform parameter values for the entire watershed. For example, Manning's coefficient is more identifiable using the streamflow data. The Manning's n value identified here ($\sim$0.06) is representative of floodplains with light brush and trees (Chow et al., 1988). On the other hand, the Priestley-Taylor α and log-permeability values for deeper soil layers are less identifiable using streamflow data alone. Additional types of data are needed to help constrain the estimated parameter values.

We performed additional sensitivity analyses to assess the effect of parameter uncertainty using `DeepONet−P` as a baseline. Specifically, all parameter values were varied between 20% and 50% from their calibrated values and hydrographs at each gage, and then simulated using the surrogate model $\mathcal{G}_\theta^P$. Example hydrographs are shown for four selected UNRW gages (two most downstream gages 0143500 and 01434498, and two upstream gages 0143402265 and 0143400680) over the test period (Figure 5). Results suggest the calibrated model is relatively robust to parameter variations in this case. The `DeepONet−P` model captures the daily streamflow variations for most years, but it also misses some high- and low-flow events, for example, the low-flow period in 07/1993–10/1993 and peak flows around 04/1992. The latter observation is further confirmed on the scatter plots of `DeepONet−P` outputs versus gage data (Figure S3 in Supporting Information S1). The Pearson's correlation coefficient ranges from 0.77 to 0.80 for the four gages, but low flows are generally more scattered. Biases in the `DeepONet−P` model were further corrected through the DeepONet seq2seq learning as shown below.

4.3. Seq2seq Learning

Without loss of generality, we took model outputs from the same `DeepONet−P` model that was trained using the setup of Exp 3 (Table 2). As mentioned previously in Section 3.3, two schemes of training were tested, one learning the outflow directly and the other aiming to learn the mismatch between surrogate model predictions and observations. For each training scheme, a total of 10 `DeepONet−F` models were trained with the dropout strategy to account for model uncertainty. In all experiments, the look back period (T_B) was set to 90 days and the forecast period (T_F) was set to 3 days. Although longer look back periods might be used, our exploratory studies suggest that the 90-day period is sufficient to cover the variations needed for daily forecasts in this case.

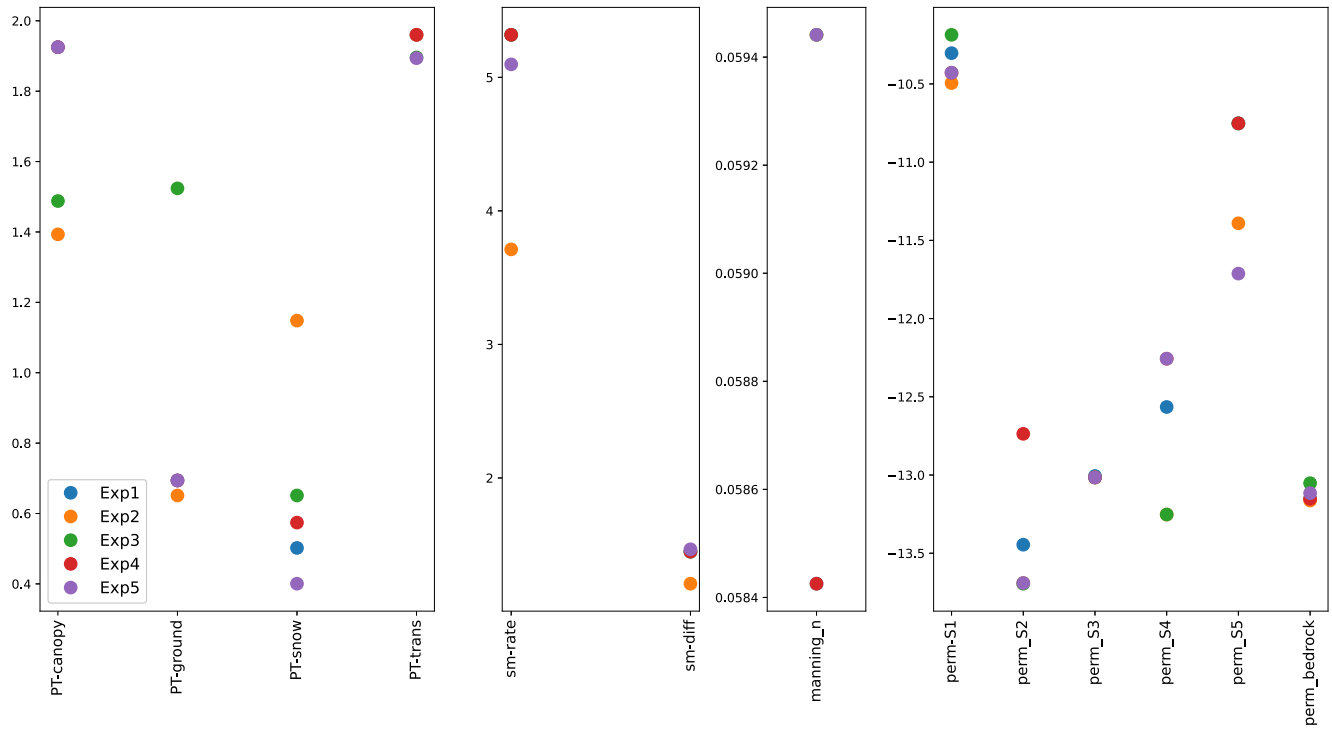

Figure 4. Comparison of parameter values estimated from the five experiments in Table 2. Symbol colors correspond to different experiments.

Results are shown for the UNRW basin outlet (Gage 01435000) for one-day-ahead prediction. Here training a `DeepONet-F` to predict the observed streamflow directly has limited or even detrimental effect on test performance (Figure 6a). The KGE calculated between the mean of 10-member ensemble and observations is 0.43, as opposed to 0.73 between the original `DeepONet-P` model predictions and observations. A closer examination of Figure 6a suggests that `DeepONet-F` actually improves the low-flow match significantly (dark solid line), but the peak flows are underestimated, leading to inferior performance compared to the original model. When the mismatch learning scheme is used instead, the test performance during both low- and high-flow periods is improved. The KGE between the ensemble mean and observations is increased to 0.82 (Figure 6b). Scatter plots (Figure S4 in Supporting Information S1) of the `DeepONet-F` ensemble mean versus observed streamflow, also confirm that low-flow match is significantly improved compared to the original `DeepONet-P` model shown in Figure S3 in Supporting Information S1. The CRPSS value, calculated using the 10-member ensemble, gage observations, and the daily streamflow climatology (reference data set), is 72%, indicating the ensemble forecast performance improved significantly over that of the simple streamflow climatology.

Test results obtained here suggest that model bias reduction through hybrid ML can be sensitive to problem formulation—it is generally easier to learn the residuals than learning to predict observations directly. The results may also be explained from a transfer learning perspective. Learning to predict observations directly using a pre-trained model may encounter the well-known issue of domain shift, which in our case is caused by shifting from the ATS model output space to observation space. Learning the residuals, however, inherently provides the ML model with stronger constraints and thus leads to better performance. We emphasize that the latter training scheme is only reasonable for hindcasting or gap-filling applications, where the "future" model simulations are available. For real-time forecasting applications, the physics-based predictors for T_F need to be generated by running the calibrated ATS using predicted forcing variables (e.g., data from Quantitative Precipitation Forecasts).

Other potential factors not examined here are related to uncertainties in the Daymet forcing data, the daily temporal resolution used in simulation, and the snowmelt parameterization used in ATS, which may all affect the magnitude and timing of simulated peaks.

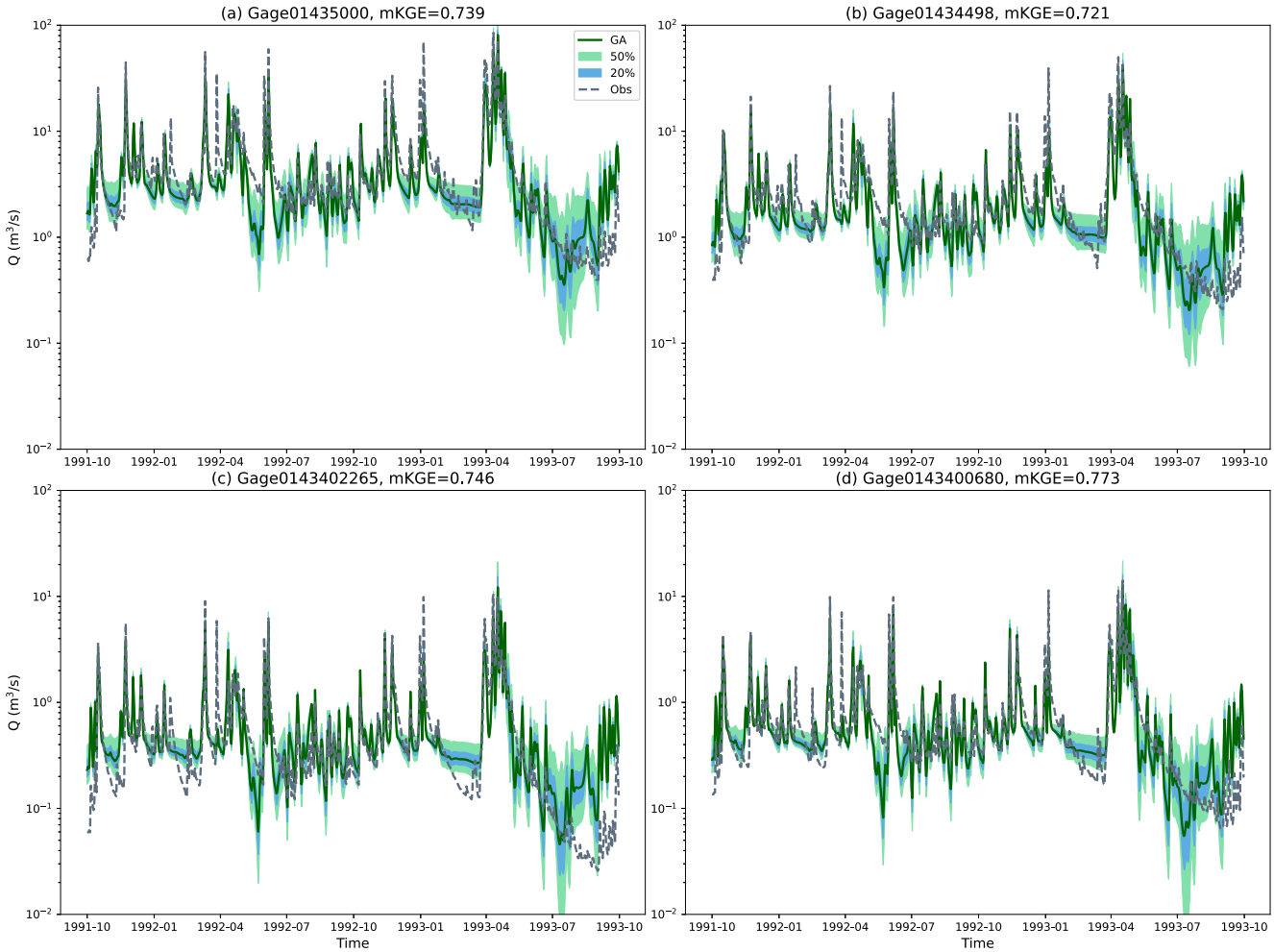

Figure 5. Example hydrographs for selected upper Neversink River watershed gages. On each plot, GA result (dark green solid line) is obtained by running `DeepONet − P` using parameters estimated from Exp 3 (see Table 2), gage observations are shown as gray dashed line, and blue and green shaded areas are bounds corresponding to 20% and 50% perturbations of Exp 3 parameter values.

5. Conclusions

Deep neural operator learning has appeared as a versatile tool for solving PDE problems. Its potential for hydrology remains largely under explored. For the first time this work presents an integrated workflow using DeepONet, a type of deep neural operator, for learning from an ensemble of hydrological model simulations. Ensemble models are routinely generated in hydrological, weather, and climate modeling studies for either forecasting or model inter-comparison purposes. In recent years, single-model large ensembles have also been used extensively for quantifying internal variability of the climate systems (Kay et al., 2015). Ensemble learning using DeepONet provides a natural means for extending the value of many of these existing large model ensembles, which can be very costly to create.

Using an ensemble simulated using the coupled surface/subsurface simulator ATS, we show that DeepONet models can be trained to learn both forward and inverse mappings effectively. Importantly, DeepONet learns mappings between infinite dimensional input/output spaces by using a finite collection of paired samples from the respective spaces. With a slight modification, we show that DeepONet may also be used to handle the seq2seq learning problem. In general, DeepONet achieves relatively high efficacy (KGE > 0.9) as a surrogate model. In our case, we found the performance of model calibration is relatively robust to the amount of observation data used, as long as data from the most downstream gage is always included. Nevertheless, physics-informed inverse mapping learning is dependent on the underlying model and parameter structures assumed in the original process-based model, thus significant model biases may remain after calibration. A major contribution of this study is we

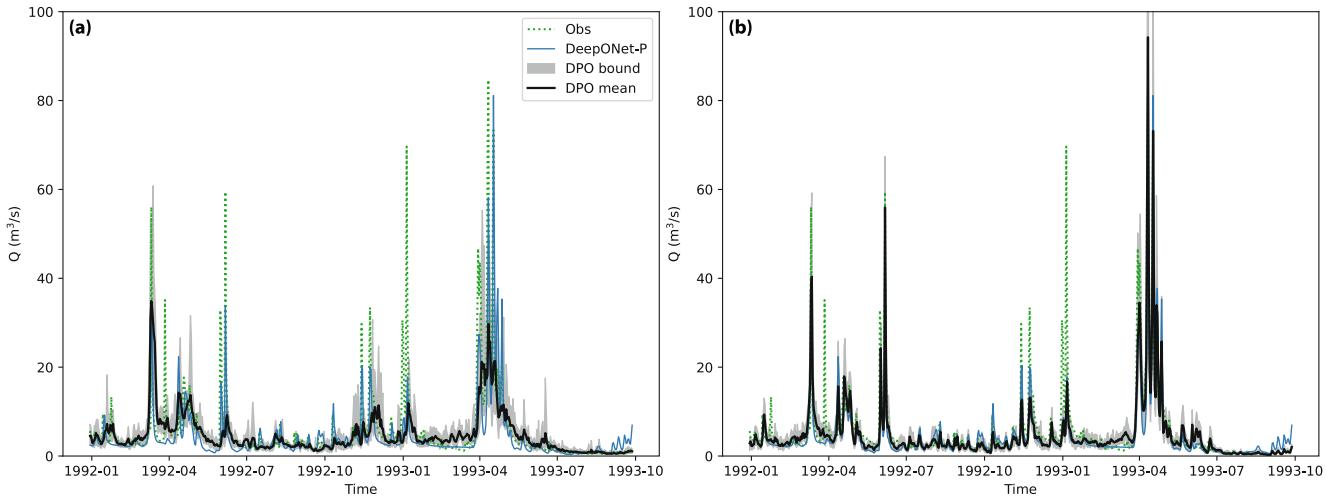

Figure 6. Comparison of `DeepONet − F` test performance at basin outlet (Gage 01435000) for (a) predicting streamflow directly (KGE = 0.43) and (b) predicting the streamflow mismatch (KGE = 0.82). The KGE values are calculated between the mean of a 10-`DeepONet − F`-model ensemble (dark solid line) and observations (green dotted line). For reference, the KGE between `DeepONet − P` (blue solid line) and observations is 0.73. Prediction bound is shown as shaded area.

use DeepONet seq2seq learning as an ML-based scheme to further reduce model biases while providing uncertainty quantification. Our results show the seq2seq learning on model-data mismatches can further improve model performance.

In this study, we mainly utilized the streamflow data, but several processes (e.g., snowpack, ET, and deep subsurface runoff) may benefit from other forms of calibration data. The ensemble outputs available in this study mainly consist of outputs from a small number of gage locations. Prediction in ungaged locations is demonstrated using a limited number of holdout gages. Nevertheless, one of the main strengths of DeepONet resides in its capability to query at sample locations not used during training. Future studies may explore the possibility of querying model outputs at dense grid locations, thus maximizing the real potential of DeepONet.

Appendix A: List of USGS Gages and Calibrated Model Parameters

Table A1 lists the USGS id and location of upper Neversink River watershed (UNRW) gages used in this study. In Table A2, the ATS parameters used in the UNRW model calibration are listed.

Table A1
List of the USGS Gage for UNSW, Where Lat and Lon Are in UTM Zone 18N

Gage ID	No. records	Lat (m)	Lon (m)
01435000	10,258	534,059.0	4,637,647.1
01434498	9,955	535,267.2	4,641,018.7
01434176	1,115	537,748.6	4,645,171.2
01434105	1,825	539,611.7	4,647,365.1
01434025	10,258	541,322.0	4,649,355.4
0143402265	1,176	542,994.0	4,648,846.3
01434021	8,126	548,468.2	4,651,189.5
01434017	9,955	538,144.9	4,641,636.1
01434013	1,175	541,032.1	4,643,041.4
0143400680	8,218	545,722.5	4,646,253.7

Table A2
List of the 13 Parameters Used in Inversion

Parameter name	Description (unit)	Min	Max
perm-S1	Permeability of the soil layer s1 (log m^2)	−13.669	−11.669
perm-S2	Permeability of the soil layer s2 (log m^2)	−14.183	−12.183
perm-S3	Permeability of the soil layer s3 (log m^2)	−14.598	−12.598
perm-S4	PermeaKGEbility of the soil layer s4 (log m^2)	−15.559	−13.559
perm-S5	Permeability of the soil layer s5 (log m^2)	−13.193	−11.193
perm-bedrock	Permeability of the fractured bedrock (log m^2)	−14.000	−13.000
sm-rate	Snow melt rate (mm day^{-1} $°C^{-1}$)	1.37	5.48
sm-diff	Air-snow temperature difference ($°C$)	1	5
manning_n	Manning's coefficient (−)	0.05	0.2
PT-trans	Priestley Taylor coefficient of canopy transpiration (−)	0.3	2.0
PT-snow	Priestley Taylor coefficient of snow evaporation (−)	0.3	2.0
PT-canopy	Priestley Taylor coefficient of canopy evaporation (−)	0.3	2.0
PT-ground	Priestley Taylor coefficient of ground evaporation (−)	0.3	2.0

Appendix B: Data Processing

Raw parameter values are scaled to range $[-1,1]$ using the `MinMaxScaler` from `scikit-learn` (Pedregosa et al., 2011). Streamflow rates are normalized by log-transform $\log(Q + 1 \times 10^{-4})$. Spatial coordinates are normalized by first setting the most downstream Gage 01435000 as the origin, calculating the offset of each of the remaining gages from the origin, and then normalizing the offsets using `MinMaxScaler`.

Appendix C: Performance Metrics Used

The modified Kling–Gupta efficiency (KGE) decomposes the performance of a model into linear correlation, bias and variability parameters (Kling et al., 2012),

$$\mathrm{KGE} = 1 - \sqrt{(r-1)^2 + (\gamma-1)^2 + (\beta-1)^2} \tag{A1}$$

where r is Pearson's correlation coefficient, $\beta = \mu_s/\mu_o$ is the ratio between simulated (μ_s) and observed (μ_o) mean flows, and γ is the ratio between the simulated and observed coefficients of variation (CV), $\gamma = (\sigma_s/\mu_s)/(\sigma_o/\mu_o)$. KGE ranges from $-\infty$ to 1.

The Continuous Ranked Probability Skill Score (CRPSS) is a measure of prediction accuracy relative to some reference data set (e.g., climatology),

$$\mathrm{CRPSS} = \left(1 - \frac{\overline{\mathrm{CRPS}}_{\mathrm{forecast}}}{\overline{\mathrm{CRPS}}_{\mathrm{ref}}}\right) \times 100\%, \tag{A2}$$

where CRPS (continuous ranked probability score) is a scoring function that compares a predicted value to groundtruth value (Gneiting & Raftery, 2007),

$$\mathrm{CRPS} = \int \left(F(x) - \mathbb{1}_{\{x \geq y\}}\right)^2 dx, \tag{A3}$$

in which y is an observation, F is the empirical cumulative distribution function associated with predictions, and $\mathbb{1}$ is a Heaviside step function that has a value of 1 if the predicted value x is greater or equal to observation y, and 0 otherwise. CRPS is commonly used to measure the quality of ensemble forecasts. The range of CRPS is from

0 to ∞, with 0 indicating a perfect match. In Equation A2, the mean of CRPS is used to calculate CRPSS. CRPSS ranges from −∞ to 1 (perfect prediction), but only values greater than zero indicate positive prediction skills.

For this work, the metrics KGE and CRPS were evaluated using `kge_2012` and `ens_crps` from the Python package Hydrostats (Roberts et al., 2018). The average daily streamflow calculated from gage observations was used as the reference data to calculate $CRPS_{ref}$.

Data Availability Statement

USGS streamflow data are downloadable from National Water Information System https://waterdata.usgs.gov/nwis. Daymet data are freely available from https://daymet.ornl.gov (Thornton et al., 2024). The ATS ensemble used in this study is available on Zenodo (https://doi.org/10.5281/zenodo.10834984). Source codes are available at https://github.com/dialuser/deeponet.

Acknowledgments
This work was funded by the ExaSheds project, which was supported by the U.S. Department of Energy (DOE), Office of Science, Office of Biological and Environmental Research, Earth and Environmental Systems Sciences Division, Data Management Program. Pacific Northwest National Laboratory is operated for the DOE by Battelle Memorial Institute under contract DE-AC05-76RL01830. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. A. Sun was also supported by the Department of Energy Advanced Scientific Computing Research Program under Grant DE-SC0022211.

References

Alfonso, L., He, L., Lobbrecht, A., & Price, R. (2013). Information theory applied to evaluate the discharge monitoring network of the Magdalena River. *Journal of Hydroinformatics*, *15*(1), 211–228. https://doi.org/10.2166/hydro.2012.066

Allen, R. G., Pruitt, W. O., Wright, J. L., Howell, T. A., Ventura, F., Snyder, R., et al. (2006). A recommendation on standardized surface resistance for hourly calculation of reference ETo by the FAO56 Penman-Monteith method. *Agricultural Water Management*, *81*(1–2), 1–22. https://doi.org/10.1016/j.agwat.2005.03.007

Anderson, J. L. (2001). An ensemble adjustment Kalman filter for data assimilation. *Monthly Weather Review*, *129*(12), 2884–2903. https://doi.org/10.1175/1520-0493(2001)129<2884:aeakff>2.0.co;2

Apicella, A., Donnarumma, F., Isgrò, F., & Prevete, R. (2021). A survey on modern trainable activation functions. *Neural Networks*, *138*, 14–32. https://doi.org/10.1016/j.neunet.2021.01.026

Baldigo, B. P., & Lawrence, G. (2000). Composition of fish communities in relation to stream acidification and habitat in the Neversink River, New York. *Transactions of the American Fisheries Society*, *129*(1), 60–76. https://doi.org/10.1577/1548-8659(2000)129<0060:cofcir>2.0.co;2

Burns, D. A., Riva-Murray, K., Bode, R. W., & Passy, S. (2008). Changes in stream chemistry and biology in response to reduced levels of acid deposition during 1987–2003 in the Neversink River basin, Catskill Mountains. *Ecological Indicators*, *8*(3), 191–203. https://doi.org/10.1016/j.ecolind.2007.01.003

Castelletti, A., Ficchì, A., Cominola, A., Segovia, P., Giuliani, M., Wu, W., et al. (2023). Model predictive control of water resources systems: A review and research agenda. *Annual Reviews in Control*, *55*, 442–465. https://doi.org/10.1016/j.arcontrol.2023.03.013

Chen, X., Hammond, G. E., Murray, C. J., Rockhold, M. L., Vermeul, V. R., & Zachara, J. M. (2013). Application of ensemble-based data assimilation techniques for aquifer characterization using tracer data at Hanford 300 area. *Water Resources Research*, *49*(10), 7064–7076. https://doi.org/10.1002/2012wr013285

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). Applied hydrology.

Clark, M. P., Rupp, D. E., Woods, R. A., Zheng, X., Ibbitt, R. P., Slater, A. G., et al. (2008). Hydrological data assimilation with the ensemble Kalman filter: Use of streamflow observations to update states in a distributed hydrological model. *Advances in Water Resources*, *31*(10), 1309–1324. https://doi.org/10.1016/j.advwatres.2008.06.005

Coon, E. T., Moulton, J. D., Kikinzon, E., Berndt, M., Manzini, G., Garimella, R., et al. (2020). Coupling surface flow and subsurface flow in complex soil structures using mimetic finite differences. *Advances in Water Resources*, *144*, 103701. https://doi.org/10.1016/j.advwatres.2020.103701

Coon, E. T., & Shuai, P. (2022). Watershed workflow: A toolset for parameterizing data-intensive, integrated hydrologic models. *Environmental Modelling & Software*, *157*, 105502. https://doi.org/10.1016/j.envsoft.2022.105502

Coon, E. T., Svyatsky, D., Jan, A., Kikinzon, E., Berndt, M., Atchley, A., et al. (2019). Advanced terrestrial simulator. Computer Software, USDOE Office of Science (SC), Biological and Environmental Research (BER)(SC-23). https://doi.org/10.11578/DC.20190911

Daly, C., Taylor, G., & Gibson, W. (1997). The prism approach to mapping precipitation and temperature. In *Proc., 10th AMS Conf. on Applied Climatology* (pp. 20–23).

Deser, C., Lehner, F., Rodgers, K. B., Ault, T., Delworth, T. L., DiNezio, P. N., et al. (2020). Insights from earth system model initial-condition large ensembles and future prospects. *Nature Climate Change*, *10*(4), 277–286. https://doi.org/10.1038/s41558-020-0731-2

Dong, X., Yu, Z., Cao, W., Shi, Y., & Ma, Q. (2020). A survey on ensemble learning. *Frontiers of Computer Science*, *14*(2), 241–258. https://doi.org/10.1007/s11704-019-8208-z

Duan, Q., Ajami, N. K., Gao, X., & Sorooshian, S. (2007). Multi-model ensemble hydrologic prediction using Bayesian model averaging. *Advances in Water Resources*, *30*(5), 1371–1386. https://doi.org/10.1016/j.advwatres.2006.11.014

Evensen, G. (2003). The ensemble Kalman filter: Theoretical formulation and practical implementation. *Ocean Dynamics*, *53*(4), 343–367. https://doi.org/10.1007/s10236-003-0036-9

Flint, A. L., & Childs, S. W. (1991). Use of the Priestley-Taylor evaporation equation for soil water limited conditions in a small forest clearcut. *Agricultural and Forest Meteorology*, *56*(3–4), 247–260. https://doi.org/10.1016/0168-1923(91)90094-7

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., & Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, *13*, 2171–2175.

Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International conference on machine learning* (pp. 1050–1059).

Giuliani, M., Lamontagne, J., Reed, P., & Castelletti, A. (2021). A state-of-the-art review of optimal reservoir control for managing conflicting demands in a changing world. *Water Resources Research*, *57*(12), e2021WR029927. https://doi.org/10.1029/2021wr029927

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*(477), 359–378. https://doi.org/10.1198/016214506000001437

Gupta, H. V., Sorooshian, S., & Yapo, P. O. (1998). Toward improved calibration of hydrologic models: Multiple and noncommensurable measures of information. *Water Resources Research*, *34*(4), 751–763. https://doi.org/10.1029/97wr03495

Harpold, A. A., Burns, D. A., Walter, T., Shaw, S. B., & Steenhuis, T. S. (2010). Relating hydrogeomorphic properties to stream buffering chemistry in the Neversink River watershed, New York State, USA. *Hydrological Processes*, *24*(26), 3759–3771. https://doi.org/10.1002/hyp.7802

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66–73. https://doi.org/10.1038/scientificamerican0792-66

Homer, C. H., Fry, J. A., & Barnes, C. A. (2012). The national land cover database. *US geological survey fact sheet*, *3020*(4), 1–4.

Houtekamer, P. L., Lefaivre, L., Derome, J., Ritchie, H., & Mitchell, H. L. (1996). A system simulation approach to ensemble prediction. *Monthly Weather Review*, *124*(6), 1225–1242. https://doi.org/10.1175/1520-0493(1996)124<1225:assate>2.0.co;2

Humphrey, G. B., Gibbs, M. S., Dandy, G. C., & Maier, H. R. (2016). A hybrid approach to monthly streamflow forecasting: Integrating hydrological model outputs into a Bayesian artificial neural network. *Journal of Hydrology*, *540*, 623–640. https://doi.org/10.1016/j.jhydrol.2016.06.026

Jiang, P., Shuai, P., Sun, A., Mudunuru, M. K., & Chen, X. (2023). Knowledge-informed deep learning for hydrological model calibration: An application to coal creek watershed in Colorado. *Hydrology and Earth System Sciences*, *27*(14), 2621–2644. https://doi.org/10.5194/hess-27-2621-2023

Jiang, P., Shuai, P., Sun, A. Y., & Chen, X. (2024). Optimizing parameter learning and calibration in an integrated hydrological model: Impact of observation length and information. *Journal of Hydrology*, *643*, 131889. https://doi.org/10.1016/j.jhydrol.2024.131889

Jiang, P., Yang, Z., Wang, J., Huang, C., Xue, P., Chakraborty, T., et al. (2023). Efficient super-resolution of near-surface climate modeling using the Fourier neural operator. *Journal of Advances in Modeling Earth Systems*, *15*(7), e2023MS003800. https://doi.org/10.1029/2023ms003800

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), 422–440. https://doi.org/10.1038/s42254-021-00314-5

Kay, J. E., Deser, C., Phillips, A., Mai, A., Hannay, C., Strand, G., et al. (2015). The community earth system model (CESM) large ensemble project: A community resource for studying climate change in the presence of internal climate variability. *Bulletin of the American Meteorological Society*, *96*(8), 1333–1349. https://doi.org/10.1175/bams-d-13-00255.1

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kling, H., Fuchs, M., & Paulin, M. (2012). Runoff conditions in the upper Danube basin under an ensemble of climate change scenarios. *Journal of Hydrology*, *424*, 264–277. https://doi.org/10.1016/j.jhydrol.2012.01.011

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2023). Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, *24*(89), 1–97.

Li, Z., & Arora, S. (2019). An exponential learning rate schedule for deep learning. arXiv preprint arXiv:1910.07454.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:2010.08895.

Liu, Y., & Gupta, H. V. (2007). Uncertainty in hydrologic modeling: Toward an integrated data assimilation framework. *Water Resources Research*, *43*(7). https://doi.org/10.1029/2006wr005756

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, *3*(3), 218–229. https://doi.org/10.1038/s42256-021-00302-5

Lu, L., Pestourie, R., Johnson, S. G., & Romano, G. (2022). Multifidelity deep neural operators for efficient learning of partial differential equations with application to fast inverse design of nanoscale heat transport. *Physical Review Research*, *4*(2), 023210. https://doi.org/10.1103/physrevresearch.4.023210

Moradkhani, H., Sorooshian, S., Gupta, H. V., & Houser, P. R. (2005). Dual state–parameter estimation of hydrological models using ensemble Kalman filter. *Advances in Water Resources*, *28*(2), 135–147. https://doi.org/10.1016/j.advwatres.2004.09.002

Moya, C., Zhang, S., Lin, G., & Yue, M. (2023). Deeponet-grid-uq: A trustworthy deep operator framework for predicting the power grid's post-fault trajectories. *Neurocomputing*, *535*, 166–182. https://doi.org/10.1016/j.neucom.2023.03.015

NERSC. (2024). Retrieved from https://docs.nersc.gov/systems/perlmutter/architecture

NRCS. (2024). Soil survey geographic database. Retrieved from https://www.nrcs.usda.gov/resources/data-and-reports/soil-survey-geographic-database-ssurgo

Nvidia. (2024). Modulus 22.09 user's manual. Retrieved from https://docs.nvidia.com/deeplearning/modulus/modulus-v2209/index.html

Pathak, J., Subramanian, S., Harrington, P., Raja, S., Chattopadhyay, A., Mardani, M., et al. (2022). Fourcastnet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. arXiv preprint arXiv:2202.11214.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Priestley, C. H. B., & Taylor, R. J. (1972). On the assessment of surface heat flux and evaporation using large-scale parameters. *Monthly Weather Review*, *100*(2), 81–92. https://doi.org/10.1175/1520-0493(1972)100<0081:otaosh>2.3.co;2

Psaros, A. F., Meng, X., Zou, Z., Guo, L., & Karniadakis, G. E. (2023). Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *Journal of Computational Physics*, *477*, 111902. https://doi.org/10.1016/j.jcp.2022.111902

Roberts, W., Williams, G. P., Jackson, E., Nelson, E. J., & Ames, D. P. (2018). Hydrostats: A python package for characterizing errors between observed and predicted time series. *Hydrology*, *5*(4), 66. https://doi.org/10.3390/hydrology5040066

Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *8*(4), e1249. https://doi.org/10.1002/widm.1249

Shuai, P., Chen, X., Mital, U., Coon, E. T., & Dwivedi, D. (2022). The effects of spatial and temporal resolution of gridded meteorological forcing on watershed hydrological responses. *Hydrology and Earth System Sciences*, *26*(8), 2245–2276. https://doi.org/10.5194/hess-26-2245-2022

Sobol', I. M. (1967). On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel noi Matematiki i Matematicheskoi Fiziki*, *7*(4), 784–802. https://doi.org/10.1016/0041-5553(67)90144-9

Sun, A. Y., Jiang, P., Yang, Z.-L., Xie, Y., & Chen, X. (2022). A graph neural network (GNN) approach to basin-scale river network learning: The role of physics-based connectivity and data fusion. *Hydrology and Earth System Sciences*, *26*(19), 5163–5184. https://doi.org/10.5194/hess-26-5163-2022

Sun, A. Y., Li, Z., Lee, W., Huang, Q., Scanlon, B. R., & Dawson, C. (2023). Rapid flood inundation forecast using Fourier neural operator. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 3733–3739).

Sun, N.-Z., & Sun, A. Y. (2015). *Model calibration and parameter estimation: For environmental and water resource systems*. Springer.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, *27*.

Taccari, M. L., Wang, H., Goswami, S., De Florio, M., Nuttall, J., Chen, X., & Jimack, P. K. (2024). Developing a cost-effective emulator for groundwater flow modeling using deep neural operators. *Journal of Hydrology*, *630*, 130551. https://doi.org/10.1016/j.jhydrol.2023.130551

Tebaldi, C., & Knutti, R. (2007). The use of the multi-model ensemble in probabilistic climate projections. *Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences*, *365*(1857), 2053–2075. https://doi.org/10.1098/rsta.2007.2076

Thornton, M., Shrestha, R., Wei, Y., Thornton, P., Kao, S., & Wilson, B. (2024). *Daymet: Daily surface weather data on a 1-km grid for north America, version 4*. Ornl Daac.

Vrugt, J. A., & Robinson, B. A. (2007). Treatment of uncertainty using ensemble methods: Comparison of sequential data assimilation and Bayesian model averaging. *Water Resources Research*, *43*(1). https://doi.org/10.1029/2005wr004838

Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A., & Benson, S. M. (2022). U-fno an enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, *163*, 104180. https://doi.org/10.1016/j.advwatres.2022.104180

Whitaker, J. S., & Hamill, T. M. (2002). Ensemble data assimilation without perturbed observations. *Monthly Weather Review*, *130*(7), 1913–1924. https://doi.org/10.1175/1520-0493(2002)130<1913:edawpo>2.0.co;2

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). Integrating scientific knowledge with machine learning for engineering and environmental systems. *ACM Computing Surveys*, *55*(4), 1–37. https://doi.org/10.1145/3514228

Xia, Y., Mitchell, K., Ek, M., Sheffield, J., Cosgrove, B., Wood, E., et al. (2012). Continental-scale water and energy flux analysis and validation for the North American land data assimilation system project phase 2 (NLDAS-2): 1. Intercomparison and application of model products. *Journal of Geophysical Research*, *117*(D3). https://doi.org/10.1029/2011jd016048

Yan, B., Chen, B., Harp, D. R., Jia, W., & Pawar, R. J. (2022). A robust deep learning workflow to predict multiphase flow behavior during geological CO_2 sequestration injection and post-injection periods. *Journal of Hydrology*, *607*, 127542. https://doi.org/10.1016/j.jhydrol.2022.127542

Yang, Y., & Burn, D. H. (1994). An entropy approach to data collection network design. *Journal of Hydrology*, *157*(1–4), 307–324. https://doi.org/10.1016/0022-1694(94)90111-2

Yang, Y., Lv, H., & Chen, N. (2023). A survey on ensemble learning under the era of deep learning. *Artificial Intelligence Review*, *56*(6), 5545–5589. https://doi.org/10.1007/s10462-022-10283-5

Zounemat-Kermani, M., Batelaan, O., Fadaee, M., & Hinkelmann, R. (2021). Ensemble machine learning paradigms in hydrology: A review. *Journal of Hydrology*, *598*, 126266. https://doi.org/10.1016/j.jhydrol.2021.126266