Spatio-temporal machine learning for continental scale terrestrial hydrology

Andrew Bennett[1], Hoang Tran[2], Luis De la Fuente[1], Amanda Triplett[1], Yueling Ma[3], Peter Melchior[3], Reed M. Maxwell[3], and Laura Elizabeth Condon[1]

[1]University of Arizona
[2]Pacific Northwest National Laboratory
[3]Princeton University

November 24, 2023

Posted on 24 Nov 2023 — CC-BY 4.0 — https://doi.org/10.22541/au.170079536.69065283/v1 — This is a preprint and has not been peer-reviewed. Data may be preliminary.

Abstract

Integrated hydrologic models can simulate coupled surface and subsurface processes but are computationally expensive to run at high resolutions over large domains. Here we develop a novel deep learning model to emulate continental-scale subsurface flows simulated by the integrated ParFlow-CLM model. We compare convolutional neural networks like ResNet and UNet run autoregressively against our novel architecture called the Forced SpatioTemporal RNN (FSTR). The FSTR model incorporates separate encoding of initial conditions, static parameters, and meteorological forcings, which are fused in a recurrent loop to produce spatiotemporal predictions of groundwater. We evaluate the model architectures on their ability to reproduce 4D pressure heads, water table depths, and surface soil moisture over the contiguous US at 1km resolution and daily time steps over the course of a full water year. The FSTR model shows superior performance to the baseline models, producing stable simulations that capture both seasonal and event-scale dynamics across a wide array of hydroclimatic regimes. The emulators provide over 1000x speedup compared to the original physical model, which will enable new capabilities like uncertainty quantification and data assimilation for integrated hydrologic modeling that were not previously possible. Our results demonstrate the promise of using specialized deep learning architectures like FSTR for emulating complex process-based models without sacrificing fidelity.

Hosted file

977690_0_art_file_11525486_s33hq9.docx available at https://authorea.com/users/677920/articles/674778-spatio-temporal-machine-learning-for-continental-scale-terrestrial-hydrology

Spatio-temporal machine learning for continental scale terrestrial hydrology

Enter authors here: Andrew Bennett[1], Hoang Tran[2], Luis De la Fuente[1], Amanda Triplett[1], Yueling Ma[3,4] , Peter Melchior[5,6], Reed M. Maxwell[3,4], Laura E. Condon[1]

[1]Department of Hydrology and Atmospheric Sciences, University of Arizona, Tucson, AZ, USA.

[2]Atmospheric Science & Global Change Division, Pacific Northwest National Laboratory, Richland, WA, USA.

[3]Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ, USA

[4]High Meadows Environmental Institute, Princeton University, Princeton, NJ, USA.

[5]Department of Astrophysical Sciences, Princeton University, Princeton, NJ, USA.

[6]Center for Statistics and Machine Learning, Princeton University, Princeton, NJ, USA.

Corresponding author: Andrew Bennett (andrbenn@arizona.edu)

Key Points:

- Deep learning models can emulate a complex integrated hydrology model that simulates groundwater over the United States
- We developed a new model architecture that is more robust over longer simulation periods than off-the-shelf neural networks
- Deep-learning based emulators of complex models enable new applications such as real-time forecasting and estimating uncertainties

Abstract

Integrated hydrologic models can simulate coupled surface and subsurface processes but are computationally expensive to run at high resolutions over large domains. Here we develop a novel deep learning model to emulate continental-scale subsurface flows simulated by the integrated ParFlow-CLM model. We compare convolutional neural networks like ResNet and UNet run autoregressively against our novel architecture called the Forced SpatioTemporal RNN (FSTR). The FSTR model incorporates separate encoding of initial conditions, static parameters, and meteorological forcings, which are fused in a recurrent loop to produce spatiotemporal predictions of groundwater. We evaluate the model architectures on their ability to reproduce 4D pressure heads, water table depths, and surface soil moisture over the contiguous US at 1km resolution and daily time steps over the course of a full water year. The FSTR model shows superior performance to the baseline models, producing stable simulations that capture both seasonal and event-scale dynamics across a wide array of hydroclimatic regimes. The emulators provide over 1000x speedup compared to the original physical model, which will enable new capabilities like uncertainty quantification and data assimilation for integrated hydrologic modeling that were not previously possible. Our results demonstrate the promise of using specialized deep learning architectures like FSTR for emulating complex process-based models without sacrificing fidelity.

Plain Language Summary

Computational models are important for understanding and predicting terrestrial hydrology, but our most physically detailed models can be time-consuming and expensive to run over large regions. In this study, we trained deep learning models to emulate a complex hydrology model that simulates groundwater flow over the contiguous US. We developed a new model architecture called FSTR that captures spatiotemporal patterns better than standard deep learning models. FSTR is over 1000 times faster than ParFlow, the original hydrologic model. This enables new possibilities like forecasting groundwater changes and estimating uncertainties. Our results show that specialized deep learning architectures can accurately emulate complex hydrologic models while drastically reducing computation time.

1 Introduction

Computational models have been hugely successful in predicting and building understand of Earth and environmental systems. Since their early use in the mid twentieth century these models have increased in complexity to account for higher spatiotemporal resolution and physical process complexity. To achieve the highest possible degree of physical realism scientists and engineers often must run these models on high-performance computing clusters and supercomputers, which require large institutional investment to build and maintain. In addition, the computational complexity of using these models on such platforms requires considerable expertise to use effectively. As a result, these large-scale and highly complex models are typically only used by a small subset of researchers.

Emulation (also referred to as reduced order models (ROMs) or surrogate models) has long been a popular method to reduce the computational complexity of running simulations in a number of domains (Astrid et al., 2008; Razavi et al., 2012; C. Wang et al., 2014). Reducing the computational complexity of process based models makes it possible to build more complex workflows such as building model chains, performing parameter calibration or sensitivity experiments (Cheng et al., 2023), and running more scenarios to better understand uncertainties (Kasim et al., 2021). One avenue that is becoming increasingly popular is the use of deep

learning (DL) based methods for building emulators of process based models (Doury et al., 2023; Leonarduzzi et al., 2022; Reichstein et al., 2019; Tran et al., 2021).

Deep learning has recently and quickly become a standard piece of the computational modeling toolkit and offers almost universal applicability to modeling tasks (Jordan & Mitchell, 2015). It has also proven very powerful in allowing researchers to take models from disparate applications and apply them to new problems (Khan et al., 2022). This is true in the Earth system sciences and related fields, where deep learning models have been used to simulate atmospheric phenomena (Brenowitz et al., 2020), predict flow in rivers (Kratzert et al., 2018), and monitor land use (Xu et al., 2017) among many other applications. Specifically in hydrology the majority of applications are based around streamflow modeling or other forms of "point-scale" applications where the models are trained on individual sites (Bennett & Nijssen, 2021; de la Fuente et al., 2023; Gauch et al., 2021). However, we know that in subsurface hydrology lateral flow occurs, and can have large impacts on both human and natural systems (Condon & Maxwell, 2019; Fan, 2015).

Hydrologic models that treat both the coupled surface and groundwater systems as well as account for lateral flow in the subsurface are commonly referred to as integrated hydrologic models. These models are among the most complex and comprehensive representations of the terrestrial hydrologic cycle that have been developed. However, they are often difficult to run because they are data-hungry and computationally heavy. This is particularly true in the subsurface, where observations are sparse and parameters are hard to measure (Blöschl et al., 2019). Even when data is available, it is often difficult to calibrate these models because of the computational complexity and large-dimensional search space over parameter configurations (O'Neill et al., 2021). It should also be noted that the lack of spatiotemporally complete observations/reanalysis data for groundwater is a large reason that purely data-driven approaches have not emerged as they have in the weather forecasting domain (Chen et al., 2023; Keisler, 2022; Lam et al., 2022). Because of these challenges the use of emulator or surrogate models is an appealing approach to improving the usability of integrated hydrologic models.

In this study we demonstrate the use of modern deep learning based emulators of a continental scale integrated hydrologic model, without sacrificing spatiotemporal or process fidelity. Specifically, we emulate subsurface flow of the ParFlow-CLM model, developed over a large portion of the contiguous US at a high spatiotemporal resolution (Maxwell et al., 2015; Maxwell & Condon, 2016; O'Neill et al., 2021). Previous work has shown that deep learning is an effective approach to this problem, showing good performance on synthetic benchmarks (Maxwell et al., 2021)and on smaller domains (Leonarduzzi et al., 2022; Tran et al., 2021).

In this work, we compare the ability of three different deep learning model architectures to emulate 3d subsurface pressure fields simulated by ParFlow. We compare ResNet, UNet, and Forced-SpatioTemporal-RNN (FSTR) architectures; the first two are off-the-shelf convolutional neural networks (CNN) that we apply in an autoregressive fashion to build up spatiotemporal predictions. That is, we feed the previous output of the model as an input to the next step of the prediction process, in addition to other features.

The FSTR model is a novel adaptation of the PredRNN model with action-conditioning, which is a video-prediction model that has proven capable for atmospheric & hydrologic modeling as well as robotics (Tran et al., 2021; Wang et al., 2017). We make use of the robotics terminology of "action-conditioning" to explicitly account for the meteorological forcings acting on the hydrology in a way that is separate from the subsurface parameters/geology and the initial state of the domain that we are simulating.

We compare the performance of the three model architectures for emulating the evolution of both pressure heads at multiple depths as well as the derived soil moisture states and water table depths. We find that the ResNet produces unstable results over long simulation rollouts, while the UNet and FSTR both show good overall capabilities at matching spatial and temporal patterns. Our FSTR architecture consistently shows the best performance results, and is capable of simulating a year of the entire domain in less than an hour on a single 40 GB Nvidia A100 GPU, showing a >1000 times speedup as compared to the original simulations run on >3000 CPU cores. Based on these findings we believe that our FSTR architecture could form the basis for a new set of modeling capabilities to fine tune model parameters and enable real-time ensemble-based forecasting.

2 Methods

2.1 Modeling domain and data

Our modeling domain is based on the ParFlow CONUS1.0 model which is documented by (O'Neill et al., 2021). The domain covers the majority of the contiguous United States, plus some small portions of Canada and Mexico (Figure 1). We are primarily interested in representing the subsurface hydrology in this domain at a 1km gridded spatial resolution, with a daily timestep. The gridded domain amounts to a regular grid of 3342 by 1888 km in the longitudinal and latitudinal directions, respectively. The depth layers of our simulations increase in the downward direction, starting with shallow surface layers and a large groundwater layer. The depths of each layer are 0.1, 0.3, 0.6, 1, and 100 m from the surface to the bottom for a total of 5 layers.

The simulations that we use for training/validation/testing have previously been validated against many observational datasets across multiple variables (e.g. streamflow, evapotranspiration, snow) with favorable results given the default parameter sets (O'Neill et al., 2021). These simulations cover water years 2003-2006 at an hourly timescale. We aggregate all the data to a daily timescale by taking the daily means, totals, minimums, and maximums where appropriate.

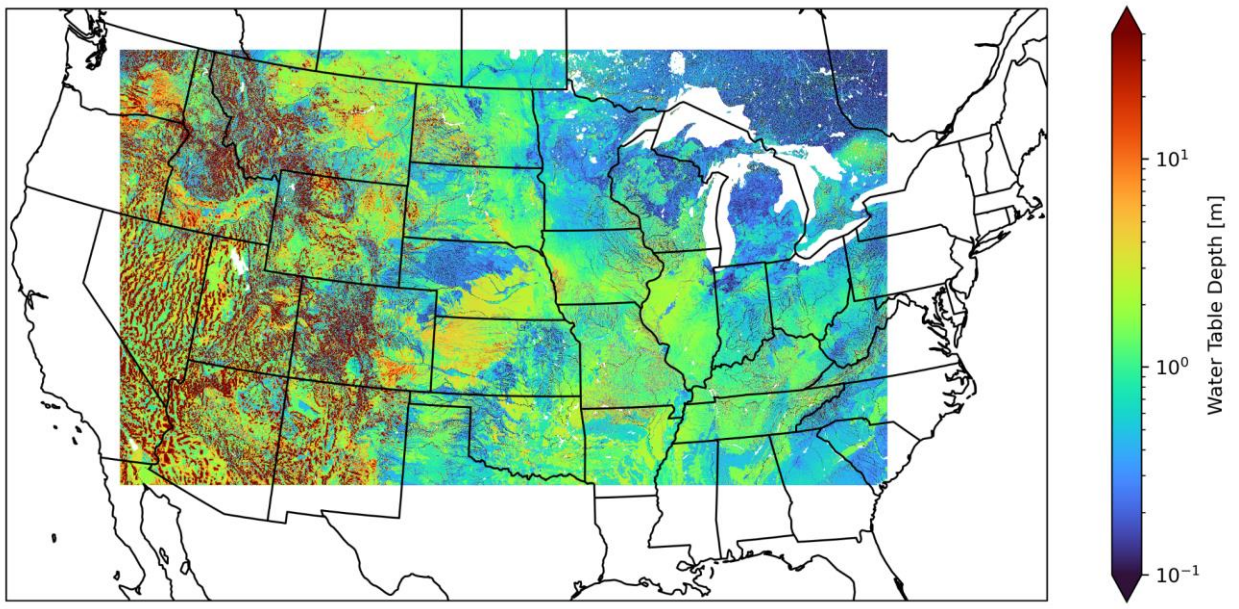

Figure 1. The extent of our modeling domain, covering most of the Contiguous United States (CONUS). We show the long-term average water table depth to highlight the spatial variability of the domain.

In this study we are primarily concerned with modeling subsurface and surface water flow, which is represented in ParFlow with Richard's equation (Richards, 1931) parameterized by the van Genuchten closure relations between pressure heads and saturation content (van Genuchten, 1980). Specifically, our emulation target is the full four-dimensional (time+space) pressure head field. From this pressure head field we can use the closure equations and additional calculations to calculate soil moisture and water table depth. The governing equations that describe the dynamics of the subsurface flows are given as:

$$S_s S(\psi)\frac{\partial \psi}{\partial t} + \phi\frac{\partial S(\psi)}{\partial t} = \Delta\left(-K_s(x)k(\psi)\cdot\nabla(\psi - z)\right) + q$$

Where S_s is the specific storage $[L^{-1}]$, S is the relative saturation $[-]$, ψ is the pressure head $[L]$, K_s is the saturated hydraulic conductivity $[LT^{-1}]$, k is the relative permeability $[-]$, ϕ is the porosity $[-]$, and q is a source-sink term$[L^3T^{-1}]$.

2.2 Model architectures

We explore three deep learning model architectures to emulate the simulations described previously. Here we will describe the overall structure of each of these models, but will leave the detailed input/output setup of them for the following section. We employ three model architectures in this study: a Residual Network (ResNet; He et al., 2015), a UNet (Ronneberger et al., 2015), and a newly developed architecture that we refer to as the ForcedSpatioTemporalRNN (FSTR) model. The first two of these are relatively "standard" models in the deep learning literature at this point and have been used for a large array of tasks. Their architectures of the ResNet and UNet are shown in schematic form in figure 2, while the FSTR architecture is shown in figure 3.

The development of the ResNet architecture is considered a milestone in the development of machine learning to the task of image classification and paved the way for the modern deep learning revolution (He et al., 2015). We consider the ResNet a strong baseline architecture to compare against as has been adopted by other studies with similar approaches (Haber & Ruthotto, 2018; Kochkov et al., 2021; Ott et al., 2020). This model consists of stacks of "residual blocks", usually convolutional layers followed by a nonlinear activation function. Following these stacks is a residual connection, which adds the input back to the output of the residual block. Architectures with residual connections have been found to train more effectively, especially in very deep networks. We stack two residual blocks, each consisting of 1 depthwise-separable convolutional layer with a layer norm and activation function, and a final convolutional layer. We use four layers with each layer having a hidden dimension of 256 channels for this study.

The second architecture that we consider is the UNet, which is named as such because of its use of successive downsampling and upsampling layers (along with skip connections) which allow the model to capture spatial relationships at varying resolutions. This type of architecture is considered state of the art in image segmentation which has applications in both remote sensing (Yuan et al., 2021) and medical imaging (Ronneberger et al., 2015). Like the ResNet, the UNet is mainly composed of stacks of convolutional layers. However, the UNet makes use of downsampling and upsampling to get a "multi-resolution" view of the data. In our model, we use stacks of these downsampling and upsampling layers which are composed of convolutional layers that are either preceded by MaxPool layers or followed by bilinear interpolation layers for down and upsampling, respectively. Each downsampling and upsampling layer consists of two convolutions followed by an activation. At parallel levels in the downsampling/upsampling skip connections are used to transfer information at multiple resolutions to the upsampling, which helps to preserve spatial structure in the data. Here we set the base dimension for the convolutional layer to be eight at the input and output and double/halve the hidden dimension for each down/up sampling layer respectively.

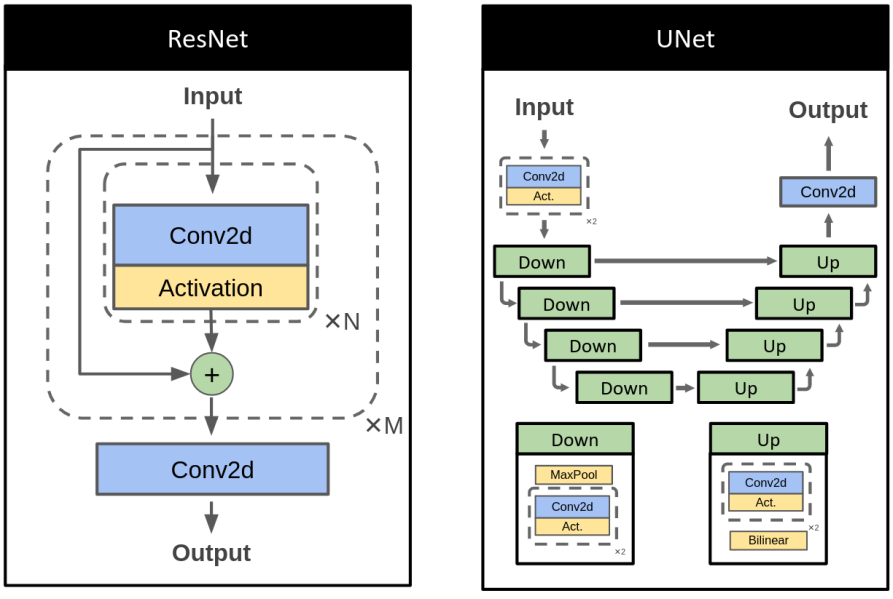

Figure 2. Diagrams of the two baseline model architectures used in our experimental setup. Left shows a ResNet architecture, which stacks convolutional blocks with residual connections that

propagate the input signal into deeper layers. The right panel shows the UNet architecture, which also consists of convolutional blocks, but differs from the ResNet by using them in successive downsampling and upsampling configurations, which consider multiple resolutions of the input data.

We also develop a novel architecture based on the PredRNN model, which is itself based on the Convolutional LSTM model (ConvLSTM; Shi et al., 2015). It attempts to take best practices from image modeling via CNNs and sequence modeling from recurrent neural networks (RNNs), particularly with the application of Long Short Term Memory networks (LSTMs; Hochreiter & Schmidhuber, 1997). The key insights that the developers of PredRNN and associated models had over the ConvLSTM architectures was that an additional memory/hidden state would better reflect the spatiotemporal state of the system, and could be shared amongst model layers. This, along with several other procedural training techniques led the PredRNN model to be one of the most performant video prediction models available (Wang et al., 2017). In the original PredRNN paper, the authors also introduced an "Action-Conditioned" variant, which allows the input sequence of a robotic arm to be used as a part of the prediction algorithm. We make use of this modification because the input to the robotic system "acts" on the video stream in a similar way that meteorological forcings "act" on the evolution of hydrologic states.

The main modification that we make to the PredRNN structure is the use of encoders to initialize the memory and cell states for the model. These states are updated throughout the recurrent loop, and by default, are initialized as zeros in the PredRNN structure. However, our insight is that the initial conditions and subsurface parameters can be considered a sort of byproduct of the true memory of the natural environment, and thus can be used to initialize these hidden states. We use the initial conditions (i.e. the 3-dimensional pressure heads for the domain being simulated) to initialize the memory state and the parameter values (e.g. porosity and permeability for each cell) to initialize the cell states. Both encoders consist of convolutional layers that project the inputs into a higher dimensional space that matches the hidden states of the Action-Conditioned ST-LSTM which forms the backbone of the FSTR model. The initial conditions are used as input to the memory encoder as well as used as the starting input pressure field to the model. We use two layers of the AC-ST-LSTM layer in our model, each having a hidden dimension of 64.

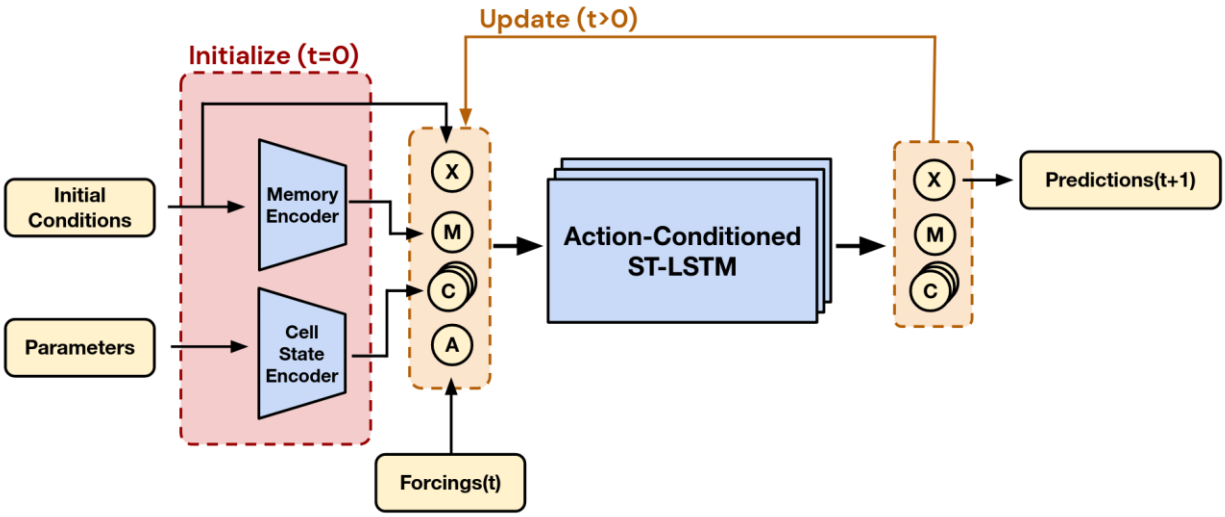

Figure 3. An architecture diagram of our proposed model architecture, the Forced Spatio Temporal RNN (FSTR). Tensor variables are represented in yellow colors, while neural-network layers with trainable weights are drawn in blue. The red "initialization" phase is only run a single time per training example, while the update arrow is run in a recurrent loop. Quantities in the orange outlined boxes represent the hidden states and model inputs.

The core of the FSTR model is the Action-Conditioned SpatioTemporal LSTM layer (AC-ST-LSTM), which was introduced in Wang et al. (2017). This layer modifies the ConvLSTM model in ways that allow it to take in external inputs on the system in the form of an action that is fused to the hidden state by an elementwise multiplication. The equations for the AC-ST-LSTM layer are given by

$$
\begin{aligned}
g_t &= tanh\left(W_{xg} * X_t + W_{hg} * H_{t-1}^l\right) \\
i_t &= \sigma\left(W_{xi} * X_t + W_{hi} * H_{t-1}^l\right) \\
f_t &= \sigma\left(W_{xf} * X_t + W_{hf} * H_{t-1}^l\right) \\
C_t^l &= f_t \odot C_{t-1}^l + i_t \odot g_t \\
g_t' &= tanh\left(W_{xi}' * X_t + W_{mg} * M_t^{l-1}\right) \\
i_t' &= \sigma\left(W_{xi}' * X_t + W_{mi} * M_t^{l-1}\right) \\
f_t' &= \sigma\left(W_{xf}' * X_t + W_{mf} * M_t^{l-1}\right) \\
M_t^l &= f_t' \odot M_t^{l-1} + i_t' \odot g_t' \\
o_t &= \sigma\left(W_{xo} * X_t + W_{ho} * H_{t-1}^l + W_{co} * C_t^l + W_{mo} * M_t^l\right) \\
H_t^l &= o_t \odot tanh\left(W_{1\times1} * [C_t^l, M_t^l]\right) \\
V_t^l &= \left(W_{hv} * H_{t-1}^l\right) \odot \left(W_{av} * A_{t-1}\right)
\end{aligned}
$$

Where X_t is the input, $W_{..}$ are the weight matrices, H_t^l is the hidden state, C_t^l is the cell state, M_t^l is the spatiotemporal memory state, A is the action tensor, i, f, g are the gating functions (along with their primed counterparts), V_t^l is the action fusion which allows for external inputs to modify the system, and o_t is the output for timestep t, and l is the depth layer of the network. Finally, the update step for the AC-ST-LSTM layer is

$$H_t^l, C_t^l, M_t^l = ACSTLSTM\left(X_t, V_t^l, C_{t-1}^l, M_t^{l-1}\right)$$

Our model structure considers the meteorological forcings to be "action" tensors rather than model inputs directly. This delineation of information partitioning between the initial conditions, parameter values, and meteorological forcings in FSTR is very similar to how ParFlow and many other hydrologic models operate.

All models take in daily minimum temperature, daily maximum temperature, total precipitation, snowmelt, and bulk evapotranspiration as the boundary condition forcing. Static parameters that the models take as input are porosity, permeability, van Genuchten n, van Genuchten alpha, topographic index, elevation, and a measure of distance to the nearest stream based on the digital elevation map. The static parameters were chosen to include the major parameters required to solve Richard's equation as well as represent major topographic features at both point and aggregated scales. The models output 4-dimensional spatiotemporal predictions of the subsurface pressure heads for all grid cells in the input domain. All models are run in an auto-regressive fashion, meaning they start with the initial conditions as a starting point and then evolve their output in response to the forcings and parameter value inputs, at which point they use their own prediction as the initial state for the next time step.

At inference time (that is, after the models have been trained) we use the models to produce the full 4D pressure head field, which is the quantity that is treated most fundamentally by the form of Richard's equation that is used in the ParFlow simulations. However, when using ParFlow model outputs it is often more useful to calculate other quantities such as soil moisture and water table depth from the subsurface pressure head field. We make a similar conversion to the model outputs by processing them into water table depth and surface soil moisture. These quantities are calculated in the same way that they are in standalone ParFlow simulations, but rather than using the functionality to compute this translation from ParFlow we re-implemented the routines in a PyTorch compatible layer, which in theory could provide these translations at training time. We do not take this approach here, however, due to several technical challenges which will be discussed later but could form the basis for training a fully differentiable model directly to data in the future.

2.3 Experimental setup

In this study we explore the ability of the three neural network architectures to reproduce the 3d ParFlow pressure heads across the CONUS domain. To quantify this, we train each model on water years 2003 and 2004 and validate them on water year 2005. The testing dataset that we evaluate against is from the water year 2006, and all results shown here are from this set. As mentioned previously, we aggregate variables to daily timesteps to reduce the overall complexity of the data. While we are interested in building emulators that maintain high spatiotemporal resolution we are primarily focused on seasonal to annual modeling, reflecting timescales that groundwater interactions tend to take place at.

We train all models in two phases. For the first phase we use a one-cycle learning rate scheduler with a maximum learning rate of 1e-3. We train on square chips of 64 by 64 pixels in size, and rollout horizons of 35 timesteps. This balance between medium spatial sizes and time horizon provides the model a good baseline for matching both spatial and temporal patterns, which resulted in the best overall training performance. During this portion of the training we augment the L2 loss with an additional term that penalizes gradients in the spatial directions of the predictions (Serifi et al., 2021). This modified loss function is given as:

$$\mathcal{L}(y, \hat{y}) = |y - \hat{y}|_2 + |\nabla y - \nabla \hat{y}|_2$$

Following an initial training epoch on this setup we then set the loss function to a pure L2 loss and continue training. At this point we train for another epoch using 48 pixel chips and a rollout horizon of 90 days. This helps stabilize the autoregressive loop of the model, as well as learn seasonal trends. Following this, we then train a final epoch using a 14-day rollout and 48 pixel chips. This final epoch of training is aimed at improving the model's ability to reproduce fast dynamics, particularly around responding to individual storm events. We found that training across all three phases improved the model prediction in terms of both stability of the predictions and the overall accuracy for all model types. However, the final training stage using the shorter rollouts helped the least. We also found that changing the ordering of the training phases led to degradation in performance. All models were trained on a single Nvidia A100 40GB GPU. Training times varied by model, but generally took roughly 50 hours each, with the ResNet being cheapest and FSTR being the most expensive.

3 Results

To analyze the performance of each of the emulator architectures we calculated the overall root mean square error (RMSE) and Pearson correlation (hereafter, just correlation) for each grid cell in the domain with respect to the original simulations. The resulting spatial maps for these measures are shown in figure 4. From figure 4a we see that the ResNet shows pockets of high error, particularly in the Western portion of the domain, but also in the upper midwest and southern central regions. These spots of high error are due to model instability at the full lead-time. We will further explore this later. The UNet and FSTR models show much lower RMSE in water table depth across the domain, with only some regions of higher error in the western portion of the domain. Overall, the FSTR model has the lowest RMSE, and highest correlations. All models had relatively low correlations on the western edge and sections in the central area of the domain. Interestingly, there are some differences in where the UNet and FSTR models show higher errors. For instance, FSTR has a region of low correlation in the plains on the western sides of Nebraska and Kansas, while the UNet seems to capture these correlations well (Although the FSTR model still showed lower overall errors in this region). The model performance improvement of FSTR over the UNet was primarily marked by reductions in RMSE over the eastern two thirds of the domain.

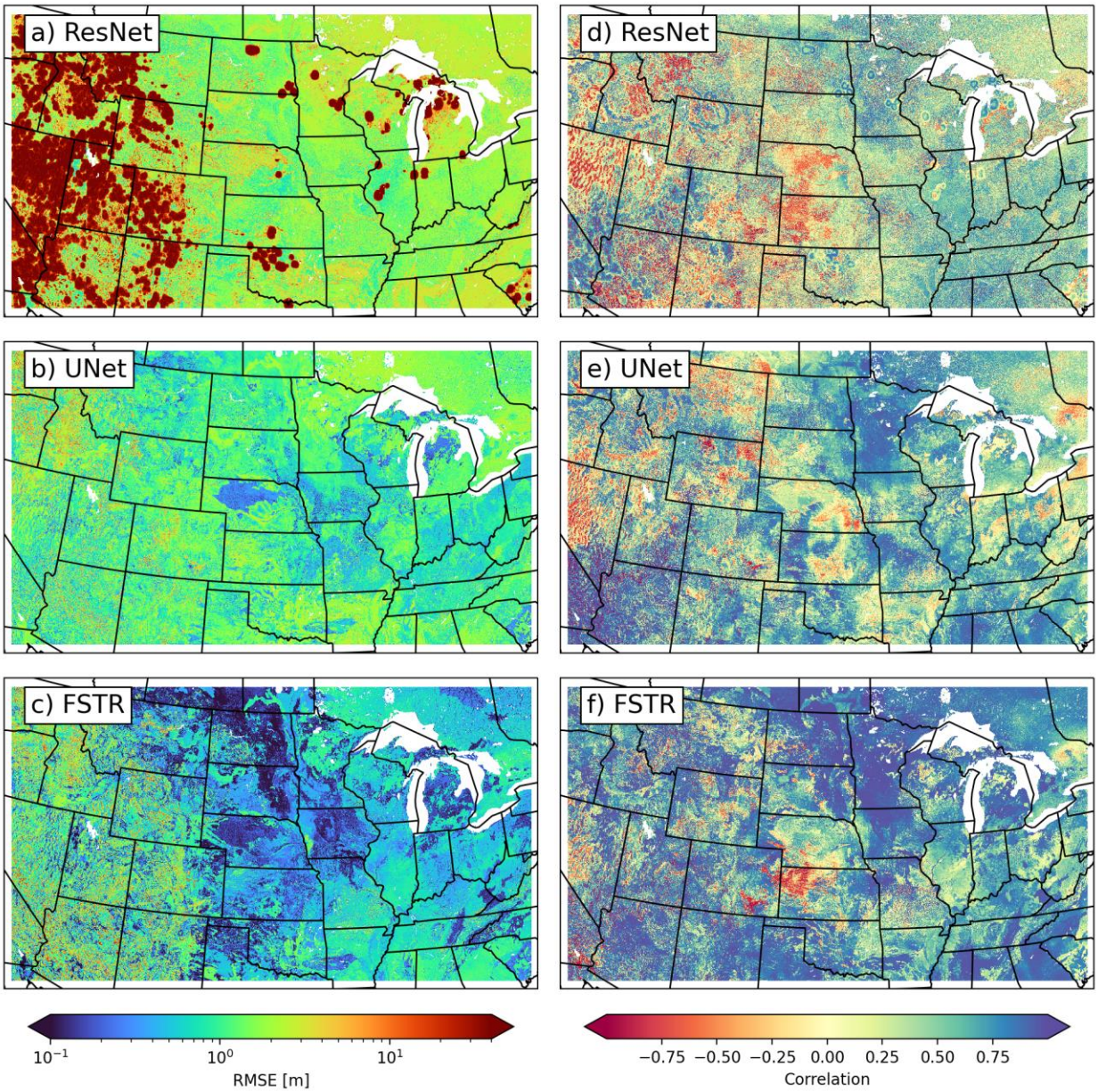

Figure 4. Spatial metrics of performance of each model architecture's ability to emulate water table depth over the entire testing period of water year 2006. The left column (a-c) shows the root mean square error (RMSE), while the right (d-f) shows the Pearson correlation.

Similarly, in figure 5 we examine the models' ability to emulate the surface saturation levels as well. Here we find again that the ResNet is the worst performing overall, with the UNet in the middle, and FSTR performing best. The areas of highest error for surface moisture tend to be the same for all the models, unlike the results for the water table depth. Of note is the northeastern portion of the domain above and around the Great Lakes, which have been masked out, as well as some regions throughout the central part of the domain and in the southeast corner. These are quite different hydrologic systems, which suggests deficiencies in the input data or model training procedure are the underlying cause, though we were not able to diagnose the exact reasons for these patterns.

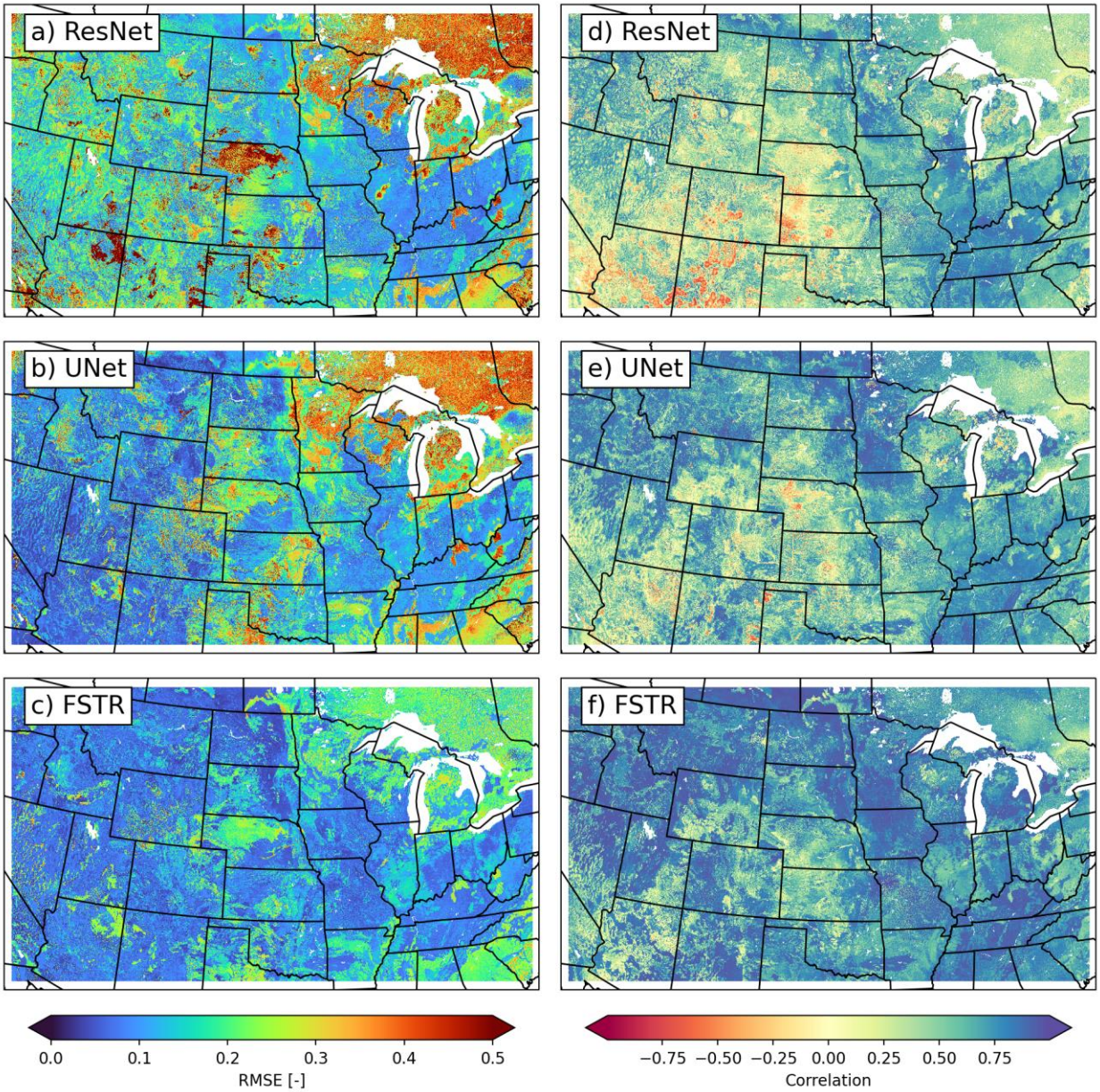

Figure 5. Spatial metrics of performance of each model architecture's ability to emulate surface soil moisture over the entire testing period of water year 2006. The left column (a-c) shows the root mean square error (RMSE), while the right (d-f) shows the Pearson correlation.

To better understand the temporal nature of error growth of the emulators, we also show the forecast error at increasing lead times in figure 6. Here we find that the ResNet tends to have reasonable forecast error out to about day 200, before growing rapidly, leading to the pockmark error patterns from figures 4 and 5. However, both the UNet and FSTR models have significantly lower error rates and no exponential blowup over the simulation period. The FSTR model maintains the lowest errors over the entire period, like what we saw in the spatial error analysis.

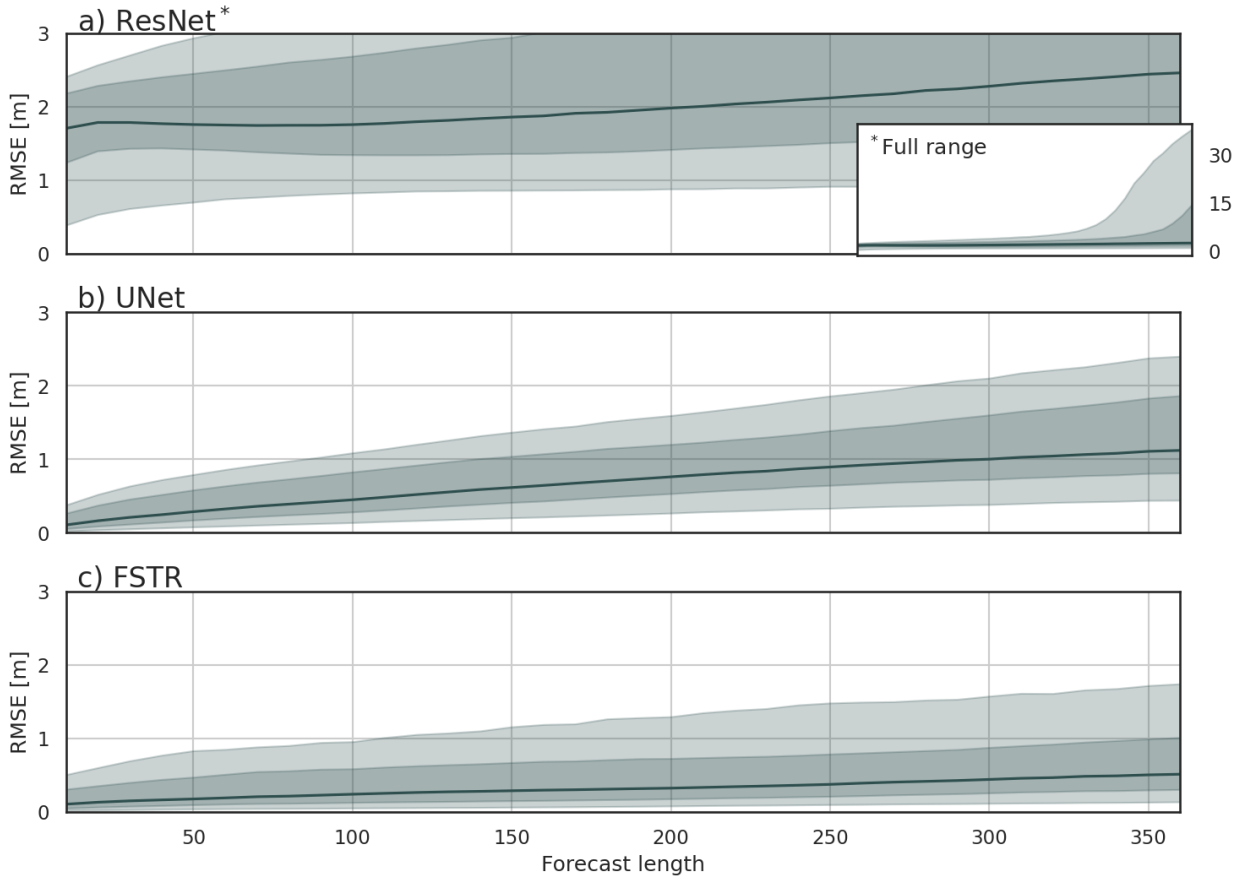

Figure 6. Growth of the error distribution of water table depth with increasing forecast length. The inset plot shows the full range of the error growth for the ResNet, which is more than ten times the error of the other two model architectures. The central lines show the median RMSE, while the shaded lines show the interquartile and 10-90% range of errors.

So far, we have looked at the overall error characteristics in the forecasts across the full domain, but it is worth zooming in on some particular regions to see local performance spatially and temporally. First, in figure 7 we show a 256 by 256 km sub-domain in the midwestern US. We omit the ResNet results here because of its instability and low accuracy. Comparing the time series for this region we once again see that the FSTR model is able to much more accurately capture the dynamics of the system, as compared to the UNet. Most notably, the FSTR model is able to capture the seasonal dynamics in a much more robust way than the UNet, which shows good overall correlation, but drifts in magnitude from the ParFlow results. Similarly, the spatial plots show that FSTR is much more able to represent the spatial heterogeneity across seasons, particularly in regions with shallow water tables. This connects back to the timeseries picture, where we see the UNet consistently predicts a deeper water table, thus making it hard to form the surface river network. Neither model can perfectly capture the structure of the river network, although both show features that correlate well with the river network from ParFlow.

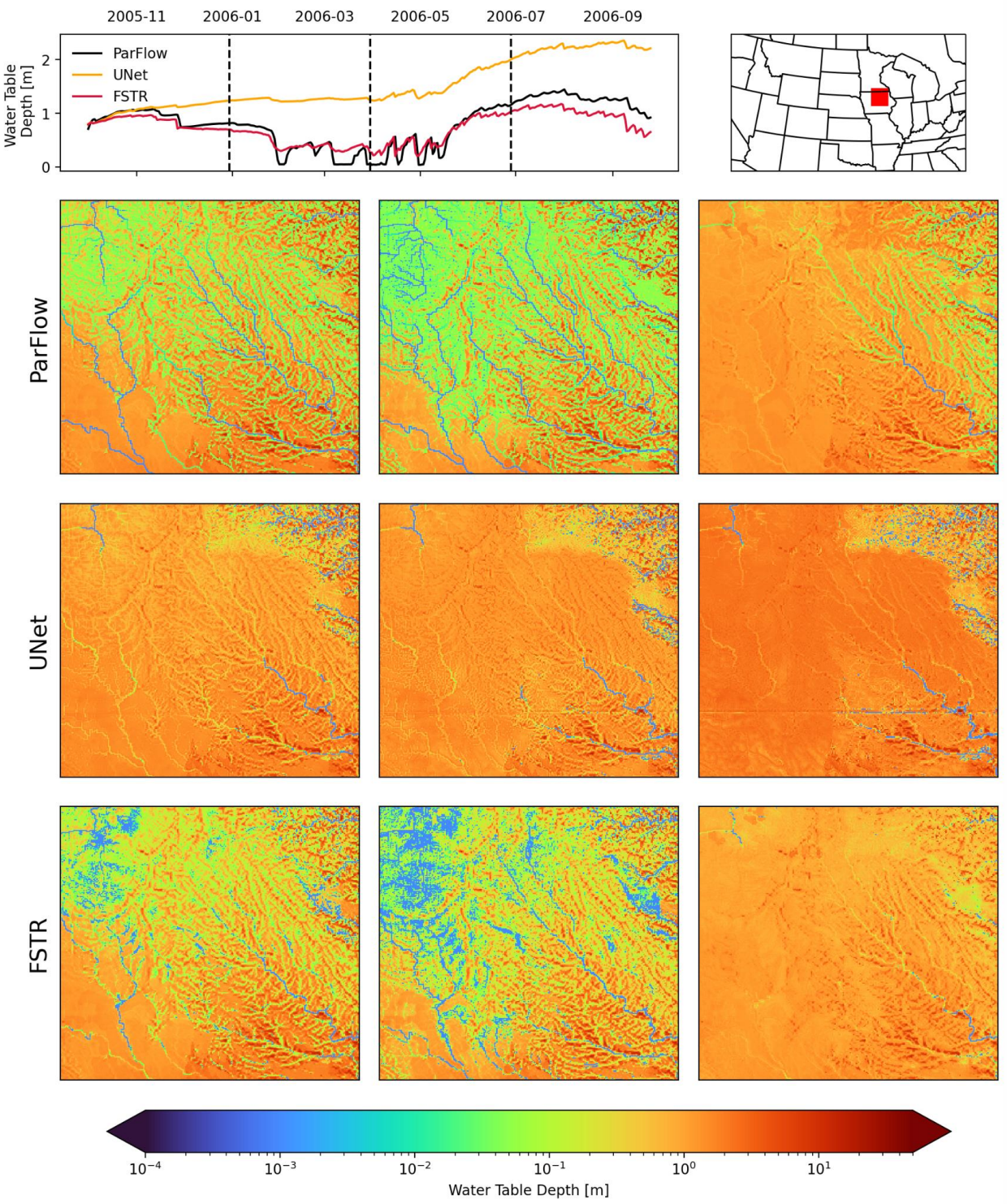

Figure 7. A zoom in of a region of the domain in the midwest highlighted in red on the upper right. Timeseries shown on the top are median values for the region, while the spatial plots correspond to time slices designated by dashed vertical lines in the timeseries.

Similarly, to figure 7 we show a zoom in to the southwestern portion of the domain in figure 8. This region is much more arid and has deeper water table depths on average. Here we see that

both the UNet and FSTR models can capture the long-term dynamics of the region and maintain spatial coherence over the simulation period as well. However, we do see that the FSTR model is better able to capture the fine-scale structure where shallow water table depths and rivers form in the basins.

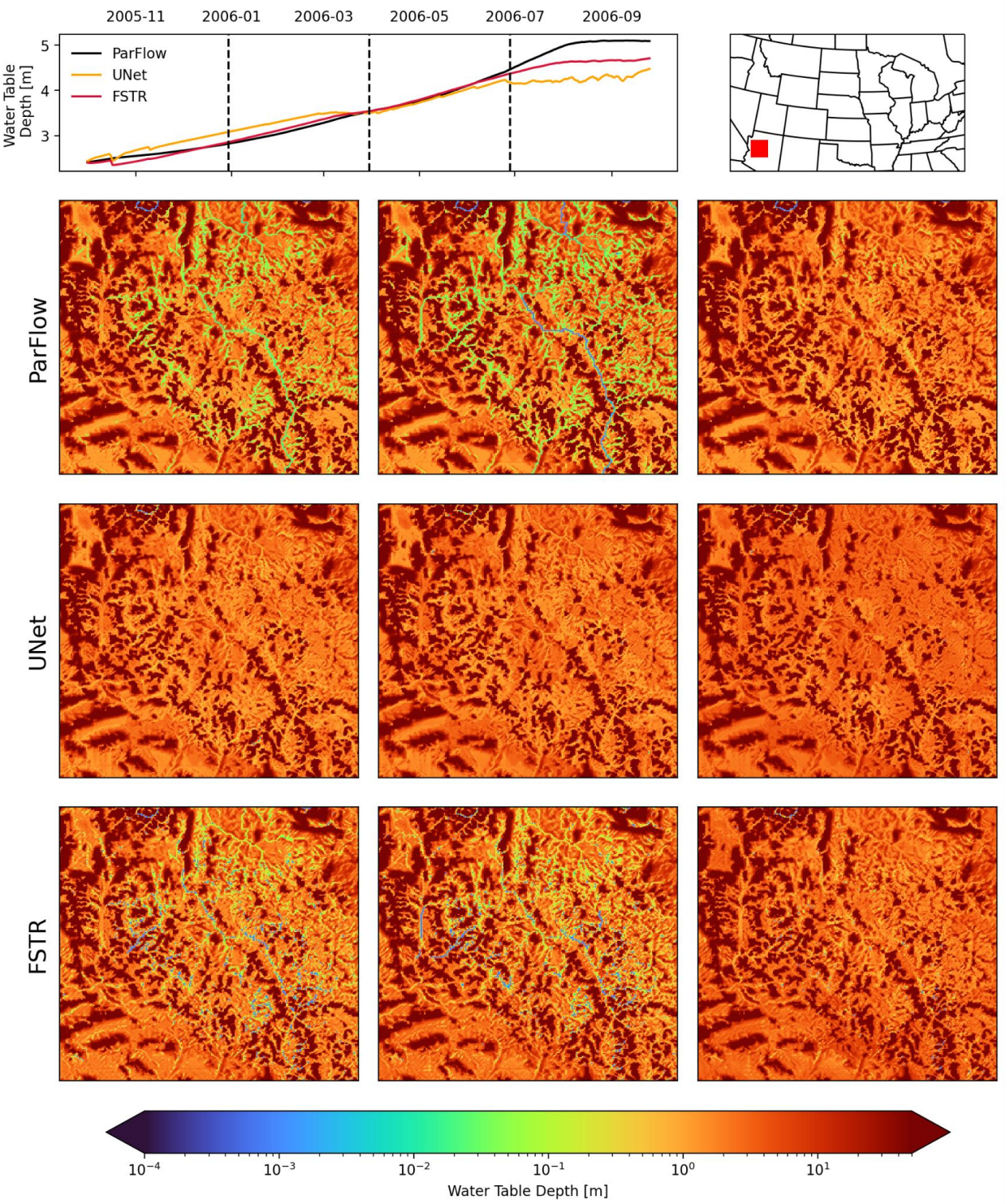

Figure 8. A zoom in of a region of the domain in the southwest highlighted in red on the upper right. Timeseries shown on the top are median values for the region, while the spatial plots correspond to time slices designated by dashed vertical lines in the timeseries.

In figure 7 we saw that one of the main deficiencies in the UNet output is drift in the longer-term dynamics, which we found in other regions as well. To better understand what drives the accumulation of errors over time we looked at how the UNet and FSTR models respond to precipitation events. We accomplished this by selecting grid cells at varying levels of precipitation and then comparing the response of the surface level pressure heads before and after the storm events (Figure 9). Overall, we found that both model architectures show similar sensitivities to precipitation events, even in the extreme events. The FSTR model does show a closer match to the response of ParFlow at the 90th percentile and above, which this is likely contributes to the improved performance compared to the UNet. The ability of the emulators to respond accurately across a wide range of events is a promising result indicating potential ability to use them in evaluating the impacts of extreme precipitation events.

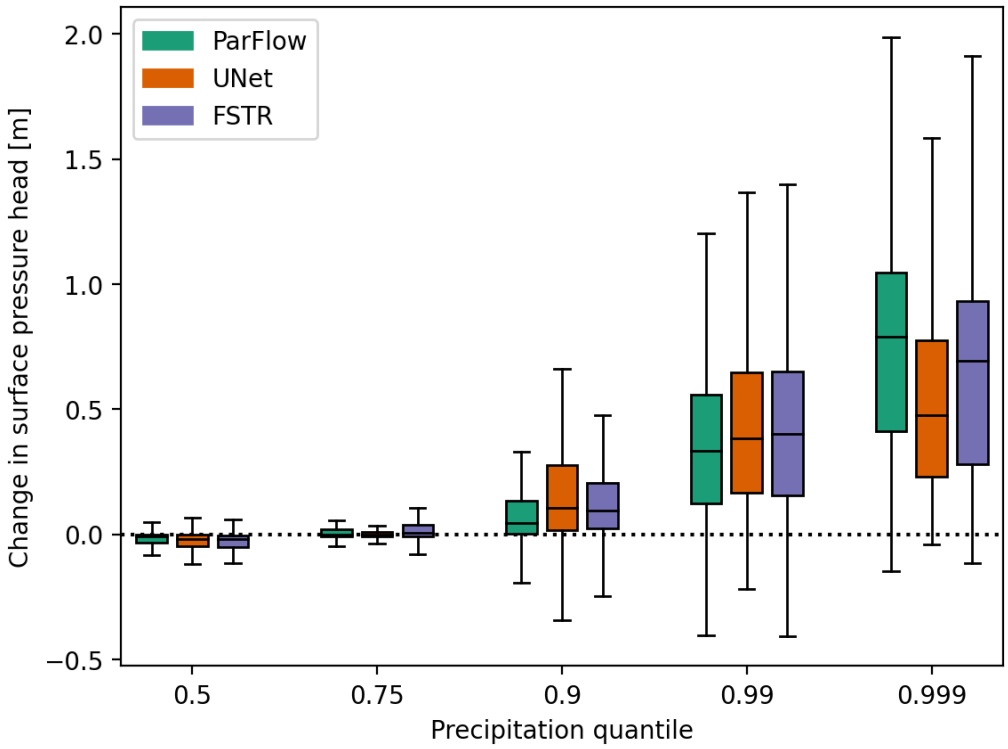

Figure 9. Comparison of the response of the surface layer pressure head to different precipitation inputs.

4 Discussion and future work

In this study we chose to focus entirely on the subsurface and treat the snow and ET as inputs to the model. We chose this because ParFlow is substantially more computationally expensive to run than the land surface model CLM, which provides the simulation of the snow, evaporation, and vegetation processes. We plan to incorporate sub-modules that simulate these processes and can be connected to the subsurface component in another study. These processes could also be

added to the FSTR model, but this would require the ability to estimate their gridded initial conditions of snowpack and ET to be used in this framework. There are also many additional experiments that could be performed using the framework that we have developed here to further improve the capabilities of the emulators.

One opportunity for further exploration is to modify the training routines that we use to produce more suitable emulators for specific time or spatial scales. Our aim of enabling subseasonal to seasonal groundwater prediction was our initial choice for the daily timestep that our models operate at, but we showed they can be rolled out to annual scales with little degradation in performance with our FSTR model. It may even be possible to combine training methodologies into a multi-stage model, similar to the approach taken by the FuXi weather forecasting architecture (Chen et al., 2023).

Another aspect that we have not yet explored is in the translation from the pressure heads to water table depth and soil moisture during training time. The equations we used are implemented in the same way as they are in ParFlow, but are written as PyTorch layers which, in theory, could be appended to our model structures and trained on directly. It is possible that this could yield better predictions for these quantities but may sacrifice the fidelity of the more fundamental pressure heads. In future work we may explore this along with multi-objective training to capture a wider range of conditions. There are some technical challenges that remain before such experiments can be performed, though we believe this will be a necessary step in order to successfully emulate the overland flow component of ParFlow as evidenced by the overall difficulty of even the FSTR model to accurately reproduce the stream network via the surface soil moisture.

One other potential limitation of our current emulation method is that the models are dependent on the soil and vegetation classifications that were used in the original simulations. Work is ongoing to develop strategies build ParFlow ensembles with varied parameters which would provide the basis for model training to be varied in more robust ways. We hope that this will provide an even stronger model which can be used to optimize the subsurface parameter values via simulation-based inference, in turn giving better simulation results as compared to observations.

Going beyond improving the emulation of ParFlow simulations, this work enables new applications for large-scale simulations. For example, having a fast and stable emulator at the seasonal timescale allows for ensemble predictions, uncertainty analysis, and coupling to land surface and atmospheric models. If future work shows that such emulation approaches can be stable over the annual-to-decadal periods this will also enable more robust studies on the effects of climate change on groundwater. Additionally, our FSTR architecture should be suitable for modeling in other domains where external forcings act on the state of the system, particularly other areas of hydrologic and land surface modeling. Additionally, we believe that the action-conditioning portion of the architecture provides a natural coupling point to other model types.

5 Conclusions

In this study we developed a deep-learning model architecture that is a viable approach to full-system emulation of a complex spatiotemporal groundwater model at high resolution. Our results show that off-the-shelf neural network architectures like the ResNet and UNet do not have the predictive capability needed for four-dimensional hydrologic simulations. Our FSTR architecture is more accurate at reconstructing groundwater dynamics and is stable over long rollout times. Our emulators exhibit much lower computational cost compared to the original

simulations. This opens up many new opportunities to use emulated results for ensemble forecasting and model calibration through simulation-based inference.

The FSTR model architecture developed in this study not only shows good overall performance at simulating continental-scale pressure heads, water table depth, and soil moisture but is also fast and scalable to run. As a side-benefit the model architecture is easy to conceptually understand because the model inputs directly correspond to the input data types used in most distributed hydrologic models. Initial conditions are used to set the model up for simulation, static physical parameters are used to modulate the time evolution of the system, and meteorologic forcings act on the internal state of the system. Currently each of these subsystems are processed via convolutional layers before being fed into the core AC-ST-LSTM model, but fundamentally could be any neural network component.

The use of deep learning for geophysical modeling is still a rapidly developing field where most applications for land and subsurface modeling are either point-scale timeseries models or static spatially distributed models. In this work we have demonstrated a modeling approach that is not only fully spatiotemporal, but also can be used to represent multiple processes simultaneously. We consider this work to be a step in moving towards more advanced representations of the subsurface, which is currently lacking compared to capabilities for weather and atmospheric predictions.

Acknowledgments

This research was funded by the US National Science Foundation Convergence Accelerator Program, Grant No. CA-2040542. The authors would also like to thank Bill Hasling, Amy Defnet, Amy Johnson, and Will Lytle for their assistance in developing software to deploy our models to https://hydrogen.princeton.edu/.

Open Research

The code to train the emulators and produce the figures associated with this manuscript can be found at: https://github.com/HydroFrame-ML/hydrogen-emulator-configurable
Additionally, the raw datasets used to train the emulators can be accessed via this python package: https://github.com/hydroframe/hf_hydrodata

References

Astrid, P., Weiland, S., Willcox, K., & Backx, T. (2008). Missing Point Estimation in Models Described by Proper Orthogonal Decomposition. *IEEE Transactions on Automatic Control*, *53*(10), 2237–2251. https://doi.org/10.1109/TAC.2008.2006102

Bennett, A., & Nijssen, B. (2021). Deep Learned Process Parameterizations Provide Better Representations of Turbulent Heat Fluxes in Hydrologic Models. *Water Resources Research*, *57*(5), e2020WR029328. https://doi.org/10.1029/2020WR029328

Blöschl, G., Bierkens, M. F. P., Chambel, A., Cudennec, C., Destouni, G., Fiori, A., et al. (2019). Twenty-three unsolved problems in hydrology (UPH) – a community perspective. *Hydrological Sciences Journal*, *64*(10), 1141–1158. https://doi.org/10.1080/02626667.2019.1620507

Brenowitz, N. D., Beucler, T., Pritchard, M., & Bretherton, C. S. (2020). Interpreting and Stabilizing Machine-Learning Parametrizations of Convection. *Journal of the Atmospheric Sciences*, *77*(12), 4357–4375. https://doi.org/10.1175/JAS-D-20-0082.1

Chen, L., Zhong, X., Zhang, F., Cheng, Y., Xu, Y., Qi, Y., & Li, H. (2023, June 27). FuXi: A cascade machine learning forecasting system for 15-day global weather forecast. arXiv. Retrieved from http://arxiv.org/abs/2306.12873

Cheng, Y., Musselman, K. N., Swenson, S., Lawrence, D., Hamman, J., Dagon, K., et al. (2023). Moving Land Models Toward More Actionable Science: A Novel Application of the Community Terrestrial Systems Model Across Alaska and the Yukon River Basin. *Water Resources Research*, *59*(1), e2022WR032204. https://doi.org/10.1029/2022WR032204

Condon, L. E., & Maxwell, R. M. (2019). Simulating the sensitivity of evapotranspiration and streamflow to large-scale groundwater depletion. *Science Advances*, *5*(6), eaav4574. https://doi.org/10.1126/sciadv.aav4574

De la Fuente, L. A., Gupta, H. V., & Condon, L. E. (2023). Toward a Multi-Representational Approach to Prediction and Understanding, in Support of Discovery in Hydrology. *Water Resources Research*, *59*(1), e2021WR031548. https://doi.org/10.1029/2021WR031548

Doury, A., Somot, S., Gadat, S., Ribes, A., & Corre, L. (2023). Regional climate model emulator based on deep learning: concept and first evaluation of a novel hybrid downscaling approach. *Climate Dynamics*, *60*(5), 1751–1779. https://doi.org/10.1007/s00382-022-06343-9

Fan, Y. (2015). Groundwater in the Earth's critical zone: Relevance to large-scale patterns and processes. *Water Resources Research*, *51*(5), 3052–3069. https://doi.org/10.1002/2015WR017037

Gauch, M., Kratzert, F., Klotz, D., Nearing, G., Lin, J., & Hochreiter, S. (2021). Rainfall–runoff prediction at multiple timescales with a single Long Short-Term Memory network. *Hydrology and Earth System Sciences*, *25*(4), 2045–2062. https://doi.org/10.5194/hess-25-2045-2021

van Genuchten, M. Th. (1980). A Closed-form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. *Soil Science Society of America Journal*, *44*(5), 892–898. https://doi.org/10.2136/sssaj1980.03615995004400050002x

Haber, E., & Ruthotto, L. (2018). Stable Architectures for Deep Neural Networks. *Inverse Problems*, *34*(1), 014004. https://doi.org/10.1088/1361-6420/aa9a90

He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). Deep Residual Learning for Image Recognition. arXiv. Retrieved from http://arxiv.org/abs/1512.03385

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260. https://doi.org/10.1126/science.aaa8415

Kasim, M. F., Watson-Parris, D., Deaconu, L., Oliver, S., Hatfield, P., Froula, D. H., et al. (2021). Building high accuracy emulators for scientific simulations with deep neural architecture search. *Machine Learning: Science and Technology*, *3*(1), 015013. https://doi.org/10.1088/2632-2153/ac3ffa

Keisler, R. (2022, February 15). Forecasting Global Weather with Graph Neural Networks. arXiv. Retrieved from http://arxiv.org/abs/2202.07575

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in Vision: A Survey. *ACM Computing Surveys*, *54*(10s), 200:1-200:41. https://doi.org/10.1145/3505244

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, *118*(21), e2101784118. https://doi.org/10.1073/pnas.2101784118

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., & Herrnegger, M. (2018). Rainfall-Runoff modelling using Long-Short-Term-Memory (LSTM) networks. *Hydrology and Earth System Sciences Discussions*, 1–26. https://doi.org/10.5194/hess-2018-247

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Pritzel, A., et al. (2022, December 24). GraphCast: Learning skillful medium-range global weather forecasting. arXiv. Retrieved from http://arxiv.org/abs/2212.12794

Leonarduzzi, E., Tran, H., Bansal, V., Hull, R., De La Fuente, L., Bearup, L., et al. (2022). Training machine learning with physics-based simulations to predict 2D soil moisture fields in changing climate. *Frontiers in Water*, *4*. Retrieved from https://www.frontiersin.org/articles/10.3389/frwa.2022.927113

Maxwell, R. M., Condon, L. E., & Kollet, S. J. (2015). A high-resolution simulation of groundwater and surface water over most of the continental US with the integrated hydrologic model ParFlow v3. *Geoscientific Model Development*, *8*(3), 923–937. https://doi.org/10.5194/gmd-8-923-2015

Maxwell, Reed M., & Condon, L. E. (2016). Connections between groundwater flow and transpiration partitioning. *Science*, *353*(6297), 377–380. https://doi.org/10.1126/science.aaf7891

Maxwell, Reed M., Condon, L. E., & Melchior, P. (2021). A Physics-Informed, Machine Learning Emulator of a 2D Surface Water Model: What Temporal Networks and Simulation-Based Inference Can Help Us Learn about Hydrologic Processes. *Water*, *13*(24), 3633. https://doi.org/10.3390/w13243633

O'Neill, M. M. F., Tijerina, D. T., Condon, L. E., & Maxwell, R. M. (2021). Assessment of the ParFlow–CLM CONUS 1.0 integrated hydrologic model: evaluation of hyper-resolution water balance components across the contiguous United States. *Geoscientific Model Development*, *14*(12), 7223–7254. https://doi.org/10.5194/gmd-14-7223-2021

Ott, K., Katiyar, P., Hennig, P., & Tiemann, M. (2020). ResNet After All: Neural ODEs and Their Numerical Solution. Presented at the International Conference on Learning Representations. Retrieved from https://openreview.net/forum?id=HxzSxSxLOJZ

Razavi, S., Tolson, B. A., & Burn, D. H. (2012). Review of surrogate modeling in water resources. *Water Resources Research*, *48*(7). https://doi.org/10.1029/2011WR011527

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., & Prabhat. (2019). Deep learning and process understanding for data-driven Earth system science. *Nature*, *566*(7743), 195–204. https://doi.org/10.1038/s41586-019-0912-1

Richards, L. A. (1931). CAPILLARY CONDUCTION OF LIQUIDS THROUGH POROUS MEDIUMS. *Physics*, *1*(5), 318–333. https://doi.org/10.1063/1.1745010

Ronneberger, O., Fischer, P., & Brox, T. (2015, May 18). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv. Retrieved from http://arxiv.org/abs/1505.04597

Serifi, A., Günther, T., & Ban, N. (2021). Spatio-Temporal Downscaling of Climate Data Using Convolutional and Error-Predicting Neural Networks. *Frontiers in Climate*, *3*. Retrieved from https://www.frontiersin.org/articles/10.3389/fclim.2021.656479

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W., & Woo, W. (2015). Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *arXiv:1506.04214 [Cs]*. Retrieved from http://arxiv.org/abs/1506.04214

Tran, H., Leonarduzzi, E., De la Fuente, L., Hull, R. B., Bansal, V., Chennault, C., et al. (2021). Development of a Deep Learning Emulator for a Distributed Groundwater–Surface Water Model: ParFlow-ML. *Water*, *13*(23), 3393. https://doi.org/10.3390/w13233393

Wang, C., Duan, Q., Gong, W., Ye, A., Di, Z., & Miao, C. (2014). An evaluation of adaptive surrogate modeling based optimization with two benchmark problems. *Environmental Modelling & Software*, *60*, 167–179. https://doi.org/10.1016/j.envsoft.2014.05.026

Wang, Y., Long, M., Wang, J., Gao, Z., & Yu, P. S. (2017). PredRNN: Recurrent Neural Networks for Predictive Learning using Spatiotemporal LSTMs. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc. Retrieved from https://papers.nips.cc/paper_files/paper/2017/hash/e5f6ad6ce374177eef023bf5d0c018b6-Abstract.html

Xu, G., Zhu, X., Fu, D., Dong, J., & Xiao, X. (2017). Automatic land cover classification of geo-tagged field photos by deep learning. *Environmental Modelling & Software*, *91*, 127–134. https://doi.org/10.1016/j.envsoft.2017.02.004

Yuan, X., Shi, J., & Gu, L. (2021). A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Systems with Applications*, *169*, 114417. https://doi.org/10.1016/j.eswa.2020.114417