https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

USING SIMULATION-BASED INFERENCE TO DETERMINE THE PARAMETERS OF AN INTEGRATED HYDROLOGIC MODEL: A CASE STUDY FROM THE UPPER COLORADO RIVER BASIN

Robert Hull[1], Elena Leonarduzzi[2], Luis De La Fuente[1], Hoang Viet Tran[3,4], Andrew Bennett[1], Peter Melchior[5,6], Reed M. Maxwell[2,3,7], Laura E. Condon[1]

[1] Hydrology and Atmospheric Sciences, University of Arizona, Tucson, AZ, USA
[2] High Meadows Environmental Institute, Princeton University, Princeton, NJ, USA
[3] Civil & Environmental Engineering, Princeton University, Princeton, NJ, USA
[4] Atmospheric Sciences & Global Change Division, Pacific Northwest National Laboratory, Richland, WA, USA
[5] Center for Statistics and Machine Learning, Princeton University, Princeton, NJ, USA
[6] Department of Astrophysical Sciences, Princeton University, Princeton, NJ, USA
[7] Integrated GroundWater Modeling Center, Princeton University, Princeton, NJ, USA
Correspondence to: Robert Hull (roberthull@email.arizona.edu)

Abstract. High-resolution, spatially distributed process-based models are a well-established tool to explore complex watershed processes and how they may evolve under a changing climate. While these models are powerful, calibrating them can be difficult because they are costly to run and have many unknown parameters. To solve this problem, we need a state-of-the-art, data- driven approach to model calibration that can scale to the high-compute, high-dimensional hydrologic simulators that drive innovation in our field today. Simulation- Based Inference (SBI) uses deep learning methods to learn a probability distribution of simulation parameters by comparing simulator outputs to observed data. The inferred parameters can then be used to run calibrated model simulations. This approach has pushed boundaries in simulator-intensive research from cosmology, particle physics, and neuroscience, but is less familiar to hydrology. The goal of this paper is to introduce SBI to the field of watershed modeling by benchmarking and exploring its performance in a set of synthetic experiments. We use SBI to infer two common physical parameters of hydrologic process-based models, Manning's Coefficient and Hydraulic Conductivity, in a snowmelt-dominated catchment in Colorado, USA. We employ a process-based simulator (ParFlow), streamflow observations, and several deep learning components to confront two recalcitrant issues related to calibrating watershed models: 1) the high cost of running enough simulations to do a calibration; 2) finding 'correct' parameters when our understanding of the system is uncertain or incomplete. In a series of experiments, we demonstrate the power of SBI to conduct rapid and precise parameter inference for model calibration. The workflow we present is general-purpose, and we discuss how this can be adapted to other hydrology-related problems.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

1 Introduction

Robust hydrologic tools are necessary to understand and predict watershed behaviors in a changing climate (Condon, 2022). This need is underscored by long-term drought in the American West (Williams et al., 2022), which has led to the withering of water supplies from the Colorado River (Santos and Patno, 2022), increased groundwater pumping (Castle et al., 2014), and uncertainty about what is next (Tenney, 2022). Hydrologic simulators that represent processes and connections within the hydrologic cycle (Paniconi and Putti, 2015) is one such toolset. These 'process-based' (PB) simulators explicitly represent hydrologic states and fluxes at multiple scales based upon physics first-principles (Fatichi et al., 2016). Watershed scientists can use PB simulators to predict the behavior of watershed snowpack, soil moisture, and streamflow in a changed future because they encode fundamental processes, and not just historical data (Maxwell et al., 2021).

The behaviors and skills of these PB watershed simulators strongly depend on spatially varying parameters (Tsai et al., 2021). Parameters represent physical properties of the hydrologic system, such as the roughness of the land surface (i.e., Manning's Coefficient, M) or the water-transmitting properties of the subsurface (i.e., Hydraulic Conductivity, K). There are many approaches to parameter determination in hydrology (Hunt et al., 2007): sensitivity analysis (Bastidas et al., 1999); parameter estimation (PEST) based on the Gauss-Marquardt-Levenberg method (White et al., 2020); techniques such as Approximate Bayesian Computation (Vrugt and Sadegh, 2013) to constrain the uncertainty of inferred parameters; and more recently neural network-based approaches to ‘parameter learning’ (Tsai et al., 2021). The large current body of work underscores that there is “no obvious formulation of [parameter determination] that previous generations of modelers have overlooked” (Hunt et al., 2007). However, the question of how best to infer parameters for PB simulators is not settled.

Parameter determination remains a challenge with watershed PB simulators, and an impediment to robust, physics-informed hydrologic predictions. Two related and ongoing difficulties are: (1) How to deal with the problem of 'equifinality' - wherein multiple simulator configurations might be possible. Many have argued it may be preferable to simulate distributions of hydrologic variables and the underlying parameters that give rise to them (Vrugt and Sadegh, 2013). (2) How to generate PB simulations quickly to explore many such configurations. Large-scale, high-resolution PB simulations can require massively parallel, high-performance computing (e.g., Maxwell et al., 2015) making multiple runs challenging.

Deep learning (DL) may provide new opportunities vis-à-vis parameter determination. In DL, behaviors are learned from data, as opposed to PB approaches, which derive behavior from established theory. The Earth Sciences have recently seen greater adoption of DL approaches (Wilkinson et al., 2016), for example in streamflow prediction (Kratzert et al., 2018). However, DL methods are not widely used in watershed prediction due to the “inadequacy of available data in representing the complex spaces of hypotheses” (Karpatne et al., 2017), including the impacts of climate change on watershed behavior. Recently, there has been a push for methods that can incorporate process understanding into DL models (e.g., Zhao et al.,

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

2019; Jiang et al., 2020). Still, studies are rare that employ DL to improve PB simulator performance by aiding in the hunt for better parameters[1] (e.g., Tsai et al., 2021).

This research uses a DL-informed workflow to PB parameter determination called surrogate-derived Simulation-Based Inference (SBI). This workflow explicitly addresses the challenges to PB parameter determination: (1) SBI determines distributions of parameters. In SBI, a neural network learns the conditional density of PB simulator parameters and hydrologic variables. When combined with an observation, this generates a probability-weighted distribution of plausible parameter configurations (Cranmer et al., 2020). (2) DL 'surrogates' for PB simulators speed up the exploration of parameter space. Surrogate simulators are neural networks that emulate the complex interdependence of variables, inputs, and parameters encoded in PB simulators, such as watershed simulators (Maxwell et al., 2021; Tran et al., 2021). Once trained, surrogate simulators can preserve fidelity to the PB simulator and run at a fraction of the cost. In a nutshell, our approach uses one neural network (the 'surrogate') to quickly generate thousands of watershed simulations that are utilized by another neural network (via SBI) to infer a distribution of PB parameter values conditioned on watershed observations. Surrogate-derived SBI is further described in the background section.

While SBI for parameter determination has shown promise in particle physics (Cranmer et al., 2020), cosmology (Alsing et al., 2019), and neural dynamics (Lueckmann et al., 2017), the applications in hydrology have been limited (Maxwell et al., 2021). As such, our field is somewhat behind the 'renaissance' in simulation-based inference (Cranmer et al., 2020) that has catalyzed scientific inquiry in other simulator-intensive fields. We believe this research is among the first to apply contemporary simulation-based inference approaches to watershed modeling. We demonstrate that SBI can generate good estimates of the spatially distributed parameters of a hydrologic simulator where calibration using conventional methods might struggle. Our work is a framework to tackle harder inference problems in watershed modeling, and other domains of the Earth Sciences where complex PB simulators are used.

Here, we determine the physical parameters of a headwater subbasin of the Upper Colorado River Basin by calibrating a PB watershed simulator to historical streamflow observations. We utilize SBI in tandem with a Long Short-Term Memory (LSTM) surrogate for the PB simulator ParFlow to rapidly generate probable configurations of Hydraulic Conductivity (K) and Manning's Coefficient (M). Furthermore, we use the inferred distribution of parameters to generate streamflow predictions. We explore the influence of observed data on parameter inference with a set of synthetic experiments that systematically vary the degree of uncertainty associated with how mock observations relate to the simulator.

2 Background of inference-based approaches to hydrologic parameter determination

This section gives background about methods for parameter determination that have been used in watershed modeling and elsewhere. We provide context relevant to understanding the "point of convergence" (Cranmer et al., 2020) we call

[1] We make a distinction between the parameters of PB simulators and the parameters embedded in neural networks, which are optimized during training by backpropagation. In this report, we almost-exclusively refer to the parameters of PB simulators even as we discuss the capacity of neural networks to learn and represent them.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences Discussions

simulation-based inference (SBI), and how it is similar to and different from some other approaches to inference. We start with a general overview of inference in hydrology; next, we discuss some traditional approaches to statistical inference and their limitations (section 2.1); we then introduce what sets SBI apart from these traditional approaches (section 2.2); and we discuss the role of machine learning in SBI (section 2.3). Here we define 'inference' as using data (observations) and a simulator to describe some unobserved characteristic of the system we are interested in (Cranmer, Brehmer, and Louppe 2020; Wikle and Berliner 2007).

2.1 Likelihood-based inference

We follow the Bayesian paradigm of statistical inference to extract information from observations. The essence of this formulation of inference unfolds in three steps (Wikle and Berliner, 2007): 1. Formulate a 'full probability model', which emerges from the joint probability distribution of observable and unobservable parameters; 2. Infer the conditional distribution of the parameters given observed data; 3. Evaluate the fit of the simulator (given parameters inferred in step 2) and its ability to adequately characterize the process(es) of interest.

Traditionally, to tackle inference problems we apply Bayes' Theorem. For illustration, let θ denote unobserved parameters of interest (such as Hydraulic Conductivity); and let Y represent simulated or observed data of the variable of interest (such as streamflow). The joint probability $p(\theta, Y)$ can be factored into the conditional and marginal distribution by applying Bayes' Rule, such that we obtain:

$$p(\theta|Y) = \frac{p(Y|\theta)\,p(\theta)}{p(Y)} \quad (1)$$

Where,

- The *data distribution*, $p(Y|\theta)$, is the distribution of data given unobservable parameters. This distribution is referred to as a likelihood function when viewed as a function of θ for a fixed Y. The likelihood usually assumes a known functional form.
- The *prior distribution*, $p(\theta)$, is our *a priori* understanding of unobservable parameters. The prior often results from a choice made by the domain expert. For example, in watershed modeling the prior distribution arises from a belief about the possible structures and magnitudes of parameters (for example, hydraulic conductivity) in a study domain, as well as the probability that they could be observed.
- The *marginal distribution*, $p(Y)$, can be thought of as a normalizing constant or 'evidence'. In practice, this distribution is rarely computed as it contains no information about the parameters. As such, we do not include $P(Y)$ and instead work with the unnormalized density provided by Equation 2:

$$p(\theta|Y) \propto p(Y|\,\theta)\,p(\theta) \quad (2)$$

- The *posterior distribution*, $p(\theta|Y)$, which is the distribution of unobservable parameters given the data. The posterior is the primary goal of Bayesian inference; it is proportional to the product of our prior knowledge of parameters and the information provided in our observations.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

Inference conducted using a Bayesian paradigm has a long history in computational hydrology (Vrugt and Sadegh, 2013). However, applications have been somewhat limited due to challenges centering on the likelihood function. The likelihood function of "implicit" simulators (such as those used in watershed modeling) is often regarded as 'intractable' – i.e., its form cannot be evaluated (integrated), at least not in a computationally-feasible way (Cranmer et al., 2020). When inference needs to be done an adequate likelihood function is therefore often 'chosen' for mathematical convenience by those attempting to apply a Bayesian framework.

2.2 Likelihood-free Inference (simulation-based inference)

SBI, also sometimes referred to as likelihood- free inference, is a set of methods that attempt to overcome the intractability of the likelihood function by learning the form of the posterior (or likelihood) distribution directly from the behavior of the simulator itself (Tejero-Cantero et al., 2020). The classic approach is Approximate Bayesian Computation (ABC), which compares observed and simulated data, rejecting and accepting simulation results based on some distance measure (Fenicia et al., 2018; Vrugt and Sadegh, 2013; Weiss and von Haeseler, 1998). While this approach has been widely used, it suffers from a range of issues, including poor scaling to high-dimensional problems (resulting in the need for summary statistics), and uncertainty arising from the selection of a distance threshold (Alsing et al., 2019). Additionally, in traditional ABC it is necessary to restart the inference process as new data become available (Papamakarios and Murray, 2016), making it inefficient to evaluate large numbers of observations (Cranmer et al., 2020).

SBI methods predicated on density estimation enable an alternative that does not suffer from the same shortcomings of ABC. The density estimation approach aims to train a flexible density estimator of the posterior parameter distribution from a set of simulated data-parameter pairs (Alsing et al., 2019). Some of the key advantages of a density estimation approach over ABC: (a) it represents the posterior[2] distribution parametrically (as a trained neural network) that can be reused to evaluate new data as it comes available; (b) it drops the need for a distance threshold by targeting an exact approximation of the posterior; (c) it more efficiently uses simulations by adaptively focusing on the plausible parameter region (Papamakarios and Murray, 2016).

One general purpose workflow that we employ in this paper uses a neural density estimator to learn the distribution of streamflow data as a function of the physical parameters of the simulator and employs active learning algorithms to run simulations in the most relevant regions of parameter space (Alsing et al., 2019; Lueckmann et al., 2017). The SBI workflow is further described in Sect. 3.5, and the neural density estimator described in Sect. 3.6.

[2] We share the literature's tendency to use 'conditional' and 'posterior' density interchangeably; denotations of $p(\theta \mid Y = Y_{True})$, for the posterior density evaluated at an observation Y_{Obs}; and $p(\theta \mid Y)$, for conditional density representative of a large set of simulated $\{\theta, Y\}$, are used when possible to reduce ambiguity.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

2.3 The role of Machine Learning in SBI

Due to advances in the capacity of neural networks to learn complex relationships, we can learn high-dimensional probability distributions from data in a way that was hardly possible before (Cranmer et al., 2020). This has led to strong claims in other fields, including cosmology and computational neuroscience, regarding the potential of SBI to "shift the way observational [science] is done in practice" (Alsing et al., 2019). While our implementation is described in more detail throughout the methods section, we direct readers to the literature for a broader (Cranmer et al., 2020) and deeper (Papamakarios and Murray, 2016) understanding of density based SBI.

Learning the full conditional density $p(\theta|Y)$ requires many simulated parameter-data pairs: thousands (or hundreds of thousands) of forward simulations. This presents a challenge with some high-resolution PB simulators, where each forward simulation can take hours of compute time to run. Many have noted that deep-learned surrogate simulators can help; after an initial simulation and training phase, these simulators can be run forward very efficiently. "Surrogate-derived approaches benefit from imposing suitable inductive bias for a given problem" (Cranmer et al., 2020). In our case, this "inductive bias" is applied by learning the rainfall-runoff response of our PB domain using a Long Short-Term Memory (LSTM) model, a type of neural network that is suited for learning temporal patterns in data (Kratzert et al., 2018). The surrogate simulator is described in more detail in Sect. 3.3. Surrogate simulators can be used directly in the construction of viable posterior distributions of physical parameters and run at low-cost relative to the PB simulator.

It should be noted that inference is always done within the context of a simulator (Cranmer, 2022). As such, if a simulator is 'mis-specified' it will affect inference in undesirable ways, for example by leading to biased or erroneous parameter estimates. Simulator misspecification arises in the case when a simulator does not capture the behavior of the dynamical system, giving rise to mismatch between simulated and observed data (Cranmer et al., 2020).

3 Materials and Methods

This section describes our implementation of surrogate-derived SBI, and three experiments undertaken to test it. We first introduce those experiments, and the goals associated with them (Sect. 3.1). Then, we describe the domain of interest, the Taylor River watershed (Sect. 3.2). The rest of the methods subsections describe the components, implementation, and validation of SBI, as outlined in Table 1.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

Table 1. Outline of the components described in the methods section.

Section	Name	Description
3.1	Experiments	
3.2	Taylor River Basin	Domain of study
3.3	ParFlow	Process-Based simulator
3.4	Long-Short Term Memory (LSTM) Network	Surrogate simulator
3.5	Simulation-Based Inference (SBI)	Method for parameter inference
3.6	Conditional Density Estimator, $q_\phi(\theta \mid Y)$	Learns distribution of parameters
3.7	Posterior Predictive Check	From inferred parameters, make prediction
3.8	Evaluation Metrics	Assess performance of SBI

Figure 1 shows how the components of surrogate-derived SBI interrelate. In Fig. 1A, a small set of process-based simulations are generated by ParFlow. A LSTM neural network learns from these simulations to mimic the behavior of ParFlow, interpolating the relationship between climate forcings, watershed parameters M and K and output streamflow time series. The LSTM can be used as a ParFlow surrogate to quickly explore the streamflow response to different parameter configurations and forcing scenarios.

We leverage the efficiency of the surrogate to conduct SBI on parameters, as depicted by Fig. 1B. Our goal with SBI is to estimate probable values for the watershed parameters M and K given the occurrence of a particular streamflow observation. To that end, we randomly sample many (n=5000) parameter configurations from a prior distribution $p(\theta)$ and from the LSTM simulate an equivalent number of streamflow timeseries Y. This set of simulated parameter-data pairs is used to train a neural density estimator $q_\phi(\theta|Y)$, which is a model of the full conditional density of parameters given data $p(\theta|Y)$. Once trained, the neural density estimator is evaluated with a given observation to produce a distribution of parameters, the posterior distribution $p(\theta \mid Y = Y_{Obs})$, which represent our 'best guess' of what the parameters should be.

Finally, a predictive check (Fig. 1C) ensures that the parameter estimates generate a calibrated model. The simplest version of this check is to put the estimates of parameters from the previous step back into the LSTM, which generates a new ensemble of streamflow simulations. The simulations should resemble the observation closely if the simulator captures the behavior of the dynamical system well, and parameter inference was done correctly.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

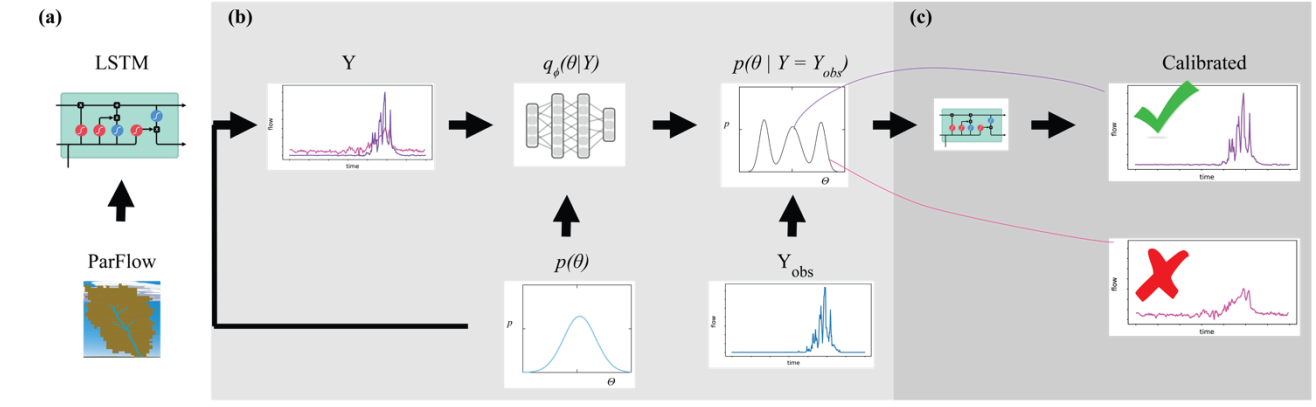


Figure 1. An illustration of surrogate-derived simulation-based inference (SBI). In subplot (a), a Long Short-Term Memory (LSTM) neural network learns watershed behavior from ParFlow, a process-based simulator. The implementation of SBI is shown in subplot (b), where the objective is to estimate watershed parameters θ given an observation Y_{obs}. This parameter estimate is formally known as the posterior parameter distribution $p(\theta \mid Y = Y_{Obs})$. We randomly sample many parameter configurations from a prior distribution $p(\theta)$ and from the LSTM simulate an equivalent number of streamflow timeseries Y. This set of simulated parameter-data pairs is used to train a neural density estimator $q_\phi(\theta|Y)$. Subplot (c) shows the posterior predictive check, which involves using the parameter estimate to (ideally) generate a calibrated model.

3.1 Experiments

We explore the performance of SBI in three experiments. These experiments test the success of SBI at accurately and precisely estimating parameters for simulator calibration. The subject of interest in these experiments is the potential mismatch between observations and the simulator. To test this, we vary the degree (and sources) of uncertainty associated with how observations relate to the simulator. Synthetic observations with known parameters are used to conduct the experiments because they are easier to benchmark. These experiments are further described below and in Table 2, and the results explored in Sect. 4:

1. 'Best Case': Find $p(\theta \mid Y = Y_{Obs\_LSTM})$. We use as observation the streamflow generated by a surrogate simulator (e.g., with a given combination of parameters) and use SBI to infer the parameters. As the simulator can (by definition) generate data identical to the observation, this experiment serves as a sanity check for our SBI workflow.
2. 'Tough Case': Find $p(\theta \mid Y = Y_{Obs\_ParFlow})$. We use a ParFlow simulation as observation and use SBI to infer the values of the parameters. This experiment tests whether the proposed framework, where SBI is carried out with the surrogate simulator, can be successful if there is a slight mismatch between observed and simulated data.
3. 'Boosted Case': Find more accurate $p(\theta \mid Y = Y_{Obs\_ParFlow})$. Building from the 'Tough Case', we again use a ParFlow simulation as observation but instead use an ensemble ('boosted') surrogate simulator to infer the known parameters. In this case we're testing whether the proposed framework can be made more robust to mismatch between observed and simulated data if a multi-model surrogate is used.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

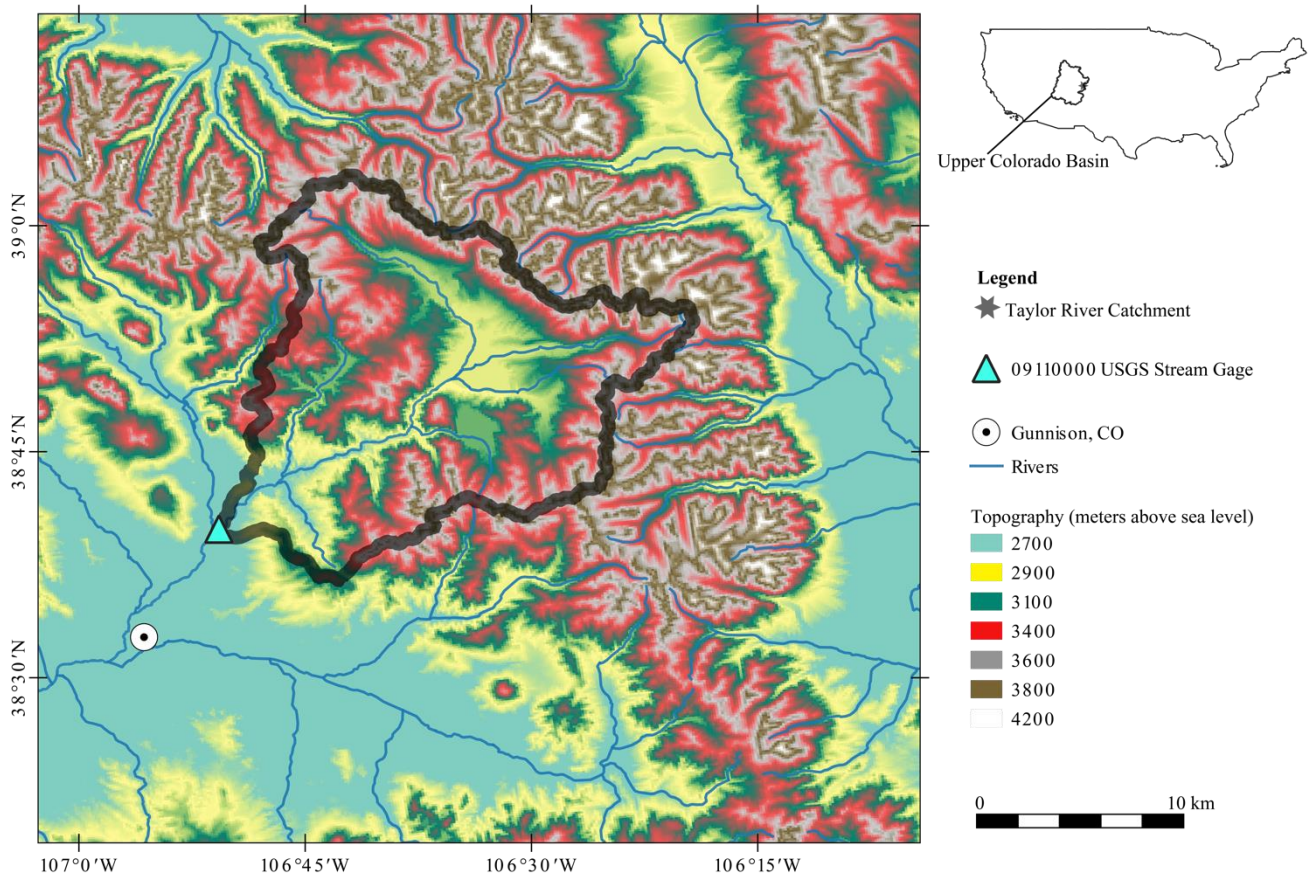
Table 2. The three experiments explore how the observation and simulator type affects the quality of parameter inference.

Experiment #	Name	Goal
1	*Best Case*	Infer parameters given no mismatch between observed and simulated data
2	*Tough Case*	Infer parameters given some mismatch between observed and simulated data
3	*Boosted Case*	Infer parameters given some mismatch between observed and multi-model simulated data.

3.2 Taylor River – The Domain

235 The physical area of study is the Taylor River headwater catchment located in the Upper Colorado River Basin (Figure 2). The Taylor is an important mountain headwater system for flood control and water supply in the Upper Colorado River Basin (Leonarduzzi et al., 2022). This catchment is at an elevation of between 2451 and 3958 meters above mean sea level and has a surface area of around. 1144 km2. This catchment is snowmelt-dominated in summer. The geographical extent of the watershed is defined by the USGS streamflow gage in Almont, Colorado (ID: 09110000) at the basin outlet. Over the full
240 period of record (1910 - 2022), the lowest average monthly discharges are recorded in January and February, with values of ca. 100 [cfs] (3 [m3/s]), after which there is a steady increase of discharge and generally wetness in the catchment up until June, when an average discharge of ca. 900 [cfs] (25 [m3/s]) is recorded. The prior information is provided as context; for the purposes of this analysis, synthetic data corresponding to the Almont gage location are used, as described in Sect. 3.1.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022



245 **Figure 2. Map showing the study domain Taylor River catchment near Almont, Colorado.**

3.3 The Process-Based Simulations (ParFlow)

We use the integrated hydrologic model ParFlow-CLM to simulate groundwater and surface water flow in our domain. ParFlow-CLM is designed to capture dynamically evolving interactions between groundwater, surface water and land
250 surface fluxes (Jones and Woodward, 2001; Maxwell and Kollet, 2006; Maxwell et al., 2015a). In the subsurface, variably saturated flow is solved using the mixed form of Richards Equation. Overland flow is solved by the kinematic wave approximation and Manning's equation. ParFlow is coupled to the Common Land Model (CLM). CLM is a land surface model which handles the surface water-energy balance (Maxwell and Miller, 2005; Kollet and Maxwell, 2008). It is thus well-suited to examine evolving watershed dynamics at the large scales (e.g., Maxwell et al., 2015b), as in the Taylor River Basin in
255 Colorado, USA.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

The Taylor catchment is represented by ParFlow at 1 kilometer resolution, with five vertical layers of total depth 102 meters (Leonarduzzi et al., 2022). As with Leonarduzzi et al., 2022, all the required input files - including soil properties, landcover, and meteorological forcings - are subset from Upper Colorado River Basin ParFlow-CLM simulations of Tran et al. 2022. The subsurface contains 23 separate soil and geological units.

We explore the sensitivity of streamflow to a large ensemble of different configurations of Manning's roughness coefficient (*M*), and hydraulic conductivity (*K*). For the baseline configuration of the model, *K* ranges between 6.16e-03 and 2.69e-01 [m/h] across the 23 spatial units; *M* is constant across the domain surface at 2.4e-06 [h/m^(1/3)]. An ensemble of 183 simulations is generated by systematically varying *M* and *K*. For *M* since the values are spatially constant it is easy to adjust this single value. *K* is spatially variable; therefore, we apply a single scaling factor to all three dimensions (Table A1). To make the distinction clear, we call these 'single' scalar representations K_s and M_s, respectively. The values K_s and M_s used in this study are shown in Table A2. A sensitivity analysis of streamflow to parameter configurations is shown in Fig. A1.

All simulations are run for a one-year period (8760 hours) using forcings from water year 1995 taken from Tran et al., 2020. Surface pressure outputs are converted to runoff using the overland flow utility built into ParFlow. This study focuses on runoff at the cell closest to USGS gage 09110000. We convert to cubic feet per second (cfs) for direct comparison to gaged data and rescale from 0 to 1. Streamflow simulations from ParFlow are relatively more sensitive to changes in K than M, as shown in Fig. A1. The purpose of generating this ParFlow ensemble is not to create the most diverse set of system realizations but provide a foundation from which to train the surrogate model and test performance of the simulation-based inference approach.

3.4 The Surrogate Simulator (LSTM)

We employ a Long Short-Term Memory (LSTM) network to learn from our process-based simulator ParFlow. LSTM networks are neural networks that are designed to learn temporal relationships (Rumelhart et al., 1986; Hochreiter and Schmidhuber, 1997). They have had some use for prediction in hydrology (Kratzert et al., 2018) to learn how sequences of previous meteorological forcing data affect streamflow at the basin outflow. In our study, an LSTM network learns the response of streamflow at gaged location 09110000 to forcings and parameters in the Taylor River basin, as defined by the ensemble of ParFlow simulations described in Sect. 3.3.

Throughout our experiments, we use an LSTM with 10 input features containing forcings *X* and parameters θ, and one output class containing streamflow *Y*. As in Kratzert et al. 2018, we employ a 'look-back' approach. For each sample, the LSTM ingests '*l*'=14 days of previous forcings weighted by scalar representations of ParFlow parameters (K_s, M_s) and returns streamflow the next day. More explicitly:

$$Y_{t+1} = LSTM(X_{t\rightarrow(t-1)}, K_s, M_s) \quad (6)$$

where Y_{t+1} is the streamflow the next day, *l* is the 'look back' which controls the length of the input sequence used for prediction, $X_{t\rightarrow(t-1)}$ are vectors containing sequences of forcing data from today (i.e., day *t*) back to day *t* minus *l* for each of the 8 forcing variables. K_s and M_s are scalar representations of the ParFlow parameters hydraulic conductivity (*K*) and

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

Manning's roughness (*M*). Since these values do not vary over time each is ingested as a vector repeated *'l'* times by the LSTM.

The relevant hyperparameters used to fit the LSTM surrogate are further defined in Table A1 and B1. Fig. B1A shows the distribution of train-validation and test sets across parameter space and the performance of the LSTM relative to ParFlow on a streamflow time series generated by a randomly selected test parameter set, θ_A. θ_A is used throughout the results section for benchmarking. The LSTM captures the general streamflow behavior quite well, but not quite perfectly (Figure B1B). We emphasize that the goal here is to produce a surrogate simulator adequate for the simulation-based inference of parameters K_s and M_s.

3.5 Implementation of Simulation-Based Inference

The goal of SBI is to infer appropriate values flexibly and efficiently for simulator parameters, given a particular observation. SBI is illustrated in Fig. 1B. Take θ to be a vector of parameters that control a simulator, and let Y be a vector of simulated data. The simulator implicitly defines a conditional probability $p(Y|\theta)$, which may very well be analytically intractable. $p(\theta)$ encodes our prior beliefs about parameters. We are interested in inferring the parameters θ given an observation Y_{Obs}, i.e., we would like to know the posterior probability density $p(\theta|Y=Y_{Obs})$, after Papamakarios and Murray (2016):

$$p(\theta|Y = Y_{Obs}) \propto p(Y = Y_{Obs}|\,\theta)\, p(\theta) \quad (3)$$

where θ contains K_s and M_s, and Y_{Obs} is an 'observed' streamflow timeseries. Y is a set of simulated outputs that are formally equivalent but not identical to the observation Y_{Obs}. Here, parameter-data pairs are simulated by a surrogate (Sect. 3.4) of ParFlow. Simulations are from just one forcing scenario to limit the degrees of freedom of parameter inference.

A conditional density estimator $q_\phi(\theta|Y)$ learns the posterior density directly from simulations generated by the surrogate. q_ϕ is a learnable model - often a neural network - that fits to $p(\theta \mid Y)$ and can be evaluated to approximate $p(\theta \mid Y = Y_{Obs})$. (See section 3.6 for details about q_ϕ). The procedure can be summarized as follows, after Papamakarios and Murray, 2016:

1. Propose a *prior* set of parameter vectors $\{\theta\}$, sampled from $p(\theta)$.
2. For each θ run the simulator to obtain the corresponding data vector, Y.
3. Train the neural density estimator $q_\phi(\theta|Y)$ on the simulated set from $\{\theta, Y\}$.
4. Evaluate q_ϕ at observed data vector Y_{Obs} to generate a *posterior* set of parameter vectors $\{\theta\}$ proportional to $p(\theta \mid Y = Y_{Obs})$.

The SBI workflow and architectures used in this study are derived from a python toolbox for simulation-based inference (Tejero-Cantero et al., 2020). We direct the reader to Papamakarios and Murray (2016) for a detailed description of the structure, training, and evaluation of a neural conditional density estimator for simulation-based inference. Others (Lueckmann

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

et al. 2017; Greenberg, Nonnenmacher, and Macke 2019) have built on this idea to introduce MCMC-like approaches to sequential learning of the posterior at observations to make inference more efficient. We employ a sequential learning procedure in our workflow, as described in Appendix C.2. The hyperparameters and architectures used in SBI are shown in Table C1.

3.6 Neural Conditional Density Estimators for SBI

The conditional density estimator $q_\phi(\theta|Y)$ is an essential ingredient of SBI. The neural conditional density estimator differs from conventional neural networks (such as the LSTM) in important ways: 1. It learns a conditional probability distribution, as opposed to a function; 2. It represents the 'inverse' model – the probability of parameters given data $p(\theta \mid Y)$ – as opposed to the dependency of data on parameters, which is encoded in 'forward' simulators like ParFlow and its surrogate, the LSTM. Once trained, the neural conditional density estimator is evaluated with an observation to infer a distribution of plausible parameters, the posterior distribution $p(\theta \mid Y = Y_{Obs})$ (Fig. 1B).

Conditional density estimators create a model for "a flexible family of conditional densities", parameterized by a vector of parameters (ϕ) (Papamakarios and Murray, 2016). Density estimator parameters are not to be confused with the simulator parameters, θ. The latter are the target of inference while the former parameterize the density-estimated posterior probability and must be learned or derived to conduct inference of simulation parameters. Deep neural networks provide new opportunities to learn ϕ for complex classes of densities, which gives rise to the term *neural* conditional density estimator.

Mixture Density Networks (MDNs) are an intuitive class of conditional density estimators capable of modeling any arbitrary conditional density (Bishop, 1994). They take the form of a mixture of k (not hydraulic conductivity, K) Gaussian components, as below.

$$q_\phi(\theta|Y) = \sum_k \alpha_k \mathcal{N}(\theta|m_k, S_k) \quad (4)$$

where the mixing coefficients (α), means (m), and covariance matrices (S) comprise the neural density parameterization, ϕ. They can be computed by a feedforward neural network.

Training an MDN is a maximum likelihood optimization problem (Bishop, 1994). Given a training set of N simulation parameters and data pairs, $\{\theta, Y\}$, the objective is to maximize the average log probability (or minimize the negative log probability) with respect to the parameters, ϕ.

$$\text{argmax}_\phi \frac{1}{\text{N}} \sum_n \log q_\phi(\theta_n|Y_n) \quad (5)$$

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

For a fuller description of the parameterization and training of neural density estimators, see the supplementary material in Papamakarios and Murray (2016) or the original write-up in Bishop (1994). This study uses a specialization of this family of neural networks called a Masked Autoencoder for Density Estimation, further described in Appendix C.1.

3.7 Posterior Predictive Check

A crucial diagnostic step in the SBI workflow is to check the ability of the simulator to characterize process(es) of interest after inference has been conducted (Cranmer et al., 2020). To be more explicit, this step checks that parameters from the inferred posterior $p(\theta \mid Y = Y_{Obs})$ can simulate streamflow data (Y) consistent with the observation (Y_{Obs}) when plugged back into the simulator. The simulated data should 'look similar' to the observation (Tejero-Cantero et al., 2020). Gabry et al. 2019 describe this type of model evaluation as a 'posterior predictive check'. This predictive check is represented by the Fig. 1C.

Here, we conduct posterior predictive checks by drawing a small number of parameter sets from our inferred parameter posterior density. In our workflow, the inferred posterior parameter density is represented by an array containing thousands (n=5000) of plausible parameter sets. The frequency of their occurrence is 'probability weighted', in the sense that there are very few occurrences of parameter sets in the 'tails' and many occurrences close to the mean, and improbable parameter sets do not exist at all. For our posterior predictive check, we randomly sample (n=50) parameter sets from this frequency-weighted parameter posterior array. We use these parameter samples to generate an ensemble of 'predicted' streamflow time series using the LSTM.

3.8 Evaluation Metrics

The performance of simulation-based inference is evaluated in terms of accuracy and precision. First, we evaluate performance with respect to the parameter posterior (the inferred parameters); and second with respect to the posterior predictive check (the ability to generate realistic data using the inferred parameters).

3.8.1 Evaluating the Posterior Parameter Density

Accuracy of parameter inference is evaluated using the Mahalanobis distance, $D_M(\theta_{True})$. Mahalanobis distance measures the distance between a point and a distribution of values after Maesschalck et al. (2000), such that:

$$D_M(\theta_{True}) = \sqrt{(\theta_{True} - \theta_\mu)^T \Sigma^{-1} (\theta_{True} - \theta_\mu)} \quad (6)$$

where θ_{True} is the set of observed or 'true' parameters; θ_μ is the mean of the posterior distribution $p(\theta \mid Y = Y_{Obs})$; and Σ is the covariance matrix of $p(\theta \mid Y = Y_{Obs})$. In essence, Mahalanobis distance measures how far off our parameter estimate is from the 'truth'. For this study values less than two are defined as acceptable (within ~two standard deviations); this threshold was identified via trial and error.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

Precision of parameter inference is evaluated in terms of the determinant of the covariance matrix of the inferred parameter posterior, |Σ|. The determinant can be interpreted geometrically as the 'volume' contained by the covariance matrix, and by extension the inferred parameter posterior distribution. Larger determinant values are less precise; smaller values more precise (4.3 Determinants and Volumes). In this study we define values less than 10^{-6} as acceptable, identified via trial and error.

3.8.2 Evaluating the Posterior Predictive Check

We evaluate the ability of the simulated ensemble of streamflow to adequately characterize the observed streamflow using the root mean squared error (RMSE) between each (n=50) simulated streamflow time series (Y) and the observed streamflow time series (Y_{Obs}). RMSE is calculated for each predication as the square root of the mean squared error, such that:

$$RMSE(Y) = \sqrt{\frac{\sum_{t=1}^{T}\left(Y_t - Y_{Obs_t}\right)^2}{T}} \quad (7)$$

where Y_{pred_t} is the simulator-predicted streamflow at time t, taken from Y_{pred}; Y_{Obs_t} is the observed or true streamflow at time t, taken from Y_{Obs}; T is the number of times (days) in the streamflow time series.

Accuracy of the simulator characterization of streamflow is the mean of the RMSE calculated for all n=50 Y relative to Y_{Obs} ($RMSE_{Ave}$). Precision of the simulator characterization of streamflow is assessed as the standard deviation of the RMSE calculated for all n=50 Y_{pred} relative to Y_{Obs} ($RMSE_{std}$). For both the mean and variance RMSE values less than 0.01 [scaled streamflow units], identified via trial and error, are acceptable.

4 Results

Here we present the outcomes of the three experiments described in Sect. 3.1. The first two experiments showcase inference problems that increase in difficulty from the easy *best case* (Sect. 4.1) to the hard *tough case* (Section 4.2). The final experiment offers a workaround by way of the *boosted case* (Sect. 4.3). The performance of the methods explored in the three experiments is first discussed in terms of one shared benchmark scenario. Then, we show the results of the three experiments on a larger shared set (n=18) of benchmark scenarios (Sect. 4.4).

4.1 Experiment 1 – *Best Case*

For the *Best Case* scenario, we attempt to infer the parameters of synthetic observation(s) taken from the trained surrogate simulator, such that $p(\theta \mid Y = Y_{Obs\_LSTM})$. We first infer the parameters of just one randomly selected streamflow observation, denoted with an 'A' ($Y_{Obs\_LSTM\_A}$). The set of 'benchmark' parameters (θ_A) used to generate the underlying simulation are approximately 0.60 for K_s, and 0.85 for M_s. θ_A is also our benchmark in parameter space for Experiments 2 and 3.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

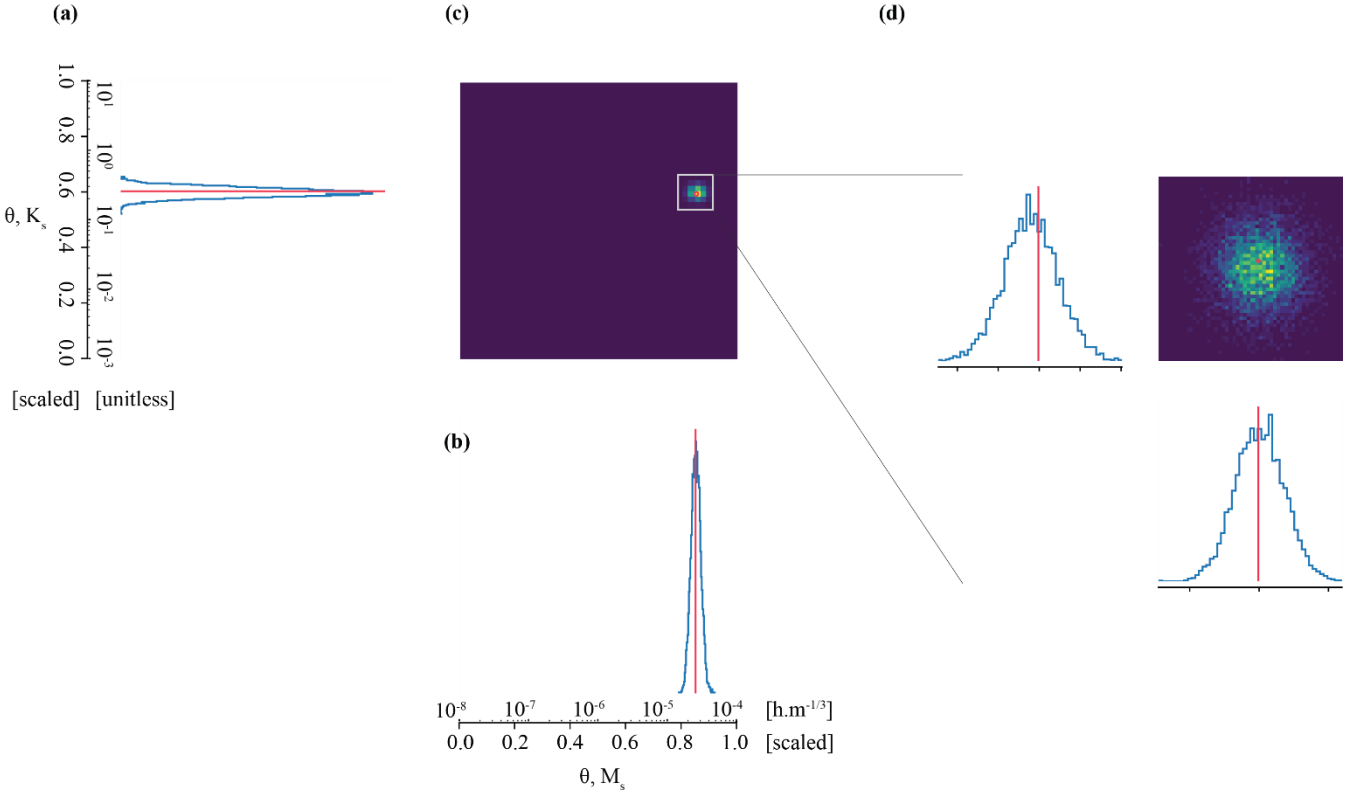


Figure 3. The parameter posterior estimate for observation $Y_{Obs\_LSTM\_A}$ closely matches the true parameter values in the *'best' case.* Subplots (a), (b) and (c) comprise a pair plot of posterior densities across the full possible parameter space; subplot (d) is zoomed in for detail. The posterior density of M_S (a) and K_S (b) are shown individually, and together (c). Axes are expressed in both the scale/transformed and unscaled units of the parameters. The 'true' parameters are denoted by the red line and circle, respectively.

We accurately and precisely estimate parameters for our benchmark case (Figure 3)**.** The pair plot approximates the posterior parameter density evaluated by the neural density estimator at the observation. In individual parameter space, narrower peaks (in blue) correspond with more confident and precise parameter estimates. In shared parameter space (c), zones of deep purple are effectively zones of no probability; zones of blue-green-yellow are zones of high probability. The benchmark parameters (i.e., the parameters used to generate the simulation) are denoted by the red line and circle, respectively. Accuracy is evaluated by the Mahalanobis Distance, which is 3e-01; thus, the 'true' parameter set can be thought of as less than one 'standard deviation' from the central tendency of the inferred distribution. Precision is estimated by taking the determinant of the covariance matrix. The determinant of the covariance matrix is 9e-08. This is well below our threshold of 1e-06 for sufficiently precise parameter inference.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

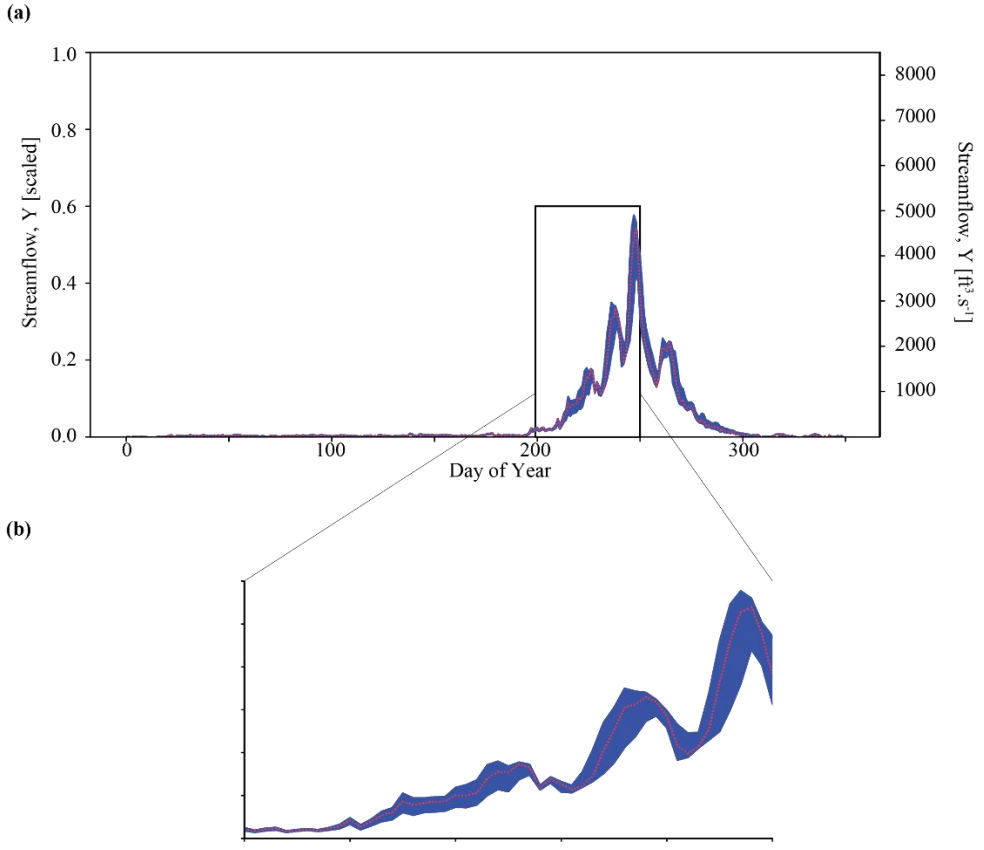


Figure 4. Results of the posterior predictive check on synthetic observation A_{Yobs} in Experiment 1 ('base' case). Subplots (a) shows streamflow simulations resulting from inference of $p(\theta|Y = A_{Yobs})$. The ensemble of predictions is bounded by blue, and observation in red. Blue lines represent time series of upper- and lower- streamflow values in this ensemble, and the red line represents the observation $Y_{Obs\_LSTM\_A}$. In subplot (b), we zoom into the area of greatest uncertainty between days 200 and 300, which correspond to the spring snow melt-off.

Taking this one step further we can use the inferred parameter distributions to generate an ensemble of streamflow simulations using the LSTM model and compare this to the observed streamflow (referred to as our posterior predictive check). As show in Figure 4a, the inferred parameters generate simulations results that characterize the observed streamflow observation reasonably well. Greater uncertainty exists around higher streamflow values over the course of the water year, as shown by the increasing width of the uncertainty envelope after day 200 (Figure 4B). Note that this is the time of year during which snow melt-off occurs in the Taylor River Basin. Mean and standard deviation of streamflow error are approximately 6e-03 and 4e-03 [scaled streamflow units], respectively.

4.1.1 Inference for many observations

In addition to conducting this analysis for one observation as described, an advantage of SBI is the low computational expense of evaluating new observations. Simulations from the process-based simulations (i.e., ParFlow) are slow and scale

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

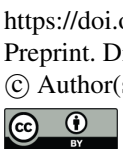

Hydrology and Earth System Sciences
Discussions

linearly with the number of simulations. It takes ~10^5 times longer to generate a ParFlow simulation (1680 seconds) than to evaluate one observation Y_{Obs} using a trained neural density estimator (0.045 seconds). Put another way, after an upfront sunk cost to learn the distributions, we can evaluate new observations, Y_{Obs}, practically for free. Many other techniques to parameter determination are not 'amortized' in this way (Cranmer et al., 2020). For example, Approximate Bayesian Computation (ABC) requires restarting most steps in the inference process when new data comes available (Vrugt and Sadegh, 2013). This property of SBI can be handy in domains where the system structure (parameters) stays the same, but new observations come available all the time - as can be the case in watershed hydrology. In Appendix D, we extend Experiment 1 to evaluate the posterior parameter density for many synthetic observations ($Y_{Obs\_LSTM\_i}$) quickly and effectively.

4.2 Experiment 2 – Tough Case

Experiment 2 is our *tough case*. We attempt to infer the parameters of synthetic observations from ParFlow, such that $p(\theta \mid Y = Y_{Obs\_ParFlow})$. We do this using the same realization of the neural density estimator from Experiment 1 (the *best case*). The 'tough' case is a realistic test of the robustness of parameter inference. Specifically, it tests our ability to evaluate data from a different source. Unlike in the *best case*, we must deal with uncertainties related to the goodness of fit between the simulator (the LSTM surrogate) and 'observation' (the underlying ParFlow model). We generate the posterior parameter and predictive densities to the benchmark case (θ_A) explored in Experiment 1. The only difference is that $Y_{Obs\_ParFlow\_A}$ is a simulation generated by ParFlow, and not the surrogate.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022



Hydrology and Earth System Sciences
Discussions

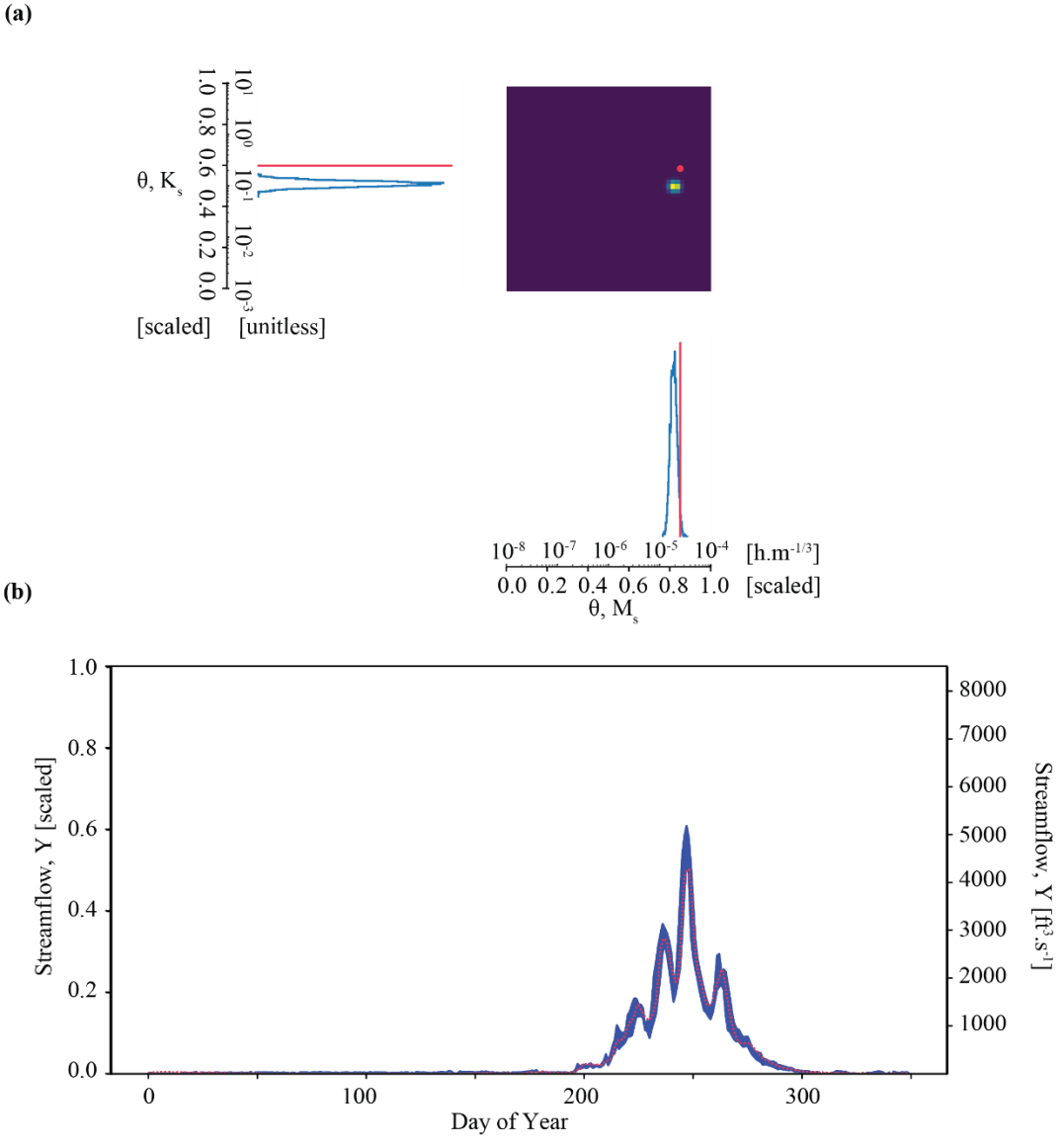


Figure 5. Results of parameter inference and posterior predictive check on synthetic observation $Y_{Obs\_ParFlow\_A}$ in Experiment 2 ('tough' case). Subplots (a) and (b) show overconfident parameter inference that still results in well-constrained posterior predictive check.

Figure 5 plots the results of experiment two. Here we see that the quality of inference is somewhat degraded for the *tough case* compared to the *best case*. Parameter inference here is overconfident; it is precise but biased as indicated by the tight probability distributions and the difference between the peak probability and the observation (indicted by the red line in Figure 7A). The true parameter value does not plot in the area corresponding to highest probability. The determinant is 6e-08, which is within the same order of magnitude as the *best case*. However, the Mahalanobis Distance is much higher, at 7e0. Thus, the 'true' parameter set can be thought of heuristically as approximately seven 'standard deviations' from the central

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

tendency of the inferred distribution. Visual inspection of Figure 7B shows that streamflow simulations yielded by inferred parameters still characterize the synthetic streamflow observation well. However, average error is roughly twice as high for the *tough case* compared to the *best case* (1e-02 compared to 6e-03), which is approximately equal the acceptability criterion described in Sect. 3.7.

Overconfident posterior estimates are a result of the misfit between our LSTM surrogate compared to ParFlow (Figure B1B). One interpretation of biased parameter inference is that the relationship between data (streamflow) and parameters (M_s, K_s) in the LSTM surrogate does not *quite* represent their relationship as it exists in ParFlow. These differences are not unexpected, because ParFlow has parameters that vary across a three-dimensional domain but are lumped together in the LSTM (See also Appendix A). This bias in the surrogate simulator increases the bias in the conditional density learned by the neural density estimator. We consider this suboptimal performance in parameter inference a consequence of 'surrogate misspecification', as described further in Sect. 6.

4.3 Experiment 3 – Boosted Case

A desirable approach to circumventing overconfident parameter posteriors is to make the LSTM surrogate simulator less biased. In our study, we utilize an ensemble of surrogate LSTM simulators with distinct biases subject to the initialization and selection of training data. That ensemble is then used to generate the set of simulated pairs $\{\theta, Y\}$ to train a new neural density estimator. The underlying principle is that the overall behavior of an ensemble of surrogate simulators *in aggregate* is not biased, even if each individual simulator has its own bias.

Experiment 3 is our *boosted* case. As in Experiment 2, we attempt to infer the parameters of synthetic observation(s) reserved from ParFlow, $p(\theta \mid Y = Y_{Obs\_ParFlow})$. As opposed to Experiments 1-2, we learn the conditional probability from an ensemble of 10 surrogate LSTM simulators instead of just one. We refer to the LSTM ensemble as a 'boosted' surrogate. Compared to the LSTM used in Experiment 1 and 2, these LSTMs are trained for fewer epochs (100, as compared to 300) and on a smaller random split of the data (0.7, as compared to 0.6). The reserved test data is the same across the LSTMs for Experiments 1, 2, and 3. Note that we don't use an adaptive learning algorithm such as AdaBoost (Freund and Schapire, 1997), and instead we equally weight each 'weak' LSTM simulator. The neural conditional density estimator is trained by taking a random draw from the ensemble of LSTMs and using the selected LSTM to generate a forward simulation of streamflow from a randomized parameter combination. Thousands of such draws are repeated until the conditional density has been sufficiently learned (see Appendix B for details), at which point it can be utilized for parameter inference.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

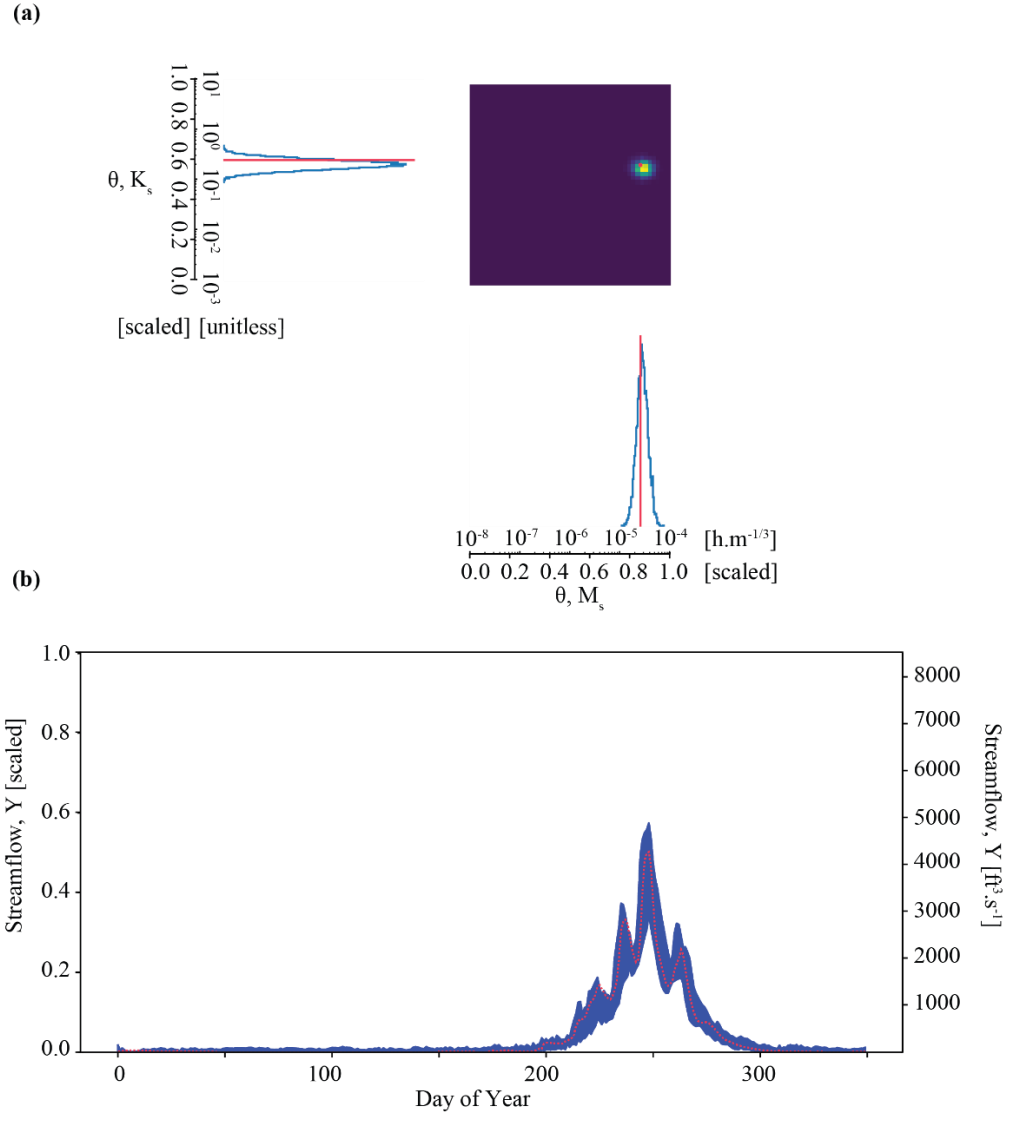


Figure 6. Results of parameter inference and posterior predictive check on synthetic observation $Y_{Obs\_ParFlow\_A}$ in Experiment 3 ('boosted' case). Subplots (a) and (b) show accurate parameter inference that is somewhat less precise, resulting in a wider but still well-constrained posterior predictive check.

Results of the *boosted case* in Experiment 3 show that we may be able to work around the issue of overconfident posteriors encountered in the *tough case* in Experiment 2. Fig. 6A shows precise and accurate parameter inference for our benchmark case in Experiment 3. The benchmark parameter values are in the area identified by the highest probability, as opposed to in Experiment 2. We note that the area of highest density is somewhat larger than in Experiment 2. The determinant is 5e-07, which is about an order of magnitude higher than the *tough case*, 6e-08. The Mahalanobis Distance is 1e0. For comparison, Mahalanobis Distance in the previous 'overconfident' experiment was 7e0. The inferred parameters generate

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

streamflow simulations that characterize the synthetic streamflow observation well, as shown by the posterior predictive check (Fig. 6B). We note that compared to Experiment 2 (Figure 5B) our simulations are somewhat more variable, as shown by the larger distance between the larger uncertainty envelope. The average streamflow error is about twice as high for the *boosted case* as compared to the *tough case*, (2e-02 compared to 1e-02). The standard deviation if the error is also greater (5e-03 compared to 2e-03). The sacrifice in precision with respect to both parameter inference and the posterior prediction is a consequence of using an ensemble of surrogates to simulate each parameter set.

4.4 Summary of Experiments 1, 2, and 3

Previously, we compared the performance of simulation-based inference in Experiments 1 (*best case*), 2 (*tough case*), and 3 (*boosted case*) on only one benchmark parameter set. In this section, we expand the comparison of SBI across the experiments to a larger number (n=18) of parameter sets and corresponding observations. In the case of Experiments 1 and 2, the same neural density estimator was utilized to conduct inference; for Experiment 3, an ensemble approach was used to create a new neural density estimator. In the case of Experiments 2 and 3, the mock data are the same benchmark streamflow simulations from ParFlow; for Experiment 1, the observations are taken from the surrogate. All three experiments utilize mock data corresponding to the same test parameter sets, to make an apples-to-apples comparison. For reference, those test parameter sets are plotted relative to parameter space in the Fig. B1A. The results of the analysis of multiple (n=18) parameter sets are shown by the box plots in Fig. 7.

4.4.1 The precision and accuracy of parameter inference

In general, the parameter estimates from the three experiments are accurate and precise, as shown in Fig. 7A and 7B. The *best case* (Experiment 1) tends to be both precise and accurate; the *tough case* (Experiment 2) tends to be just as precise but less accurate; and the *boosted case* (Experiment 3) tends to be less precise but more accurate. A couple of second-order discussion points arise from Figs. 7A and 7B.

The resulting box plots of the determinant, a metric for the precision of inference, are shown in Fig. 7A. Here we see that the training of the conditional density estimator – and not the source of the observations – seems to define the precision of inference. The box plots show parameter inference is more precise (i.e., the determinant smaller) for Experiments 1 and 2, compared to Experiment 3. Experiments 1 and 2 use synthetic observations from different sources (the LSTM surrogate and ParFlow, respectively), however are evaluated using the same neural conditional density estimator; note the similar behavior of the determinant in the first two experiments. On the other hand, the determinant behaves quite differently in Experiment 2 compared to Experiment 3; both experiments use synthetic observations from ParFlow, but a different configuration of the neural conditional density estimator. Fundamentally, the precision of parameter inference seems to reflect the simulator (i.e.,

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

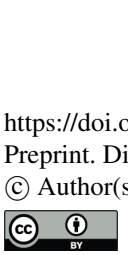

Hydrology and Earth System Sciences
Discussions

the variety in simulated responses, Y, to parameter configurations, θ), and not the goodness-of-fit between observations, $Y_{obs.}$ and simulated data, Y.[3]

Box plots of the Mahalanobis[4] distance, a metric of the accuracy of inference, are shown in Fig. 7B. The box plots show that parameter inference in Experiment 1 is more accurate than Experiments 2 and 3. The box plots also demonstrate that parameter inference is in general more accurate for the *boosted case* (Experiment 3) compared to the *tough case* (Experiment 2). However, the Mahalanobis distance is greater at some outlier points in the *boosted case* (Figure 7B). What this means is that while boosting yielded more accurate inference in some parts of parameter space (for example, the benchmark parameter set θ_A explored throughout the earlier results sections), our implementation is no silver bullet for averting biased parameter estimates.

[3] This behavior is also observed in Figure D1A, which shows that the determinant exhibits a fixed pattern across parameter space.
[4] Note that Mahalnobis distance is a precision-weighted metric of distance, unlike Euclidean distance. These numbers should not be considered raw distance.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022



Hydrology and Earth System Sciences
Discussions

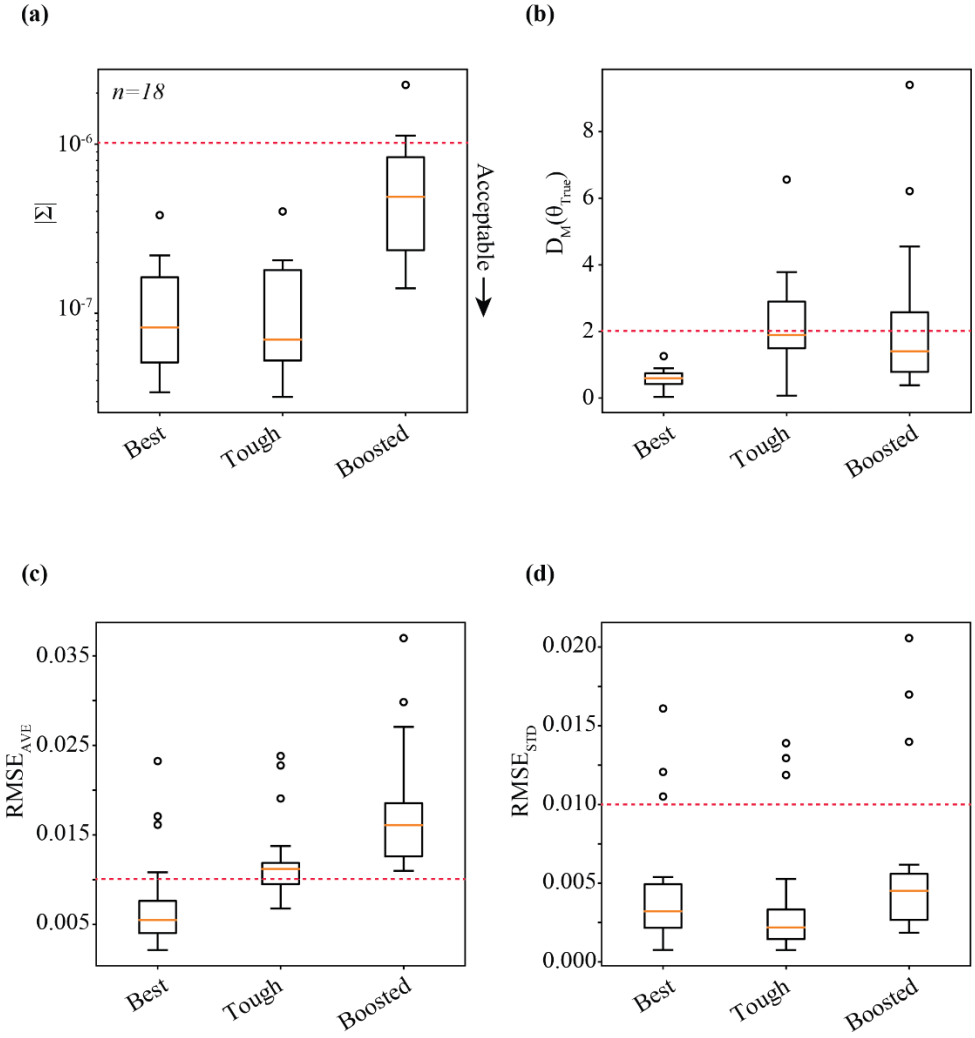


Figure 7: Comparative plots showing the performance of simulation-based inference of parameters and predicted quantities across a set of n=18 test data. We compare the results of Experiments 1 ('base' case), 2 ('tough' case), and 3 ('boosted' case). Subplots (a) and (b) show the precision and accuracy of parameter inference. Precision is shown in subplot (a) via the Determinant, $|\Sigma|$ of the posterior parameter density. Accuracy is shown in subplot (b) via the Mahalanobis Distance. Subplots (c) and (d) show the accuracy and precision of the posterior predictive check. Subplot (c) shows the average of the error, $RMSE_{Ave}$ of streamflow ensembles relative to 'truth', which can be thought of as a measure of accuracy. Subplot (d) shows the standard deviation of the error, $RMSE_{std}$ of streamflow ensembles, which can be thought of as a measure of precision. The red dotted line is the 'acceptability' criterion for each metric. Values closer to the x-axis are better.

4.4.2 The precision and accuracy of posterior predictions

Taking this one step further we can use the inferred parameter distributions to generate an ensemble of streamflow simulations using the LSTM and compare this to the observed streamflow (referred to as our posterior predictive check). As shown in Fig. 7C and 7D, the posterior predictions are precise, but not always accurate. Fig. 7C shows the average of the error ($RMSE_{Ave}$) between the simulated streamflow timeseries and the observed time series, with lower average error corresponding

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

to greater accuracy. Streamflow prediction accuracy decreases between Experiments 1, 2, and 3. This is represented by the fact that the $RMSE_{AVE}$ increases nearly 3-fold across each of our experiments (median ~0.005 in *best case*, ~0.010 in *best case*, and ~0.015 in *boosted case* [scaled streamflow units]). The degradation in posterior predictive accuracy is related to degradation in the accuracy of parameter inference (Figure 7A). Fig. 7D shows the variability of the error ($RMSE_{STD}$) between the simulated streamflow timeseries and the observed time series, with lower error variability corresponding to greater precision. We see that the central tendency of the $RMSE_{STD}$ of streamflow simulations for the *base, tough,* and *boosted* cases are all similar. Streamflow posterior predictions across all three experiments remained precise, in spite of the breakdown in the accuracy.

The multi-observation comparison helps us to generalize some insights. 1. Inference results are often desirable; in particular, SBI seems to result in precise parameter inference across all conditions. 2. Parameter inference with a well-trained surrogate simulator is precise, but not always suited for conducting inference on observations with an uncertain relationship to simulated data (as in Experiment 2). 3. The performance of posterior predictive checks is dependent on both the performance of the simulator and the neural density estimator. As such it can be a valuable tool in assessing the performance of parameter inference. 4. Ensemble simulators (as in Experiment 3) may yield more accurate parameter inference. 5. Biased parameter estimates are a recalcitrant problem for some observed data, even with the boosted surrogate.

5 Discussion

For hydrologic problems, we typically can't be sure that a simulator configuration consistent with observations exists. We refer to this issue as misspecification (Cranmer et al., 2020). Misspecification in this work is underscored by the misfit between the process based ParFlow and the surrogate LSTM simulators. As seen in a scatter plot of test points that relate ParFlow and LSTM streamflow simulations, the performance of the LSTM surrogate is quite good (Figure 8A). But it isn't perfect. This is highlighted by boxplots of RMSE (Figure 8B) and other metrics (Figure 8C) comparing LSTM and ParFlow simulated streamflow for n=18 test series. This misfit shows that the causal relationship of parameters M_S, K_S to streamflow is not quite identical for the LSTM and ParFlow, even if we desire them to be as such. We call this special situation surrogate misspecification.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

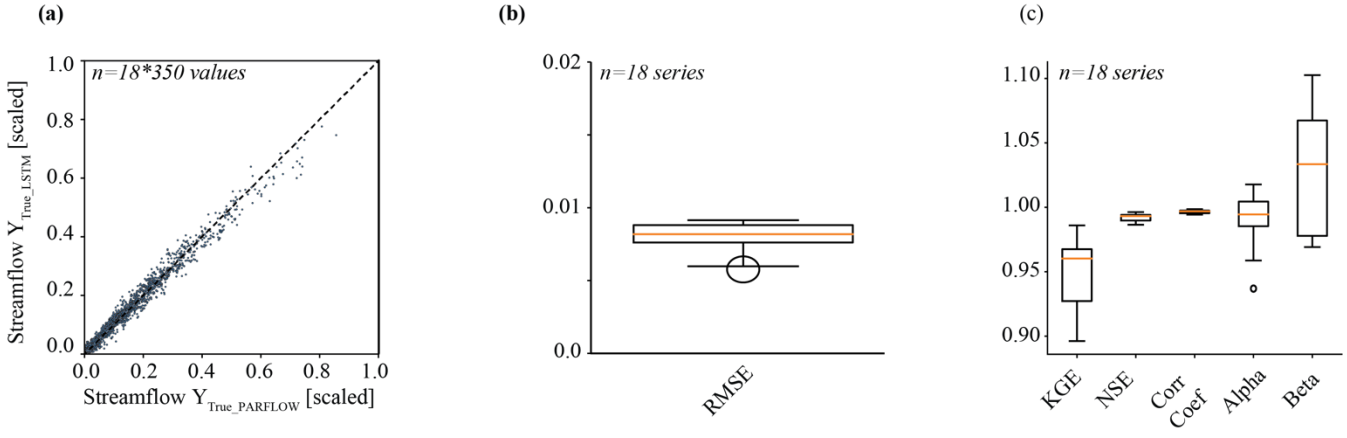


Figure 8. Relationship between streamflow simulated by ParFlow and by the surrogate LSTM, reserved for testing. Subplot (a) is a scatter plot relating each individual ParFlow and LSTM data pair. Perfect fit along the 1:1 line. Subplot (b) and (c) show bar plots of metrics relating the fit of the surrogate LSTM to ParFlow for all n=18 test data series. RMSE close to 0 is better in (b). All other metrics close to 1 are better in (c).

Our research shows that using a mis specified surrogate to conduct inference for a process-based simulator can yield erroneous parameter estimates. The neural density estimator learns the conditional relationship between parameters and data from the surrogate simulator. Thus, SBI explicitly infers inputs to the surrogate and *not* parameters of the process-based simulator. Given surrogate misspecification, careful accounting of bias in the surrogate must be done to translate the inferred values into parameters that retain their physical significance to the process-based simulator.

Despite the negative consequences of misspecification on parameter estimates, the posterior predictions of streamflow are overall still quite good, even for the 'tough case' (Figure 7). In many 'real world' applications, a calibrated estimate of the hydrologic variable (i.e., streamflow) is what watershed scientists strive for. Moreover, the actual values for physical parameters (i.e., K and M) are almost never known at the scale we care about, making them impossible to validate (Oreskes et al., 1994). Thus, it is important to acknowledge the success of surrogate-derived SBI at improving predictions of streamflow in this study. It's an important test-case that even if we have the wrong model, we can still get reasonably good answers, with the caveat that no prediction is ever perfectly 'true' (Van Fraassen and others, 1980).

We remind the reader that our goal has been to introduce and test SBI in a watershed setting using synthetic observations. A logical next step would be to conduct inference on real observations. Simulator configurations that could be used for defensible inference on observations assuredly exist, but they require expanding the information used for inference beyond the observations and simulator configurations explored in this research.

Including additional watershed observation types (i.e., groundwater, soil moisture) could improve our ability to accurately infer the physical parameters for real systems. However, observations in hydrology - particularly about groundwater systems - are generally sparse. This presents a problem. One option is to observe that complexity *better*. New spatially

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

distributed 'big data' products that leverage remote sensing to offer new opportunities to observe hydrologic variables like soil moisture (Mohanty et al., 2017; Petropoulos et al., 2015).

Adding additional complexity to the training set for the surrogate simulator (i.e., exploring spatially variable parameter configurations, or multiple forcing scenarios) may help yield better accounting of the uncertainty of inferred parameters. Many of the practitioners of simulation-based inference advocate packing as much complexity into our models as possible (Alsing and Wandelt, 2019). If we buy this argument, then our task in employing SBI on real hydrological systems is to build a "faithful" spatially distributed, transient, simulators of heterogeneous, time-varying simulator. High-resolution process-based simulators (such as ParFlow) can be used to explore the real-like behaviors of watersheds across many more variable and parameter configurations than presented here.

6 Conclusion

Our investigation implements simulation-based inference (SBI) to determine parameters for a spatially distributed, process-based watershed simulator. We believe this research is among the first to apply contemporary SBI to watershed modeling. The implementation employed here has a couple of noteworthy features:

a. We use deep learning to train a surrogate Long Short-Term Memory (LSTM) on the original physically based simulations (from ParFlow). This allows for quick and comprehensive exploration of simulation results for which we have corresponding observations, such as streamflow at a basin outflow in a watershed.
b. A density-based neural network leverages the capacity of the surrogate to generate simulations quickly to learn a 'model' for the full conditional density, $p(\theta|Y)$, of parameters given data. This learned model can be evaluated using observations to determine the parameter posterior density, $p(\theta|Y = Y_{Obs})$. This parameter posterior represents our 'best guess' of what the parameters for our simulator should be.

We demonstrate that this approach to SBI can generate reasonable estimates of the parameters of a hydrologic simulator, ParFlow. We show in Experiment 1 (the *best case*) that SBI works well in controlled settings in which we assume that our surrogate LSTM simulator is accurate. Moreover, this experiment highlights how, once learned, the model of the conditional density can be used to determine the process-based parameters rapidly and effectively for many observations without the need for additional process-based simulations. That's particularly valuable when simulations are costly, as is often the case with high-resolution, transient simulators used in the field of watershed modeling.

In Experiments 2 and 3 (the *tough* and *boosted cases*, respectively), we show that SBI can successfully be implemented under conditions of greater uncertainty. In Experiment 2 (the *tough case*), SBI determines a set of probable parameters with precision. These inferred parameters are used to generate streamflow simulations that reproduce observations well. However, the *tough case* shows that parameter inference is not always accurate with respect to the physics-based simulator that was used to train the surrogate. This undesirable characteristic (of 'precision but not accuracy') arises from the problem that the simulator does not represent the underlying system generating the observation perfectly. We call this problem 'simulator misspecification', which is well-recognized in the literature as an impediment for accurate parameter inference

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

(Cranmer, 2020). The controlled nature of Experiment 2 explores the special case of 'surrogate misspecification'. This special case arises from a mismatch between the surrogate and the process-based simulations from ParFlow. In inference, surrogate misspecification gives rise to bias with respect to the physical parameters. We show that this bias can be quite difficult to diagnose, although conducting a posterior predictive check is a qualitative way of ascertaining if simulator bias may exist.

In Experiment 3 (the *boosted case*), we attempt to solve the issue of 'precise but biased' parameter inference. We use an ensemble of 'weak' surrogate simulators (instead of just one 'strong' surrogate simulator) to learn the full conditional density. The underlying principle is that the behavior of an ensemble of surrogate simulators *in aggregate* is not biased, even if each individual simulator has its own bias. This may 'wash out' the negative effects of surrogate misspecification on parameter inference. But it is not a silver bullet for conducting accurate inference. The results of Experiments 2 and 3 demonstrate progress towards being able to implement SBI in hydrological domains subject to uncertainty we can benchmark (i.e., the misspecification of the surrogate). We have work to do to implement SBI in problems with 'unaccounted for' uncertainty. This 'unaccounted for' uncertainty often exists in watershed modeling, where (e.g.) natural heterogeneities in the subsurface or bias in the meteorological input data may give rise to simulators mis specified in ways that are hard to diagnose. An obvious next step would be to expand the simulator to explore more and different configurations of parameters and input variables.

Despite the negatives of misspecification on parameter estimates, the posterior predictions of streamflow are overall still quite good. In many 'real world' applications, a calibrated estimate of the hydrologic variable (i.e., streamflow) is what watershed scientists strive for. Thus, it is important to acknowledge the success of surrogate-derived SBI at improving predictions of streamflow in this study. We show that simulation-based inference may not be a panacea for parameter determination and model calibration in hydrology, but it provides many opportunities. Our work is a framework that can be built up to tackle harder and more complex inference problems in watershed modeling. More work needs to be done to utilize rapidly emerging best practices in SBI to make inference more robust and useful for hydrologic inquiry.

Appendix A The Process-Based Simulations (ParFlow)

Table A1: The relationship between ParFlow and LSTM static inputs (e.g., parameters, θ), dynamic inputs (e.g., meteorological forcings, X), and dynamic outputs (e.g. streamflow, Y). ParFlow variables must be 'compressed' into lower-dimensional representations in order to be used in the LSTM.

	ParFlow Description	**LSTM Description**
Parameters, θ	a) 2-dimensional homogeneous Manning's Roughness, M b) 3-dimensional heterogeneous Hydraulic Conductivity, K *(Other static inputs, such as soil properties and land cover, are not used by LSTM)*	a) Scalar value, M_s, set for all values of M b) Scalar factor, K_s, multiplied by all values of K *(Both are log transformed and re-normalized to be between 0 and 1)*

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

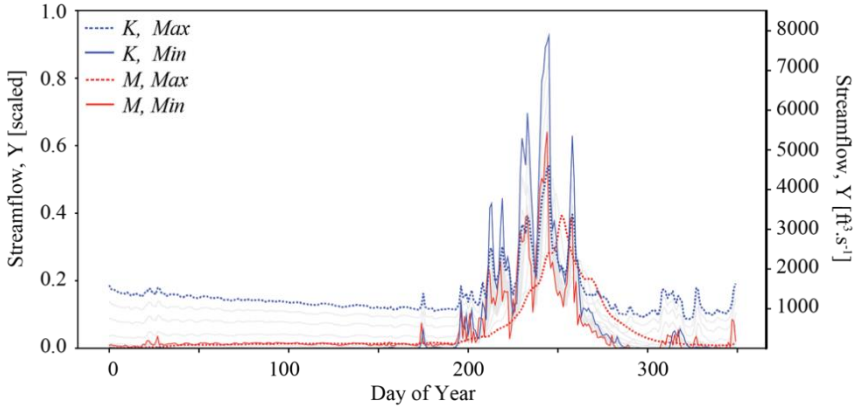
Dynamic Outputs, *Y*	Hourly, 3D spatially distributed pressure field	Daily, 1-dimensional discharge time series (length=350) at i,j location corresponding to USGS gage 09110000, as follows: 1. *Gridded discharge calculated using surface pressure, slopes, Manning's, resolution via the overland flow equation for each hourly time step (n=8,760) of one year of ParFlow results* 2. *Slice at i,j location and calculate daily average* 3. *Remove first 15 days of record (burn in time), and renormalize values between 0 and 1*
Dynamic Inputs, *X*	Hourly, 2D spatially distributed meteorological forcings, including: ● *DLWR: Long Wave Radiation [W.m-2]* ● *DSWR: Short Wave Radiation [W.m-2]* ● *Press: Atmospheric pressure [pa]* ● *APCP: Precipitation [mm.s-1]* ● *Temp: Air Temperature [K]* ● *SPFH: Specific humidity [kg.kg-1]* ● *UGRD: East-west wind speed [m.s-1]* ● *VGRD: South-to-North wind speed [m.s-1]*	Daily, 1D time series (length=350) for each (n=8) forcing: *(Except for APCP, forcings are averages taken over space and time for all hours (n=24) in each day. APCP is the sum over space and time for all hours (n=24) of precipitation each day.)*

Table A2: ParFlow was run many times under different parameter configurations. This table shows the scalar factors used to modify spatially distributed Manning's Coefficient and Hydraulic Conductivity. We call these factors K_s and M_s, respectively, to keep the distinction between them and ParFlow's parameters clear.

	K_s (Scaling factor times whole domain)[unitless]	**M_s (Constant across domain), [h/m^(1/3)]**
Scalar Parameters	0.001, 0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1, 2.5, 5, 7.5, 10	1e-8, 1e-7, 2.5e-7, 5e-7, 7.5e-7, 1e-6, 2.5e-6, 5e-6, 7.5e-6, 1e-5, 2.5e-5, 5e-5, 1e-4

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions



675

Figure A1: Sensitivity of ParFlow-generated streamflow time series for water year 1995 to perturbations of Hydraulic Conductivity and Mannings. We show sensitivity holding each of K_s and M_s constant at 0.1 and 5e-6, respectively, while varying the other across the range of parameters explored in Table A2.

Appendix B The Surrogate Simulator (LSTM)

680 **Table B1: Relevant notes on architecture, training, and hyperparameters for the surrogate LSTM simulator.**

	LSTM	**Further Description**
Number of Epochs	300	Number of times iterating through training loops
Batch Size	50	Batching during training
Input Size	10	Number of input features
Hidden Layers	1	Number of hidden layers
Hidden Size	10	Number of hidden nodes / layers
Number of Classes	1	Number of nodes in output
Objective Function	MSE	Mean Squared Error
Optimizer	Adam	
Learning Rate	0.001	

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

Train-Validation-Test Split	0.7, 0.2, 0.1	Simulations were divided into sets based on their parameters, such that each member characterizes the streamflow response (encoded as a year-long timeseries) to an individual pair of parameter values K_s and M_s. We conduct the train-validation-test split in a pseudo-Latin hypercube manner across parameters space.

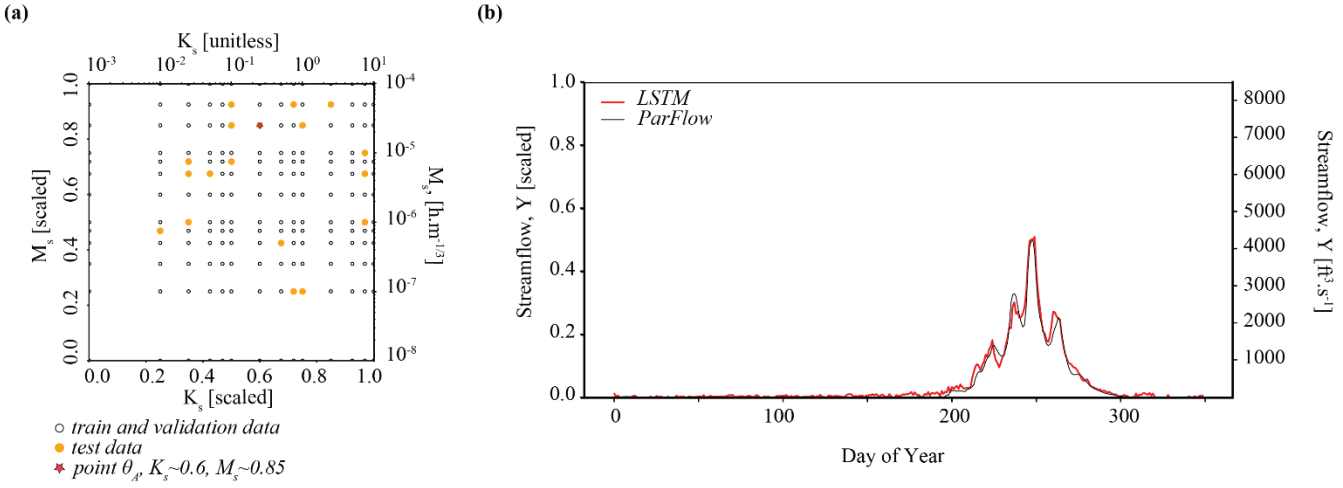


Figure B1: Plots show the train/validation and test split for the LSTM surrogate trained on n=183 ParFlow simulations. In (a), the locations in parameter space where ParFlow simulations were run. The surrogate is trained and tested at orange dots. In (b), a comparison of ParFlow to LSTM streamflow simulation generated at benchmark parameter set θ_A K_s~0.6, M_s~0.85. The fit between ParFlow and LSTM is explored more in the results.

Appendix C Improved Components for SBI

Deriving implicit statistical models using density estimation techniques is not new (Diggle and Gratton, 1984). However, these traditional approaches suffer from some shortcomings, including sample efficiency and inference quality, as described further in Cranmer, Brehmer, and Louppe 2020. We show two components of the density based SBI workflow utilized here that have benefited due to recent innovations: Masked Autoencoders for Density Estimation (MADEs) and sequential neural posterior sampling.

C.1 Masked Autoencoder for Density Estimation (MADE)

While mixture density networks have a long operational history, there have been more recent innovations in using neural networks to learn and represent conditional probability distributions. This study utilizes a class of neural density estimators called Masked Autoregressive Flows (Alsing et al., 2019), which share some of the underlying principles described

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

for Mixture Density Networks. Masked Autoregressive Flows arise from the principle that "any probability density can be factorized as a product of one-dimensional conditionals" via the chain rule (Alsing et al., 2019); these one-dimensional conditionals are parameterized by a fully connected neural network known as a Masked Autoencoder for Density Estimation (MADE) (Uria et al., 2016). Masked Autoregressive Flows are composed of 'stacks' of Masked Autoencoder for Density Estimations, to add flexibility (Papamakarios et al., 2018) . A detailed description of these methods is beyond the scope of this paper.

C.2 Sequential Neural Posterior Estimation

We use a sampling technique called Sequential Neural Posterior Estimation (SNPE) to speed up and improve the evaluation of a trained neural conditional density estimator. By evaluation, we here mean using data Y (most typically observed data, Y_{Obs}) to generate a posterior estimate $p(\theta \mid Y = Y_{Obs})$ (step 4 in Sect. 3.5). The need for SNPE arises from the challenge that drawing simulation parameters from the full prior distribution is wasteful (Papamakarios et al., 2018; Lueckmann et al., 2017; Greenberg et al., 2019). This is due to the fact that data simulated from some parts of parameter space have higher or lower posterior density for Y_{Obs}. SNPE iteratively refines the posterior estimate to make inference more efficient and flexible, as described by Greenberg et al, 2019.

Details related to the architectures, hyperparameters, training, and evaluation of neural density estimators are shown in Table C1. Decisions about hyperparameters were made via trial and error. It's important to note that the goal of our work is not to create the most robust neural density estimator model, but to explore inference under a variety of different conditions.

Table C1: Hyperparameters and model architecture for neural density estimation. See also (Tejero-Cantero et al., 2020).

Hyper- parameter	Value	Significance
Inference Method	SNPE_C	Sequential Neural Posterior Estimator (see text)
Neural Density Model, $q_\phi(\theta \mid Y)$	MAF	Masked Autoregressive Flow (see text)
Hidden Features	10	number of hidden layers in each MADE of $q_\phi(\theta \mid Y)$
Number of Transforms	2	Number of flows (transforms) between MADEs in $q_\phi(\theta \mid Y)$, MAF
Prior_min, Prior_max	0.0, 1.0	Minimum and Maximum possible values of $q_\phi(\theta \mid Y)$, K_s and M_s
Prior Function	Uniform	All values *a priori* equally possible in parameter space

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

Number of simulations	1000	Number of simulated $\{\theta, Y\}$ pairs; used to train $q_\phi(\theta\|Y)$
Number of samples	5000	Number of sampled $\{\theta, Y\}$ pairs; used to evaluate $q_\phi(\theta\|Y)$

Appendix D Inference for many observations, $Y_{Obs\_LSTM\_i}$

A trained neural density estimator can be used to infer the parameters of an observation without the need for additional simulation runs. In this section, we extend Experiment 1 (the 'best' case) to evaluate the posterior parameter density for many synthetic observations ($Y_{Obs\_LSTM\_i}$) quickly and effectively. We use many parameter sets (θ_i) of K_s and M_s sampled uniformly across parameter space to generate an equivalent number of synthetic observations, where $i=1, 2, ..., 441$.

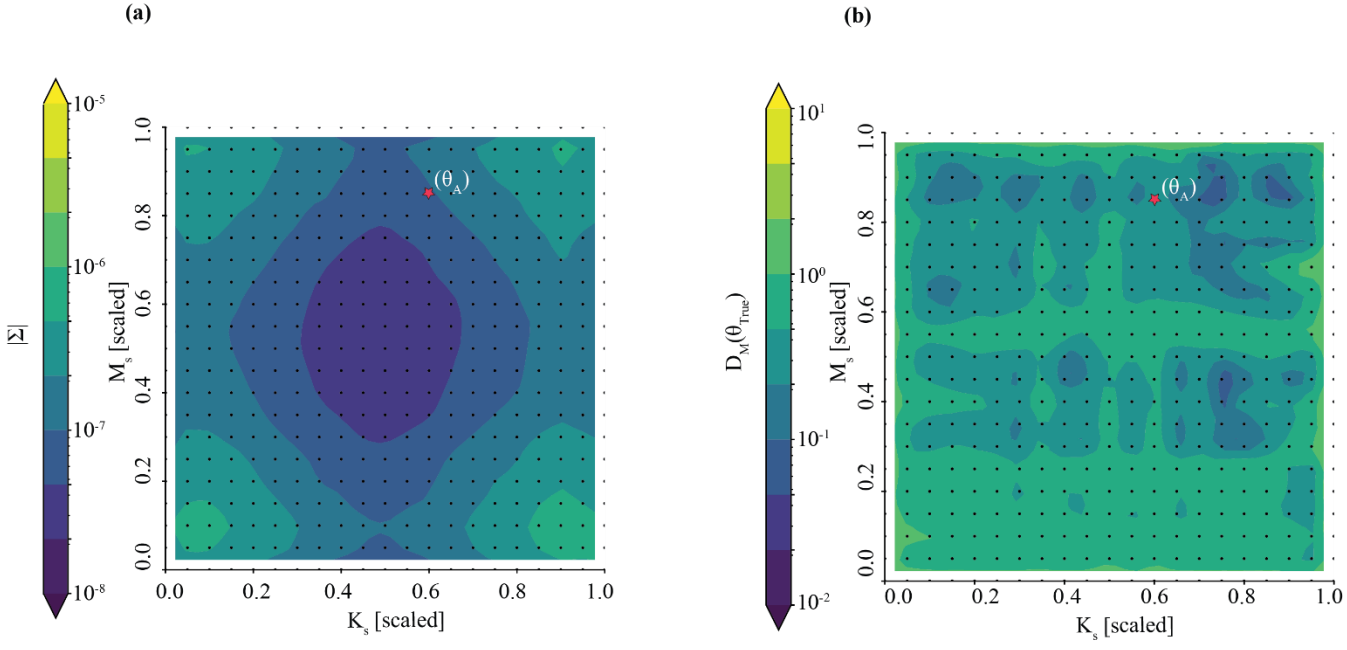


Figure D1. Once the neural conditional density estimator is trained, it can be evaluated quickly and effectively given new data. This figure shows the performance of SBI of Mannings (M_s), and Hydraulic Conductivity (K_s) given synthetic streamflow data generated by the surrogate from across 441 locations across parameter space. Subplot (a) shows the Determinant, $|\Sigma|$ of the posterior parameter estimate, which quantifies the precision of parameter inference. Subplot (b) shows the Mahalanobis distance, $D_M(\theta_{True})$ between the inferred distribution and true parameter values, which quantifies the accuracy of inference. These values are shown across the entirety of parameter space investigated, where purple is better. The red star in subplots corresponds with benchmark location θ_A in parameter space of the analysis shown in Figure 3.

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022

Hydrology and Earth System Sciences
Discussions

SBI can infer the parameters from many diverse and different synthetic observations well, as shown in Figure D1. The precision of inference of the posterior parameter densities is explored in Figure D1A as a map of determinants across parameter space. Parameter inference is more precise (with a smaller determinant) in the center than at the edges of the parameter space; it is below our precision threshold of 1e-06 everywhere. Parameter inference is accurate across parameter space, as shown by the map of Mahalanobis Distance in Fig. D1B. There are some pockets of parameter space characterized by more- and less- accurate parameter inference. The structure of the Mahalanobis distances across parameter space doesn't seem to be as well-defined as that of the determinant and are likely a consequence of randomness in the initialization of the neural density estimator (confirmed by many independent trials). We note that evaluating each of the synthetic observations in Fig. D1 took only a few seconds.

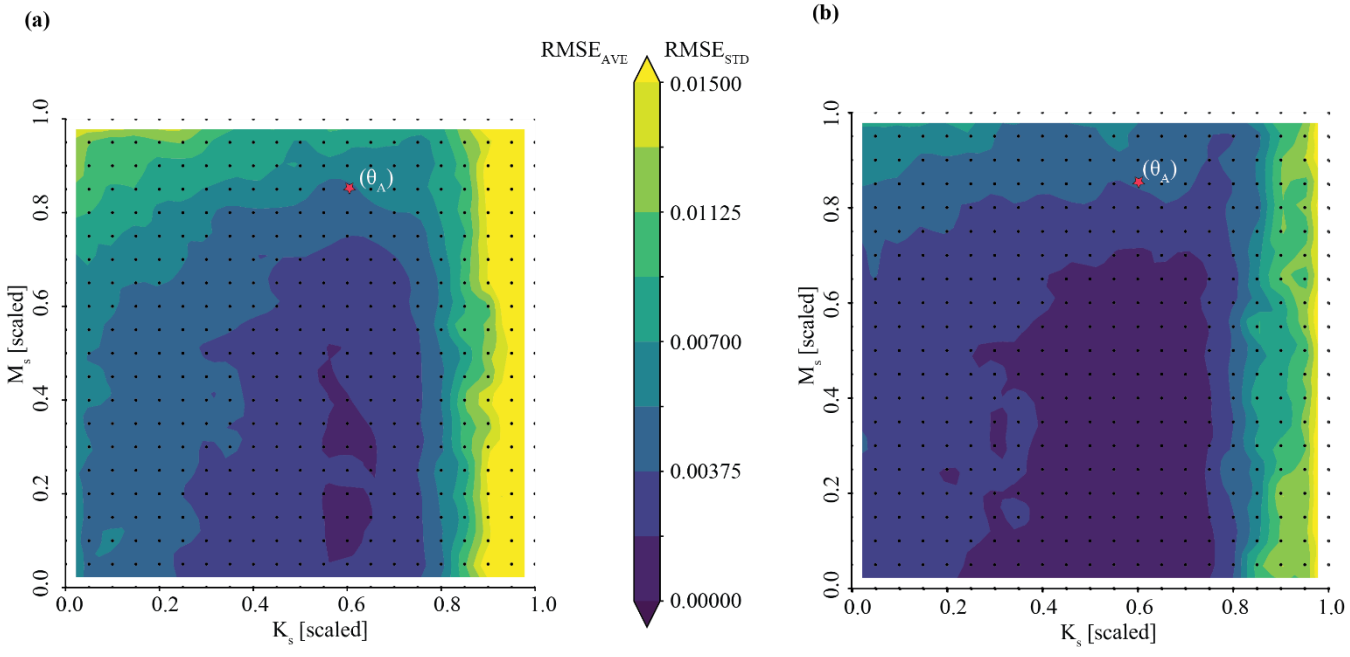


Figure D2. Posterior predictive check for many observations: Once parameters are inferred, the posterior can be drawn (n=50) to generate probabilistic streamflow ensembles. This figure shows the performance of streamflow ensembles derived from SBI at 441 locations across parameter space. Subplot (a) shows the average of the error ($RMSE_{Ave}$) of streamflow ensembles relative to 'truth', which can be thought of as a measure of accuracy. Subplot (b) shows the standard deviation of the error ($RMSE_{std}$) of streamflow ensembles, which can be thought of as a measure of precision. Streamflow ensembles are evaluated against the 'true' synthetic streamflow time series generated by the surrogate simulator, where blue is better.

The posterior predictive check shows that streamflow characterization is generally both precise and accurate. This required drawing a subset of parameters from *each* of the 441 posterior parameter densities represented as points in Fig. D1 and generating an ensemble of simulated streamflow time series using the surrogate simulator. The accuracy of the posterior predictions is explored in Fig. D2A as a map across parameter space. In general, the posterior predictions have an average error of less than 0.01. Accuracy is highest in the middle of the parameter space and seems to degrade towards the upper

https://doi.org/10.5194/hess-2022-345
Preprint. Discussion started: 7 November 2022


Hydrology and Earth System Sciences
Discussions

boundaries where parameters K_s and M_s are large. The precision of the posterior predictions is explored in Fig. D2B as a map across parameter space. In general, the posterior predictions are precise, with standard deviation of the error less than 0.01. We note that both the average and standard deviation of error increase at large parameter values, in particular large values of hydraulic conductivity. Overall, Fig. D1 and D2 show that SBI can reliably infer parameters and characterize streamflow processes for *many* streamflow observations that span the parameter space we investigated.

References

Alsing, J. and Wandelt, B. D.: Nuisance hardened data compression for fast likelihood-free inference, Monthly Notices of the Royal Astronomical Society, 2019.

Alsing, J., Charnock, T., Feeney, S., and Wandelt, B.: Fast likelihood-free cosmology with neural density estimators and active learning, Monthly Notices of the Royal Astronomical Society, https://doi.org/10.1093/mnras/stz1960, 2019.

Bastidas, L., Gupta, H., Sorooshian, S., Shuttleworth, W., and Yang, Z.-L.: Sensitivity analysis of a land surface scheme using multicriteria methods, Journal of Geophysical Research, 104, 19481–19490, https://doi.org/10.1029/1999JD900155, 1999.

Bishop, C.: Mixture Density Networks, 1994.

Castle, S. L., Thomas, B. F., Reager, J. T., Rodell, M., Swenson, S. C., and Famiglietti, J. S.: Groundwater depletion during drought threatens future water security of the Colorado River Basin, Geophysical Research Letters, 41, 5904–5911, https://doi.org/10.1002/2014GL061055, 2014.

Condon, L. E.: Scientist Spotlight: Laura Condon, 2022.

Cranmer, K.: A3D3 Seminar: Accelerating Simulation-based Inference | Kyle S. Cranmer, 2022.

Cranmer, K., Brehmer, J., and Louppe, G.: The frontier of simulation-based inference, Proceedings of the National Academy of Sciences, 117, 30055–30062, https://doi.org/10.1073/pnas.1912789117, 2020.

Diggle, P. J. and Gratton, R. J.: Monte Carlo methods of inference for implicit statistical models, Journal of the Royal Statistical Society: Series B (Methodological), 46, 193–212, 1984.

Fatichi, S., Vivoni, E. R., Ogden, F. L., Ivanov, V. Y., Mirus, B., Gochis, D., Downer, C. W., Camporese, M., Davison, J. H., Ebel, B., Jones, N., Kim, J., Mascaro, G., Niswonger, R., Restrepo, P., Rigon, R., Shen, C., Sulis, M., and Tarboton, D.: An overview of current applications, challenges, and future trends in distributed process-based models in hydrology, Journal of Hydrology, 537, 45–60, https://doi.org/10.1016/j.jhydrol.2016.03.026, 2016.

Fenicia, F., Kavetski, D., Reichert, P., and Albert, C.: Signature-Domain Calibration of Hydrological Models Using Approximate Bayesian Computation: Empirical Analysis of Fundamental Properties, Water Resources Research, 54, 3958–3987, https://doi.org/10.1002/2017WR021616, 2018.

Freund, Y. and Schapire, R. E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences, 55, 119–139, https://doi.org/10.1006/jcss.1997.1504, 1997.

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A.: Visualization in Bayesian workflow, Journal of the Royal Statistical Society: Series A (Statistics in Society), 182, 389–402, https://doi.org/10.1111/rssa.12378, 2019.

Greenberg, D. S., Nonnenmacher, M., and Macke, J. H.: Automatic Posterior Transformation for Likelihood-Free Inference, , https://doi.org/10.48550/ARXIV.1905.07488, 2019.

Hochreiter, S. and Schmidhuber, J.: Long Short-Term Memory, Neural computation, 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735, 1997.

Hunt, R. J., Doherty, J., and Tonkin, M. J.: Are Models Too Simple? Arguments for Increased Parameterization, Groundwater, 45, 254–262, https://doi.org/10.1111/j.1745-6584.2007.00316.x, 2007.

Jiang, S., Zheng, Y., and Solomatine, D.: Improving AI system awareness of geoscience knowledge: Symbiotic integration of physical approaches and deep learning, Geophysical Research Letters, 47, e2020GL088229, 2020.

Jones, J. E. and Woodward, C. S.: Newton-Krylov-multigrid solvers for large-scale, highly heterogenous, variably saturated flow problems, Advances in Water Resources, 763–774, https://doi.org/10.1016/S0309-1708(00)00075-0, 2001.

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., and Kumar, V.: Theory-guided data science: A new paradigm for scientific discovery from data, IEEE Transactions on knowledge and data engineering, 29, 2318–2331, 2017.

Kollet, S. J. and Maxwell, R. M.: Capturing the influence of groundwater dynamics on land surface processes using an integrated, distributed watershed model, Water Resources Research, https://doi.org/10.1029/2007wr006004, 2008.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., and Herrnegger, M.: Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks, Hydrology and Earth System Sciences, 22, 6005–6022, https://doi.org/10.5194/hess-22-6005-2018, 2018.

Leonarduzzi, E., Tran, H., Bansal, V., Hull, R. B., De la Fuente, L., Bearup, L. A., Melchior, P., Condon, L. E., and Maxwell, R. M.: Combining Machine learning and Physics-based hydrological Modeling to predict 2D soil moisture fields in a changing climate, Frontiers, Publication Forthcoming, 2022.

Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H.: Flexible statistical inference for mechanistic models of neural dynamics, , https://doi.org/10.48550/ARXIV.1711.01861, 2017.

Maesschalck, R. D., Jouan-Rimbaud, D., and Massart, D. L.: The Mahalanobis distance, Chemometrics and Intelligent Laboratory Systems, 50, 1–18, https://doi.org/10.1016/S0169-7439(99)00047-7, 2000.

4.3 Determinants and Volumes: https://textbooks.math.gatech.edu/ila/determinants-volumes.html.

Maxwell, R. M. and Kollet, S. J.: Integrated surface–groundwater flow modeling: A free-surface overland flow boundary condition in a parallel groundwater flow model, Advances in Water Resources, 945–958, https://doi.org/10.1016/j.advwatres.2005.08.006, 2006.

Maxwell, R. M. and Miller, N. L.: Development of a Coupled Land Surface and Groundwater Model, Journal of Hydrometeorology, 233–247, https://doi.org/10.1175/JHM422.1, 2005.

Maxwell, R. M., Condon, L. E., and Kollet, S. J.: 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, 923–937, https://doi.org/10.5194/gmd-8-923-2015, 2015a.

Maxwell, R. M., Condon, L. E., and Kollet, S. J.: 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, 923–937, https://doi.org/10.5194/gmd-8-923-2015, 2015b.

Maxwell, R. M., Condon, L. E., and Melchior, P.: 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, https://doi.org/10.3390/w13243633, 2021.

Mohanty, B. P., Cosh, M. H., Lakshmi, V., and Montzka, C.: Soil moisture remote sensing: State-of-the-science, Vadose Zone Journal, 16, 1–9, 2017.

Oreskes, N., Shrader-Frechette, K., and Belitz, K.: Verification, Validation, and Confirmation of Numerical Models in the Earth Science, Science (New York, N.Y.), 263, 641–6, https://doi.org/10.1126/science.263.5147.641, 1994.

Paniconi, C. and Putti, M.: Physically based modeling in catchment hydrology at 50: Survey and outlook, Water Resources Research, 51, 7090–7129, https://doi.org/10.1002/2015WR017780, 2015.

Papamakarios, G. and Murray, I.: Fast $ε$-free Inference of Simulation Models with Bayesian Conditional Density Estimation, , https://doi.org/10.48550/ARXIV.1605.06376, 2016.

Papamakarios, G., Sterratt, D. C., and Murray, I.: Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows, , https://doi.org/10.48550/ARXIV.1805.07226, 2018.

Petropoulos, G. P., Ireland, G., and Barrett, B.: Surface soil moisture retrievals from remote sensing: Current status, products & future trends, Physics and Chemistry of the Earth, Parts A/B/C, 83, 36–56, 2015.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J.: Learning representations by back-propagating errors, Nature, 323, 533–536, https://doi.org/10.1038/323533a0, 1986.

Santos, N. and Patno, H.: Operations Plan for Colorado River Reservoirs, Bureau of Reclamation, 2022.

Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Gonçalves, P. J., Greenberg, D. S., and Macke, J. H.: sbi: A toolkit for simulation-based inference, Journal of Open Source Software, 5, 2505, https://doi.org/10.21105/joss.02505, 2020.

Tenney, W.: A Tier 2a Shortage is confirmed, but uncertainty remains, Arizona Municipal Water Users Association, 2022.

Tran, H., Leonarduzzi, E., De la Fuente, L., Hull, R. B., Bansal, V., Chennault, C., Gentine, P., Melchior, P., Condon, L. E., and Maxwell, R. M.: Development of a Deep Learning Emulator for a Distributed Groundwater–Surface Water Model: ParFlow-ML, Water, 13, https://doi.org/10.3390/w13233393, 2021.

Tran, H., Zhang, J., O'Neill, M. M., Ryken, A., Condon, L. E., and Maxwell, R. M.: A hydrological simulation dataset of the Upper Colorado River Basin from 1983 to 2019, Scientific Data, 9, 16, https://doi.org/10.1038/s41597-022-01123-w, 2022.

Tsai, W.-P., Feng, D., Pan, M., Beck, H., Lawson, K., Yang, Y., Liu, J., and Shen, C.: From calibration to parameter learning: Harnessing the scaling effects of big data in geoscientific modeling, Nature Communications, 12, 5988, https://doi.org/10.1038/s41467-021-26107-z, 2021.

Uria, B., Côté, M.-A., Gregor, K., Murray, I., and Larochelle, H.: Neural Autoregressive Distribution Estimation, Journal of Machine Learning Research, 17, 1–37, 2016.

Van Fraassen, B. C. and others: The scientific image, Oxford University Press, 1980.

Vrugt, J. A. and Sadegh, M.: Toward diagnostic model calibration and evaluation: Approximate Bayesian computation, Water Resources Research, 49, 4335–4345, https://doi.org/10.1002/wrcr.20354, 2013.

Weiss, G. and von Haeseler, A.: Inference of Population History Using a Likelihood Approach, Genetics, 149, 1539–46, https://doi.org/10.1093/genetics/149.3.1539, 1998.

White, J. T., Hunt, R. J., Fienen, M. N., and Doherty, J. E.: Approaches to highly parameterized inversion: PEST++ Version 5, a software suite for parameter estimation, uncertainty analysis, management optimization and sensitivity analysis, Reston, VA, https://doi.org/10.3133/tm7C26, 2020.

Wikle, C. K. and Berliner, L. M.: A Bayesian tutorial for data assimilation, Physica D: Nonlinear Phenomena, 230, 1–16, https://doi.org/10.1016/j.physd.2006.09.017, 2007.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J. G., Groth, P., Goble, C., Grethe, J. S., Heringa, J., 't Hoen, P. A. C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M. A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., and Mons, B.: The FAIR Guiding Principles for scientific data management and stewardship., Sci Data, 3, 160018, https://doi.org/10.1038/sdata.2016.18, 2016.

Williams, A. P., Cook, B. I., and Smerdon, J. E.: Rapid intensification of the emerging southwestern North American megadrought in 2020–2021, Nature Climate Change, 12, 232–234, https://doi.org/10.1038/s41558-022-01290-z, 2022.

Zhao, W. L., Gentine, P., Reichstein, M., Zhang, Y., Zhou, S., Wen, Y., Lin, C., Li, X., and Qiu, G. Y.: Physics-constrained machine learning of evapotranspiration, Geophysical Research Letters, 46, 14496–14507, 2019.