https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Sensitivity-Aware Gradient Estimation (SAGE) for Rapid Continental-Scale Training of Hydrologic Models

Vrugt Jasper A.[1] and Frame Jonathan M.[2]

[1]Department of Civil and Environmental Engineering, University of California, Irvine, CA 92697, USA.
[2]Department of Geological Sciences, University of Alabama, Tuscaloosa, AL 35487, USA.

Correspondence: Jasper A. Vrugt (jasper@uci.edu)

Abstract. We introduce SAGE (Sensitivity-Aware Gradient Estimation), a new framework for scalable and physics-consistent training of hydrologic models that leverages analytic forward sensitivities to enable exact and efficient gradient-based learning of model parameters from catchment attributes. Unlike existing approaches that rely on finite-difference approximations, automatic differentiation, or surrogate emulators, SAGE propagates exact derivatives through physically based dynamical systems using analytically derived sensitivity equations. This eliminates the need for repeated model evaluations, substantially reduces computational cost, and preserves the interpretability and structural integrity of process-based hydrologic models. We demonstrate SAGE in a large-sample hydrology experiment using the CAMELS data set, comprising 531 hydrologically valid catchments across the contiguous United States. A feedforward neural network maps static catchment attributes to the parameter space of a conceptual rainfall-runoff model, while exact gradients of the loss function with respect to network weights are computed through analytic sensitivity propagation of the governing ordinary differential equations. Compared to conventional training strategies based on numerical differentiation or automatic differentiation, SAGE achieves machine-precision agreement with reference gradients while reducing computational cost by several orders of magnitude. To assess cross-basin model performance, we further introduce a new integrated distributional skill score based on the empirical cumulative distribution function of Nash-Sutcliffe efficiency (NSE) values across basins. Rather than summarizing performance using a single quantile such as the median NSE, the proposed score quantifies the distance between the observed basin-wise NSE distribution and the ideal degenerate distribution at NSE = 1. This distributional skill score provides a more robust and informative measure of large-sample model skill and enables objective comparison of learning strategies at continental scale. Together, SAGE and the proposed Vrugt–Frame loss score form a unified framework for both training and evaluating physics-based hydrologic models in large-sample settings and offer a new pathway toward continental-scale, attribute-conditioned calibration that is both computationally tractable and physically interpretable.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

1 Introduction and Scope

Hydrologic models play a central role in understanding and predicting terrestrial water fluxes, supporting applications ranging from flood forecasting and drought monitoring to climate impact assessments and water resource management. Over the past decades, a wide spectrum of modeling paradigms has emerged, spanning from conceptual rainfall-runoff models (Gupta et al., 1998; Beven and Freer, 2001; Wagener et al., 2003), to distributed land surface models (Liang et al., 1994; Clark et al., 2015), to more recent data-driven and hybrid approaches (Kratzert et al., 2019b, a; Nearing et al., 2021; Reichstein et al., 2019). A persistent challenge across these paradigms is parameter estimation. Traditional calibration approaches treat each catchment independently, solving a nonlinear inverse problem that often involves expensive optimization or sampling procedures (Duan et al., 1992; Vrugt et al., 2003; Vrugt, 2016). These methods scale poorly to large spatial domains, particularly when long time series, high temporal resolution, or computationally intensive models are involved.

Recent years have witnessed growing interest in regionalization and attribute-conditioned parameter inference, where model parameters are expressed as functions of catchment attributes (Samaniego et al., 2010; Parajka et al., 2007; Beck et al., 2017; Kratzert et al., 2019c). These approaches aim to exploit shared hydrologic behavior across basins and to improve generalization to ungauged or data-scarce regions. In this context, artificial neural networks have emerged as a powerful nonlinear regression method, enabling flexible nonlinear mappings from physiographic attributes to model parameters or directly to hydrologic states and fluxes (Kratzert et al., 2018, 2019b; Feng et al., 2022). However, most existing approaches either replace physical models entirely or embed them in hybrid architectures where the training process relies on automatic differentiation or finite-difference approximations (Feng et al., 2022, 2023; Shen et al., 2023). These strategies face three fundamental limitations:

1. Scalability: Automatic differentiation through long time integrations can be memory-intensive and computationally expensive, particularly when applied to large-sample hydrology problems.

2. Numerical stability: Finite-difference approximations introduce truncation and round-off errors that can corrupt gradient-based learning.

3. Loss of physical transparency: Surrogate and black-box models obscure the causal structure encoded in physical equations.

Sensitivity analysis has long been a cornerstone of hydrologic modeling, supporting uncertainty quantification, parameter identifiability, and experimental design (Hornberger and Spear, 1981; Pianosi et al., 2016). In particular, forward sensitivity equations provide an exact and deterministic route to derivatives by augmenting the governing ODEs with additional sensitivity states that evolve alongside the physical states (Cacuci, 1981; Walter and Pronzato, 1997; Perumal and Gunawan, 2011; Vrugt et al., 2026). A key practical advantage is that state and sensitivity errors are controlled simultaneously by the same adaptive time-stepping and tolerance criteria, yielding stable gradients without relying on finite-difference perturbations that are prone to truncation and round-off effects. Adjoint-based alternatives are also available and can be attractive for very high-dimensional parameterizations, but forward sensitivities are especially well suited when one requires the full Jacobian with respect to a moderate number of parameters as in learning problems that repeatedly map basin attributes to parameter vectors across many catchments.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Motivated by these considerations, this paper leverages analytic forward sensitivities to make attribute-conditioned parameter learning both scalable and numerically robust. By coupling a neural network regionalization map with process-based hydrologic dynamics and exact Jacobians, the proposed framework avoids backpropagation through long integrations and reduces sensitivity to solver and differencing hyperparameters, while retaining the transparency of the underlying physical model structure.

Forward sensitivity equations have rarely been exploited as a core component of modern hydrologic learning workflows. Instead, most physics-informed and hybrid hydrologic learning frameworks rely on automatic differentiation (AD) (e.g., Raissi et al., 2019; Feng et al., 2022; Wang and Gupta, 2024), typically implemented through machine-learning libraries such as PyTorch (Paszke et al., 2019), JAX (Bradbury et al., 2018), and TensorFlow (Abadi et al., 2016). While these tools are powerful and convenient, they were developed primarily for general machine-learning applications and can be conceptually distant from the modeling practices and physical reasoning common in hydrology (Vrugt et al., 2026).

In AD-based workflows, hydrologic models are embedded in large computational graphs that obscure the underlying mathematics and make derivative operations difficult to inspect or interpret (Griewank and Walther, 2008). Moreover, the numerical solver is tightly coupled to the AD engine, so even minor changes in time stepping, state representation, or control flow may require costly re-tracing or recompilation of the computational graph (Innes et al., 2019; Rackauckas et al., 2020). These frameworks also impose nontrivial overhead in terms of memory usage, dependency management, and debugging complexity, particularly for long time series and stiff dynamical systems (Griewank and Walther, 2008; Rackauckas et al., 2020). As a result, although AD provides exact derivatives, it often does so in a way that limits transparency, flexibility, and computational efficiency for process-based hydrologic models. This motivates the development of alternative approaches that retain mathematical clarity while enabling scalable gradient-based learning.

In this article, we propose Sensitivity-Aware Gradient Estimation (SAGE), a new framework that integrates analytic forward sensitivities into the training of neural networks that parameterize process-based hydrologic models (see Figure 1). In SAGE:

1. A neural network maps static catchment attributes to model parameters.
2. A process-based hydrologic model evolves according to its governing differential equations.
3. Forward sensitivity equations propagate exact derivatives of model states with respect to parameters.
4. The chain rule is applied analytically to compute exact gradients of the loss function with respect to network weights and biases.

This eliminates the need for finite-difference perturbations, reverse-mode automatic differentiation through time and surrogate emulation of physical models. As a result, SAGE offers machine-precision gradients at a dramatically reduced computational cost with preservation of physical interpretability and scalability to continental-scale problems.

The remainder of this paper develops the proposed sensitivity-aware learning framework from theory to large-scale application. Section 2 establishes sensitivity analysis as the foundation for learning in dynamical hydrologic systems, deriving the chain-rule decomposition that links neural-network parameter mappings, analytic forward sensitivities, and process-based hydrologic models. This section also specifies the ANN architecture and its analytic Jacobian, the training procedure, and the loss functions

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

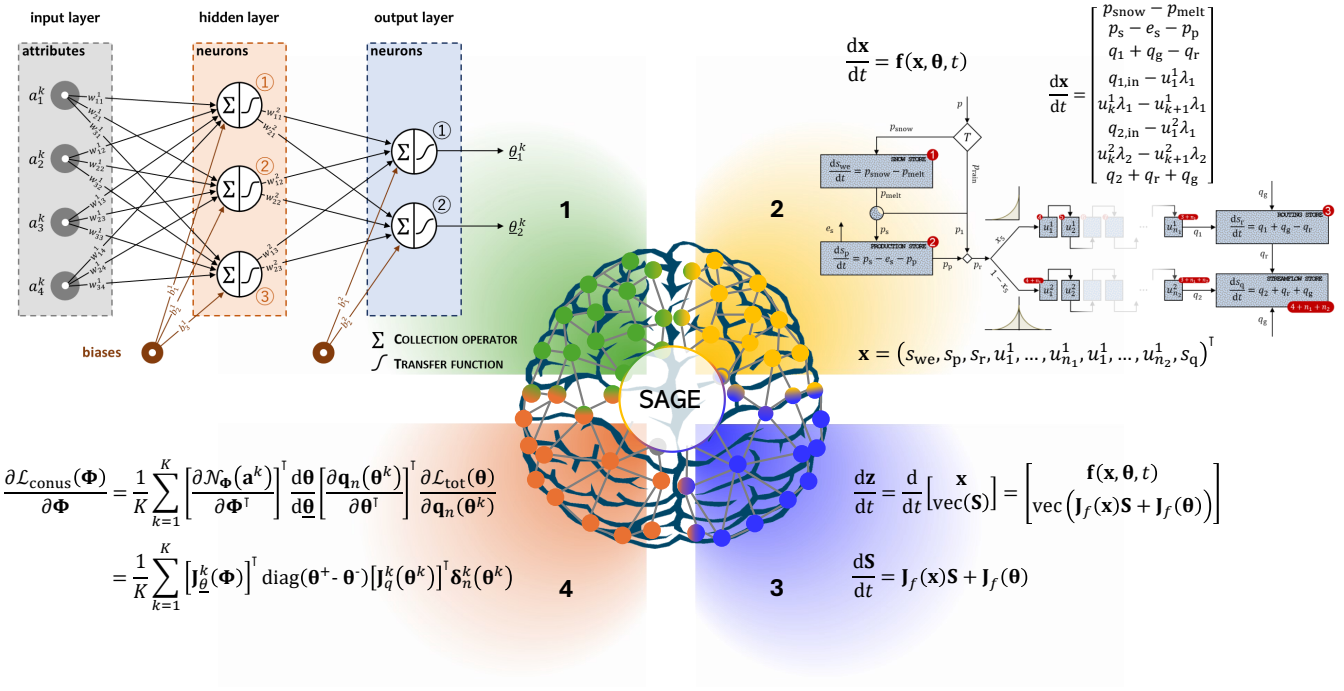

Figure 1. Overview of the SAGE framework. A neural network maps static catchment attributes to hydrologic model parameters. The process-based model is integrated forward in time together with its associated forward sensitivity equations, yielding both hydrologic states and exact parameter Jacobians in a single solve. These sensitivities are combined analytically with the loss gradient using the chain rule to compute exact gradients with respect to network weights, enabling efficient and interpretable training.

used herein. Building on this foundation, Section 3 introduces an integrated basin loss score as a more informative alternative to pointwise summaries of large-sample performance (e.g., the median Nash–Sutcliffe efficiency). Section 4 then describes the CONUS/CAMELS data and details the experimental design and model implementations used in Parts A–C, while Section 5 documents the SAGE software implementation in MATLAB and C++. Section 6 presents results for six conceptual hydrologic models under traditional single-site calibration (Part A), temporal validation with SAGE (Part B), and spatial cross-validation with SAGE (Part C). Section 7 synthesizes the main insights and discusses SAGE implications and limitations. Section 8 discusses the broader implications of distributional model evaluation and sensitivity-based learning for large-sample hydrology and outlines promising directions for future research. We conclude this paper in Section 9 with a summary of our main findings. Complete mathematical formulations of the hydrologic models and their analytic sensitivity equations are provided in Appendices A and B to ensure transparency and reproducibility.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

2 Sensitivity analysis as a foundation for learning

2.1 Chain-rule decomposition of the joint gradient

The methodological foundation of this study follows earlier theoretical developments on forward sensitivity analysis (sensitivity equations augmented ODE systems) of hydrologic models. Here, we provide a brief overview and refer the reader to our companion theory paper (Vrugt et al., 2026) for full derivations and descriptions.

Suppose we have hydrologic data $\mathcal{D}$ and static catchment attributes $\mathcal{A} \in \mathbb{R}^{r\times K}$ of K watersheds. For each basin $k = 1,\dots,K$, we wish to train a hydrologic model $\mathbf{y}_n^k = \mathcal{M}(\mathcal{D}^k;\boldsymbol{\theta}^k)$ to describe as closely and consistently as possible a historical record of measured daily river discharge $\mathbf{q}_n^k = (q_1^k,\dots,q_n^k)^\top$. This necessitates estimating the parameters $\boldsymbol{\theta}^k = (\theta_1^k,\dots,\theta_d^k)^\top$ for each kth basin. We wish to minimize the distance between the observed and simulated streamflows. We can express this distance using a so-called pointwise loss function $\mathcal{L}_t(\cdot)$

$$\mathcal{L}_{\text{tot}}^k(\boldsymbol{\theta}^k) = \sum_{t=1}^{n} \mathcal{L}_t(y_t^k, q_t^k),$$

where $y_1^k,\dots,y_n^k$ are simulated streamflows for basin k under parameter vector $\boldsymbol{\theta}^k$ and $\mathcal{L}_{\text{tot}}^k(\cdot)$ is referred to as the total loss for basin k. Commonly used pointwise loss functions are the absolute error $\mathcal{L}_t = |y_t - q_t|$ and squared error $\mathcal{L}_t = \frac{1}{2}(y_t - q_t)^2$, and reward-based goodness-of-fit metrics such as the Nash-Sutcliffe efficiency or NSE (Nash and Sutcliffe, 1970) and Kling-Gupta efficiency (KGE) (Gupta et al., 2009; Kling et al., 2012).

Instead of performing independent calibrations for each basin, we train an artificial neural network (ANN) whose weights and biases are optimized to predict basin-specific parameter sets from catchment attributes (Feng et al., 2022; Shen et al., 2023). We can cast this ANN in a regression framework and write

$$\underline{\boldsymbol{\Theta}} = \mathcal{N}_{\boldsymbol{\Phi}}(\mathcal{A})$$

where $\underline{\boldsymbol{\Theta}} = [\underline{\boldsymbol{\theta}}^1 \dots \underline{\boldsymbol{\theta}}^K] \in [\mathbf{0},\mathbf{1}]$ is a $d\times K$ matrix of normalized hydrologic parameter values, $\mathcal{A}$ is the $r\times K$ matrix of catchment attributes

$$\mathcal{A} = \begin{bmatrix} \mathbf{a}^1 & \mathbf{a}^2 & \dots & \mathbf{a}^K \end{bmatrix} = \begin{bmatrix} a_1^1 & a_1^2 & \dots & a_1^K \\ a_2^1 & a_2^2 & \dots & a_2^K \\ \vdots & \vdots & \ddots & \vdots \\ a_r^1 & a_r^2 & \dots & a_r^K \end{bmatrix} \in \mathbb{R}^{r\times K}$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

and $\boldsymbol{\phi}$ is a $l \times 1$-vector of network weights and biases that link basin attributes $\mathcal{A}$ to hydrologic parameter values. $\underline{\boldsymbol{\Theta}}$ can be turned into a equal-sized matrix of hydrologic parameter values $\boldsymbol{\Theta} = [\boldsymbol{\theta}^1 \dots \boldsymbol{\theta}^K]$ using the affine transformation

$$\boldsymbol{\theta}^k = \boldsymbol{\theta}^- + \underline{\boldsymbol{\theta}}^k \odot (\boldsymbol{\theta}^+ - \boldsymbol{\theta}^-) \tag{1}$$

where $\boldsymbol{\theta}^-$ and $\boldsymbol{\theta}^+$ are $d \times 1$ vectors with minimum and maximum values of the hydrologic parameters, $\odot$ is the element-wise (Hadamard) product and $k = 1, \dots, K$. The Jacobian of this component-wise affine transformation is

$$\begin{aligned} \mathbf{J}_\theta(\underline{\boldsymbol{\theta}}^k) = \frac{\mathrm{d}\boldsymbol{\theta}^k}{\mathrm{d}(\underline{\boldsymbol{\theta}}^k)^\top} &= \begin{bmatrix} \theta_1^+ - \theta_1^- & 0 & \dots & 0 \\ 0 & \theta_2^+ - \theta_2^- & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \dots & \theta_d^+ - \theta_d^- \end{bmatrix} \in \mathbb{R}^{d \times d} \\ &= \mathrm{diag}(\boldsymbol{\theta}^+ - \boldsymbol{\theta}^-). \end{aligned} \tag{2}$$

For each watershed k, an ODE system comprised of state equations augmented with sensitivity equations is integrated forward in time using a 2[nd]-order Runge Kutta numerical solver and returns simulated discharge $\mathbf{q}_n^k = (q_1^k, \dots, q_n^k)^\top$ and a $n \times d$ Jacobian matrix $\mathbf{J}_q^k(\boldsymbol{\theta}^k)$ of first-order partial derivatives of $q_1^k, \dots, q_n^k$ with respect to model parameters $\theta_1^k, \dots, \theta_d^k$. Theory and application of this procedure are described in Vrugt et al. (2026) and interested readers are referred to this publication for further details. Given observed streamflows $\mathbf{y}_n^k = (y_1^k, \dots, y_n^k)^\top$ we compute the total loss $\mathcal{L}_{\mathrm{tot}}^k(\boldsymbol{\theta}^k)$ for basin k and its associated gradient (Vrugt et al., 2026)

$$\mathbf{g}_n^k(\boldsymbol{\theta}^k) = \frac{\partial \mathcal{L}_{\mathrm{tot}}^k(\boldsymbol{\theta}^k)}{\partial \boldsymbol{\theta}^k} = \left[\mathbf{J}_q^k(\boldsymbol{\theta}^k)\right]^\top \boldsymbol{\delta}_n^k(\boldsymbol{\theta}^k), \tag{3}$$

where $\boldsymbol{\delta}_n^k(\boldsymbol{\theta}^k) \in \mathbb{R}^n$ is referred to as the *loss-sensitivity vector* or *error-propagation vector*. The above expression shows that the gradient vector $\mathbf{g}_n(\boldsymbol{\theta}) \in \mathbb{R}^{d \times 1}$ decomposes into two components: (i) the discharge Jacobian $\mathbf{J}_q(\boldsymbol{\theta})$, which depends only on the hydrologic model and its parameter values and (ii) the loss-sensitivity vector $\boldsymbol{\delta}_n(\boldsymbol{\theta})$, which depends exclusively on the form of the loss function and encodes how discrepancies between observations and simulations are propagated back through the model. Importantly, for any differentiable pointwise loss function $\mathcal{L}(y_t, q_t)$, we can derive the entries of $\boldsymbol{\delta}_n(\boldsymbol{\theta})$ by analytic means. Table 1 of Vrugt et al. (2026) lists such analytic expressions for commonly used goodness-of-fit metrics such as the sum of absolute residuals, weighted sum of squared residuals, NSE, KGE and less well-known objective functions including Huber loss (Huber, 1964; Huber and Ronchetti, 2011) and the flow duration curve score divergence (Vrugt, 2024). Thus, with negligible additional computational cost, analytic gradients $\mathbf{g}_n^1, \dots, \mathbf{g}_n^K$ of the basin-specific total losses $\mathcal{L}_{\mathrm{tot}}^k$ with respect to the hydrologic model parameters $\theta_1, \dots, \theta_d$ are obtained for all K watersheds.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

The foregoing development yields analytic gradients of the loss function for each individual basin. However, training the neural network requires gradients (i) of an aggregate objective function defined over all K watersheds and (ii) with respect to the network weights and biases $\boldsymbol{\phi}$. With respect to (i), we introduce the mean aggregate loss

$$\mathcal{L}_{\text{conus}}(\boldsymbol{\phi}) = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{\text{tot}}^{k}\big(\boldsymbol{\theta}^{k}(\boldsymbol{\phi})\big), \tag{4}$$

which summarizes cross-basin model performance in a single differentiable objective function. Normalizing by K ensures that gradient magnitudes remain approximately invariant with respect to batch size, thereby improving numerical stability and reducing sensitivity to learning-rate selection.

The summation in Equation 4 is mathematically convenient and plays a pivotal role in enabling scalable, attribute-conditioned learning across hundreds of catchments. For loss functions such as the sum of absolute residuals, weighted residual sum of squares and the Huber loss, the additive form of $\mathcal{L}_{\text{conus}}$ is mathematically equivalent to computing the loss on the pooled residual vector across all catchments. In these cases, Equation (4) corresponds to a joint residual norm or, under suitable assumptions, a negative log-likelihood function, and yields gradients that are consistent with maximum likelihood or robust estimation principles. This composite loss will naturally emphasize basins with larger residual variance and is sensitive to the temporal structure of the discharge residuals. However, for normalized or nonlinear performance metrics such as the NSE, KGE, and the flow-duration-curve divergence score, simple averaging of per-catchment values is generally *not* equivalent to computing the metric on the pooled discharge records or pooled residual time series. These metrics rely on catchment-specific moments (e.g., variance and mean), and averaging them implicitly changes the relative weighting of residuals across basins. Composite training under these metrics defines a distributional performance objective rather than a likelihood-based objective, emphasizing uniform skill across heterogeneous catchments instead of strict minimization of discharge residuals. The consequences of loss function aggregation in large-sample learning are discussed further in Section 8.

For each basin k, the ANN maps static attributes $\mathbf{a}^k$ to normalized hydrologic parameters $\underline{\boldsymbol{\theta}}^k \in [0,1]^d$, which are then rescaled to physical parameters $\boldsymbol{\theta}^k \in \mathbb{R}^d$ using Equation 1. During training, the per-catchment total loss function $\mathcal{L}_{\text{tot}}$ depends on the simulated discharge $\mathbf{q}_n$, which itself depends on the hydrologic model parameters $\boldsymbol{\theta}$. Hence, gradients with respect to the network parameters $\boldsymbol{\phi}$ are obtained by repeated application of the chain rule to Equation 4. For a single watershed we can write out the chain rule to yield

$$\frac{\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}} = \overbrace{\underbrace{\left[\frac{\partial \mathcal{N}_{\boldsymbol{\phi}}(\mathbf{a})}{\partial \boldsymbol{\phi}^{\top}}\right]^{\top}}_{\mathbf{J}_{\underline{\theta}}(\boldsymbol{\phi})}}^{l\times d}\overbrace{\underbrace{\left[\frac{\mathrm{d}\boldsymbol{\theta}}{\mathrm{d}\underline{\boldsymbol{\theta}}^{\top}}\right]}_{\mathbf{J}_{\theta}(\underline{\boldsymbol{\theta}})}}^{d\times d}\overbrace{\underbrace{\left[\frac{\partial \mathbf{q}_n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^{\top}}\right]^{\top}}_{\mathbf{J}_{q}(\boldsymbol{\theta})}}^{d\times n}\overbrace{\underbrace{\left[\frac{\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\theta})}{\partial \mathbf{q}_n(\boldsymbol{\theta})}\right]}_{\boldsymbol{\delta}_n(\boldsymbol{\theta})}}^{n\times 1}$$

$$= \mathbf{J}_{\underline{\theta}}^{\top}(\boldsymbol{\phi})\mathbf{J}_{\theta}(\underline{\boldsymbol{\theta}})\mathbf{J}_{q}^{\top}(\boldsymbol{\theta})\boldsymbol{\delta}_n(\underline{\boldsymbol{\theta}}) \tag{5}$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

where $\mathbf{J}_{\underline{\theta}}(\boldsymbol{\phi}) = \partial \mathcal{N}_{\boldsymbol{\phi}}(\mathbf{a}) / \partial \boldsymbol{\phi}^{\top}$ is the $d \times l$ Jacobian matrix of the ANN-predicted normalized hydrologic parameter values $\underline{\boldsymbol{\theta}}$ with respect to its weights and biases $\boldsymbol{\phi}$.

The above decomposition expresses the gradient vector $\partial \mathcal{L}_{\text{conus}}(\boldsymbol{\phi}) / \partial \boldsymbol{\phi} \in \mathbb{R}^{l}$ as a sequence of Jacobian–vector multiplications that propagates sensitivities from discharge, through the hydrologic parameters, and back to the ANN weights and biases without finite differences or automatic differentiation of the hydrologic core. By standard matrix-algebra rules, the dimensions are compatible at each step, and the product maps ultimately to a single gradient vector in parameter space. Indeed, multiplying an $l \times d$ matrix by a $d \times d$ matrix, then by a $d \times n$ matrix, and finally by an $n \times 1$ vector yields an $l \times 1$ vector. Hence, the chain-rule decomposition collapses naturally into one compact matrix–matrix–matrix–vector product that returns the joint-loss gradient with respect to all l network parameters $\boldsymbol{\phi}$.

In practice, the gradient $\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi}) / \partial \boldsymbol{\phi}$ need not be computed by explicitly constructing intermediate Jacobian matrices. Starting from the chain rule in Equation 5, we exploit associativity by working with the transpose form $\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi}) / \partial \boldsymbol{\phi}^{\top} \in \mathbb{R}^{1 \times l}$ of the gradient $\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi}) / \partial \boldsymbol{\phi}$. Using $(\mathbf{ABCD})^{\top} = \mathbf{D}^{\top}\mathbf{C}^{\top}\mathbf{B}^{\top}\mathbf{A}^{\top}$, we obtain

$$\frac{\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^{\top}} = \boldsymbol{\delta}_n^{\top}(\boldsymbol{\theta})\, \mathbf{J}_q(\boldsymbol{\theta})\, \mathbf{J}_{\theta}(\underline{\boldsymbol{\theta}})\, \mathbf{J}_{\underline{\theta}}(\boldsymbol{\phi}),$$

and then form the loss-sensitivity vector-Jacobian product

$$\mathbf{v}_n(\boldsymbol{\theta}) = \boldsymbol{\delta}_n^{\top}(\boldsymbol{\theta})\, \mathbf{J}_q(\boldsymbol{\theta}) \in \mathbb{R}^{1 \times d}, \tag{6}$$

which is the row-vector representation of the parameter-space gradient in Equation 3, i.e., $\mathbf{v}_n(\boldsymbol{\theta}) = \mathbf{g}_n^{\top}(\boldsymbol{\theta})$. Importantly, $\mathbf{v}_n(\boldsymbol{\theta})$ can be evaluated directly from the forward-sensitivity states by contracting them with $\boldsymbol{\delta}_n(\boldsymbol{\theta})$, without explicitly forming the discharge Jacobian $\mathbf{J}_q(\boldsymbol{\theta})$. The remaining factors are then applied as

$$\frac{\partial \mathcal{L}_{\text{tot}}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^{\top}} = \mathbf{v}_n(\boldsymbol{\theta}) \operatorname{diag}(\boldsymbol{\theta}^{+} - \boldsymbol{\theta}^{-})\, \mathbf{J}_{\underline{\theta}}(\boldsymbol{\phi}), \tag{7}$$

where $\mathbf{J}_{\underline{\theta}}(\boldsymbol{\phi})$ is applied implicitly via analytic backpropagation through the ANN. This matrix-free evaluation avoids storing large dense arrays while preserving exactness because all products are obtained from analytic sensitivity dynamics and closed-form network derivatives.

For $K > 1$ watersheds, the gradient of the multi-basin mean loss $\mathcal{L}_{\text{conus}}(\boldsymbol{\phi})$ is simply the average of the basin-wise gradients

$$\frac{\partial \mathcal{L}_{\text{conus}}(\boldsymbol{\phi})}{\partial \boldsymbol{\phi}^{\top}} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{v}_n^{k}(\boldsymbol{\theta}^{k})\, \operatorname{diag}(\boldsymbol{\theta}^{+} - \boldsymbol{\theta}^{-})\, \mathbf{J}_{\underline{\theta}}^{k}(\boldsymbol{\phi}), \tag{8}$$

with $\mathbf{v}_n^{k}(\boldsymbol{\theta}^{k}) = [\boldsymbol{\delta}_n^{k}(\boldsymbol{\theta}^{k})]^{\top} \mathbf{J}_q^{k}(\boldsymbol{\theta}^{k}) \in \mathbb{R}^{1 \times d}$. The summation follows directly from the additive form of the joint loss in Equation 4.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Remark.

Equations 6-8 describe a matrix-free (Jacobian-vector) evaluation of the network gradient. In the present SAGE software, however, we implement the explicit chain-rule factorization of Equation 5 in terms of the individual Jacobians. This choice keeps the hydrologic model core fully decoupled from the learning objective. The $n \times d$ discharge Jacobian $\mathbf{J}_q(\boldsymbol{\theta})$ and the $n \times 1$ loss-sensitivity vector $\boldsymbol{\delta}_n(\boldsymbol{\theta})$ are formed and combined *outside* the forward model using measured and simulated discharges and the forward-sensitivity states (Vrugt et al., 2026).

A fully matrix-free implementation would instead require forming the vector-Jacobian product $\mathbf{v}_n(\boldsymbol{\theta}) = \boldsymbol{\delta}_n^\top(\underline{\boldsymbol{\theta}})\mathbf{J}_q(\boldsymbol{\theta})$ on-the-fly as the sensitivity states become available from the ODE solver. This would necessitate passing the observed discharge time series into the hydrologic model so that each component of $\boldsymbol{\delta}_n(\boldsymbol{\theta})$ can be evaluated as soon as discharge q_t is produced, and immediately contracted with the corresponding sensitivity rows. While straightforward in principle, we defer this refactorization to avoid complicating the source code of the hydrologic model suite and to preserve a clean separation between process model and learning machinery. Future SAGE releases will include an optional matrix-free mode. For daily applications, any additional speed-up is expected to be negligible because $\mathbf{J}_q(\boldsymbol{\theta})$ and $\boldsymbol{\delta}_n(\boldsymbol{\theta})$ are comparatively small. For multi-year, hourly time series, however, the memory and runtime savings can be more substantial.

2.2 Specific ANN architecture and analytic network Jacobian

By the universal approximation theorem, a neural network of three layers with sufficiently many neurons in the second layer can approximate any continuous function on a compact domain to an arbitrary degree of accuracy (Cybenko, 1989; Hornik et al., 1989; Hornik, 1991). Figure 2 displays such feed-forward network or multi-layer perceptron. The first (input) layer consists of $r = 4$ catchment attributes, which are connected to $h = 3$ neurons in the second (hidden) layer. These neurons, in turn, connect to $d = 2$ nodes in the third layer. This output layer represents the normalized values of the hydrologic model parameters. This 4-3-2 architecture is intentionally simple and is used solely for illustrative purposes. In our numerical experiments, we employ substantially larger values of r, h and d.

Let $\mathcal{N}_{\boldsymbol{\Phi}} : \mathbb{R}^r \rightarrow [0,1]^d$ denote a three-layer feed-forward network or multi-layer perceptron. Given a vector of static catchment attributes $\mathbf{a}^k \in \mathbb{R}^r$ for basin k the network predicts a vector of normalized hydrologic model parameters $\underline{\boldsymbol{\theta}}^k \in [0,1]^d$. The hidden layer employs a nonlinear activation function, either the hyperbolic tangent

$$\tanh(z) = \frac{\exp(2z) - 1}{\exp(2z) + 1}$$

or the rectified linear unit

$$\mathrm{ReLU}(z) = \max(0, z),$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

while the output layer uses a logistic sigmoid transfer function

$$\sigma(z) = \frac{1}{1+\exp(-z)},$$

to ensure that all predicted normalized parameters lie within the unit hypercube.

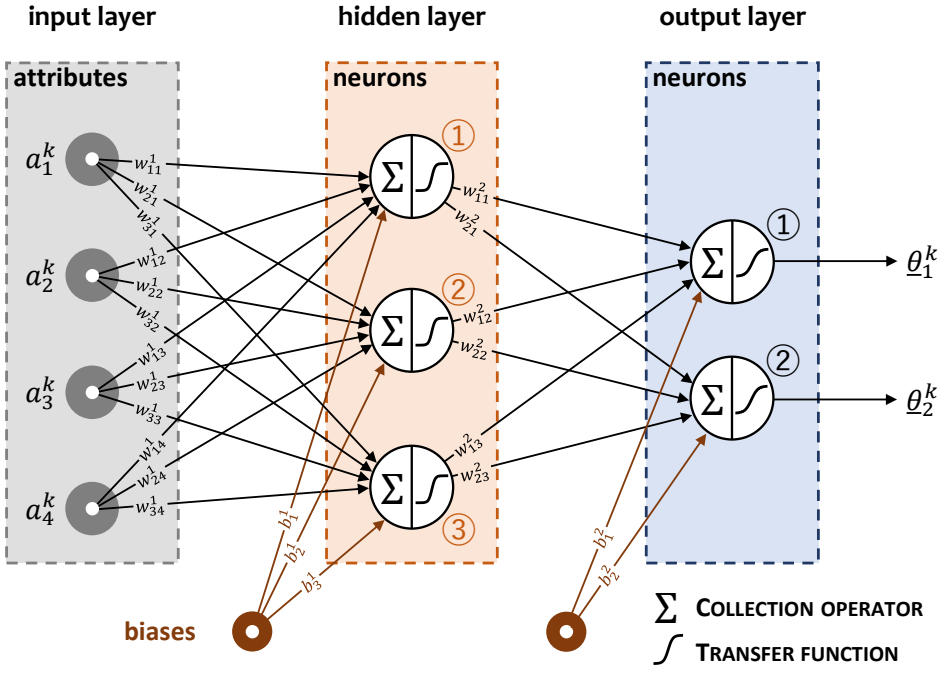

Figure 2. Illustrative feedforward neural network with $r = 4$ static catchment attributes as inputs, a single hidden layer with $h = 3$ neurons, and an output layer that predicts the normalized values of $d = 2$ hydrologic model parameters. The hidden layer uses either a hyperbolic tangent activation, $\tanh : \mathbb{R} \to [0, 1]$, or a rectified linear unit, $\mathrm{ReLU} : \mathbb{R} \to [0, \infty)$, while the output layer applies a logistic sigmoid, $\sigma : \mathbb{R} \to [0, 1]$, to constrain predictions to the unit interval. Network weights and biases are learned jointly across all basins by minimizing the multi-basin loss function in Equation 4 using the SAGE framework coupled with gradient-descent training.

The network is fully connected and parameterized by

$$\boldsymbol{\phi} = \left\{ \mathbf{W}^1 \in \mathbb{R}^{h \times r},\ \mathbf{b}^1 \in \mathbb{R}^h,\ \ \mathbf{W}^2 \in \mathbb{R}^{d \times h},\ \mathbf{b}^2 \in \mathbb{R}^d \right\},$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

where $\mathbf{W}^1$ and $\mathbf{b}^1$ store the weights and biases of the first (hidden) layer

$$\mathbf{W}^1 = \begin{bmatrix} w_{11}^1 & w_{12}^1 & \dots & w_{1r}^1 \\ w_{21}^1 & w_{22}^1 & \dots & w_{2r}^1 \\ \vdots & \vdots & \ddots & \vdots \\ w_{h1}^1 & w_{h2}^1 & \dots & w_{hr}^1 \end{bmatrix} \in \mathbb{R}^{h \times r} \qquad \mathbf{b}^1 = \begin{bmatrix} b_1^1 \\ b_2^1 \\ \vdots \\ b_h^1 \end{bmatrix} \in \mathbb{R}^{h \times 1},$$

and $\mathbf{W}^2$ and $\mathbf{b}^2$ denote the weights and biases of the second (output) layer

$$\mathbf{W}^2 = \begin{bmatrix} w_{11}^2 & w_{12}^2 & \dots & w_{1h}^2 \\ w_{21}^2 & w_{22}^2 & \dots & w_{2h}^2 \\ \vdots & \vdots & \ddots & \vdots \\ w_{d1}^2 & w_{d2}^2 & \dots & w_{dh}^1 \end{bmatrix} \in \mathbb{R}^{d \times h} \qquad \mathbf{b}^2 = \begin{bmatrix} b_1^2 \\ b_2^2 \\ \vdots \\ b_d^2 \end{bmatrix} \in \mathbb{R}^{d \times 1}.$$

The total number of trainable ANN parameters is therefore equal to

$$l = hr + h + dh + d = h(r+1) + d(h+1). \tag{9}$$

A three-layer network is relatively easy to train and the hidden layer width h can be adjusted to maximize hydrologic model performance. Deeper architectures with multiple hidden layers may be beneficial when the attribute-parameter relationship is strongly hierarchical or compositional, highly nonlinear across different regimes or when r is on the order of hundreds or more. With CAMELS-attributes, this is rarely necessary.

Forward pass:

For our three-layer network, the forward mapping for the joint set of K basins is given by

$$\begin{aligned} \mathbf{Z}^1 &= \mathbf{W}^1 \mathcal{A} + \mathbf{b}^1 \mathbf{1}_K^\top, & \mathbf{Z}^1 &\in \mathbb{R}^{h \times K}, \\ \mathbf{H} &= \boldsymbol{\psi}(\mathbf{Z}^1), & \mathbf{H} &\in \mathbb{R}^{h \times K}, \\ \mathbf{Z}^2 &= \mathbf{W}^2 \mathbf{H} + \mathbf{b}^2 \mathbf{1}_K^\top, & \mathbf{Z}^2 &\in \mathbb{R}^{d \times K}, \\ \underline{\boldsymbol{\Theta}} &= \boldsymbol{\sigma}(\mathbf{Z}^2), & \underline{\boldsymbol{\Theta}} &\in [0,1]^{d \times K}, \end{aligned}$$

where $\boldsymbol{\psi}(\cdot)$ is the element-wise hidden-layer activation (e.g., $\tanh$ or ReLU), $\boldsymbol{\sigma}(\cdot)$ is the matrix-valued sigmoidal output activation function enforcing $\underline{\boldsymbol{\Theta}} \in [0,1]^d$, and, again, $\mathcal{A}$ is the $r \times K$ matrix of catchment attributes. The $1 \times K$ vector of ones,

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

$\mathbf{1}_K^\top$, expands the bias vectors $\mathbf{b}^1 \in \mathbb{R}^h$ and $\mathbf{b}^2 \in \mathbb{R}^d$ into $h \times K$ and $d \times K$ matrices, respectively, so that the biases are added column-wise, following standard broadcasting conventions.

Block form of the network Jacobian:

Matrix $\mathbf{J}_{\underline{\theta}}^k(\boldsymbol{\phi})$ in Equation 8 can be written explicitly in four blocks corresponding to the weights and biases of the input and hidden layers

$$\mathbf{J}_{\underline{\theta}}^k(\boldsymbol{\phi}) = \frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial \boldsymbol{\phi}^\top} = \left[\frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial \mathrm{vec}(\mathbf{W}^1)^\top} \quad \frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial (\mathbf{b}^1)^\top} \quad \frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial \mathrm{vec}(\mathbf{W}^2)^\top} \quad \frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial (\mathbf{b}^2)^\top} \right] \in \mathbb{R}^{d \times l}, \tag{10}$$

where $\mathrm{vec}(\cdot)$ denotes column-wise vectorization. This structured Jacobian preserves the interpretation of $\mathbf{J}_{\underline{\theta}}^k(\boldsymbol{\phi})$ as a $d \times l$ matrix, while explicitly reflecting the layered architecture of the neural network.

The different blocks of $\mathbf{J}_{\underline{\theta}}^k(\boldsymbol{\phi})$ can be computed analytically. Let

$$\begin{aligned} \mathbf{z}^{1,k} &= \mathbf{W}^1 \mathbf{a}^k + \mathbf{b}^1, & \mathbf{z}^{1,k} &\in \mathbb{R}^h, \\ \mathbf{h}^k &= \boldsymbol{\psi}(\mathbf{z}^{1,k}), & \mathbf{h}^k &\in \mathbb{R}^h, \\ \mathbf{z}^{2,k} &= \mathbf{W}^2 \mathbf{h}^k + \mathbf{b}^2, & \mathbf{z}^{2,k} &\in \mathbb{R}^d, \end{aligned}$$

be the pre-activations and hidden activations for basin k. Furthermore, let the diagonal matrices $\mathbf{D}_\psi^k = \mathrm{diag}\big(\boldsymbol{\psi}'(\mathbf{z}^{1,k})\big) \in \mathbb{R}^{h \times h}$ and $\mathbf{D}_\sigma^k = \mathrm{diag}\big(\boldsymbol{\sigma}'(\mathbf{z}^{2,k})\big) \in \mathbb{R}^{d \times d}$ be the square Jacobians of the hidden- and output-layer activation functions, respectively, and

$$\psi'(z) = \begin{cases} 1 - \tanh^2(z), & \text{tanh}, \\ \mathbb{I}(z > 0), & \text{ReLU}, \end{cases}$$

$$\sigma'(z) = \sigma(z)\big(1 - \sigma(z)\big),$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

are the analytic derivatives of the tanh and ReLU transfer functions of the hidden layer and sigmoidal activation of the output layer. The four Jacobian blocks in Equation 10 now equal

$$\frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial \operatorname{vec}(\mathbf{W}^1)^\top} = \mathbf{D}_\sigma^k \mathbf{W}^2 \mathbf{D}_\psi^k \left(\mathbf{I}_h \otimes \left[\mathbf{a}^k\right]^\top \right) \in \mathbb{R}^{d \times (h \cdot r)}, \tag{11}$$

$$\frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial (\mathbf{b}^1)^\top} = \mathbf{D}_\sigma^k \mathbf{W}^2 \mathbf{D}_\psi^k \in \mathbb{R}^{d \times h}, \tag{12}$$

$$\frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial \operatorname{vec}(\mathbf{W}^2)^\top} = \mathbf{D}_\sigma^k \left(\mathbf{I}_d \otimes \left[\mathbf{h}^k\right]^\top \right) \in \mathbb{R}^{d \times (d \cdot h)}, \tag{13}$$

$$\frac{\partial \underline{\boldsymbol{\theta}}^k}{\partial (\mathbf{b}^2)^\top} = \mathbf{D}_\sigma^k \in \mathbb{R}^{d \times d}, \tag{14}$$

where $\otimes$ signifies the Kronecker product and $\mathbf{I}_p$ is the $p \times p$ identity matrix. Each output component depends linearly on all weights connected to it, and the Kronecker products arrange these partial derivatives into the appropriate block structure. The dimensions of the individual blocks confirm that $\mathbf{J}_{\underline{\theta}}^k(\boldsymbol{\phi})$ has d rows and $l = hr + h + dh + d$ columns, matching the total number of unknown ANN parameters in Equation 9.

Connection to backpropagation:

Equations 11–14 are equivalent to standard backpropagation for a single hidden layer, and match the implementation used in our MATLAB `demo_SAGE` Live Script. When inserted into Equation 8, they provide an analytic mapping from hydrologic model gradients $\mathbf{g}_n^k(\boldsymbol{\theta}^k)$ to gradients with respect to the network weights and biases, enabling end-to-end training without numerical differentiation of the hydrologic core.

2.3 Neural network training

The weights and biases of the neural network, collected in the parameter vector $\boldsymbol{\phi}$, are optimized using the Adam algorithm (Kingma and Ba, 2015). Adam is an adaptive first-order optimization method that rescales gradient updates using exponentially weighted estimates of the first and second moments of the gradients, thereby improving numerical stability and accelerating convergence. Appendix A provides an algorithmic description of our implementation. Compared to the original formulation of Kingma and Ba (2015), our implementation incorporates two modifications: (i) gradients are averaged over the number of training basins K, rendering the updates invariant to mini-batch size, and (ii) ℓ_2-norm clipping is applied to prevent occasional spikes in the hydrologic Jacobian from destabilizing network training.

During training, the network weights and biases are updated iteratively using gradients propagated from the hydrologic model through the differentiable mapping between catchment attributes and model parameters. This end-to-end training framework enables the network to learn parameterizations that minimize the chosen hydrologic loss function jointly across all basins.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

2.4 Hydrologic models and analytic gradients

We consider six widely used conceptual hydrologic models: `hymod`, `hmodel`, `sacsma`, `Xinanjiang`, `gr4j`, and `hbv`. Each model is formulated as an augmented system of ordinary differential equations (ODEs) that couples the governing state equations with their corresponding forward sensitivity equations. These coupled systems are integrated using explicit Runge–Kutta time stepping schemes implemented in both MATLAB and `C++`. The resulting discharge simulations and analytic Jacobians are returned to MATLAB, where they are used for end-to-end neural network training within the SAGE framework.

Complete model descriptions of `gr4j` and `hbv`, including their state equations and analytic sensitivity formulations, are provided in Appendix B. Detailed ODE and sensitivity formulations for `hymod`, `hmodel`, `sacsma`, and `Xinanjiang` are given in Appendix B of Vrugt et al. (2026) and are therefore not repeated here. The original formulations of these four models did not include explicit representations of snow accumulation. We briefly explain this module below.

2.4.1 Smooth temperature-index snow module

We adopt a temperature-index (degree-day) snow formulation that is widely used in conceptual hydrologic models (Anderson, 1973; Martinec et al., 1992; Bergström, 1995; Perrin et al., 2003). Let s_{we} (mm) denote snow water equivalent (SWE). At each time t, precipitation p_t is partitioned into snowfall p_{s} and rainfall p_{r} using a smooth transition around a temperature threshold T_{tr} $(^\circ\mathrm{C})$

$$f_{\mathrm{snow}}(T_t) = \frac{1}{2}\left[1 - \tanh\left(\frac{T_t - T_{\mathrm{tr}}}{T_{\mathrm{sm}}}\right)\right]$$

$$p_{\mathrm{s}} = p_t f_{\mathrm{snow}}$$

$$p_{\mathrm{r}} = p_t(1 - f_{\mathrm{snow}}),$$

where $T_{\mathrm{sm}} > 0$ $(^\circ\mathrm{C})$ controls the smoothness of the phase transition between snowfall and rainfall. A similar temperature-based partitioning scheme is used in the snow module of the `hbv` model and related large-sample hydrologic frameworks.

Snowmelt is computed using a degree-day formulation with degree-day factor f_{dd} in units of $\mathrm{mm/d/^\circ C}$

$$p_{\mathrm{pmelt}} = f_{\mathrm{dd}}\ \mathrm{pos}_{T_{\mathrm{sm}}}(T_t - T_{\mathrm{tr}}),$$

$$p_{\mathrm{amelt}} = \min_{\varepsilon_{\mathrm{s}}}(s_{\mathrm{we}}, p_{\mathrm{pmelt}}),$$

where pos and $\min$ denote smooth positivity and smooth minimum operators (Appendix A). The SWE state evolves according to

$$\frac{\mathrm{d}s_{\mathrm{we}}}{\mathrm{d}t} = p_{\mathrm{s}} - p_{\mathrm{amelt}}.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

The liquid water input supplied to the rainfall–runoff model core is $p_{\mathrm{liq}} = p_{\mathrm{r}} + p_{\mathrm{amelt}}$. This formulation preserves the conceptual structure of classical temperature-index snow models (Anderson, 1973; Martinec et al., 1992) while introducing smooth transitions that ensure continuous differentiability of all fluxes with respect to both states and parameters. This property is essential for analytic sensitivity propagation and gradient-based learning in the SAGE framework.

We use parameter bounds $-2 \leq T_{\mathrm{tr}} \leq 2\ ^{\circ}\mathrm{C}$ and $0.1 \leq f_{\mathrm{dd}} \leq 10\ \mathrm{mm/d/^{\circ}C}$, consistent with commonly adopted ranges in temperature-index snow formulations (Bergström, 1995; Perrin et al., 2003; Clark et al., 2015).

2.5 Loss functions

Building on the general relationship $\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \mathbf{J}_{\mathbf{q}}^{\top}(\boldsymbol{\theta})\, \boldsymbol{\delta}_n(\boldsymbol{\theta})$, the SAGE framework accommodates a broad class of pointwise differentiable loss functions through analytically derived $\boldsymbol{\delta}_n$-vectors. These include absolute and squared residual losses, widely used hydrologic performance metrics such as the Nash–Sutcliffe efficiency (Nash and Sutcliffe, 1970) and Kling–Gupta efficiency (Gupta et al., 2009; Kling et al., 2012), robust M-estimators that down-weight outliers (Vrugt et al., 2025; Vrugt and Diks, 2025), and hydrograph-based functionals such as flow-duration and recession curves (Vrugt, 2024). The explicit definitions of these objective functions and their analytic derivatives with respect to the simulated discharge time series are given in Vrugt et al. (2026). In particular, Table 1 of that study summarizes the corresponding *loss-sensitivity vectors*, that is, the partial derivatives $\partial \mathcal{L} / \partial \mathbf{q}_n$ with respect to the discharge vector $\mathbf{q}_n$. Combined with the analytic Jacobians $\partial \mathbf{q}_n / \partial \boldsymbol{\theta}$ of the hydrologic models derived in this work, the total gradient, $\mathbf{g}_n(\boldsymbol{\theta}) = \mathbf{v}_n^{\top}(\boldsymbol{\theta})$, is obtained by direct application of the chain rule

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{tot}}(\boldsymbol{\theta}) = \left(\frac{\partial \mathbf{q}_n(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^{\top}} \right)^{\top} \frac{\partial \mathcal{L}_{\mathrm{tot}}(\boldsymbol{\theta})}{\partial \mathbf{q}_n(\boldsymbol{\theta})} \tag{15}$$

at negligible additional computational cost. This enables efficient end-to-end training of the neural networks without the need for numerical differentiation or automatic differentiation through the hydrologic solvers.

3 Distributional performance metric across basins

Large-sample hydrologic studies commonly summarize cross-basin model performance using a single point statistic, most often the median NSE computed across all basins (e.g., Newman et al., 2015; Kratzert et al., 2019b). While intuitive, such summaries ignore the full distribution of model skill and can mask substantial differences in performance among basins. Two models may exhibit identical median NSE values while differing markedly in the fraction of poorly simulated catchments or in the shape of the performance distribution.

To address this limitation, we follow our earlier work published in Vrugt (2024) and introduce a distributional performance metric of basin-wide NSE values. Suppose $F(x)$ is the hypothetical but unknown "true" distribution of basin-wide NSE values, x, across the contiguous US. The ideal model (data-generating process) corresponds to the degenerate target distribution

$$G(x) = \mathbb{1}\{x \geq 1\},$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

that is, all basins achieve unit NSE. We define a distributional loss score $\mathcal{S}_{\mathrm{IB}}$ as the integrated deviation of the empirical NSE distribution from this ideal target (Figure 3)

$$\mathcal{S}_{\mathrm{IB}}(F,G) = \int_{-\infty}^{1} F(x)\,\mathrm{d}x \tag{16}$$

where smaller values indicate better overall performance.

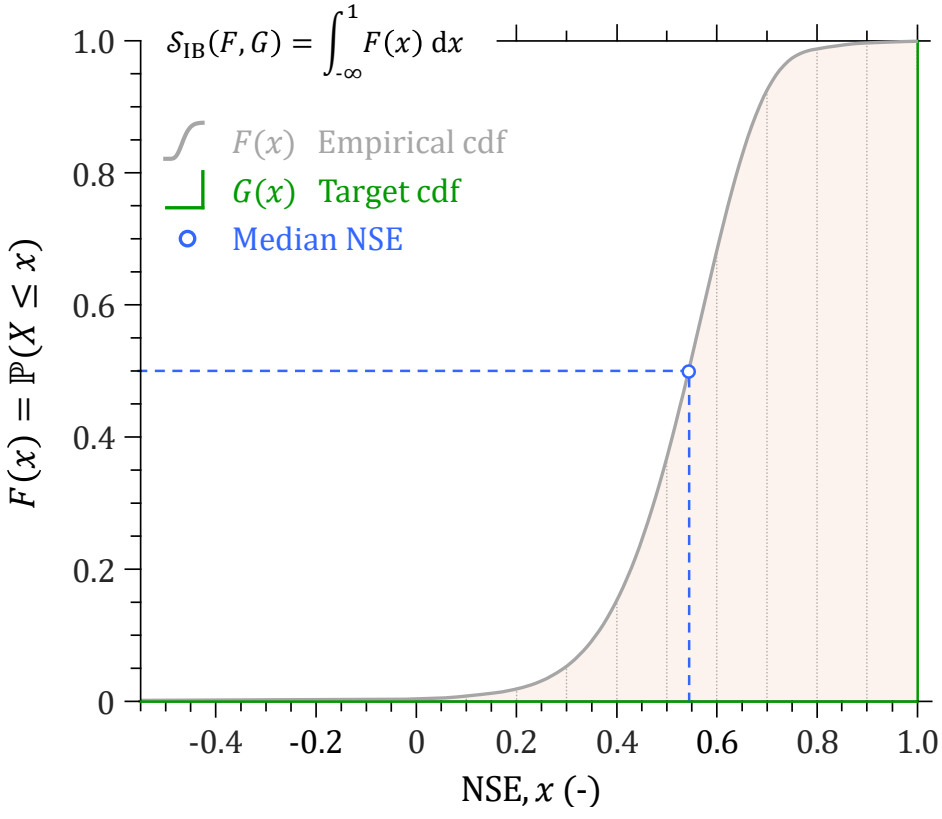

Figure 3. Illustration of the integrated basin loss score $\mathcal{S}_{\mathrm{IB}}$ as the area under the empirical cumulative distribution function (ECDF) of NSE values (shaded region). The proposed metric summarizes performance across all basins and provides a more informative measure of large-sample hydrologic model skill than the point-wise median NSE (blue marker).

In the limiting case where all basins achieve $\mathrm{NSE} = 1$, the ECDF collapses to a step function at $x = 1$ and the integrated basin loss score satisfies $\mathcal{S}_{\mathrm{IB}} = 0$. Conversely, larger values of $\mathcal{S}_{\mathrm{IB}}$ reflect increasing mass of the NSE distribution away from perfect skill. Whereas traditional hydrologic performance metrics such as the median NSE are positively oriented (larger is better), the proposed distributional metric is naturally formulated as a loss or distance measure, with smaller values indicating improved cross-basin performance. For interpretability, this loss can optionally be transformed into a positively oriented skill score by monotone rescaling or by comparison with a reference distribution.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

The proposed loss score admits an interpretation as the first-order Wasserstein distance (Peyré and Cuturi, 2019) between the empirical NSE distribution F and the degenerate target distribution G concentrated at $\mathrm{NSE} = 1$

$$\mathcal{S}_{\mathrm{IB}}(F, G) = \int_{-\infty}^{1} |x - 1| \, \mathrm{d}F(x), \tag{17}$$

which measures the average deviation of basin-wise performance from perfect skill. Since NSE values are bounded above by unity, the integral reduces to the domain $(-\infty, 1]$ in Equation 16. Equivalently, $\mathcal{S}_{\mathrm{IB}}$ is a special case of the Continuous Ranked Probability Score (CRPS) (Matheson and Winkler, 1976) evaluated between the empirical distribution $F(x)$ and the degenerate target distribution $G(x)$

$$\mathcal{S}_{\mathrm{CRPS}}(F, G) = \int_{-\infty}^{1} \big(F(x) - G(x)\big)^2 \, \mathrm{d}x. \tag{18}$$

As such, the proposed metric inherits the theoretical properties of proper scoring rules (Gneiting and Raftery, 2007; Vrugt, 2024). It is minimized in expectation when the predicted distribution matches the target distribution and accounts for the full distribution of basin-wise performance rather than a single quantile or moment. This sensitivity to the entire performance distribution is particularly relevant for continental-scale studies, where model deficiencies often manifest in specific hydroclimatic regimes or physiographic settings. Alternative variants of the integrated basin loss score may be defined, for example

$$\mathcal{S}_{\mathrm{IB}}^{(2)}(F, G) = \int_{-\infty}^{1} F(x)^2 \, \mathrm{d}x,$$

which penalizes the lower tail of the NSE distribution more strongly.

In practice, we work with a finite sample of K watersheds. Let $\{\mathrm{NSE}_k\}_{k=1}^{K}$ denote the corresponding NSE values and let $F(x)$ denote their empirical cumulative distribution function (ECDF)

$$F(x) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}\{\mathrm{NSE}_k \le x\}, \tag{19}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function. Substituting this empirical distribution into the Wasserstein form in Equation 18 yields the simple plug-in estimator

$$\widehat{\mathcal{S}}_{\mathrm{IB}}(F, G) = \frac{1}{K} \sum_{k=1}^{K} \big(1 - \mathrm{NSE}_k\big), \tag{20}$$

which is the sample mean shortfall from perfect skill. This quantity is not new: it corresponds to using the NSE in "loss" form and has been employed, for example, in our recent work on single-basin gradient-based calibration with forward sensitivities

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

(Vrugt et al., 2026). In multi-basin settings, the same average loss has also appeared as a composite objective (Huynh et al., 2026), but typically without an explicit scoring-rule interpretation. The contribution here is to place this estimator within a principled distributional framework, thereby clarifying when and why it constitutes an appropriate objective for large-sample model comparison and training.

Alternatively, we can use a compact ECDF representation as a Riemann–Stieltjes integral. Let $\widehat{F}_{\mathrm{NSE}}(x)$ denote the ECDF of the basin-wise NSE values. Since $\widehat{F}_{\mathrm{NSE}}$ is a step function with point masses at the observed NSE values, Equation 18 holds for a finite number of basins K. For the stepwise ECDF, the integral reduces to a weighted sum over the ECDF jump locations. Let $x_{(1)} < x_{(2)} < \cdots < x_{(m)}$ denote the *distinct* NSE values and let $\Delta\widehat{F}_{\mathrm{NSE}}(x_{(i)}) = \widehat{F}_{\mathrm{NSE}}(x_{(i)}) - \widehat{F}_{\mathrm{NSE}}(x_{(i)}^{-})$ be the jump size at $x_{(i)}$, where $i = 1, \ldots, m$ and $\sum_{i=1}^{m} \Delta\widehat{F}_{\mathrm{NSE}}(x_{(i)}) = 1$. Then the ECDF estimator of $\mathcal{S}_{\mathrm{IB}}$ is

$$\widehat{\mathcal{S}}_{\mathrm{IB}}(F, G) = \sum_{i=1}^{m} (1 - x_{(i)}) \, \Delta\widehat{F}_{\mathrm{NSE}}(x_{(i)}). \tag{21}$$

This integrated distributional loss naturally complements ECDF visualizations and facilitates objective ranking of competing models based on their entire NSE distribution. In the remainder of this paper, we refer to $\mathcal{S}_{\mathrm{IB}}$ as the Vrugt–Frame integrated basin loss score.

4 Data and Experimental Setup

This section briefly reviews the CONUS dataset and outlines our preliminary experiments used to illustrate our SAGE framework.

4.1 CONUS dataset

The experimental domain comprises 671 watersheds distributed across the conterminous United States, as defined in the Catchment Attributes and Meteorology for Large-sample Studies (CAMELS) dataset (Newman et al., 2014, 2015; Addor et al., 2017a, b). For each basin, a set of static attributes describing climate, geology, soil, topography, vegetation, and land cover is assembled from publicly available datasets. Meteorological forcing data are obtained from the Daymet, Maurer, and NLDAS products, and observed streamflow is taken from U.S. Geological Survey (USGS) records. Prior to network training, all r basin attributes are standardized to zero mean and unit variance.

In keeping with other large-sample machine learning applications, we consider only a subset of 531 of the 671 CONUS watersheds for model training and benchmarking (e.g., Kratzert et al., 2019b; Newman et al., 2017). This reduced set excludes basins with engineered flow control structures (e.g., dams, canals, levees), large continuous gaps in streamflow records, exceedingly large drainage areas ($> 2{,}000$ km^2), and substantial inconsistencies in reported catchment areas. The selected $K = 531$ basins are predominantly headwater catchments with close-to-natural rainfall-runoff dynamics, allowing us to focus on intrinsic hydrologic behavior rather than on the impacts of hydraulic regulation.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

4.2 Training and validation strategy

We evaluate the proposed SAGE framework using three complementary experimental paradigms that progressively increase the difficulty of the learning task: (i) direct single-site calibration of model parameters for each catchment separately, (ii) temporal validation of SAGE within catchments, and (iii) spatial cross-validation of SAGE across basins. Together, these experiments quantify (a) the attainable skill of each hydrologic model when calibrated individually, (b) the out-of-sample predictive skill of SAGE in time, and (c) the generalization capability of the neural network mapping from catchment attributes to model parameters in space.

4.2.1 Direct single-site calibration (Part A).

In the first experiment, we establish a single-site baseline by calibrating the parameters of each hydrologic model independently for each basin. All $K = 531$ CAMELS catchments are considered. For each catchment, model parameters are optimized directly against observed daily discharge over the training period (water years 1999–2008), and performance is subsequently evaluated on an independent validation period (water years 1989–1999). Parameter estimation is performed using a robust limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm with bound constraints (L-BFGS-B; Byrd et al., 1995) and the residual sum-of-squares loss function. For each basin and model, we conduct 25 independent L-BFGS-B optimizations from different initial parameter values and retain the parameter set yielding the lowest loss. This retained solution is used to compute basin-wise performance over the 9-year training period and the preceding 10-year validation period.

A one-year spin-up is applied at the start of the training interval. This design mirrors standard large-sample hydrology practices (e.g., Newman et al., 2015; Kratzert et al., 2019b; Addor et al., 2017b) and provides an upper-bound reference for model skill when parameters are tuned specifically to each basin. Because the training and validation intervals are consecutive in time, the final model states from the training phase are carried forward directly into the validation phase, avoiding the need for a second spin-up period and ensuring continuity of the simulated hydrologic states. Model parameters are estimated from the training period and then held fixed during the validation period, yielding a direct measure of out-of-sample predictive skill under basin-specific calibration.

4.2.2 Temporal validation with SAGE (Part B).

In the second experiment, we evaluate SAGE under temporal generalization while leveraging all available basins for training the attribute-to-parameter mapping. All $K = 531$ CAMELS catchments are used to train the neural network, using discharge observations from the training period spanning water years 1999–2008. Model performance is then evaluated on the independent validation period covering water years 1989–1999, again with a one-year spin-up applied at the start of the training interval. During validation, the ANN-predicted model parameters are held fixed and the final model states from the training phase are carried forward into the validation phase, providing a direct measure of out-of-sample predictive skill in time for the SAGE-trained parameter fields.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

4.2.3 Spatial cross-validation with SAGE (Part C).

In the third experiment, we assess the ability of the neural network to generalize across basins by performing spatial cross-validation. From the full set of 531 hydrologically valid CAMELS catchments, a fixed subset of $K_{\mathrm{t}} = 400$ basins is used for training, while the remaining $K_{\mathrm{v}} = 131$ basins are reserved exclusively for validation. The assignment of training and validation basins is deterministic and follows the predefined ordering of the CAMELS dataset to ensure reproducibility. In this setting, the neural network is trained to map static catchment attributes to hydrologic model parameters using only the training basins and their associated discharge records from the training period. The trained network is then applied to the unseen validation basins to predict parameter values solely from their attributes. Hydrologic model simulations driven by these predicted parameters are evaluated against observed discharge to quantify out-of-sample performance in space.

4.2.4 Implementation

Hydrologic models are evaluated either sequentially or in parallel across basins, enabling scalable training over the entire CONUS dataset. Model performance is assessed using basin-wise efficiency metrics, empirical cumulative distribution functions, and the proposed integrated basin loss score introduced in Section 3. These complementary diagnostics provide both pointwise and distributional perspectives on large-sample model skill.

Together, Parts A–C provide a rigorous assessment of sensitivity-aware learning under three regimes: basin-wise parameter optimization (single-site baseline), extrapolation across time within individual catchments (temporal validation), and extrapolation across physiographically diverse catchments (spatial cross-validation). The next section describes the computational implementation of SAGE and its integration with the neural network training environment.

5 Computer Implementation and Software

We now present the MATLAB implementation of the proposed end-to-end learning framework, designed to closely mirror the theoretical developments in Sections 2.1-2.5. The code is fully modular, transparent, and uses consistent notation with the mathematical formulation.

The first code block in box 1 defines the experimental settings and configuration, including directory paths of the meteorological and USGS streamflow data, training and validation periods, neural network architecture, loss function, and optimization settings. Examples include the number of training and validation basins K, the number of basin attributes r, the number of hydrologic model parameters d, the hidden-layer width h, and the loss function $\mathcal{L}$.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Box 1: SAGE – Specify main settings

```
dir_A = 'C:\Users\jaspe\Dropbox\MATLAB_Drive\Data';
dir_meteo = [dir_A,'\daily\v1p2\forcing'];
dir_Q = [dir_A,'\daily\v1p2\streamflow'];
file = '531_basins.txt';    % user assignment train/val basins
logfile = 'CONUS_iter.txt'; % log file training/iteration progress

K_cns = 531;                % # CONUS basins [671: CAMELS; 531: restricted]
K_t = 400;                  % # training watersheds
K_v = 131;                  % # validation watersheds
dt = 1;                     % 1:daily/24:hourly data
dmeteo = 1;                 % 1:daymet/2:maurer/3:nldas
ET = 1;                     % 1:Penman-Monteith/2:Priestley-Taylor/3:Makkink
period = struct('dt',dt,...
    'yr_ts',1999, ...       % start of training period
    'yr_te',2008, ...       % end of training period
    'yr_vs',1989, ...       % start of testing period
    'yr_ve',1999, ...       % end of testing period
    'spin_up',1, ...        % spin up period in years
    'valB',K_v > 0);        % validation basins: no validation period
model = 1;                  % 1:hymod/2:hmodel/3:sacsma/4:xnjng/5:gr4j/6:hbv
mcode = 4;                  % 1:RK/2:ODE/3:Euler [Matlab]/4:RK [C++]
calc_mthd = 2;              % 1:sequential/2:parallel model evaluation
h = 32;                     % # neurons hidden layer
tf = 'tanh';                % transfer function hidden layer: 'tanh'/'relu'
loss = 2;                   % 1:SAR/2:GLS/3:NSE/4:KGE/5:Huber/6:FDC
alg = 2;                    % 1:Gradient-descent/2:Adam
i_max = 1000;               % maximum number of descent iterations
n_print = 10;               % # basins printed on one line
prt_file = 1;               % print CPU time, par values basin/iteration
id_a = [1:5 12 13 16 18 ... % choice of basin attributes
    33 34 39 45:47 49 50 55];
r = numel(id_a);            % # basin attributes
```

The code in box 2 performs the initialization of the computational pipeline. This includes compilation of the C++ hydrologic core, sampling of training and validation basins, reading and standardizing basin attributes, loading meteorological forcing and USGS discharge observations, and initializing the neural network parameters $\boldsymbol{\phi}$ and Adam moments. Hydrologic data, initial state, parameter bounds, ODE settings and Runge-Kutta solver tolerances are stored in the `Data` cell structure. The weights and biases of the neural network are initialized by calling `net_theta('init',...)` and the first- and second-order moments of the Adam algorithm are initialized following Algorithm A.1 using `descent('init',...)`.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Box 2: SAGE - Initialization

```
flag = compile_model(model,mcode);          % Compile model
[A_cns,id_USGS] = read_attr(dir_A,id_attr); % Read basin attributes
[A,K,id_t,id_v,id_Data] = sample_basins(... % Sample train/val basins
    A_cns,id_USGS,K_cns,K_t,K_v,file);
[Data,d] = read_model(model,K,period);      % Read model parameter info
Data = read_numsetting(Data,K);             % ODE solver settings
[Data,meta] = read_meteo(Data,dir_meteo,... % Read CONUS meteo data
    dmeteo,id_Data,ET,period);
[Data,meta] = read_Q(Data,dir_Q,id_Data,... % Read USGS discharge data
    meta,period,K_t,id_t,id_v);
[phi,l] = net_theta('init',r,h,d,tf);       % Init l network weights/biases
[~,opts] = descent('init',alg,phi);         % Init descent algorithm
[RSSt,RSSv,mNSEt,mNSEv,Sibt,Sibv] = ...     % Init return arguments
    deal(nan(i_max,1)); ax = [];
```

The third block in box 3 implements the end-to-end training strategy described in Section 2.4. At each iteration i, the neural network maps standardized catchment attributes $\mathbf{a}^k$ to normalized hydrologic model parameters $\underline{\theta}^k \in [0,1]^d$. The hydrologic models are then evaluated for all training and validation basins using the compiled ODE solvers, returning both simulated discharge and the analytic Jacobians $\partial \mathbf{q}_n / \partial \boldsymbol{\theta}^\top$. These Jacobians are combined with the loss-sensitivity vectors $\boldsymbol{\delta}_n(\boldsymbol{\theta}) = \partial \mathcal{L}_{\text{tot}}(\boldsymbol{\theta}) / \partial \mathbf{q}_n$ via the chain rule to obtain $\partial \mathcal{L}_{\text{conus}}(\boldsymbol{\theta}) / \partial \boldsymbol{\theta}$, which is subsequently backpropagated through the neural network to compute $\partial \mathcal{L}_{\text{conus}}(\boldsymbol{\phi}) / \partial \boldsymbol{\phi}$. The ANN weights and biases are then updated following the Adam algorithm A.2 by calling `[phi,opts] = descent('dyn',alg,phi,dLdphi,K,i,opts)`. Then, the functions `print_nse`, `print_stats`, and `print_figs` summarize and visualize the results. `print_nse` reports basin-wise NSE values (and summary statistics) for training and validation, `print_stats` prints key optimization and run-time diagnostics, and `print_figs` produces the ECDFs of NSE and the iteration histories of the residual sum of squares (RSS) and the Vrugt–Frame loss score (Sib) for both data splits (t and v). Note that we adopt the same mathematical symbols as in the theoretical development in Sections 2.4-2.5, with Greek symbols written out explicitly (e.g., `phi` for $\boldsymbol{\phi}$ and `nTh` for $\underline{\theta}$).

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Box 3: SAGE - Dynamic part

```
for i = 1:i_max
    sT = tic; nTh = net_theta('eval',phi,A,tf);     % predict nTheta [0,1]^d
    [G,rt,rv] = run_CONUS(nTh,Data,d,K,K_t, ...     % Run CONUS watersheds
        model,loss,mcode,calc_mthd,i,prt_file);
    [RSSt(i),NSEt,mNSEt(i),Sibt(i),RSSv(i),NSEv,... % Performance train/val
        mNSEv(i),Sibv(i)] = pmetrics(Data,K,K_t,rt,rv);
    print_nse(NSEt,id_t,n_print);                   % NSE train period/basins
    dLdphi = net_theta('grad',phi,A(:,1:K_t),tf,G); % Gradient dL/dPhi
    [dLdphi,flg] = check_grad('dLdPhi',dLdphi);     % Clip gradient if large
    if flg == 1, break; end
    [phi,opts] = descent('dyn',alg,phi,dLdphi, ...  % Update weights & biases
        K,i,opts); cpuT = toc(sT);
    print_stats(K,K_t,Data{1}.id_val,i,l,cpuT, ...  % Print perfmnce metrics
        RSSt(i),RSSv(i),mNSEt(i),mNSEv(i), ...
        Sibt(i),Sibv(i));
    ax = print_figs(K,K_t,RSSt(1:i),NSEt, ...       % Print loss & ecdf NSE
        mNSEt(i),Sibt(1:i),RSSv(1:i),NSEv, ...
        mNSEv(i),Sibv(1:i),model,ax);
end
```

The fourth and final box postprocesses the SAGE output to produce discharge simulations and NSE values for the training and validation periods (Part B) or training and validation basins (Part C).

Box 4: SAGE - Postprocessing

```
[Qt,Qv,NSEt,NSEv] = pproc_SAGE('sage',nTh, ...      % train/val discharge/NSE
    model,Data,id_Data,K_t,K_v,period,mcode,loss);
```

The postprocessing routine can readily be extended to return additional diagnostics, such as alternative performance metrics (e.g., KGE, RMSE) and other simulated fluxes or state variables.

Our implementation relies almost entirely on custom-written routines and does not require MATLAB's Deep Learning Toolbox. All components, hydrologic solvers, analytic sensitivity propagation, neural network forward and backward passes, and optimization, are implemented explicitly, ensuring full transparency, reproducibility, and consistency with the mathematical formulations presented in this paper. To facilitate broader adoption and integration with existing machine-learning workflows, we also provide an equivalent PYTHON implementation (Vrugt and Frame, 2026). This version mirrors the MATLAB code in functionality and notation, while enabling seamless interaction with standard PYTHON-based deep learning ecosystems.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

6 Results

This section presents the results of the numerical experiments conducted to evaluate the proposed SAGE framework. Throughout this section, results for the different hydrologic models are distinguished using a consistent color scheme, as indicated in the figure legends.

We organize the results into three complementary parts corresponding to the experimental paradigms described in Section 4.2. Part A reports the outcomes of direct single-site calibration, in which the parameters of each hydrologic model are optimized independently for each CAMELS catchment using the prescribed training period and then evaluated on the independent validation period. This baseline provides a reference for the attainable skill of each model when calibrated directly to individual basins and quantifies out-of-sample predictive performance in time under basin-specific parameter estimation. Part B focuses on temporal validation with SAGE, in which all $K = 531$ basins are used to train the attribute-to-parameter mapping and performance is evaluated on an independent multi-year validation period. This experiment quantifies out-of-sample predictive skill in time when model parameters are predicted from catchment attributes rather than optimized separately for each basin, providing a benchmark for large-sample, attribute-conditioned calibration under fixed basin membership. Part C reports the outcomes of spatial cross-validation with SAGE, in which the CAMELS basins are partitioned into disjoint training and validation subsets. Here, the neural network is trained using only the training basins and then applied to unseen validation catchments to predict parameter values solely from their static attributes. This experiment evaluates the ability of SAGE to generalize in space and assesses the robustness of sensitivity-aware learning under physiographic extrapolation.

Together, Parts A–C provide complementary insights into (i) the model-specific upper-bound skill of direct basin-wise calibration, (ii) temporal generalization of attribute-conditioned parameter learning, and (iii) spatial generalization to unseen catchments, thereby characterizing the scalability, robustness, and generalization capability of sensitivity-aware learning for process-based hydrologic models.

6.1 Part A: Single-site training and validation

For benchmarking the SAGE framework, Table 1 summarizes cross-basin performance of six conceptual hydrologic models during the training and validation periods using two complementary metrics: the median Nash–Sutcliffe efficiency $\widehat{T}_{\mathrm{nse}}$ and the Vrugt–Frame integrated basin loss score $\widehat{\mathcal{S}}_{\mathrm{IB}}$. Whereas $\widehat{T}_{\mathrm{nse}}$ compresses basin-wise skill to a single quantile, $\widehat{\mathcal{S}}_{\mathrm{IB}}$ evaluates the entire empirical distribution of performance relative to the ideal reference distribution concentrated at $\mathrm{NSE} = 1$ (Equation 16). This distributional view provides a more robust basis for inter-model comparison, particularly in large-sample settings where differences often arise in the tails (i.e., the prevalence and severity of poorly simulated basins).

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Table 1. Summary of large-sample model performance across $K = 531$ CAMELS basins for the training and validation periods. Reported metrics include the number of model parameters (d), the number of state variables (m), the median Nash–Sutcliffe efficiency $\widehat{T}_{\mathrm{nse}}$, and the Vrugt–Frame integrated basin loss score $\widehat{\mathcal{S}}_{\mathrm{IB}}$ (Equation 16). Model ranks are based on $\widehat{\mathcal{S}}_{\mathrm{IB}}$, where lower values correspond to performance distributions more concentrated near perfect skill.

Model	d	m	Training period			Validation period		
			$\widehat{T}_{\mathrm{nse}}$	$\widehat{\mathcal{S}}_{\mathrm{IB}}$	Rank	$\widehat{T}_{\mathrm{nse}}$	$\widehat{\mathcal{S}}_{\mathrm{IB}}$	Rank
`hymod`	7	7	0.635	0.384	5	0.574	0.447	5
`hmodel`	9	6	0.657	0.369	4	0.589	0.431	3
`sacsma`	15	10	0.670	0.355	3	0.598	0.439	4
`Xinanjiang`	16	9	0.744	0.294	1	0.664	0.376	1
`gr4j`	8	10	0.629	0.400	6	0.578	0.452	6
`hbv`	13	5	0.686	0.342	2	0.605	0.420	2

Across both periods, `Xinanjiang` exhibits the strongest and most consistent performance. It attains the highest median NSE values (0.744 during training and 0.664 during validation) and the lowest integrated basin loss scores (0.294 and 0.376, respectively), ranking first under $\widehat{\mathcal{S}}_{\mathrm{IB}}$ for both periods. The low $\widehat{\mathcal{S}}_{\mathrm{IB}}$ suggests that its basin-wise performance distribution is more concentrated near high NSE values, with fewer poorly simulated catchments. The persistence of the top rank from training to validation indicates strong temporal generalization across the CAMELS basin set.

A second tier is formed by `hbv`, `sacsma`, and `hmodel`. During training, these models achieve relatively similar median NSE values (ranging from 0.657 to 0.686), yet their distributional scores provide clearer separation. `hbv` ranks second with $\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.342$ ($\widehat{T}_{\mathrm{nse}} = 0.686$), followed by `sacsma` (rank 3: $\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.355$, $\widehat{T}_{\mathrm{nse}} = 0.670$) and `hmodel` (rank 4: $\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.369$, $\widehat{T}_{\mathrm{nse}} = 0.657$). In validation, the ordering remains similar, with `hbv` retaining rank 2 ($\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.420$, $\widehat{T}_{\mathrm{nse}} = 0.605$), followed by `hmodel` (rank 3: $\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.431$, $\widehat{T}_{\mathrm{nse}} = 0.589$) and `sacsma` (rank 4: $\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.439$, $\widehat{T}_{\mathrm{nse}} = 0.598$). Importantly, these three models exhibit relatively close median NSE values in validation, yet $\widehat{\mathcal{S}}_{\mathrm{IB}}$ provides sharper discrimination by reflecting differences in the fraction and severity of low-performing basins. Such behavior is largely invisible to single-quantile summaries but is highly relevant at continental scale.

The remaining two models, `hymod` and `gr4j`, occupy the bottom of the ranking in both periods. Their median NSE values are comparable during training (0.635 and 0.629) and validation (0.574 and 0.578), and they also exhibit the largest integrated basin loss scores in each period. Specifically, $\widehat{\mathcal{S}}_{\mathrm{IB}}$ increases from 0.384 to 0.447 for `hymod` and from 0.400 to 0.452 for `gr4j`, indicating broader and less favorable distributions of basin-wise skill during validation.

A key result of Table 1 is the strong agreement between training and validation rankings based on $\widehat{\mathcal{S}}_{\mathrm{IB}}$. Moreover, the ordering is broadly consistent with model complexity. In general, models with a larger number of parameters d tend to achieve stronger basin-aggregated performance (lower $\widehat{\mathcal{S}}_{\mathrm{IB}}$ and higher $\widehat{T}_{\mathrm{nse}}$) in both the training and validation periods, suggesting that added structural and parametric flexibility can translate into improved large-sample skill when trained at scale. A second key result is that the Vrugt–Frame integrated basin loss score $\widehat{\mathcal{S}}_{\mathrm{IB}}$ complements traditional summary statistics such as the median NSE,

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

$\widehat{T}_{\text{nse}}$, by incorporating information from the full empirical distribution of basin-wise skill. As a result, it yields a more nuanced and discriminating assessment of large-sample hydrologic performance and supports objective ranking of competing model structures based on their aggregate behavior across heterogeneous catchments.

6.2 Part B: Training & validation periods

Figure 4 presents trace plots of the residual sum of squares (RSS) loss function of the six hydrologic models for the (a) training and (b) validation periods. Hydrologic model parameters are predicted by a feedforward neural network with $h = 32$ hidden neurons and $r = 18$ catchment attributes. According to Equation 9 this requires the estimation of $l = 1{,}037$ weights and biases.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

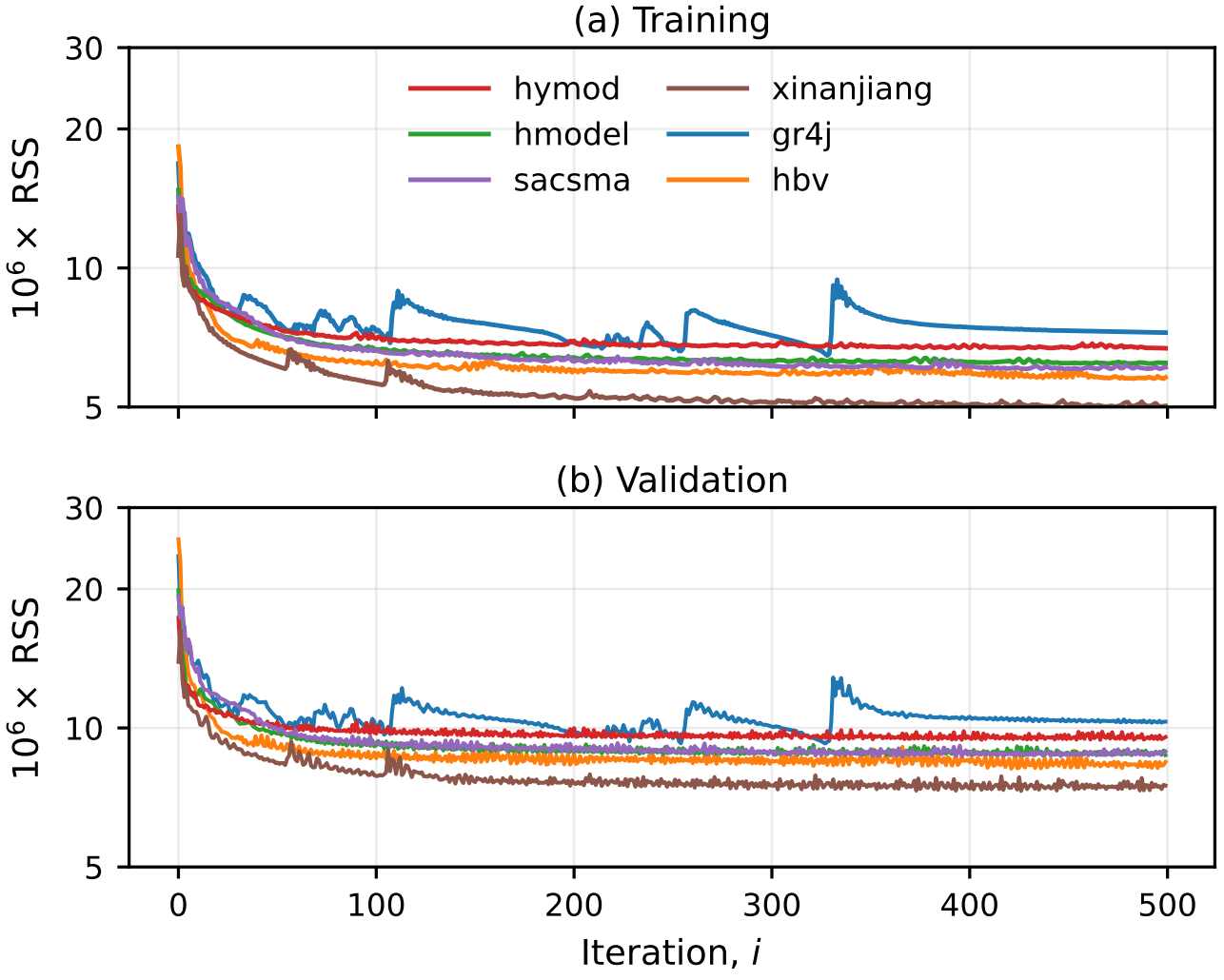

Figure 4. Trace plots of the residual sum of squares for the (a) 9-year training and (b) 10-year validation periods as a function of Adam iteration $i \in [0, 500]$. Results are shown for the six conceptual hydrologic models `hymod` (red), `hmodel` (green), `sacsma` (purple), `Xinanjiang` (brown), `gr4j` (blue), and `hbv` (orange). We use default values of the Adam hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$ for SAGE training of network weights and biases.

All models exhibit a characteristic *learning-curve* behavior, with a rapid initial reduction of the loss followed by a gradual leveling off as training progresses. In the first $\sim$ 100-150 Adam iterations, the residual sum of squares (RSS) decreases sharply for both the training period (Fig. 4a) and the validation period (Fig. 4b), indicating that early parameter updates improve fit not only to the data used for optimization but also to the withheld time window. Beyond about $i \approx$ 150-200, the curves largely plateau: additional iterations yield only marginal improvements in the training RSS and produce very similar (often nearly unchanged) RSS levels in the validation period. This pattern suggests diminishing returns rather than a strong train-validation

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

divergence, and supports the use of early stopping around $i \approx 150$-200 iterations as a simple, computationally efficient choice in large-sample SAGE training.

Beyond the overall learning dynamics, Figure 4 reveals clear and persistent differences in attainable loss levels across the six model structures. The `Xinanjiang` model (brown) achieves the lowest RSS in both periods, followed by `hbv` (orange) and `sacsma` (purple), which form a consistent second tier. `hmodel` (green) and `hymod` (red) attain intermediate RSS values, with broadly similar validation-period performance, while `gr4j` (blue) yields the largest RSS and exhibits intermittent spikes during training that are also reflected (though muted) in validation. Overall, the relative ordering of models is stable across the training and validation periods, indicating that the observed performance differences are primarily structural and persist across time rather than being driven by period-specific overfitting.

Table 2 reports the CPU time required for $1{,}000$ Adam iterations for the six conceptual hydrologic models. Listed timings correspond to runs in Windows 11 on an AMD Ryzen Threadripper PRO workstation with 64-Cores and 128 threads.

Table 2. CPU time in seconds for $1{,}000$ iterations of the Adam algorithm for the six conceptual hydrologic models used herein. Run times correspond to an AMD Ryzen Threadripper PRO 9985WX CPU workstation with 32 active cores, 3.2 GHz base clock and 127 GB RAM.

Model	d	m	CPU-time (s)
`hymod`	7	7	2,060
`hmodel`	9	6	6,805
`sacsma`	15	10	5,361
`Xinanjiang`	16	9	2,879
`gr4j`	8	10	2,489
`hbv`	13	5	1,524

The results demonstrate a substantial reduction in computational cost relative to current state-of-the-art approaches. Table 2 reports the CPU time required for $1{,}000$ Adam iterations for the six conceptual hydrologic models, corresponding to per-iteration costs ranging from about 1.5 s (`hbv`) to 6.8 s (`hmodel`) on the Threadripper workstation. Since Figure 4 indicates that near-optimal validation performance is typically reached at around $i \approx 100$ iterations (and convergence is generally achieved within $i_{\max} \approx 150$–200 iterations), these timings translate to practical end-to-end training times of roughly 2.5–11.3 minutes for 100 iterations and about 3.8–17.0 minutes for 150 iterations (depending on the model). For example, `hbv` requires about 2.5 minutes for 100 iterations (and ~ 5.1 minutes for 200 iterations), whereas `hmodel` requires about 11.3 minutes for 100 iterations (and ~ 22.7 minutes for 200 iterations). More generally, the remaining models fall between these bounds and require longer training times, reflecting differences in model complexity and state dimensionality. These results underscore that wall-clock cost is influenced not only by the number of parameters d and physical states m, but also by model-specific process formulations and numerical overhead. Nonetheless, the overall computational demands remain modest compared to the burden typically associated with automatic differentiation-based training frameworks. Collectively, the reported run times demonstrate that SAGE

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

enables large-sample training with computationally efficient early stopping around $i \approx 100$ iterations, and highlight its suitability for large-sample and continental-scale hydrologic applications.

Having established the computational requirements of SAGE, we now turn to model performance. Figure 5 presents empirical cumulative distribution functions (ECDFs; solid lines) of the Nash-Sutcliffe efficiency (NSE) across the set of (a) training and (b) validation basins, with colors denoting the six hydrologic models. Dashed lines indicate results obtained from traditional single-site calibration (part A), in which model parameters are inferred independently for each basin using daily discharge observations and a robust limited-memory L-BFGS-B optimization algorithm (Byrd et al., 1995).

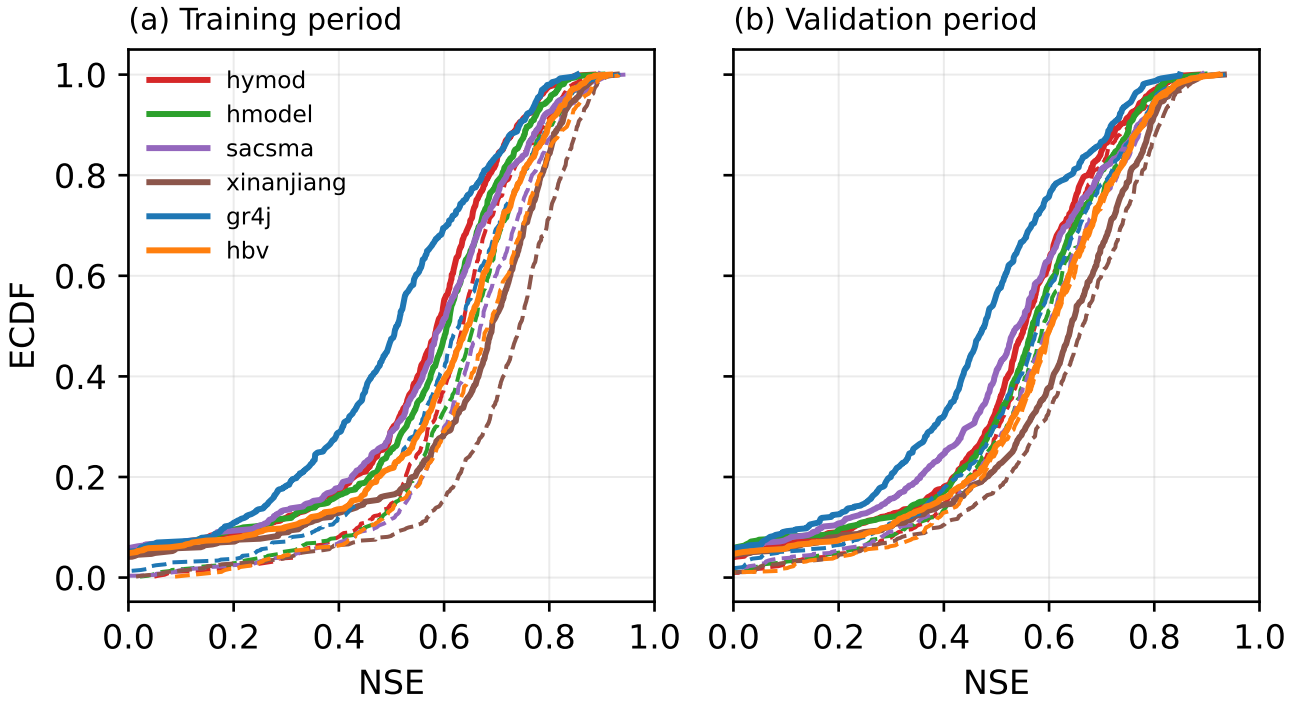

Figure 5. Empirical cumulative distribution functions (ECDFs) of the Nash–Sutcliffe efficiency (NSE) for the `hymod`, `hmodel`, `sacsma`, `Xinanjiang`, `gr4j`, and `hbv` models across the full set of $K = 531$ basins: (a) NSE values for the 9-year training period and (b) NSE values for the preceding 10-year validation period (Sections 4.2.1–4.2.2). Solid curves depict results from the proposed SAGE framework (colors distinguish the six models). Dashed curves show the corresponding ECDFs from conventional single-site calibration with gradient-based optimization.

Fig. 5a shows that for all models the ECDFs from single-site calibration (dashed) are consistently shifted to the right of the corresponding SAGE curves (solid). This behavior is expected. When each basin is calibrated independently on the training record, the parameter estimates can specialize to that period and achieve higher in-sample NSE. Within the SAGE results, the relative ordering of model structures is still clear. The `Xinanjiang` model is the most right-shifted, indicating the most favorable basin-wise NSE distribution, whereas `gr4j` exhibits the most left-shifted ECDF and thus the largest fraction of basins with

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

moderate-to-low NSE. The remaining models (`hymod`, `hmodel`, `sacsma`, and `hbv`) form a tight middle group with only modest separation around the median and upper tail.

The performance of all models degrades during the validation period (Fig. 5b). ECDFs shift leftward overall, indicating a larger proportion of basins with lower NSE. Differences in typical skill become smaller and the curves cluster more tightly in the mid-range, suggesting that tail behavior is more informative for discrimination. Notably, the training advantage of single-site calibration diminishes, as the dashed–solid gaps are generally smaller than in Fig. 5a, implying that SAGE sacrifices some in-sample fit but remains competitive out-of-sample. The overall ranking remains stable. `Xinanjiang` continues to place the largest fraction of basins at high NSE values, whereas `gr4j` retains the heaviest low-skill tail, with the other four models again forming a relatively compact second tier.

Our results highlight that the `Xinanjiang` model achieves the overall best performance during training and validation. We therefore single out this model for further analysis, focusing on how SAGE learns spatially distributed `Xinanjiang` parameter fields at continental scale. But before doing so, Figure 6 shows the training and validation traces of the Vrugt–Frame integrated basin loss score, $\widehat{\mathcal{S}}_{\mathrm{IB}}$, for the `Xinanjiang` model over Adam iterations, computed for the 9-year training period (blue) and the preceding 10-year validation period (green).

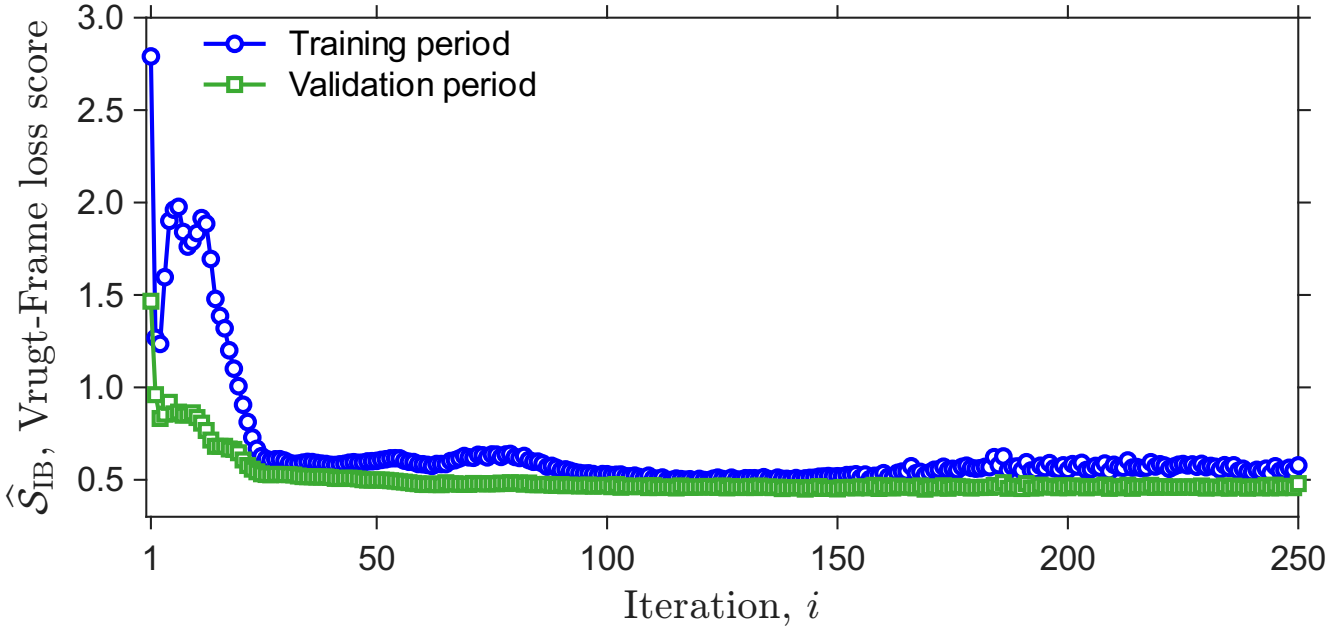

Figure 6. Trace plots of the Vrugt–Frame integrated basin loss score, $\widehat{\mathcal{S}}_{\mathrm{IB}}$, for the `Xinanjiang` model during SAGE training: training (blue) and validation (green) periods. We restrict attention to the first 250 iterations.

Both traces exhibit a rapid initial decrease, indicating that SAGE quickly improves the concentration of the NSE ECDFs toward higher skill, with the validation curve decreasing largely in tandem with the training curve during the early iterations. After this transient phase, the training loss continues to decline slowly and then stabilizes, whereas the validation loss reaches a near-minimum and subsequently meanders without sustained improvement. This divergence implies diminishing generalization

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

returns. Further refinements that sharpen the training-period ECDF no longer translate into systematic gains in the validation-period ECDF, motivating an early-stopping interpretation based on the validation trace.

Figure 7 summarizes the evolution of the `Xinanjiang` parameters predicted by SAGE during Adam training across all CONUS basins (median and 50/90% inter-basin intervals in normalized space), whereas Figure 8 displays the converged parameter fields mapped across the United States. Interpreted through the process meanings of the `Xinanjiang` parameters in Table B.4 of Vrugt et al. (2026), the figures show that SAGE rapidly identifies a small number of dominant hydrologic controls that are shared across many basins (tight envelopes and/or early stabilization), while other parameters remain heterogeneous (wide envelopes), capturing basin-specific differences in evapotranspiration limitation, runoff-generation thresholds, and flow recession time scales.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

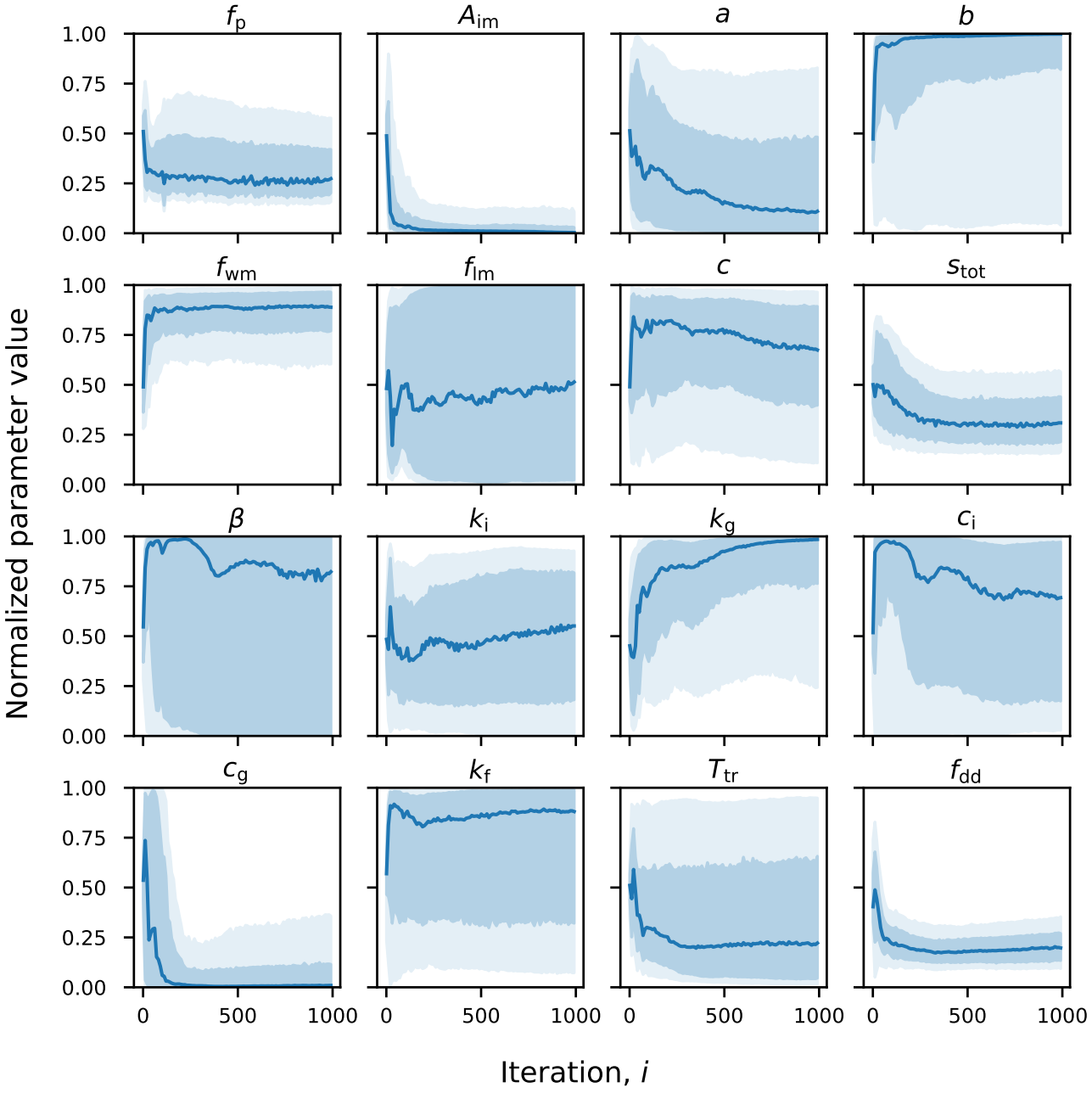

Figure 7. Evolution of the `Xinanjiang` parameters predicted by SAGE during Adam training across all CONUS basins. Each panel displays the median parameter value (solid line) together with the 50% and 90% inter-basin intervals (shaded regions) of the normalized parameter values as a function of iteration.

A salient feature of Fig. 7 is the rapid transient during approximately the first $i \approx 100$ iterations, followed by gradual stabilization of most parameter medians and uncertainty bands. This behavior is consistent with Adam quickly steering the

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

predicted parameters into a hydrologically and numerically stable subset of the admissible domain before fine-tuning. Importantly, our parameter bounds are intentionally liberal (wider than commonly reported literature ranges) to avoid overly restrictive priors and to allow the data to determine the effective parameter support. Such wide bounds are feasible here because the process model is solved in ODE form with an adaptive-step Runge–Kutta integrator that can handle substantially larger parameter domains than fixed-step implementations. Nevertheless, even robust adaptive solvers can experience convergence and efficiency problems when parameter values imply physically implausible dynamics. A classic example is effectively "torrential rain on a small reservoir," which produces extreme fluxes relative to available storage; the solver then iteratively reduces its internal time step to satisfy absolute and relative error tolerances, increasing computational cost and potentially degrading stability. The early training transient and subsequent stabilization therefore also reflect that the optimizer learns to avoid dynamically problematic regions of parameter space and concentrates learning on parameter combinations that yield physically plausible and numerically well-behaved trajectories.

Runoff production via the tension-water capacity distribution: $(a, b, f_{\mathrm{wm}}, s_{\mathrm{tot}})$.

In the `Xinanjiang` framework, runoff production is governed by a spatial distribution of tension-water capacity within a basin. The parameters b (shape) and a (inflection/scaling) control how strongly runoff generation accelerates as the catchment wets up, while s_{tot} is the soil-moisture capacity and f_{wm} controls the fraction of s_{tot} that defines the effective maximum tension-water storage ($w_{\max} = f_{\mathrm{wm}}\, s_{\mathrm{tot}}$). In our implementation, the upper-branch runoff is $r = p_{\mathrm{i}}\left(1 - c_2\, v^{\,b}\right)$, where $v = \max(1 - w/w_{\max}, 0)$ (smoothed) and $c_2 = (0.5 + a)^{1-b}$. For $0 < v < 1$, increasing b drives $v^b \to 0$ and hence $r \to p_{\mathrm{i}}$, implying that once the basin becomes moderately wet a large fraction of incremental effective precipitation is converted efficiently into runoff (i.e., limited additional retention in the tension-water store under this branch). Fig. 7 shows that b is rapidly driven toward its upper range and remains there, while f_{wm} becomes persistently large and s_{tot} settles to comparatively lower values. This combination is consistent with a sharp wetness-controlled transition in runoff production together with a relatively large "active" fraction of storage, without requiring extremely large total storage everywhere. The spatial fields in Fig. 8 reinforce this interpretation: b exhibits coherent geographic structure, indicating that the inferred nonlinearity of runoff production varies systematically with regional hydroclimatic regime rather than being random basin-to-basin noise. The more mixed spatial textures of a and s_{tot} indicate that the precise wetness thresholding and storage scale remain more catchment-dependent. Depending on the exact tension-water curve parameterization, a can shift the transition in opposite directions; here, a modulates the upper-branch scaling via c_2 and can partially compensate changes in b.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

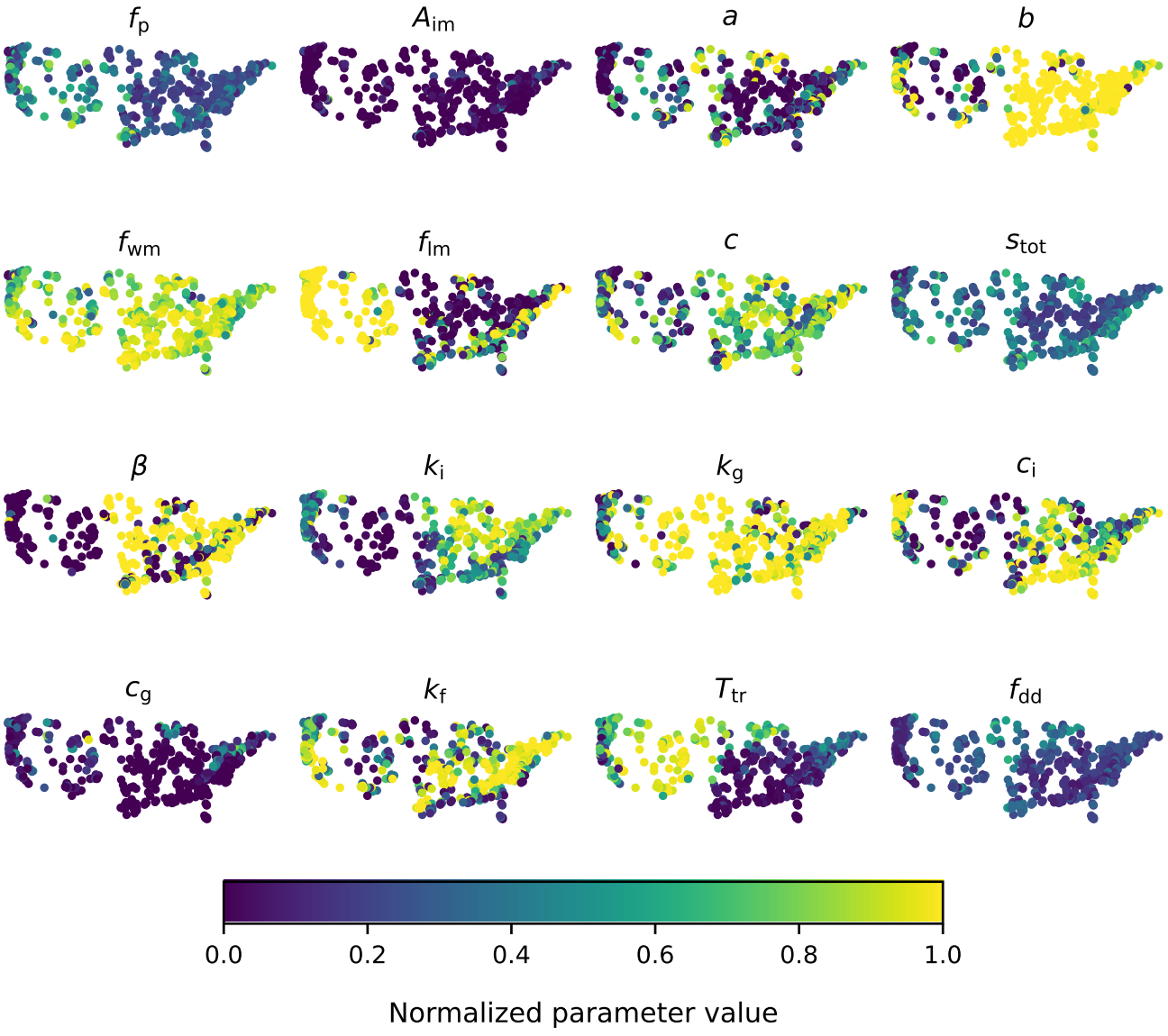

Figure 8. Spatial distribution of the SAGE trained normalized values of the `Xinanjiang` parameters across the United States. Each dot is a basin, colored by its learned parameter value.

Evapotranspiration limitation and soil-moisture thresholds: (f_p, f_{lm}, c).

The parameters f_p, f_{lm}, and c regulate how atmospheric demand is translated into actual evapotranspiration as soil moisture varies. Specifically, f_p scales potential evapotranspiration relative to pan evaporation, while f_{lm} and c define a two-threshold limitation of evapotranspiration as a function of tension-water storage: f_{lm} sets the first (early) limitation threshold as a fraction of w_{max}, and c sets a second limitation threshold as a fraction of the current tension-water storage. In Fig. 7 these parameters

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

stabilize after the initial transient, but retain non-trivial inter-basin spread, consistent with the idea that ET limitation is (i) strongly constrained by seasonality and forcing across the dataset (hence early stabilization), yet (ii) still reflects basin-specific controls related to climate, vegetation, and soils (hence persistent spread). Fig. 8 shows organized regional patterns for these parameters, consistent with large-scale gradients in aridity and evaporative demand.

Impervious runoff fraction: A_{im}.

A prominent outcome is that A_{im} collapses rapidly toward its lower bound and remains near zero (Fig. 7), with a corresponding near-uniformly low map (Fig. 8). Because A_{im} controls the impervious-area contribution (fast runoff generated without infiltration/storage), this indicates that, at the spatial scale and with the basin set used here, the data rarely require a substantial explicit impervious component to reproduce observed streamflow dynamics.

Free-water store partitioning: $(\beta, k_{\mathrm{i}}, k_{\mathrm{g}})$ and recession time scales: $(c_{\mathrm{i}}, c_{\mathrm{g}})$.

After runoff is generated, the model routes water through a free-water store whose distribution and partitioning determine how much becomes interflow versus groundwater flow. The parameter β controls the shape of the free-water distribution, while k_{i} and k_{g} govern drainage rates/partitioning to interflow and groundwater pathways. Fig. 7 shows that k_{g} rises quickly and remains near the upper end of its range, whereas k_{i} settles at intermediate values with broader spread, indicating a consistently important groundwater pathway together with more heterogeneous interflow responsiveness. The coefficients c_{i} and c_{g} set the recession time scales of the interflow and baseflow components. In Fig. 7, c_{i} exhibits persistent uncertainty, indicating that interflow timing is difficult to constrain uniquely at the continental scale and remains catchment-dependent. By contrast, c_{g} is pushed toward low values, consistent with slow baseflow recession; this is directly aligned with our groundwater representation $Q_{\mathrm{g}} = c_{\mathrm{g}} S_{\mathrm{g}}$, for which small c_{g} implies a long groundwater memory time scale ($\tau_{\mathrm{g}} \sim 1/c_{\mathrm{g}}$). Methodologically, persistent convergence to bounds (e.g., $c_{\mathrm{g}} \to 0$, $k_{\mathrm{g}} \to 1$, $b \to 1$) can reflect genuine continental-scale preference and/or compensation among correlated parameters. However, the coherent spatial organization in Fig. 8 suggests the learned fields are not purely artifacts.

Routing attenuation: k_{f}.

Finally, k_{f} controls recession in the routing reservoirs and therefore attenuation of quickflow peaks. Fig. 7 shows fast stabilization at moderately high values with non-negligible spread, indicating that routing aggressiveness is consistently required but still basin-dependent.

A notable feature of Fig. 8 is the strong spatial consistency of several parameters across large regions, including cases where parameters are effectively pinned near their admissible bounds. This boundary pinning is not confined to a few isolated basins; rather, for some parameters it occurs broadly across the dataset, consistent with the training-time behavior in Fig. 7. Importantly, the spatial organization of these near-boundary solutions is not random: the maps reveal coherent regional structure that aligns with hydroclimatically challenging areas, most prominently the central CONUS corridor extending from West Texas through the Plains into the Dakotas. The emergence of this band across multiple parameters suggests that SAGE systematically adjusts

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

(and in some cases saturates) key partitioning and recession controls there to accommodate challenging dynamics, while also highlighting where model structural limitations or forcing/observation inconsistencies may be most influential.

Figure 9 displays simulated hydrographs by the six SAGE-trained hydrologic models for representative one-year excerpts from both the training and the validation periods. The basin (ID: 01022500) was drawn at random. The model simulations are color coded consistent with earlier figures. We omit the results from single-site calibration (Part A) as model behavior is nearly indistinguishable at this resolution.

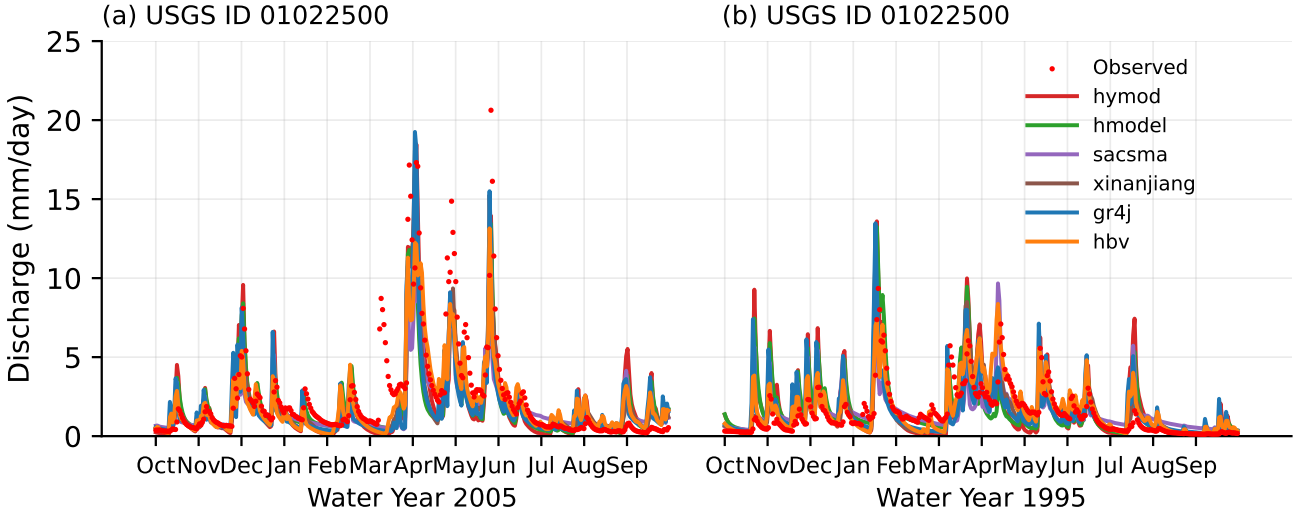

Figure 9. Simulated discharge time series for a representative USGS basin (ID: 01022500) comparing SAGE-trained hydrologic models. In each panel, the solid lines correspond to discharge simulations from the six hydrologic models (`hymod`, `hmodel`, `sacsma`, `Xinanjiang`, `gr4j`, `hbv`). The color scheme matches earlier figures. Observed discharge is shown as red points. Each panel includes both the training and validation periods for the same basin, with a broken (discontinuous) time axis used to display one-year representative portions from each period.

In both years, the simulations reproduce the dominant seasonal patterns and many event-scale responses, with relatively small discrepancies during low-flow periods and larger deviations during high-flow events. The remaining mismatches are concentrated in peak magnitudes and short-lived recessions, indicating that the dominant errors arise from storm-response and routing/storage representation rather than systematic bias in the overall water balance. Importantly, the relative ordering and spread of the six models is not constant across years. Some events compress the ensemble (models behave similarly), whereas other events reveal larger inter-model divergence, showing that structural differences become more consequential under certain forcing regimes.

From an ensemble perspective, the key feature in Fig. 9 is the degree of inter-model dispersion relative to the observations. Because all simulations are produced within the same SAGE learning framework but with different hydrologic structures, the resulting spread reflects structural diversity rather than differences in calibration strategy. When this diversity leads to partial bracketing of the observed discharge, particularly around peaks, it provides a richer basis for downstream post-processing and

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

model averaging. Conversely, when simulations remain tightly clustered and share similar structural compromises, the ensemble fails to bracket the data and any convex combination of model outputs (e.g., equal-weight averaging, Bayesian model averaging, or Mallows-type averaging with weights constrained to the unit simplex) will inherit these shared deficiencies. In such cases, post-processing can at best interpolate between similar trajectories and cannot fully correct systematic under- or over-prediction, so improvements over the best individual model are expected to be modest. These considerations motivate ensemble strategies that exploit periods of genuine inter-model diversity while acknowledging that common-mode structural error can dominate during particular events or regimes.

6.3 Part C: Training & validation basins

We now turn our attention to spatial validation and execute SAGE with 400 training basins and 100 validation basins. The trained feedforward network predicts the hydrologic model parameters of the validation basins from their catchment attributes. Figure 4 shows the characteristic two-phase learning dynamics of Adam for all six hydrologic models when training is performed on the training basins (a) and performance is monitored on the held-out validation basins (b).

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

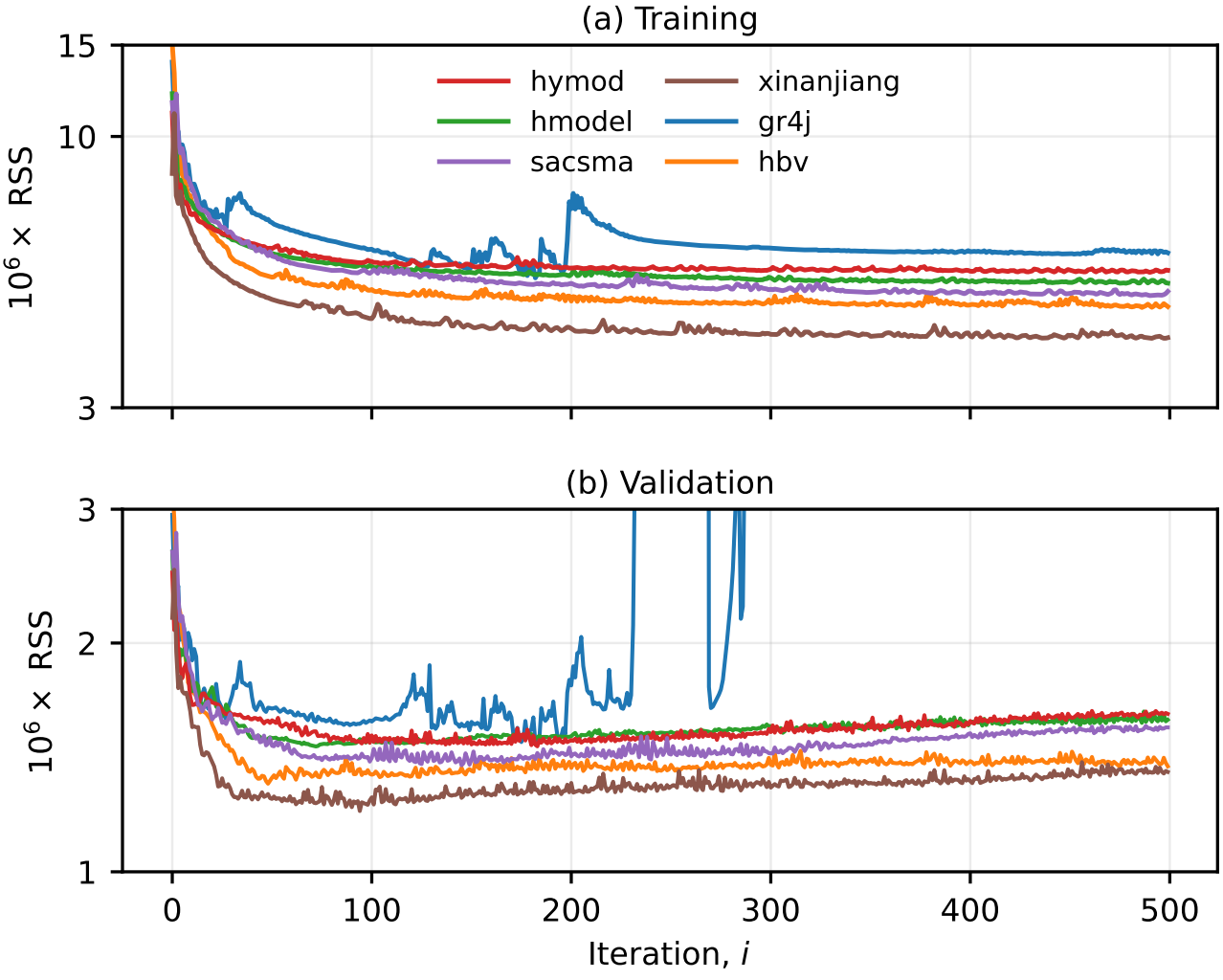

Figure 10. Trace plots of the sum of squared residuals loss function for the (a) $K = 400$ training basins and (b) 100 validation basins as a function of Adam iteration i. Results are shown for the six conceptual hydrologic models `hymod` (red), `hmodel` (green), `sacsma` (purple), `Xinanjiang` (brown), `gr4j` (blue), and `hbv` (orange). We use default values of the Adam hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 10^{-8}$ for feedforward network training.

In the first few dozen iterations, the training RSS drops sharply for every model, indicating that the optimizer rapidly moves the network weights (and thus the basin-specific parameter predictions) into a region of parameter space that explains a large fraction of the residual variance. After roughly $i \approx 50$-150 iterations, improvement becomes incremental and the curves largely plateau, with only small fluctuations. Differences between models persist in the plateau level, reflecting structural differences in how readily each model accommodates the data. The `Xinanjiang` configuration achieves the lowest training RSS throughout, followed by `hbv` and `sacsma`, whereas `gr4j` settles at the highest training RSS and exhibits occasional instability/spikes relative to the other models.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

The validation RSS in Fig. 10b provides the key generalization diagnostic. Validation performance improves substantially during the initial optimization phase (again, roughly the first 50-150 iterations), reaching a broad minimum at early-to-intermediate iteration counts. Beyond this point, additional training yields little practical gain and is typically associated with flat or slowly degrading validation behavior (and in the case of `gr4j`, pronounced excursions), consistent with the onset of overfitting. Continued reduction (or small refinements) of the training objective no longer translates into systematic improvement on unseen basins. Practically, these curves motivate early stopping around the validation minimum (often within $i \approx 100$-200 iterations), or, if longer runs are desired, stronger regularization (e.g., smaller learning rates, weight decay/priors, or tighter ODE-aware parameter bounds) to limit basin-specific compensatory parameter adjustments that improve the training fit but do not generalize. According to Table 2, early stopping at $i = 100$ iterations reduces the end-to-end training cost by roughly a factor of 10, corresponding to practical SAGE runtimes of approximately 2.5–11.3 minutes across the six conceptual models.

Figure 11 presents ECDFS (solid lines) of the NSE for the (a) training and (b) validation basins and Table 3 summarizes corresponding summary metrics. The six models are color coded. For completeness, the dashed lines correspond to the best attainable performance by each model (Part A).

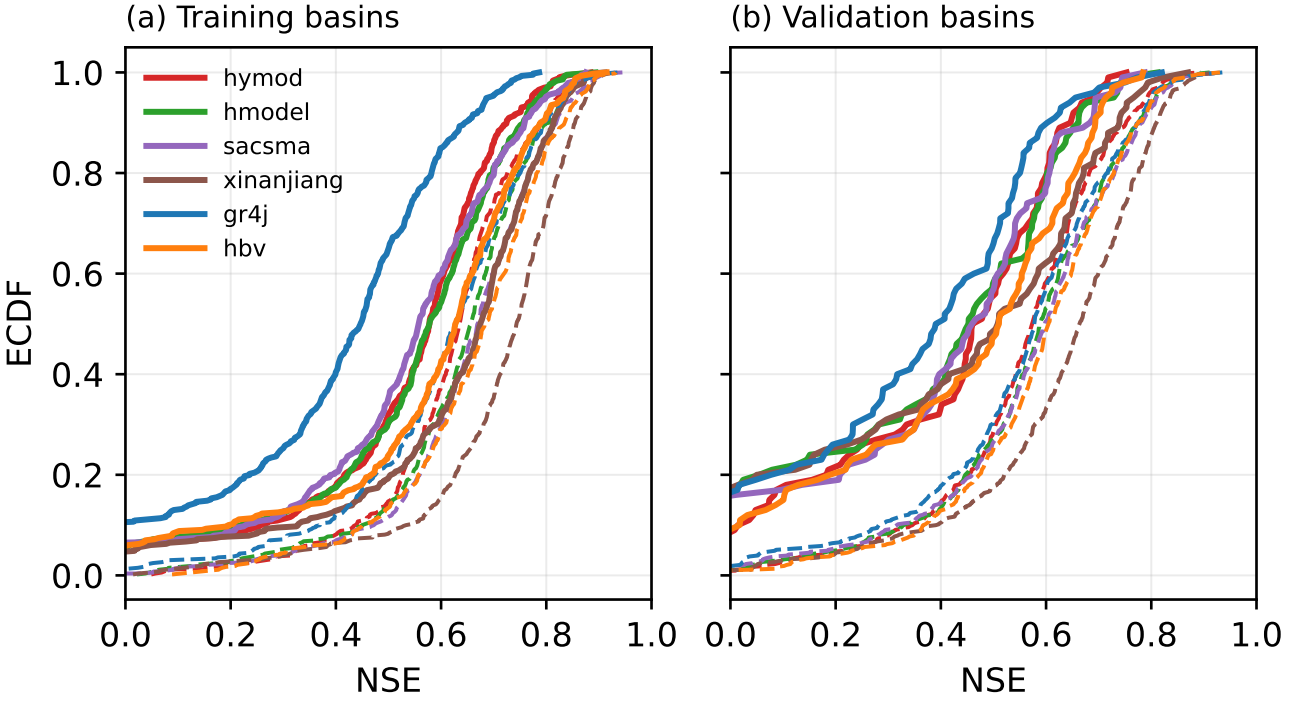

Figure 11. Empirical cumulative distribution functions (ECDFs) of the NSE for the `hymod`, `hmodel`, `sacsma`, `Xinanjiang`, `gr4j`, and `hbv` models spatial cross-validation): (a) NSE values of the $K_t = 400$ training basins for the training period, and (b) NSE values of the $K_v = 100$ held-out basins for the training period. Solid curves depict results from the proposed SAGE framework (colors distinguish the six models). Dashed curves show the corresponding ECDFs from conventional single-site calibration with gradient-based optimization, in which parameters are inferred independently for each basin using the training period only and performance is subsequently evaluated over the training and validation records.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

The ECDFs make clear that model differences are not limited to the "typical" basin but are strongly influenced by tail behavior (i.e., the fraction of basins with low NSE). This perspective is quantified in Table 3, which summarizes performance using both the median NSE, $\widehat{T}_{\mathrm{nse}}$, and the Vrugt–Frame integrated basin loss score, $\widehat{\mathcal{S}}_{\mathrm{IB}}$. Fig. 11a shows that conventional single-site calibration (dashed) yields uniformly more right-shifted ECDFs than SAGE (solid) for all six models, reflecting the expected advantage of basin-specific parameter fitting over the training record. SAGE, in contrast, imposes an attribute-conditioned parameterization that trades some in-sample skill for a shared regionalized structure. Within the SAGE curves, the separation between model structures is substantial. `gr4j` exhibits the most pronounced left shift and the heaviest low-skill tail, indicating that a large fraction of basins achieve only modest NSE values under the learned regional parameterization. By comparison, `Xinanjiang` (and to a lesser extent `hbv`/`hmodel`/`sacsma`) is shifted to higher NSE values, implying fewer poorly simulated basins and a more favorable overall distribution of basin-wise skill.

Fig. 11b highlights the more stringent test of generalization. For the validation basins, the SAGE ECDFs shift leftward and spread out, indicating both a decline in typical performance and a larger dispersion of skill across basins when evaluated out of sample. At the same time, the dashed curves remain strongly right-shifted because single-site calibration is allowed to fit each basin independently, and hence retains a large advantage in terms of attainable NSE. The key qualitative message is therefore not that SAGE outperforms single-site calibration in this setting, but that its performance degradation under spatial transfer is model dependent. Some structures (notably `Xinanjiang`) preserve a more favorable distribution with a lighter low-skill tail, whereas `gr4j` remains the least robust with the largest fraction of low-NSE basins. This underscores why distributional evaluation is useful in Part C. Models with similar mid-range behavior can differ markedly in tail performance, which dominates reliability in heterogeneous, continental-scale applications.

Table 3. Summary of SAGE performance across 400 CAMELS basins for training and 100 validation basins. Reported metrics include the number of model parameters (d), the number of state variables (m), the median Nash–Sutcliffe efficiency $\widehat{T}_{\mathrm{nse}}$, and the Vrugt–Frame integrated basin loss score $\widehat{\mathcal{S}}_{\mathrm{IB}}$ (Equation 16). Model ranks are based on $\widehat{\mathcal{S}}_{\mathrm{IB}}$, where lower values correspond to performance distributions more concentrated near perfect skill.

Model	d	m	Training period			Validation period		
			$\widehat{T}_{\mathrm{nse}}$	$\widehat{\mathcal{S}}_{\mathrm{IB}}$	Rank	$\widehat{T}_{\mathrm{nse}}$	$\widehat{\mathcal{S}}_{\mathrm{IB}}$	Rank
`hymod`	7	7	0.578	0.672	5	0.469	1.004	4
`hmodel`	9	6	0.578	1.060	6	0.453	1.967	6
`sacsma`	15	10	0.558	0.607	3	0.463	0.843	2
`Xinanjiang`	16	9	0.675	0.544	2	0.509	1.116	5
`gr4j`	8	10	0.447	1.505	4	0.399	1.590	3
`hbv`	13	5	0.628	0.463	1	0.511	0.599	1

On the training basins, Table 3 shows that `Xinanjiang` achieves the highest median NSE ($\widehat{T}_{\mathrm{nse}} = 0.675$), followed by `hbv` (0.628). However, the ECDFs in Figure 11 indicate that `hbv` has a more favorable distribution overall, with fewer poorly performing basins, which is reflected by its best (lowest) integrated score ($\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.463$: rank 1) compared to `Xinanjiang` (0.544: rank 2). This illustrates reliance on the median NSE alone can be misleading. Indeed, a model can have a high median

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

yet still exhibit a heavier low-skill tail, whereas $\widehat{\mathcal{S}}_{\mathrm{IB}}$ more directly penalizes widespread or severe failures. The same distinction helps explain why `hmodel` and `hymod` share identical training medians (0.578) but differ markedly in distributional quality ($\widehat{\mathcal{S}}_{\mathrm{IB}} = 1.060$ versus 0.672), respectively, consistent with the ECDFs showing a heavier low-NSE tail for `hmodel`.

The validation basin results reinforce these conclusions and highlight differences in generalization. Median NSE values compress into a relatively narrow range across models (Table 3), yet Figure 11 shows substantial separation in the lower and middle portions of the ECDFs, indicating different prevalence of low-performing basins. In Table 3, `hbv` is again best on validation, attaining both the highest median NSE ($\widehat{T}_{\mathrm{nse}} = 0.511$) and the lowest integrated score ($\widehat{\mathcal{S}}_{\mathrm{IB}} = 0.599$; rank 1), consistent with an ECDF that is comparatively right-shifted and rises more slowly at low NSE. In contrast, `Xinanjiang` maintains a strong validation median (0.509) but drops to rank 5 by $\widehat{\mathcal{S}}_{\mathrm{IB}}$ (1.116), indicating that its validation performance distribution contains a more substantial low-skill tail than suggested by the median alone. The relatively poor generalization of `hmodel` is also evident. It has a similar validation median to several competitors (0.453) yet a much larger integrated loss score (1.967: rank 6), consistent with the ECDF showing many basins with low NSE.

Taken together, Figure 11 and Table 3 demonstrate that $\widehat{\mathcal{S}}_{\mathrm{IB}}$ provides a more discriminating and reliability-oriented assessment than the median NSE, particularly for large-sample evaluation where tail behavior governs the fraction of basins that fail. The combined reporting of $\widehat{T}_{\mathrm{nse}}$ and $\widehat{\mathcal{S}}_{\mathrm{IB}}$ therefore yields a more complete comparison of model structures under SAGE, separating differences in typical performance from differences in robustness across basins.

Finally, Figure 12 provides two illustrative hydrographs after completion of SAGE training. The left graph shows a representative training basin, for which the ANN-parameter mapping and model states were optimized using the training period data. The right graphs displays a validation basin, in which model parameters are predicted solely from catchment attributes and the resulting discharge simulations are compared against observations.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

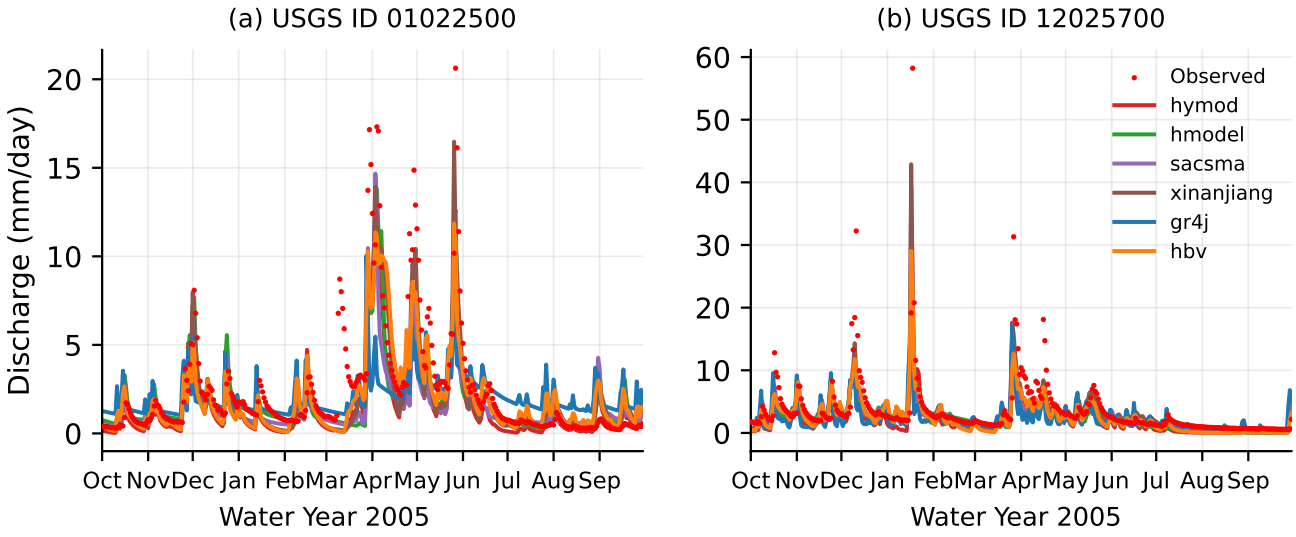

Figure 12. Example hydrographs after SAGE training for (a) a training basin (USGS ID 01022500) and (b) an out-of-sample validation basin (USGS ID 12025700). Red markers denote observed daily discharge, while solid lines show simulations from the six conceptual models `hymod`, `hmodel`, `sacsma`, `Xinanjiang`, `gr4j`, and `hbv` driven by the same meteorological forcing. In panel (a), basin-specific parameters are produced by the trained ANN and evaluated on a basin used during training. In panel (b), parameters are inferred exclusively from static catchment attributes, demonstrating regionalized prediction without site-by-site calibration.

Across both basins, the six model structures reproduce the dominant timing of runoff responses, including event-scale rises and recessions, while exhibiting noticeable differences in peak magnitude and baseflow recession behavior. For the training basins (Fig. 12a), the simulations generally track observed variability well, particularly during high-flow periods, indicating that SAGE successfully learns an attribute-to-parameter mapping that yields realistic basin dynamics when evaluated on basins seen during training. The spread among model hydrographs highlights persistent structural differences with some models produce sharper peaks and faster recessions, whereas others dampen extremes and maintain higher recession flows.

The validation basin in Fig. 12b provides a substantially stronger test of hydrologic performance because its parameters are not calibrated to that catchment; they are predicted solely from catchment attributes. Even in this out-of-sample setting, the simulated hydrographs reproduce the dominant seasonal regime and the timing of major events, indicating that SAGE can transfer learned regional attribute-parameter relationships to unseen basins. The remaining discrepancies-most apparent in peak magnitudes and low-flow levels-are consistent with limitations in (i) the information content of static attributes and (ii) the structural flexibility of the underlying model formulations, and they motivate distribution-based evaluation across many basins rather than inference from a small number of illustrative examples.

7 Discussion

The differentiable hydrologic modeling framework presented here provides a principled bridge between process-based modeling and machine learning. By combining continuous-time hydrologic ODE models with analytic sensitivity equations and an

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

attribute-to-parameter neural network, SAGE enables fast, stable, and transparent large-sample learning while producing spatially coherent parameter fields across the CONUS domain. This section synthesizes the main findings and places them in context, highlighting key strengths, limitations, and several avenues for future improvement. To structure the discussion, we organize the material into a set of focused subsections that address the most important methodological and hydrologic issues raised by our results.

Computational efficiency and transparency of SAGE

A central contribution of this work is a computationally efficient differentiable training pipeline that avoids generic automatic-differentiation toolchains. By augmenting each continuous-time hydrologic ODE model with forward sensitivity equations, SAGE propagates state trajectories and parameter sensitivities within a single forward integration. This yields exact analytic Jacobians and gradients (up to ODE solver tolerance) without finite differences and without backpropagation through long time sequences. As a result, the implementation is compact and transparent: it relies on a small number of self-contained routines and does not require specialized MATLAB toolboxes (e.g., the Deep Learning Toolbox) or external autodiff frameworks such as PyTorch, JAX, or TensorFlow.

In practical terms, SAGE reduces large-sample training from days to minutes for CONUS-scale experiments. For example, in the `hbv` configuration, a single forward pass that evaluates all $K = 531$ basins over the 20-year analysis window completes in only a few seconds on a modest Intel(R) Core(TM) i7-10700T CPU@2.00GHz desktop computer (8 cores), making end-to-end network training feasible in minutes rather than days. This computational efficiency is not merely a convenience: it enables systematic experimentation that is typically impractical in automatic-differentiation-based settings, including controlled comparisons of hydrologic model structures, network architectures, loss functions, optimizer settings, and regularization choices, as well as routine sensitivity analyses and multi-model ensembles.

SAGE also makes high-resolution forcing computationally tractable at continental scale. The software natively supports variable temporal resolution through the `dt` setting, so switching from daily to hourly learning requires only (i) reading hourly forcing/discharge files and (ii) rescaling parameters with rate or time units to consistent hourly units (e.g., converting bounds expressed per day to per hour). With piecewise-constant forcing and the requirement that the adaptive Runge–Kutta solver hits each forcing/output boundary exactly, the computational cost of an ODE simulation scales approximately linearly with the number of forcing intervals. Thus, for a fixed simulation length, moving from daily to hourly data increases runtime by roughly a factor of 24, with potentially somewhat larger costs during intense precipitation events when the solver may take shorter internal steps to meet error tolerances. This scaling contrasts with autodiff-based training, where runtime and memory demands can grow rapidly when backpropagating through long sequences, often making hourly continental-scale training prohibitively expensive.

Overall, the combination of analytic sensitivities and continuous-time integration yields orders-of-magnitude speedups relative to comparable differentiable modeling frameworks built around generic automatic differentiation. For `hbv`, full SAGE training over all 531 CONUS basins is on the order of minutes to tens of minutes on commodity hardware, and reduces further on modern multi-core workstations. This speed fundamentally changes what is feasible: it makes large-sample experimentation

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

fast enough to be routine, thereby broadening the scope of questions that can be explored within a single study and improving reproducibility through extensive ablation and sensitivity testing.

Hydrologic skill, routing sensitivity, and relation to deep-learning benchmarks

Across the six conceptual models considered, the `Xinanjiang` configuration achieved the strongest overall performance in both training and validation according to the integrated basin loss score and associated distributional diagnostics. Its validation skill ($\widehat{T}_{\mathrm{nse}} \approx 0.66$) lies within the range of values reported for large-sample deep-learning benchmarks such as Long Short-Term Memory (LSTM) models on CAMELS, which typically attain median NSE values of about 0.73-0.75 in independent studies (Kratzert et al., 2019a). While the LSTM results remain higher, this comparison indicates that structurally parsimonious hydrologic models can approach the predictive accuracy of flexible data-driven architectures when their parameters are learned consistently across large and diverse basin ensembles. Importantly, the strong cross-basin performance of `Xinanjiang` also reinforces that process representation and model structure remain decisive even in a learning-based setting, and that substantial gains can be achieved without sacrificing mechanistic interpretability.

Differences among model structures were particularly apparent in the routing component. A modified `gr4j` variant in which the standard Nash-cascade routing reservoirs were replaced by an analytic routing formulation produced marginally poorer performance than baseline `gr4j`, consistent with Mathias et al. (2026). The most likely explanation is reduced flexibility of the analytic routing scheme to represent basin-specific travel-time distributions and storage effects. This finding underscores that seemingly minor architectural choices in the transfer function from effective rainfall to discharge can measurably influence continental-scale skill.

Despite these encouraging results, the SAGE-trained conceptual models remain, on average, below the best machine-learning benchmarks reported for rainfall-runoff simulation. The single-site calibration results (Part A) with time-invariant parameters and direct basin-wise optimization therefore provide a useful reference for what is practically attainable within this class of conceptual models under the chosen forcing, periods, and objective functions. In particular, the ECDFs for training and validation periods in Part A can be interpreted as an approximate upper bound on skill for these model structures when parameters are not allowed to vary in time. Consequently, if the goal is to close the remaining gap to state-of-the-art predictive performance while retaining a physics-based SAGE workflow, two complementary avenues appear most promising: (i) improved process-model structure and (ii) ensemble post-processing.

The first avenue is to develop better physics-based model formulations that are explicitly compatible with differentiable training. The comparative differences among the six model structures provide concrete guidance for such development, including how many storages are needed, which flux parameterizations and nonlinearities appear most beneficial, and how routing/storage representations influence event response. In a SAGE context, these design choices must also respect numerical considerations: candidate model structures should remain computationally efficient at CONUS scale and should yield smooth, stable gradients when coupled to analytic sensitivity equations and trained with gradient-based optimization.

The second avenue is ensemble modeling based on the "wisdom of the crowd." Our initial experiments with Bayesian model averaging (BMA) did not yield systematic improvements in deterministic accuracy over the best individual member (typically

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

`Xinanjiang`). However, probabilistic post-processing remains attractive because mixture predictive distributions can deliver calibrated uncertainty bounds. Indeed, following the conditional-component strategy of Case Study V in Vrugt (2024), we found that while the BMA posterior mean again did not outperform `Xinanjiang`, the resulting mixture distribution provides well-calibrated and relatively sharp prediction intervals.

Why SAGE underperforms direct basin-wise calibration

While SAGE performs competitively, direct single-site calibration (Part A) remains an approximate upper bound on attainable fit for time-invariant conceptual models because parameters are optimized independently for each basin. In contrast, SAGE restricts basin-specific parameters to lie on a low-dimensional manifold induced by (i) the available static catchment attributes and (ii) the chosen attribute-to-parameter network architecture. The resulting performance gap therefore reflects a fundamental trade-off: SAGE enforces cross-basin coherence and generalization, whereas single-site calibration maximizes within-basin fit.

This trade-off is exacerbated by the well-known notion of *effective* parameters under misspecification. Even when catchment attributes encode genuine physiographic controls, calibrated parameter values often absorb compensatory effects due to structural inadequacies, errors in meteorological forcing and discharge observations, and numerical approximations. Such compensation can be strongly basin- and period-dependent, and therefore only partially predictable from static descriptors alone. A long history of regionalization studies has reported mixed outcomes when relating conceptual-model parameters to catchment attributes (Oudin et al., 2008; He et al., 2011; Razavi and Coulibaly, 2013; Arsenault and Brissette, 2014), consistent with this interpretation.

Within SAGE, limited attribute information and compensatory parameter behavior can manifest as overly smooth parameter fields. Continuity in the learned maps is partly a consequence of the finite attribute set and the regularizing effect of a shallow feedforward network, which may suppress sharp regime shifts or localized process differences that are nonetheless relevant for reproducing basin-specific hydrographs. Thus, the gap to single-site calibration should not be interpreted solely as an optimization deficiency, but rather as an intrinsic consequence of learning transferable, attribute-conditioned parameterizations in the presence of misspecification and incomplete predictors.

Evidence for limited attribute–parameter identifiability

To more directly assess the extent to which static attributes can explain basin-specific parameter variability, we conducted offline regression experiments that map the single-site optimal parameters from Part A to the available catchment attributes. We considered both regularized linear models (LASSO; Tibshirani, 1996; Friedman et al., 2010) and feedforward neural networks. Table 4 summarizes validation performance (median across parameters) using a fixed split of $K_t = 400$ training basins and the remaining basins for evaluation.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Table 4. Comparison of LASSO and multi-layer perceptron (MLP) regression for parameter regionalization from static catchment attributes. For each conceptual model, we matched available attribute vectors to the corresponding calibrated parameter sets and used the same deterministic split into $K_t = 400$ training basins and the remaining basins for validation. Attribute predictors were standardized using the training basins only, and the resulting transformation was applied to the validation basins. For LASSO, we fit an ℓ_1-regularized linear regression separately for each parameter with 10-fold cross-validation to select the regularization strength. For MLP, we fit a feedforward network separately for each parameter with one hidden layer of 12 ReLU units and ℓ_2 weight decay. Reported values summarize validation performance as the median across parameters for each model.

Model	d	m	**LASSO**			**MLP**		
			r	R^2	RMSE	r	R^2	RMSE
`hymod`	7	7	0.331	-0.238	0.299	0.197	-10.4	1.05
`hmodel`	9	6	0.381	-0.0147	0.288	0.215	-9.75	1.02
`sacsma`	15	10	0.214	-0.0518	0.304	0.0832	-23.2	1.43
`Xinanjiang`	16	9	0.169	-0.0885	0.326	0.0492	-24.3	1.51
`gr4j`	8	10	0.388	-0.0278	0.284	0.0700	-19.2	1.32
`hbv`	13	5	0.349	-0.00861	0.311	0.0789	-19.1	1.19

Notes: d denotes the number of model parameters and m the number of model states. Validation performance is summarized by the median across parameters for each hydrologic model.

Across all models, LASSO provides only modest out-of-sample skill. Correlations remain low ($r \approx 0.17$-0.39) and R^2 values are near zero or negative, indicating that a substantial fraction of basin-to-basin variability in calibrated parameter values is not explained by the available attribute set. This result is consistent with the interpretation that calibrated parameters absorb compensatory behavior linked to structural and forcing errors, which limits the existence of stable, transferable attribute–parameter relationships. With the exception of `gr4j`, the weakest predictability occurs for more highly parameterized models (e.g., `sacsma`, `Xinanjiang`, and `hbv`), suggesting that increasing parameter dimensionality further complicates attribute-based identifiability.

The neural-network regression accentuates this generalization limitation. While the MLP can fit the training-basin parameter targets extremely well, the mapping fails to transfer. Validation correlations remain small and median R^2 becomes strongly negative with substantial RMSE inflation. We explored multiple network widths and depths and observed the same qualitative behavior, indicating that the learned relationships largely reflect idiosyncrasies of the training basins rather than robust controls that generalize. These offline analyses are nonetheless useful diagnostics for identifying which attributes carry transferable signal and what model complexity is warranted. Moreover, the resulting regression weights and biases can provide a principled initialization for SAGE, potentially reducing the number of iterations required to reach convergence.

Parameter bounds, physical plausibility, and adaptive ODE efficiency

A key practical lesson from our continuous-time implementation is that computational cost is tightly coupled to parameter bounds. Many published ranges were developed for discrete-time formulations (often daily explicit update schemes) and do not necessarily transfer one-to-one to adaptive-step ODE solvers, particularly when the hydrologic core is augmented with coupled

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

forward sensitivity equations. In this work, we adopted intentionally liberal bounds to give the discharge data ample opportunity to determine suitable parameter values. Yet bounds cannot be made arbitrarily wide. Physically implausible combinations (e.g., unrealistically small storage capacities or excessively large rate constants) can drive the model into regimes with unusually rapid state dynamics. In such cases, the adaptive Runge–Kutta solver must repeatedly reduce its step size to satisfy absolute and relative error tolerances, increasing runtime and potentially degrading numerical robustness.

The per-iteration CPU times reported in Table 1 therefore reflect an inherent balance between flexibility and numerical efficiency. Narrower parameter ranges would reduce the frequency of extreme dynamics and step-size collapse, lowering runtime but at the risk of excluding viable solutions and biasing inference. Conversely, very wide bounds increase solver workload by allowing excursions into stiff regions of parameter space. This trade-off is strongly model dependent. For example, `sacsma` has more coupled states and substantially more flux pathways than `hbv`, and it relies on strongly nonlinear constitutive relations whose derivatives can spike. These features increase the likelihood of stiffness when parameters wander into fast-response regimes (e.g., large rate constants), prompting the adaptive solver to take many small steps. Such structural differences explain a substantial portion of the spread in per-iteration costs across models in Table 1. Appendix B further discusses our choice of parameter ranges for `gr4j`. The bounds reported in Tables B.1 and B.2 are enlarged relative to common literature values. Adopting narrower, discrete-time-inspired ranges would likely reduce CPU time by preventing extreme dynamics that trigger aggressive step-size reduction in the adaptive ODE solver.

SAGE runtime is also controlled by the numerical settings of the ODE solver. The timings in Table 1 correspond to $\texttt{AbsTol} = 10^{-3}$ mm and $\texttt{RelTol} = 10^{-3}$, which we found to provide a favorable compromise between efficiency and accuracy for large-sample training. More stringent tolerances can be imposed when higher numerical fidelity is desired. For instance, reducing both tolerances to 10^{-5} (consistent with the numerical configuration used by Vrugt et al. (2026)) approximately doubles the execution time in our experiments. Even under these more conservative settings, SAGE remains orders of magnitude faster than training strategies that rely on generic automatic differentiation and backpropagation through long sequences. These considerations motivate the development of ODE-aware parameter priors and/or adaptive bounding strategies that preserve physical plausibility while limiting excursions into stiff regimes that disproportionately increase computational cost.

Solver tolerances affect not only runtime but also derivative quality. Figure 13 compares the sensitivity of the analytic Jacobian and a finite-difference (numerical) Jacobian to tolerance settings. The analytic Jacobian (Fig. 13a) remains close to the reference Jacobian $\mathbf{J}_q^{\text{ref}}(\boldsymbol{\theta})$ over the full $(\texttt{AbsTol}, \texttt{RelTol})$ grid, with only modest degradation for the loosest relative tolerance ($\texttt{RelTol} = 10^{-2}$), consistent with small trajectory inaccuracies under permissive tolerances. In contrast, the numerical Jacobian in Fig. 13b deteriorates rapidly as tolerances are relaxed, with the largest deviations occurring for $\texttt{RelTol} = 10^{-2}$ and loose absolute tolerances (near $\texttt{AbsTol} = 10^{-2}$). This behavior is characteristic of finite-difference derivatives, which become dominated by integration and truncation errors unless the forward solution is computed at high accuracy.

Practically, Fig. 13 demonstrates that analytic sensitivities provide a broad "safe" tolerance regime in which gradients remain reliable without excessively strict solver settings, enabling faster and more robust gradient-based training. Numerical Jacobians, by comparison, require tight tolerances to avoid solver-induced noise, increasing computational cost and reducing optimization stability. Together, these results highlight that continuous-time differentiable hydrology benefits from parameter bounds and

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

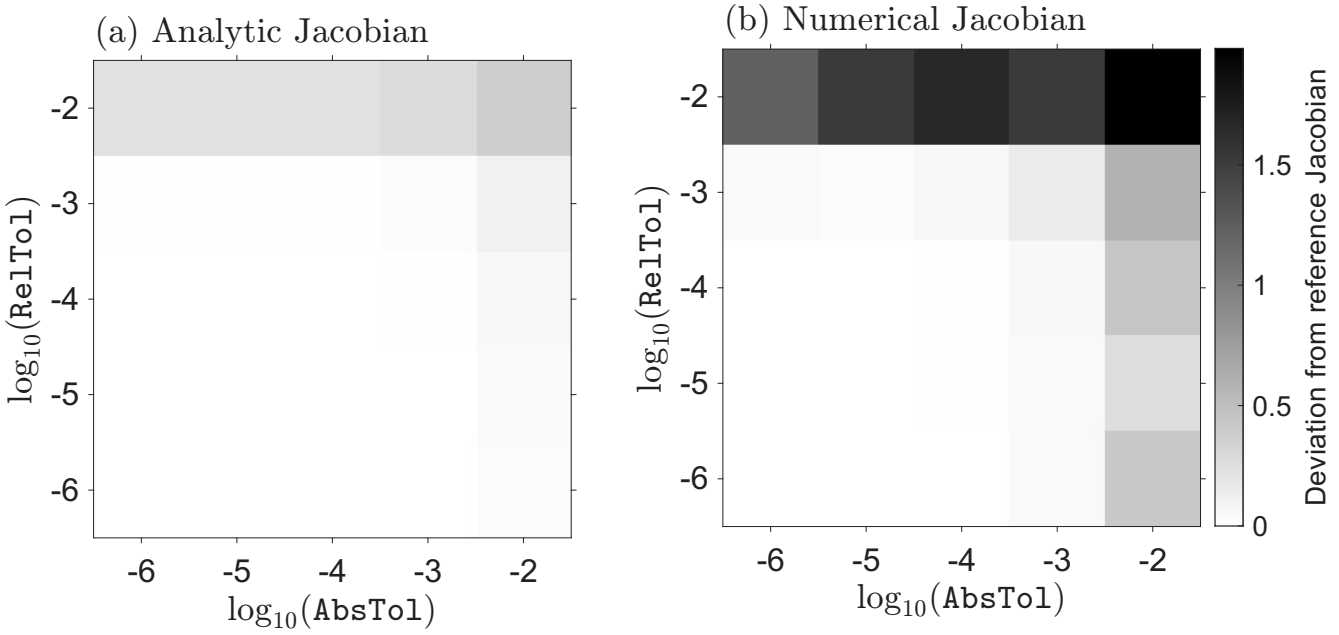

Figure 13. Heat maps of the column-wise max–normalized Frobenius deviation of the (a) analytic and (b) numerical Jacobian $\mathbf{J}_q(\boldsymbol{\theta}) \in \mathbb{R}^{n \times d}$ from a reference Jacobian $\mathbf{J}_q^{\mathrm{ref}}(\boldsymbol{\theta})$ computed with strict tolerances ($\texttt{AbsTol} = 10^{-8}$ mm, $\texttt{RelTol} = 10^{-8}$). Lighter shading indicates closer agreement with $\mathbf{J}_q^{\mathrm{ref}}(\boldsymbol{\theta})$.

numerical settings that are chosen with the ODE solver in mind, and that analytic sensitivities substantially relax the accuracy demands required for stable training.

Analytic derivatives, smoothing, and identifiability under misspecification

Analytic sensitivities are central to SAGE: they enable fast and stable gradient-based learning while avoiding the step-size tuning and numerical noise that often accompany finite-difference derivatives. This is particularly important in large-sample training, where gradients are aggregated over hundreds of basins and long time series, and where even modest derivative noise can accumulate into unstable or biased optimization trajectories. In addition, the availability of exact Jacobians makes the training pipeline transparent: every term in the gradient can be traced to a physically interpretable flux or state update, which is difficult to achieve in generic automatic-differentiation workflows.

At the same time, achieving differentiability in hydrologic process models requires careful treatment of non-smooth operators that are ubiquitous in conceptual formulations (e.g., $\max / \min$ constraints, threshold-controlled fluxes, and piecewise reservoir outflows). Here we use smooth approximations that preserve differentiability and numerical stability while maintaining the intended physical behavior. Such smoothing acts as a form of numerical regularization: it limits gradient discontinuities near thresholds and reduces the likelihood of erratic parameter updates when the optimizer traverses regions where flow partitioning switches rapidly. The trade-off is that overly aggressive smoothing can blur sharp regime transitions and may slightly alter local

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

sensitivities near thresholds. In practice, we found that modest smoothing provides a favorable compromise, yielding stable gradients while preserving the qualitative hydrologic response.

Beyond efficiency, analytic Jacobians provide diagnostic value for understanding parameter sensitivity and identifiability (Vrugt and Gao, 2022). In misspecified models, parameters often behave as *effective* quantities that compensate for structural deficiencies and forcing errors, so local sensitivity patterns may not translate directly into physical interpretability. Nevertheless, Jacobian-based diagnostics can help identify parameters (or combinations of parameters) that are weakly informed by the data, reveal where equifinality is most pronounced, and guide the design of priors, bounds, and regularization terms that discourage compensatory behavior. In this sense, analytic derivatives are not only an optimization tool but also a lens for interrogating model adequacy and for motivating targeted structural improvements within a differentiable, physics-based learning framework.

Choice of loss function and aggregation across basins

A central design choice in large-sample hydrologic learning is how basin-wise performance is aggregated into a single objective for training and model comparison. In this work, aggregation is achieved through the mean loss in Equation 4, which combines individual catchment losses into a differentiable CONUS-scale objective. While this aggregation is mathematically simple, the choice of basin-level loss function largely determines the statistical meaning and optimization behavior of the resulting objective.

For residual-based losses (e.g., weighted sum of squares, absolute residual loss, and the Huber loss), basin-wise contributions admit a natural additive structure. When the loss is defined on discharge residuals, the aggregate objective in Equation 4 can be interpreted as minimizing a joint residual norm and, under standard assumptions, corresponds to maximizing a likelihood or robust pseudo-likelihood. In contrast, commonly reported hydrologic skill metrics such as the Nash–Sutcliffe efficiency (NSE) and Kling–Gupta efficiency (KGE) are nonlinear functionals that include basin-specific normalizations. As a result, averaging NSE or KGE across basins is not equivalent to computing the same metric on pooled residuals, and it does not generally correspond to minimizing a single residual-based likelihood. Instead, it defines a composite criterion that trades off skill across heterogeneous catchments and hydroclimatic regimes.

The distinction matters because aggregation induces implicit weighting. Pooled residual objectives weight catchments by both record length and residual magnitude/variance, so high-variance or long-record basins can dominate the training signal. By contrast, averaging per-catchment metrics assigns equal weight to each basin (after normalization), which is often preferable in continental-scale studies when the goal is balanced skill across physiographic and climatic regimes rather than optimal fit for a small subset of highly variable basins.

The integrated basin loss score introduced in Section 3 provides a complementary, distributional perspective. Rather than summarizing performance by a single quantile such as the median NSE, it evaluates the full empirical distribution of basin-wise NSE values relative to an ideal reference distribution concentrated at $\mathrm{NSE} = 1$. This formulation retains sensitivity to poorly performing basins while remaining interpretable and naturally paired with ECDF-based diagnostics. Future work should explore hybrid objectives that combine residual-based likelihood terms with distributional penalties, thereby promoting both statistical coherence and balanced cross-basin skill. Because SAGE provides analytic gradients for a broad class of pointwise differentiable losses, such extensions are straightforward to implement and test.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

8 Outlook and potential improvements

A defining advantage of SAGE is that it reduces the computational cost of large-sample experimentation to the point that iterative hypothesis testing becomes routine rather than exceptional. Questions that previously required days to weeks of wall-clock time (and thus encouraged conservative experimental design) can now be explored in minutes to hours, enabling broader ablation studies, more ambitious sensitivity analyses, and faster iteration on model and learning-system design. This shift in computational tractability opens several near-term research directions, which we summarize below.

Optimization and training dynamics.

While we adopted standard Adam settings for robustness and simplicity, there is considerable headroom for improving optimization efficiency and stability. Learning-rate schedules, warm restarts, adaptive gradient clipping, and decoupled weight decay may improve convergence speed and reduce sensitivity to hyperparameters. More fundamentally, the availability of analytic Jacobians makes it feasible to explore second-order or quasi-Newton updates that exploit curvature information (e.g., Gauss–Newton or limited-memory BFGS variants) without relying on generic automatic differentiation through long sequences. Such methods may be particularly beneficial when the loss landscape is ill-conditioned due to parameter interactions or when training is performed at higher temporal resolution. Beyond point estimation, analytic derivatives also facilitate principled diagnostics of training behavior, including gradient-noise scaling with batch size (number of basins) and convergence monitoring using curvature-informed stopping rules.

Network design and regionalization capacity.

The current two-layer feedforward architecture provides a strong baseline, yet the mapping from static catchment attributes to hydrologic parameters is likely nonlinear, heterogeneous, and partially non-identifiable. Future work should therefore revisit network design with the goal of improving predictability while controlling overfitting. Candidate extensions include deeper architectures, residual connections, and normalization layers, as well as mixtures-of-experts or regionally adaptive models that allow different parameterizations in distinct hydroclimatic regimes. Such designs may better capture spatial nonstationarity (e.g., snow-dominated versus rain-dominated basins, arid versus humid regimes) while retaining a single unified training objective. Importantly, the computational efficiency of SAGE enables systematic architecture searches and ablations at CONUS scale that would be impractical under conventional backpropagation-through-time training frameworks.

Priors, constraints, and physics-aware regularization.

A complementary direction is to incorporate stronger priors and constraints that reflect physical knowledge and reduce compensatory parameter behavior. This includes ODE-aware parameter bounds (e.g., ensuring timescales and recession constants remain dynamically meaningful at the chosen temporal resolution), as well as physics-based regularization that penalizes unrealistic partitioning of fluxes or implausible storage trajectories. More generally, one can introduce penalties that discourage "compensation under misspecification," whereby different parameter combinations yield similar discharge fits while

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

implying unrealistic internal states or flux decompositions. Because SAGE provides analytic gradients for both the hydrologic model and the parameter-mapping network, such regularization terms can be added without sacrificing computational tractability, making it feasible to balance predictive skill against physical plausibility across hundreds of basins.

Richer predictors beyond static attributes.

The validation experiments indicate that part of the out-of-sample performance gap relative to single-site calibration is attributable to the limited information content of static attributes alone. A natural extension is therefore to augment the predictor set with descriptors that better summarize climate and hydrologic behavior, such as climatic indices, flow signatures, or statistics of forcing and response. More ambitious approaches could incorporate learned representations (embeddings) of forcing histories or regime indicators derived from meteorological time series. These richer predictors may substantially improve generalization by providing information about seasonality, intermittency, storm characteristics, and snow/rain partitioning that static physiographic attributes cannot fully encode. SAGE is well-suited to evaluating such design choices because it enables rapid end-to-end training and comparison across alternative feature sets under consistent experimental protocols.

Hybrid objectives and distributional training criteria.

The results also motivate exploring objectives that combine residual-based likelihood terms with distributional performance penalties. Residual-based formulations provide statistical coherence and interpretability, whereas distributional criteria (such as the integrated basin loss score) promote balanced skill across heterogeneous catchments by down-weighting solutions that succeed on average but fail systematically in specific subsets of basins. Future work should explore hybrid formulations that incorporate both perspectives within a single differentiable objective, thereby enforcing statistical fidelity while explicitly shaping the cross-basin distribution of performance. The SAGE framework readily accommodates such extensions because analytic gradients can be derived for any differentiable aggregate objective, enabling principled large-sample learning that integrates physical modeling, statistical rigor, and distribution-aware evaluation.

Some details about numerics and smooth derivatives. Discuss the aggregate or composite loss and robust M-estimators. Maybe Huber loss or other M-estimators will lead to a stronger relationships between static catchment attributes and hydrologic model parameters.

High-resolution (hourly) continental-scale training.

A practical consequence of SAGE's computational efficiency is that high-resolution meteorological forcing and discharge data become feasible at CONUS scale. The SAGE software natively supports variable temporal resolution through the `dt` setting (Box 1 in the software section), where setting `dt = 24` switches the workflow from daily to hourly inputs. The increase in data volume is substantial: a 3-year simulation at hourly resolution requires $3 \times 24 \times 365 = 26{,}280$ time steps, which is approximately $7.2\times$ the number of time steps in a 10-year daily training period ($10 \times 365 = 3{,}650$). Since forward simulation and sensitivity propagation scale approximately linearly with the number of time steps, runtime increases by a similar factor. Even so, the

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

overall burden remains modest: for a model such as hbv, SAGE can train over all CONUS basins with hourly inputs in roughly one hour on a modern workstation, whereas comparable training with current automatic-differentiation-based architectures would be prohibitively expensive due to substantially higher per-step overhead and the need to backpropagate through long sequences. Switching temporal resolution requires minimal changes beyond scaling parameters with units of inverse time (or length per time) to the hourly timestep. For example, the upper and lower bounds of the hbv recession parameter k_0 (Table B.2) should be multiplied by $1/24$ to convert units from $1/\mathrm{d}$ to $1/\mathrm{h}$. An analogous rescaling applies to other rate parameters. No further modifications are required.

Ensembles, spatial diagnostics, and acceleration.

The computational efficiency of SAGE makes ensemble strategies routine that are otherwise cost-prohibitive in large-sample hydrology, including multi-start training, bootstrap resampling over basins, and ensembles over alternative neural-network architectures, hydrologic model structures, and objective (loss/likelihood) formulations. Such ensembles can improve robustness, reduce sensitivity to initialization and sampling variability, and provide empirical uncertainty estimates for predictions and learned parameter fields. In particular, training can be repeated efficiently under alternative meteorological forcings (e.g., Maurer and NLDAS) and different loss functions, capabilities that are already supported in the SAGE software (Section **??**) and can be readily augmented with new model structures. This flexibility enables systematic ensemble post-processing and uncertainty quantification at continental scale, including event- or regime-dependent weighting rules designed to exploit periods of genuine inter-model complementarity. In parallel, future work should analyze spatial patterns in the learned parameters and relate them to hydroclimatic gradients, physiography, and data quality to identify where the attribute-to-parameter mapping is well constrained and where it breaks down, thereby motivating improved predictors, priors, or process representations. Finally, although the current implementation already provides substantial speedups, additional acceleration is possible via GPU-enabled linear algebra for the network forward pass, more efficient batching of basin evaluations, and improved parallelization, further reducing time-to-result and enabling even larger experimental campaigns.

Misspecification and uncertainty quantification.

A final direction concerns inference under model misspecification and the quantification of uncertainty in learned parameters and predictions. In large-sample hydrology, calibrated parameters are often *effective* quantities: they partially compensate for structural inadequacies, forcing-data errors, and numerical error, rather than reflecting purely physical properties. SAGE tends to produce smooth parameter fields because continuity is imposed by the two-layer attribute-to-parameter network and the limited attribute set; nonetheless, direct single-site calibration with time-invariant parameters (e.g., multi-start Gauss-Newton/Levenberg-Marquardt or SCE-UA) still provides an approximate upper bound on the best achievable within-basin fit. The persistent gap between basin-wise calibration and attribute-conditioned prediction also aligns with the broader regionalization literature, which has repeatedly found limited and unstable relationships between conceptual-model parameters and static catchment descriptors, especially when models are misspecified and the parameters implicitly compensate for model and data errors.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

A key implication is that uncertainty quantification must explicitly acknowledge misspecification. Analytic Jacobians are not only beneficial for fast and stable optimization, but also enable statistically principled inference frameworks that remain computationally feasible at continental scale. In particular, Jacobian information can be incorporated into score-based likelihood formulations that are robust to misspecification and support sandwich-adjusted uncertainty estimates (Frazier et al., 2023; Vrugt and Diks, 2025), potentially delivering reliable parameter and predictive uncertainties within a single Markov chain Monte Carlo run. At the same time, long-held assumptions that input-data errors "average out" over long records are not generally valid. Even independent rainfall errors can bias hydrologic parameter estimates (Kavetski et al., 2003), and numerical error from inadequate integration schemes can further contaminate inference. These considerations motivate future work on (i) hybrid physics-machine learning formulations that retain mechanistic interpretability while increasing flexibility where process representations are deficient, and (ii) scalable uncertainty quantification strategies that explicitly propagate forcing and numerical error alongside structural uncertainty.

Taken together, these directions emphasize that SAGE is not only an efficient training algorithm, but also an enabling technology for rapid, reproducible, and statistically grounded experimentation in large-sample hydrology. By making continental-scale learning computationally routine, the framework opens a wide design space spanning optimization, architecture, constraints, predictors, objective functions, temporal resolution, and uncertainty quantification, each of which can now be explored systematically rather than opportunistically.

9 Conclusions

This paper introduced SAGE, a sensitivity-aware learning framework that couples process-based hydrologic ODE models with analytic forward sensitivities and a neural network that maps static catchment attributes to model parameters. By propagating hydrologic states and parameter sensitivities in a single forward integration, SAGE yields exact Jacobians and gradients without finite differences or generic automatic differentiation, enabling transparent, stable, and computationally efficient training at continental scale. SAGE substantially reduces training cost from days to minutes for daily data and makes high-resolution large-sample learning with hourly meteorological forcing and discharge data computationally tractable. This computational leverage enables systematic exploration of model structures, loss functions, parameter priors, and network architectures, and opens the door to hybrid objectives that balance likelihood-based fidelity with distributional performance and regional consistency.

Across CAMELS basins and six conceptual hydrologic models, SAGE achieved strong and consistent out-of-sample performance in both temporal validation and spatial cross-validation. The `Xinanjiang` model delivered the best overall skill, and the learned parameter fields exhibit coherent regional structure rather than unstructured basin-wise variability. At the same time, direct single-site calibration remains an upper bound on attainable fit, underscoring that static attributes and smooth attribute-to-parameter mappings cannot fully capture basin-specific effective parameter variability under model misspecification and data uncertainties.

To summarize and compare large-sample performance in a statistically coherent manner, we introduced the Vrugt–Frame integrated basin loss score, which evaluates models using the full empirical distribution of basin-wise skill rather than a single

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

summary quantile (e.g., median NSE). This distributional metric complements ECDF-based visualizations and enables objective model ranking that remains sensitive to poorly performing regions while avoiding overemphasis on a small subset of basins. Overall, SAGE provides a practical pathway toward scalable, interpretable, and differentiable hydrologic modeling that combines physical process representations with modern learning across continental observational datasets.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Appendix A: Algorithmic recipes

We present an algorithmic description of the Adam optimization method of Kingma and Ba (2015), as used in this work to update the weights and biases of the hidden and output layers of the neural network. First, we summarize the steps of Adam initialization.

Algorithm A.1 Adam: initialization

Input: Number of basin attributes r
Hidden layer width h
Number of hydrologic model parameters d
Output: Initialized moment states and hyperparameters
Set learning rate $\eta \leftarrow 0.1$
Set first-moment decay rate $\beta_1 \leftarrow 0.9$
Set second-moment decay rate $\beta_2 \leftarrow 0.999$
Set stabilization constant $\varepsilon \leftarrow 10^{-8}$
Initialize hidden-layer weight moments:
$\mathbf{m}_{W^1}^{(0)} \leftarrow \mathbf{0}^{h\times r}, \quad \mathbf{v}_{W^1}^{(0)} \leftarrow \mathbf{0}^{h\times r}$
Initialize hidden-layer bias moments:
$\mathbf{m}_{b^1}^{(0)} \leftarrow \mathbf{0}^{h\times 1}, \quad \mathbf{v}_{b^1}^{(0)} \leftarrow \mathbf{0}^{h\times 1}$
Initialize output-layer weight moments:
$\mathbf{m}_{W^2}^{(0)} \leftarrow \mathbf{0}^{d\times h}, \quad \mathbf{v}_{W^2}^{(0)} \leftarrow \mathbf{0}^{d\times h}$
Initialize output-layer bias moments:
$\mathbf{m}_{b^2}^{(0)} \leftarrow \mathbf{0}^{d\times 1}, \quad \mathbf{v}_{b^2}^{(0)} \leftarrow \mathbf{0}^{d\times 1}$
Return $\eta, \beta_1, \beta_2, \varepsilon$ and $\mathbf{m}_{W^1}^{(0)}, \mathbf{v}_{W^1}^{(0)}, \mathbf{m}_{b^1}^{(0)}, \mathbf{v}_{b^1}^{(0)}, \mathbf{m}_{W^2}^{(0)}, \mathbf{v}_{W^2}^{(0)}, \mathbf{m}_{b^2}^{(0)}, \mathbf{v}_{b^2}^{(0)}$ =0

Next, we present a recipe for the dynamic update of the Adam algorithm.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Algorithm A.2 Adam: dynamic update step

Input: Current parameters $\boldsymbol{\phi} = \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$
Current gradients $\nabla_{W^1}\mathcal{L}(\boldsymbol{\phi})$, $\nabla_{b^1}\mathcal{L}(\boldsymbol{\phi})$, $\nabla_{W^2}\mathcal{L}(\boldsymbol{\phi})$, and $\nabla_{b^2}\mathcal{L}(\boldsymbol{\phi})$
Moment states $\mathbf{m}_\pi^{(i-1)}$, $\mathbf{v}_\pi^{(i-1)}$ of hidden and output layers, $\pi \in \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$
Hyperparameters η, β_1, β_2, and ε
Number of training basins K
Iteration counter $i \geq 1$
Output: Updated parameters and moment states

Step 1: Do Euclidean norm clipping, $c = 1$
for $\pi \in \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$ **do**
  $n_\pi \leftarrow \| \mathbf{g}_\pi^{(i)} \|_2$
  if $n_\pi > c$ **then**
    $\overline{\mathbf{g}}_\pi^{(i)} \leftarrow n_\pi^{-1} \mathbf{g}_\pi^{(i)}$
  else
    $\overline{\mathbf{g}}_\pi^{(i)} \leftarrow \mathbf{g}_\pi^{(i)}$
  end if
end for

Step 2: Update first- and second-moment
for $\pi \in \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$ **do**
  $\mathbf{m}_\pi^{(i)} \leftarrow \beta_1 \mathbf{m}_\pi^{(i-1)} + (1-\beta_1)\overline{\mathbf{g}}_\pi^{(i)}$
  $\mathbf{v}_\pi^{(i)} \leftarrow \beta_2 \mathbf{v}_\pi^{(i-1)} + (1-\beta_2)\left(\overline{\mathbf{g}}_\pi^{(i)} \odot \overline{\mathbf{g}}_\pi^{(i)}\right)$
end for

Step 3: Do bias correction
for $\pi \in \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$ **do**
  $\widehat{\mathbf{m}}_\pi^{(i)} \leftarrow \mathbf{m}_\pi^{(i)} / (1-\beta_1^i)$
  $\widehat{\mathbf{v}}_\pi^{(i)} \leftarrow \mathbf{v}_\pi^{(i)} / (1-\beta_2^i)$
end for

Step 4: Update parameters
for $\pi \in \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$ **do**
  $\pi \leftarrow \pi - \eta \widehat{\mathbf{m}}_\pi^{(i)} \oslash \left\{ \sqrt{\widehat{\mathbf{v}}_\pi^{(i)}} + \varepsilon \right\}$
end for

Return $\boldsymbol{\phi} = \{\mathbf{W}^1, \mathbf{b}^1, \mathbf{W}^2, \mathbf{b}^2\}$ and $\mathbf{m}_{W^1}^{(i)}$, $\mathbf{v}_{W^1}^{(i)}$, $\mathbf{m}_{b^1}^{(i)}$, $\mathbf{v}_{b^1}^{(i)}$, $\mathbf{m}_{W^2}^{(i)}$, $\mathbf{v}_{W^2}^{(i)}$, $\mathbf{m}_{b^2}^{(i)}$, $\mathbf{v}_{b^2}^{(i)}$ =0

In this algorithmic recipe, the symbol $\odot$ denotes the Hadamard (elementwise) product. Similarly, the symbol $\oslash$ signifies elementwise (Hadamard) division.

This concludes the algorithmic recipe of Adam.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Appendix B: Jacobian matrices of system dynamics

In this Appendix we review the `gr4j` and `hbv` conceptual watershed models and present analytic expressions of their Jacobian matrices of the system dynamics with respect to their states and parameters, respectively. The models are coded in MATLAB and `C++` and use a mass-conservative second-order integration method with adaptive time step. This guarantees a robust and accurate numerical solution of the simulated fluxes, state variables and sensitivity matrices. Next, we discuss each of the models separately.

B1 GR4J conceptual watershed model

The `GR4J` ("Génie Rural à 4 paramètres Journalier") is a lumped conceptual rainfall–runoff model developed within the French *Génie Rural* tradition. The present implementation follows the four-parameter structure formalized by Perrin et al. (2003), building on earlier production–routing formulations developed at Cemagref (now INRAE). We employ the continuous-time ODE formulation proposed by Mathias et al. (2026), while retaining the model's original daily time scale and its emphasis on parsimony and robustness in representing catchment-scale hydrologic dynamics.

Structurally, `gr4j` is composed of a production store that governs soil moisture accounting, a routing store that controls baseflow recession, and two unit-hydrograph-like routing components that represent fast and slow flow pathways (Figure B.1).

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

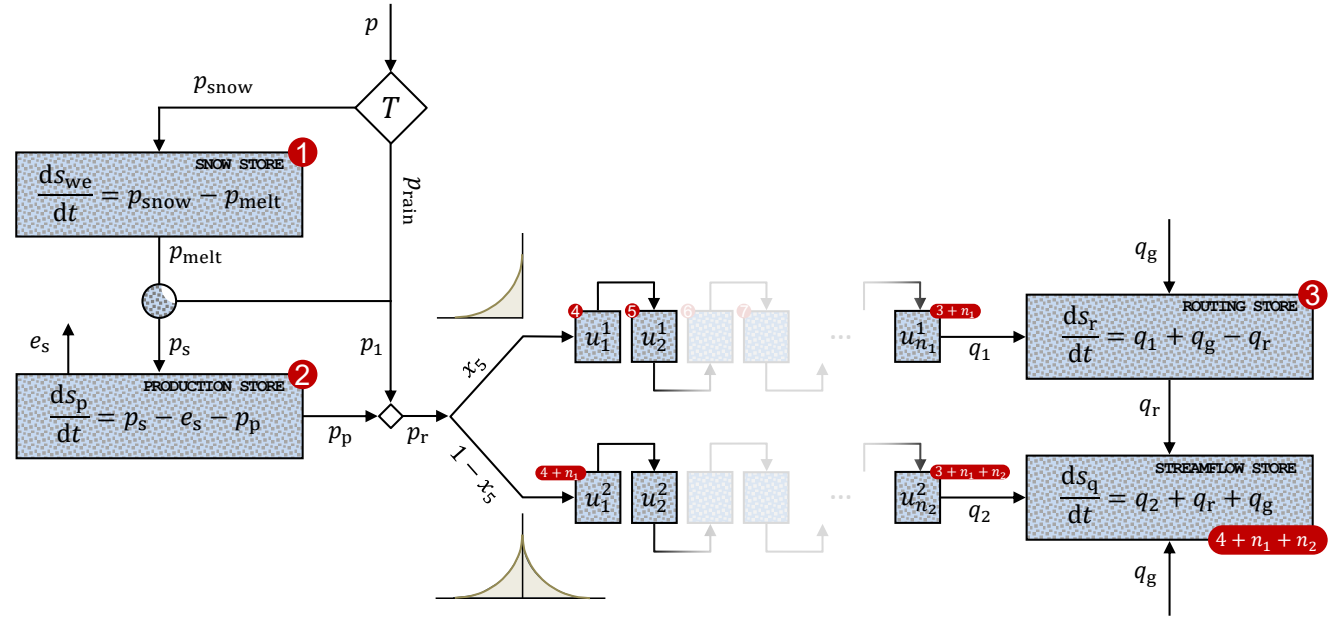

Figure B.1. Schematic illustration of the `gr4j` conceptual watershed model following the ODE formulation of Mathias et al. (2026). Blue boxes labeled in red denote fictitious control volumes governing the rainfall-runoff transformation. The model includes m state variables: the snow water equivalent s_{we} (optional snow module), the production store s_{p}, the routing store s_{r}, two unit-hydrograph-like routing cascades $\mathbf{u}^1 = (u^1 1, \ldots, u^1_{n_1})^\top$ and $\mathbf{u}2 = (u^2_1, \ldots, u^2_{n_2})^\top$ representing fast and slow flow pathways, respectively, and an auxiliary infinite reservoir s_{q} that accumulates discharge. Fluxes (arrows) describe water transfer between compartments, including gross precipitation p, gross potential evapotranspiration e_{p}, snowfall p_{snow}, rainfall p_{rain}, snowmelt p_{amelt}, net precipitation p_{n}, net potential evapotranspiration e_{n}, soil infiltration p_{s}, soil evapotranspiration e_{s}, soil percolation p_{p}, effective rainfall to routing p_{r}, groundwater exchange q_{g}, routing-store outflow q_{r}, and cascade outflows q_1 and q_2. The production store controls soil moisture dynamics and partitions liquid precipitation into infiltration, actual evapotranspiration, percolation, and storage. Effective rainfall $p_{\mathrm{r}} = p_1 + p_{\mathrm{p}}$ is divided into two branches: a fraction x_5 is routed through the fast cascade with n_1 linear reservoirs, while the remaining fraction $(1 - x_5)$ is routed through the slower cascade with n_2 reservoirs. The routing store governs nonlinear routing and groundwater exchange through the fluxes q_{r} and q_{g}. Total discharge is given by $q = q_2 + q_{\mathrm{r}} + q_{\mathrm{g}}$. The model has $d = 8$ parameters: x_1, x_2, x_3, x_4, x_5, f_{p}, T_{tr}, and f_{dd}.

This architecture enables the model to reproduce a wide range of hydroclimatic response behaviors using a small number of physically interpretable parameters. Despite its structural simplicity, `gr4j` has demonstrated strong predictive skill across a wide range of climatic regimes (Perrin et al., 2003; Coron et al., 2012) and is widely used in comparative hydrological studies and large-sample model intercomparison frameworks (Oudin et al., 2005; Knoben et al., 2019; Mathias et al., 2026).

Following recent practice (e.g., Knoben et al., 2018; Mathias et al., 2026), we implement `gr4j` in ODE form with a smooth degree-day snow module and analytic forward sensitivities. This extension introduces two additional parameters: a temperature threshold T_{tr} that governs precipitation phase partitioning and melt onset, and a degree-day factor f_{dd} that controls snowmelt intensity. The resulting model retains the conceptual structure of GR4J while enabling differentiable treatment of snow accumulation and melt processes.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.1 States, parameters, and layout

The state vector comprises the following m storage variables

$$\mathbf{x} = (s_{\mathrm{we}}, s_{\mathrm{p}}, s_{\mathrm{r}}, u_1^1, \ldots, u_{n_1}^1, u_1^2, \ldots, u_{n_2}^2, s_{\mathrm{q}})^\top \in \mathbb{R}^{m\times 1},$$

where s_{we} denotes the snow water equivalent, s_{p} the production store, and s_{r} the routing store. The variables $\{u_k^1\}_{k=1}^{n_1}$ and $\{u_k^2\}_{k=1}^{n_2}$ represents Nash-cascade states for the $100x_5\%$ and $100(1-x_5)\%$ routing branches, respectively, and s_{q} is the discharge reservoir. Following the daily-resolution `gr4j` formulation, we approximate the underlying unit hydrograph with minimal state augmentation and set $n_1 = 2$ and $n_2 = 4$ routing states. Then the vector $\mathbf{x}$ consists of $m = 10$ state variables. Larger values of n_1 and n_2 increase almost quadratically the dimensionality of the augmented ODE system and its associated sensitivity equations, leading to a disproportionate rise in computational cost with negligible impact on simulated discharge at daily resolution. The selected values therefore ensure computational efficiency without compromising model fidelity.

The `gr4j` model has eight parameters ($d = 8$)

$$\boldsymbol{\theta} = (x_1, x_2, x_3, x_4, x_5, f_{\mathrm{p}}, T_{\mathrm{tr}}, f_{\mathrm{dd}})^\top \in \mathbb{R}^{d\times 1},$$

which are summarized in Table B.1 and govern the capacities of the production and routing stores (x_1, x_3), groundwater exchange (x_2), routing time base (x_4), branch partitioning (x_5), potential evapotranspiration scaling (f_{p}), and snow accumulation and melt (T_{tr}, f_{dd}).

Table B.1. Description of the `gr4j` model parameters, including symbols, units, and bounds. The parameters x_5, f_{p}, T_{tr}, and f_{dd} extend the ODE formulation of Mathias et al. (2026) by adding flow partitioning, pan-evaporation correction, and a smooth degree-day snow module.

Symbol	Description	Units	Min.	Max.
x_1	Production store capacity	mm	1	4,000
x_2	Groundwater exchange coefficient	mm/d	−20	15
x_3	Routing store capacity	mm	1	2,000
x_4	Routing time base	d	10^{-2}	10
x_5	Flow partitioning fast and slow branch	–	10^{-3}	1
f_{p}	Ratio potential evapotranspiration to pan evaporation	–	0.3	1.5
T_{tr}	Temperature threshold snow/rain partition and melt	°C	−2	2
f_{dd}	Degree-day melt factor	mm/d/°C	0.1	10

Upper and lower bounds for the core `gr4j` parameters x_1, x_2, x_3, and x_4 were inferred from Figure 8 of Mathias et al. (2026), which presents scatter plots of calibrated parameter values obtained from two different `gr4j` model formulations across a large set of UK catchments. The extrema observed in these plots were used to define a plausible parameter domain for the continuous-time formulation. For the extended parameters x_5, f_{p}, T_{tr}, and f_{dd}, bounds were adopted from values commonly reported in the literature. The resulting parameter ranges are broader than those typically used in discrete-time `gr4j` implementations, but were

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

chosen to ensure numerical robustness of the continuous-time ODE system and its associated sensitivity equations. Narrower bounds would likely reduce the computational cost of the adaptive ODE solver by limiting extreme nonlinear responses and associated step-size reductions.

B1.2 Smooth operators

To ensure differentiability and numerical robustness in compiled code, all non-smooth operations ($\max$, $\min$, and positivity constraints) are replaced by continuously differentiable smooth approximations. Specifically, we use the smooth positivity and smooth minimum operators

$$\mathrm{pos}_\varepsilon(a) = \frac{1}{2}\left(a + \sqrt{a^2+\varepsilon^2}\right)$$

$$\min_\varepsilon(a,b) = \frac{1}{2}\left(a + b - \sqrt{(a-b)^2+\varepsilon^2}\right),$$

with partial derivatives

$$\frac{\partial \mathrm{pos}_\varepsilon}{\partial a} = \frac{1}{2}\left(1 + \frac{a}{\sqrt{a^2+\varepsilon^2}}\right),$$

$$\frac{\partial \min_\varepsilon}{\partial a} = \frac{1}{2}\left(1 - \frac{a-b}{\sqrt{(a-b)^2+\varepsilon^2}}\right), \qquad \frac{\partial \min_\varepsilon}{\partial b} = \frac{1}{2}\left(1 + \frac{a-b}{\sqrt{(a-b)^2+\varepsilon^2}}\right).$$

All raw ODE storages s^* are clamped to be nonnegative using $s = \mathrm{pos}_{\varepsilon_s}(s^*)$ and Jacobians with respect to raw states are obtained by multiplying the corresponding Jacobian columns by $\partial s/\partial s^*$.

B1.3 Model fluxes

Let $\varepsilon_\mathrm{s} > 0$ denote the smoothing parameter used throughout. Define the clamped storages

$$s_\mathrm{we} = \mathrm{pos}_{\varepsilon_\mathrm{s}}(s^*_\mathrm{we})$$

$$s_\mathrm{p} = \mathrm{pos}_{\varepsilon_\mathrm{s}}(s^*_\mathrm{p})$$

$$s_\mathrm{r} = \mathrm{pos}_{\varepsilon_\mathrm{s}}(s^*_\mathrm{r})$$

$$u^1_k = \mathrm{pos}_{\varepsilon_\mathrm{s}}(u^{*1}_k), \quad k = 1,\ldots,n_1$$

$$u^2_k = \mathrm{pos}_{\varepsilon_\mathrm{s}}(u^{*2}_k), \quad k = 1,\ldots,n_2.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.3 Snow module.

Let T_{sm} (°C) be the temperature smoothing scale and define

$$a_{\mathrm{m}} = \frac{T_t - T_{\mathrm{tr}}}{\max(T_{\mathrm{sm}}, \varepsilon_{\mathrm{m}})}.$$

The snow and rain fractions equal $f_{\mathrm{snow}} = \frac{1}{2}\big(1 - \tanh(a_{\mathrm{m}})\big)$, and $f_{\mathrm{rain}} = 1 - f_{\mathrm{snow}}$, respectively, with fluxes of snow $p_{\mathrm{snow}} = p_t f_{\mathrm{snow}}$ and rain $p_{\mathrm{rain}} = p_t f_{\mathrm{rain}}$. Potential snow melt p_{pmelt} (mm/d) is computed as follows

$$p_{\mathrm{pmelt}} = f_{\mathrm{dd}} \cdot \mathrm{pos}_{T_{\mathrm{sm}}}(T_{t\mathrm{m}}),$$

where $T_{t\mathrm{m}} = T_t - T_{\mathrm{tr}}$ in units of °C and

$$\mathrm{pos}_{T_{\mathrm{sm}}}(T_{t\mathrm{m}}) = \frac{1}{2}\left(T_{t\mathrm{m}} + \sqrt{T_{t\mathrm{m}}^2 + T_{\mathrm{sm}}^2}\right).$$

Actual snow melt p_{amelt} (mm/d) is limited by available SWE using a smooth minimum

$$p_{\mathrm{amelt}} = \min_{\varepsilon_{\mathrm{m}}}(s_{\mathrm{we}}, p_{\mathrm{pmelt}}),$$

where $p_{\mathrm{liq}} = p_{\mathrm{rain}} + p_{\mathrm{amelt}}$ (mm/d) is the liquid water flux.

B1.3 Production store.

Define the effective potential evapotranspiration $e_{\mathrm{p,eff}} = f_{\mathrm{p}}\, e_{\mathrm{p}}$ and

$$a_{\mathrm{pe}} = p_{\mathrm{liq}} - e_{\mathrm{p,eff}}.$$

Net precipitation and net evapotranspiration are

$$p_{\mathrm{n}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(a_{\mathrm{pe}}), \qquad e_{\mathrm{n}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(-a_{\mathrm{pe}}),$$

both in units of mm/d. The smooth wet/dry weight is

$$u_{\mathrm{pe}} = \frac{a_{\mathrm{pe}}}{x_1}, \qquad w_{\mathrm{wet}} = \frac{1}{2}\Big(1 + \tanh(u_{\mathrm{pe}})\Big).$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Let $\overline{s}_{\mathrm{p}} = s_{\mathrm{p}}/x_1$ denote the relative production storage. Percolation equals $p_{\mathrm{p}} = s_{\mathrm{p}}\left(1 - g^{-1/4}\right)$ with $g = 1 + \left(\kappa\,\overline{s}_{\mathrm{p}}\right)^4$. The wet-branch storage rate $p_{\mathrm{s}}^{\mathrm{wet}}$ and dry-branch evapotranspiration rate $e_{\mathrm{s}}^{\mathrm{dry}}$ are

$$p_{\mathrm{s}}^{\mathrm{wet}} = x_1 \frac{(1 - \overline{s}_{\mathrm{p}}^2)a_{\mathrm{p}}}{1 + \overline{s}_{\mathrm{p}} a_{\mathrm{p}}}, \qquad e_{\mathrm{s}}^{\mathrm{dry}} = \frac{s_{\mathrm{p}}(2 - \overline{s}_{\mathrm{p}})a_{\mathrm{e}}}{1 + (1 - \overline{s}_{\mathrm{p}})a_{\mathrm{e}}}, \tag{B.1}$$

where $a_{\mathrm{p}} = \tanh(p_{\mathrm{n}}/x_1)$ and $a_{\mathrm{e}} = \tanh(e_{\mathrm{n}}/x_1)$. The blended fluxes are $p_{\mathrm{s}} = w_{\mathrm{wet}}\, p_{\mathrm{s}}^{\mathrm{wet}}$ and $e_{\mathrm{s}} = (1 - w_{\mathrm{wet}})\, e_{\mathrm{s}}^{\mathrm{dry}}$. Finally, the effective rainfall to routing is $p_{\mathrm{r}} = p_1 + p_{\mathrm{p}}$ where $p_1 = p_{\mathrm{n}} - p_{\mathrm{s}}$.

B1.3 Routing store and Nash cascades.

The cascade time constants are

$$\tau_1 = \frac{x_4}{\max(1, n_1)}, \qquad \tau_2 = \frac{2x_4}{\max(1, n_2)}, \qquad \lambda_1 = \frac{1}{\tau_1}, \qquad \lambda_2 = \frac{1}{\tau_2}.$$

The cascade inflows are $q_{1,\mathrm{in}} = x_5 p_{\mathrm{r}}$ and $q_{2,\mathrm{in}} = (1 - x_5)p_{\mathrm{r}}$, and the cascade outflows are $q_1 = u_{n_1}^1 \lambda_1$ and $q_2 = u_{n_2}^2 \lambda_2$. Let $\overline{s}_{\mathrm{r}} = s_{\mathrm{r}}/x_3$ denote the relative routing storage. The net groundwater contribution from neighboring catchments or exchange contribution is

$$q_{\mathrm{g}} = x_2\, \overline{s}_{\mathrm{r}}^{b_{\mathrm{g}}}.$$

Following the updated continuous-time routing used in our implementation, routing-store outflow is represented by the same nonlinear power-law function proposed in Equation 12 of Mathias et al. (2026)

$$q_{\mathrm{r}} = \frac{x_3}{(b_{\mathrm{r}} - 1)\Delta t}\, \overline{s}_{\mathrm{r}}^{b_{\mathrm{r}}}, \tag{B.2}$$

so that q_{r} has units of $\mathrm{mm/d}$. Define $q_{\mathrm{d}} = q_2 + q_{\mathrm{g}}$ and enforce nonnegative direct flow $q_{\mathrm{d},+} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(q_{\mathrm{d}})$. The model discharge is then

$$q = q_{\mathrm{r}} + q_{\mathrm{d},+}.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.4 State equations

The ODE system is

$$
\begin{aligned}
\frac{\mathrm{d}s^*_{\mathrm{we}}}{\mathrm{d}t} &= p_{\mathrm{snow}} - p_{\mathrm{amelt}},\\
\frac{\mathrm{d}s^*_{\mathrm{p}}}{\mathrm{d}t} &= p_{\mathrm{s}} - e_{\mathrm{s}} - p_{\mathrm{p}},\\
\frac{\mathrm{d}s^*_{\mathrm{r}}}{\mathrm{d}t} &= q_1 + q_{\mathrm{g}} - q_{\mathrm{r}},\\
\frac{\mathrm{d}u^{*1}_1}{\mathrm{d}t} &= q_{1,\mathrm{in}} - u^1_1\lambda_1,\\
\frac{\mathrm{d}u^{*1}_k}{\mathrm{d}t} &= u^1_{k-1}\lambda_1 - u^1_k\lambda_1, \qquad k = 2,\dots,n_1,\\
\frac{\mathrm{d}u^{*2}_1}{\mathrm{d}t} &= q_{2,\mathrm{in}} - u^2_1\lambda_2,\\
\frac{\mathrm{d}u^{*2}_k}{\mathrm{d}t} &= u^2_{k-1}\lambda_2 - u^2_k\lambda_2, \qquad k = 2,\dots,n_2,\\
\frac{\mathrm{d}s_{\mathrm{q}}}{\mathrm{d}t} &= q_{\mathrm{r}} + q_{d,+}.
\end{aligned}
$$

The modeled discharge is obtained by temporal differencing of the accumulator state s_{q}.

B1.5 Augmented ODE system for analytic sensitivities

Let $\mathbf{S}(t) = \partial\mathbf{x}(t)/\partial\boldsymbol{\theta}^\top \in \mathbb{R}^{m\times d}$. The forward sensitivity equations are

$$
\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t} = \mathbf{J}_f(\mathbf{x})\,\mathbf{S} + \mathbf{J}_f(\boldsymbol{\theta}),
$$

where

$$
\mathbf{J}_f(\mathbf{x}) = \frac{\partial\mathbf{f}}{\partial\mathbf{x}^\top} \in \mathbb{R}^{m\times m}, \qquad \mathbf{J}_f(\boldsymbol{\theta}) = \frac{\partial\mathbf{f}}{\partial\boldsymbol{\theta}^\top} \in \mathbb{R}^{m\times d}.
$$

Auxiliary derivatives

The following derivatives are required to construct $\mathbf{J}_f(\mathbf{x})$ and $\mathbf{J}_f(\boldsymbol{\theta})$:

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.5 Snow derivatives.

$$\frac{\partial f_{\text{snow}}}{\partial T_{\text{tr}}} = \tfrac{1}{2}\,\operatorname{sech}^2(u_t)\,\max(T_{\text{sm}}, \varepsilon_{\text{m}})^{-1}$$
$$\frac{\partial p_{\text{snow}}}{\partial T_{\text{tr}}} = p_t\,\frac{\partial f_{\text{snow}}}{\partial T_{\text{tr}}}$$
$$\frac{\partial p_{\text{rain}}}{\partial T_{\text{tr}}} = -p_t\,\frac{\partial f_{\text{snow}}}{\partial T_{\text{tr}}}$$
$$\frac{\partial \text{pos}_t}{\partial a_t} = \tfrac{1}{2}\{1 + a_t(a_t^2 + T_{\text{sm}}^2)^{-1/2}\}$$
$$\frac{\partial \text{pos}_t}{\partial T_{\text{tr}}} = -\frac{\partial \text{pos}_t}{\partial a_t}$$
$$\frac{\partial p_{\text{pmelt}}}{\partial T_{\text{tr}}} = f_{\text{dd}}\,\frac{\partial \text{pos}_t}{\partial T_{\text{tr}}}$$
$$\frac{\partial p_{\text{pmelt}}}{\partial f_{\text{dd}}} = \text{pos}_t$$
$$\frac{\partial p_{\text{amelt}}}{\partial s_{\text{we}}} = \frac{\partial \min_{\varepsilon_{\text{m}}}}{\partial a}(s_{\text{we}}, p_{\text{pmelt}})$$
$$\frac{\partial p_{\text{amelt}}}{\partial p_{\text{pmelt}}} = \frac{\partial \min_{\varepsilon_{\text{m}}}}{\partial b}(s_{\text{we}}, p_{\text{pmelt}})$$
$$\frac{\partial p_{\text{liq}}}{\partial s_{\text{we}}} = \frac{\partial p_{\text{amelt}}}{\partial s_{\text{we}}}$$
$$\frac{\partial p_{\text{liq}}}{\partial T_{\text{tr}}} = \frac{\partial p_{\text{rain}}}{\partial T_{\text{tr}}} + \frac{\partial p_{\text{amelt}}}{\partial p_{\text{pmelt}}}\,\frac{\partial p_{\text{pmelt}}}{\partial T_{\text{tr}}}$$
$$\frac{\partial p_{\text{liq}}}{\partial f_{\text{dd}}} = \frac{\partial p_{\text{amelt}}}{\partial p_{\text{pmelt}}}\,\frac{\partial p_{\text{pmelt}}}{\partial f_{\text{dd}}}.$$

B1.5 Wet/dry split derivatives.

Recall $a_{\text{pe}} = p_{\text{liq}} - f_{\text{p}} e_{\text{p}}$ and $u_{\text{pe}} = a_{\text{pe}}/x_1$. Then

$$\frac{\partial p_{\text{n}}}{\partial a_{\text{pe}}} = \frac{\partial \text{pos}_{\varepsilon_{\text{s}}}}{\partial a}(a_{\text{pe}}), \qquad \frac{\partial e_{\text{n}}}{\partial a_{\text{pe}}} = -\frac{\partial \text{pos}_{\varepsilon_{\text{s}}}}{\partial a}(-a_{\text{pe}}),$$
$$\frac{\partial w_{\text{wet}}}{\partial a_{\text{pe}}} = \frac{1}{2}\operatorname{sech}^2(u_{\text{pe}})\frac{1}{x_1}, \qquad \frac{\partial w_{\text{wet}}}{\partial p_{\text{liq}}} = \frac{\partial w_{\text{wet}}}{\partial a_{\text{pe}}},$$
$$\frac{\partial w_{\text{wet}}}{\partial f_{\text{p}}} = \frac{\partial w_{\text{wet}}}{\partial a_{\text{pe}}}\frac{\partial a_{\text{pe}}}{\partial f_{\text{p}}} = -e_{\text{p}}\frac{\partial w_{\text{wet}}}{\partial a_{\text{pe}}}. \tag{B.3}$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.5 Percolation derivatives.

With $\overline{s}_{\mathrm{p}} = s_{\mathrm{p}}/x_1$:

$$\frac{\partial \overline{s}_{\mathrm{p}}}{\partial s_{\mathrm{p}}} = \frac{1}{x_1}$$
$$\frac{\partial \overline{s}_{\mathrm{p}}}{\partial x_1} = -\frac{s_{\mathrm{p}}}{x_1^2}$$
$$\frac{\partial p_{\mathrm{p}}}{\partial s_{\mathrm{p}}} = (1 - g^{-1/4}) + \frac{1}{4} s_{\mathrm{p}} \frac{\partial g}{\partial \overline{s}_{\mathrm{p}}} \frac{\partial \overline{s}_{\mathrm{p}}}{\partial s_{\mathrm{p}}} g^{-5/4}$$
$$\frac{\partial p_{\mathrm{p}}}{\partial x_1} = \frac{1}{4} s_{\mathrm{p}} \frac{\partial g}{\partial \overline{s}_{\mathrm{p}}} \frac{\partial \overline{s}_{\mathrm{p}}}{\partial x_1} g^{-5/4}$$
$$\frac{\partial g}{\partial \overline{s}_{\mathrm{p}}} = 4\kappa^4 \overline{s}_{\mathrm{p}}^3.$$

B1.5 Production-store wet and dry branch derivatives.

The implementation computes p_{s} and e_{s} and their partial derivatives $\partial p_{\mathrm{s}}/\partial s_{\mathrm{p}}$, $\partial e_{\mathrm{s}}/\partial s_{\mathrm{p}}$, $\partial p_{\mathrm{s}}/\partial p_{\mathrm{liq}}$, $\partial e_{\mathrm{s}}/\partial p_{\mathrm{liq}}$, and parameter derivatives $\partial p_{\mathrm{s}}/\partial x_1$, $\partial e_{\mathrm{s}}/\partial x_1$. We denote these compactly as

$$\frac{\partial p_{\mathrm{s}}}{\partial s_{\mathrm{p}}}, \frac{\partial e_{\mathrm{s}}}{\partial s_{\mathrm{p}}}, \frac{\partial p_{\mathrm{s}}}{\partial p_{\mathrm{liq}}}, \frac{\partial e_{\mathrm{s}}}{\partial p_{\mathrm{liq}}}, \frac{\partial p_{\mathrm{s}}}{\partial x_1}, \frac{\partial e_{\mathrm{s}}}{\partial x_1},$$

and similarly for p_{r}:

$$\frac{\partial p_{\mathrm{r}}}{\partial s_{\mathrm{p}}}, \frac{\partial p_{\mathrm{r}}}{\partial p_{\mathrm{liq}}}, \frac{\partial p_{\mathrm{r}}}{\partial x_1}.$$

These quantities are computed in the MATLAB and C++ source codes.

B1.5 Routing-store derivatives.

Let $\overline{s}_{\mathrm{r}} = s_{\mathrm{r}}/x_3$. With $q_{\mathrm{g}} = x_2 \overline{s}_{\mathrm{r}}^{b_{\mathrm{g}}}$ we obtain

$$\frac{\partial q_{\mathrm{g}}}{\partial s_{\mathrm{r}}} = x_2\, b_{\mathrm{g}}\, \overline{s}_{\mathrm{r}}^{(b_{\mathrm{g}}-1)} \frac{1}{x_3}, \qquad \frac{\partial q_{\mathrm{g}}}{\partial x_2} = \overline{s}_{\mathrm{r}}^{b_{\mathrm{g}}},$$
$$\frac{\partial q_{\mathrm{g}}}{\partial x_3} = x_2\, b_{\mathrm{g}}\, \overline{s}_{\mathrm{r}}^{(b_{\mathrm{g}}-1)} \left(-\frac{s_{\mathrm{r}}}{x_3^2}\right).$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

For routing-store outflow $q_\mathrm{r} = x_3/\{(b_\mathrm{r}-1)\Delta t\}\,\overline{s}_\mathrm{r}^{b_\mathrm{r}}$ we have

$$\frac{\partial q_\mathrm{r}}{\partial s_\mathrm{r}} = \frac{x_3}{(b_\mathrm{r}-1)\Delta t}\, b_\mathrm{r}\, \overline{s}_\mathrm{r}^{(b_\mathrm{r}-1)}\, \frac{1}{x_3},$$
$$\frac{\partial q_\mathrm{r}}{\partial x_3} = \frac{1}{(b_\mathrm{r}-1)\Delta t}\left[\overline{s}_\mathrm{r}^{b_\mathrm{r}} + x_3\, b_\mathrm{r}\, \overline{s}_\mathrm{r}^{(b_\mathrm{r}-1)}\left(-\frac{s_\mathrm{r}}{x_3^2}\right)\right].$$

Finally, with $q_\mathrm{d} = q_2 + q_\mathrm{g}$ and $q_{\mathrm{d},+} = \mathrm{pos}_{\varepsilon_\mathrm{s}}(q_\mathrm{d})$ we yield that

$$\frac{\partial q_{\mathrm{d},+}}{\partial q_\mathrm{d}} = \frac{\partial\, \mathrm{pos}_{\varepsilon_\mathrm{s}}}{\partial a}(q_\mathrm{d}), \qquad \frac{\partial q}{\partial s_\mathrm{r}} = \frac{\partial q_\mathrm{r}}{\partial s_\mathrm{r}} + \frac{\partial q_{\mathrm{d},+}}{\partial q_\mathrm{d}}\frac{\partial q_\mathrm{g}}{\partial s_\mathrm{r}}.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.6 Jacobian of system dynamics with respect to states

The Jacobian $\mathbf{J}_f(\mathbf{x}) = \partial \mathbf{f} / \partial \mathbf{x}^\top$ is sparse. The nonzero entries are:

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(1,1)} = -\frac{\partial p_{\mathrm{amelt}}}{\partial s_{\mathrm{we}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(2,1)} = \left(\frac{\partial p_{\mathrm{s}}}{\partial p_{\mathrm{liq}}} - \frac{\partial e_{\mathrm{s}}}{\partial p_{\mathrm{liq}}}\right)\frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(2,2)} = \frac{\partial p_{\mathrm{s}}}{\partial s_{\mathrm{p}}} - \frac{\partial e_{\mathrm{s}}}{\partial s_{\mathrm{p}}} - \frac{\partial p_{\mathrm{p}}}{\partial s_{\mathrm{p}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4,1)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4,2)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial s_{\mathrm{p}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4,4)} = -\lambda_1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+k,4+k-1)} = \lambda_1, \qquad k = 1, \ldots, n_1 - 1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+k,4+k)} = -\lambda_1, \qquad k = 1, \ldots, n_1 - 1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+n_1,1)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+n_1,2)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial s_{\mathrm{p}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+n_1,4+n_1)} = -\lambda_2$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+n_1+k,4+n_1+k-1)} = \lambda_2, \qquad k = 1, \ldots, n_2 - 1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(4+n_1+k,4+n_1+k)} = -\lambda_2, \qquad k = 1, \ldots, n_2 - 1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(3,3+n_1)} = \lambda_1$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(3,3)} = \frac{\partial q_{\mathrm{g}}}{\partial s_{\mathrm{r}}} - \frac{\partial q_{\mathrm{r}}}{\partial s_{\mathrm{r}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(m,3)} = \frac{\partial q}{\partial s_{\mathrm{r}}}$$

$$\left[\mathbf{J}_f(\mathbf{x})\right]_{(m,3+n_1+n_2)} = \lambda_2 \frac{\partial q_{\mathrm{d},+}}{\partial q_{\mathrm{d}}}.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B1.7 Jacobian of system dynamics with respect to parameters

The $m \times d$ parameter Jacobian $\mathbf{J}_f(\boldsymbol{\theta}) = \partial \mathbf{f} / \partial \boldsymbol{\theta}^\top$ has the following nonzero entries (before the chain-rule correction for clamped storages). Columns correspond to $\boldsymbol{\theta} = (x_1, x_2, x_3, x_4, x_5, f_{\mathrm{p}}, T_{\mathrm{tr}}, f_{\mathrm{dd}})^\top$.

Snow (SWE equation):

$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(1,7)} = \frac{\partial p_{\mathrm{snow}}}{\partial T_{\mathrm{tr}}} - \frac{\partial p_{\mathrm{amelt}}}{\partial T_{\mathrm{tr}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(1,8)} = -\frac{\partial p_{\mathrm{amelt}}}{\partial f_{\mathrm{dd}}}.$$

Production store equation:

$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(2,1)} = \frac{\partial p_{\mathrm{s}}}{\partial x_1} - \frac{\partial e_{\mathrm{s}}}{\partial x_1} - \frac{\partial p_{\mathrm{p}}}{\partial x_1},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(2,6)} = \frac{\partial p_{\mathrm{s}}}{\partial f_{\mathrm{p}}} - \frac{\partial e_{\mathrm{s}}}{\partial f_{\mathrm{p}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(2,7)} = \left(\frac{\partial p_{\mathrm{s}}}{\partial p_{\mathrm{liq}}} - \frac{\partial e_{\mathrm{s}}}{\partial p_{\mathrm{liq}}}\right)\frac{\partial p_{\mathrm{liq}}}{\partial T_{\mathrm{tr}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(2,8)} = \left(\frac{\partial p_{\mathrm{s}}}{\partial p_{\mathrm{liq}}} - \frac{\partial e_{\mathrm{s}}}{\partial p_{\mathrm{liq}}}\right)\frac{\partial p_{\mathrm{liq}}}{\partial f_{\mathrm{dd}}}.$$

Cascade inflow states:

$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,5)} = p_{\mathrm{r}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,5)} = -p_{\mathrm{r}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,1)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial x_1},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,1)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial x_1},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,6)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial f_{\mathrm{p}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,6)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial f_{\mathrm{p}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,7)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial T_{\mathrm{tr}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,7)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial T_{\mathrm{tr}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,8)} = x_5 \frac{\partial p_{\mathrm{r}}}{\partial f_{\mathrm{dd}}},$$
$$[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,8)} = (1 - x_5)\frac{\partial p_{\mathrm{r}}}{\partial f_{\mathrm{dd}}}.$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Exchange, routing store, discharge accumulator:

$$
\begin{aligned}
[\mathbf{J}_f(\boldsymbol{\theta})]_{(3,2)} &= \frac{\partial q_{\mathrm{g}}}{\partial x_2}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(m,2)} &= \frac{\partial q_{\mathrm{d},+}}{\partial q_{\mathrm{d}}}\frac{\partial q_{\mathrm{g}}}{\partial x_2}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(3,3)} &= \frac{\partial q_{\mathrm{g}}}{\partial x_3} - \frac{\partial q_{\mathrm{r}}}{\partial x_3}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(m,3)} &= \frac{\partial q_{\mathrm{r}}}{\partial x_3} + \frac{\partial q_{\mathrm{d},+}}{\partial q_{\mathrm{d}}}\frac{\partial q_{\mathrm{g}}}{\partial x_3}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(m,5)} &= \frac{\partial q_{\mathrm{d},+}}{\partial q_{\mathrm{d}}}\frac{\partial q_2}{\partial x_5}.
\end{aligned}
$$

In addition, the time base parameter x_4 affects λ_1 and λ_2 and therefore enters the cascade equations and the routing-store and discharge equations through the cascade outflows:

Time base x_4 :

$$
\begin{aligned}
[\mathbf{J}_f(\boldsymbol{\theta})]_{(4,4)} &= -u_1^1\frac{\partial \lambda_1}{\partial x_4}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+k,4)} &= (u_{k-1}^1 - u_k^1)\frac{\partial \lambda_1}{\partial x_4}, \quad k = 1,\dots,n_1-1, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1,4)} &= -u_1^2\frac{\partial \lambda_2}{\partial x_4}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(4+n_1+k,4)} &= (u_{k-1}^2 - u_k^2)\frac{\partial \lambda_2}{\partial x_4}, \quad k = 1,\dots,n_2-1, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(3,4)} &= u_{n_1}^1\frac{\partial \lambda_1}{\partial x_4}, \\
[\mathbf{J}_f(\boldsymbol{\theta})]_{(m,4)} &= u_{n_2}^2\frac{\partial q_{\mathrm{d},+}}{\partial q_{\mathrm{d}}}\frac{\partial \lambda_2}{\partial x_4}.
\end{aligned}
$$

with

$$
\frac{\partial \lambda_1}{\partial x_4} = -\frac{1}{\tau_1^2}\frac{1}{\max(1,n_1)}, \qquad \frac{\partial \lambda_2}{\partial x_4} = -\frac{1}{\tau_2^2}\frac{2}{\max(1,n_2)}.
$$

B1.8 Chain-rule correction

The ODE is integrated in the *raw* states $s_{\mathrm{we}}^*, s_{\mathrm{p}}^*, s_{\mathrm{r}}^*, u_k^{*1}$ and u_k^{*2}, while all fluxes are evaluated using the clamped states $s = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s^*)$. Therefore, any Jacobian entry computed with respect to the clamped variable must be converted to a Jacobian with respect to the raw state via

$$
\begin{aligned}
\frac{\partial f_i}{\partial s^*} &= \frac{\partial f_i}{\partial s}\frac{\partial s}{\partial s^*} \\
\frac{\partial s}{\partial s^*} &= \frac{1}{2}\left(1 + \frac{s^*}{\sqrt{s^{*2} + \varepsilon_{\mathrm{s}}^2}}\right).
\end{aligned}
$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

This is applied column-wise to $\mathbf{J}_f(\mathbf{x})$ exactly as in the MATLAB and C++ codes. This completes the analytic specification of the augmented ODE system for the `gr4j` model.

B2 HBV conceptual watershed model

The HBV (*Hydrologiska Byråns Vattenavdelning*) hydrologic model is a widely used conceptual rainfall–runoff model originally developed in the 1970s by Sten Bergström at the Swedish Meteorological and Hydrological Institute for operational streamflow forecasting in Scandinavia (Bergström, 1976; Bergström, 1992). The model has since undergone numerous extensions and adaptations and remains a benchmark in hydrologic research and practice (Lindström et al., 1997; Seibert, 2000; Seibert and Vis, 2012). `hbv` combines a temperature-index snow routine, a soil-moisture accounting module, and a two-reservoir response function representing fast and slow runoff components (Lindström et al., 1997). In this work, we implement `hbv` in ODE form with smooth flux definitions and analytic forward sensitivities, consistent with our MATLAB and C++ source codes (Figure B.2).

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

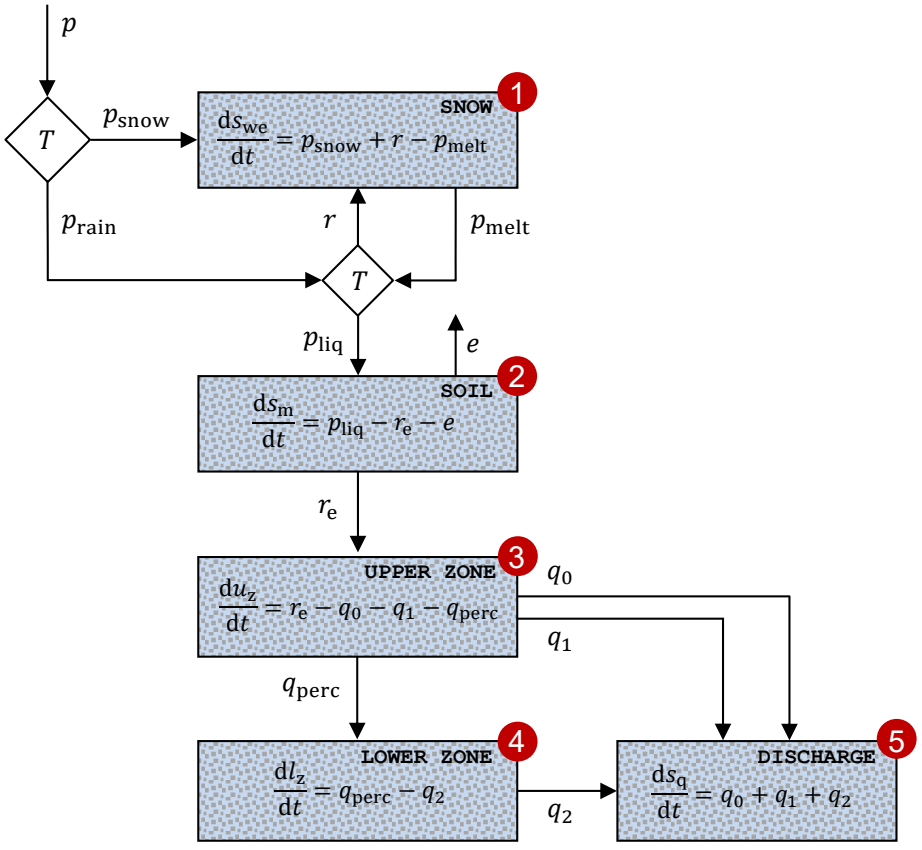

Figure B.2. Schematic illustration of the hbv conceptual watershed model in ODE form. Blue boxes labeled in red are fictitious control volumes. The model includes $m = 5$ state variables: snow water equivalent s_{we}, soil moisture storage s_{m}, upper-zone storage s_{uz}, lower-zone storage s_{lz}, and an auxiliary infinite reservoir s_{q} that accumulates discharge. Fluxes (arrows) describe water movement between compartments: precipitation partitioning into snowfall p_{s} and rainfall p_{r}, snowmelt p_{amelt}, refreezing p_{arfreez}, liquid water input to soil p_{liq}, actual evapotranspiration e_{a}, soil recharge r_{e}, fast runoff q_0 activated above an upper-zone threshold u_{zl}, interflow q_1, baseflow q_2, and percolation from upper to lower zone q_{perc}. Total runoff is $q = q_0 + q_1 + q_2$ and the modeled discharge is obtained by temporal differencing of the accumulator state s_{q}. To emulate the classical HBV routing with maxbas in a differentiable manner, we apply a continuous triangular unit-hydrograph filter as a post-processing step to the raw discharge time series (Section B2.6).

The hbv model is forced by precipitation p_t (mm/d), potential evapotranspiration e_{pt} (mm/d), and air temperature T_t (°C). Non-smooth hbv operations such as positivity constraints, min, max, and the clamp to $[0, 1]$ are replaced by continuously differentiable (smooth) approximations to yield stable compiled implementations and exact analytic sensitivities.

B2.1 States, parameters, and layout

The vector of state variables is

$$\mathbf{x} = (s_{\mathrm{we}}, s_{\mathrm{m}}, s_{\mathrm{uz}}, s_{\mathrm{lz}}, s_{\mathrm{q}})^{\top} \in \mathbb{R}^{m \times 1},$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

where s_{we} is snow water equivalent, s_{m} soil moisture, s_{uz} upper-zone storage, s_{lz} lower-zone storage, and s_{q} is an infinite reservoir that accumulates discharge. Each state has units of length (mm).

The 13 parameters of the hbv model

$$\theta = (f_{\mathrm{c}}, \beta, \ell_{\mathrm{p}}, k_0, u_{\mathrm{zl}}, k_1, k_2, \mathrm{perc}, T_{\mathrm{tr}}, f_{\mathrm{dd}}, \mathrm{sf}_{\mathrm{cf}}, f_{\mathrm{r}}, b_{\mathrm{rt}})^{\top} \in \mathbb{R}^{d\times 1},$$

are listed in Table B.2 and include the soil's field capacity, nonlinearity exponent and evapotranspiration limit, recession coefficients, percolation, temperature threshold, and multipliers (factors) for degree-day melt, snowfall correction and refreezing and routing.

Table B.2. Description of hbv model parameters, including symbols, units, and lower and upper bounds. The snow parameters T_{tr}, f_{dd}, $\mathrm{sf}_{\mathrm{cf}}$, and f_{r} implement a smooth temperature-index accumulation–melt routine. The routing parameter $b_{\mathrm{rt}} > 0$ is a continuous relaxation of the classical maxbas triangular unit-hydrograph.

Symbol	Description	Units	Min.	Max.
f_{c}	Soil field capacity	mm	50	800
β	Soil nonlinearity exponent	–	0.1	6
ℓ_{p}	Evapotranspiration limit parameter	–	0.1	1
k_0	Quick-flow recession coefficient	$1/\mathrm{d}$	10^{-3}	1
u_{zl}	Upper-zone threshold for quick flow	mm	0	200
k_1	Upper-zone recession coefficient (interflow)	$1/\mathrm{d}$	10^{-3}	1
k_2	Lower-zone recession coefficient (baseflow)	$1/\mathrm{d}$	10^{-4}	0.5
perc	Percolation from upper to lower zone	$\mathrm{mm/d}$	0	10
T_{tr}	Temperature threshold (snow/rain and melt)	$^{\circ}\mathrm{C}$	-2	2
f_{dd}	Degree-day melt factor	$\mathrm{mm/d/^{\circ}C}$	0	10
$\mathrm{sf}_{\mathrm{cf}}$	Snowfall correction factor	–	0.5	1.5
f_{r}	Refreezing factor	–	0	0.2
b_{rt}	Routing width	d	0.5	25

B2.2 Smooth operators

Let $\varepsilon_{\mathrm{s}} > 0$ be the smoothing scale used for storage and flux smoothing, $\varepsilon_{\mathrm{t}} > 0$ for temperature transitions, and $\varepsilon_{\mathrm{x}} > 0$ for "safety" lower bounds. We use smooth positivity, smooth minimum, and smooth maximum operators

$$\mathrm{pos}_{\varepsilon}(a) = \tfrac{1}{2}\left(a + \sqrt{a^2 + \varepsilon^2}\right)$$

$$\min_{\varepsilon}(a,b) = \tfrac{1}{2}\left(a + b - \sqrt{(a-b)^2 + \varepsilon^2}\right)$$

$$\max_{\varepsilon}(a,b) = \tfrac{1}{2}\left(a + b + \sqrt{(a-b)^2 + \varepsilon^2}\right)$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

with their respective derivatives

$$\frac{\partial \mathrm{pos}_\varepsilon}{\partial a} = \frac{1}{2}\left(1 + \frac{a}{\sqrt{a^2+\varepsilon^2}}\right)$$
$$\frac{\partial \min_\varepsilon}{\partial a} = \frac{1}{2}\left(1 - \frac{a-b}{\sqrt{(a-b)^2+\varepsilon^2}}\right)$$
$$\frac{\partial \min_\varepsilon}{\partial b} = \frac{1}{2}\left(1 + \frac{a-b}{\sqrt{(a-b)^2+\varepsilon^2}}\right)$$
$$\frac{\partial \max_\varepsilon}{\partial a} = \frac{1}{2}\left(1 + \frac{a-b}{\sqrt{(a-b)^2+\varepsilon^2}}\right).$$

The smooth clamp to $[0,1]$ used in the recharge block is the composition

$$\mathrm{clamp01}_\varepsilon(z) = \min_\varepsilon\big(\max_\varepsilon(z,0), 1\big),$$

with respective derivatives

$$\frac{\partial}{\partial z}\,\mathrm{clamp01}_\varepsilon(z) = \frac{\partial \min_\varepsilon}{\partial a}\big(\max_\varepsilon(z,0),1\big)\,\frac{\partial \max_\varepsilon}{\partial a}(z,0).$$

B2.3 Raw versus clamped storages

The ODE is integrated in the *raw* storages $\mathbf{x}^* = (s^*_{\mathrm{we}}, s^*_{\mathrm{sm}}, s^*_{\mathrm{uz}}, s^*_{\mathrm{lz}}, s_{\mathrm{q}})^\top$, but all fluxes are evaluated using nonnegative clamped storages

$$s_{\mathrm{we}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s^*_{\mathrm{we}})$$
$$s_{\mathrm{m}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s^*_{\mathrm{sm}})$$
$$s_{\mathrm{uz}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s^*_{\mathrm{uz}})$$
$$s_{\mathrm{lz}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s^*_{\mathrm{lz}})$$

with derivatives $\partial s_\bullet / \partial s^*_\bullet = \partial\,\mathrm{pos}_{\varepsilon_{\mathrm{s}}}/\partial a(s^*_\bullet)$ for $\bullet \in \{\mathrm{we}, \mathrm{m}, \mathrm{uz}, \mathrm{lz}\}$.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B2.4 Model fluxes

B2.4 1) Precipitation phase partitioning.

Define

$$a_{\mathrm{m}} = \frac{T_t - T_{\mathrm{tr}}}{\varepsilon_{\mathrm{t}}}, \qquad f_{\mathrm{snow}} = \tfrac{1}{2}\big(1 - \tanh(a_{\mathrm{m}})\big), \qquad f_{\mathrm{rain}} = 1 - f_{\mathrm{snow}},$$

so that

$$p_{\mathrm{s}} = \mathrm{sf}_{\mathrm{cf}}\, p_t\, f_{\mathrm{snow}} \quad \text{and } p_{\mathrm{r}} = p_t\, f_{\mathrm{rain}}.$$

The derivative needed for sensitivities is

$$\frac{\partial f_{\mathrm{snow}}}{\partial T_{\mathrm{tr}}} = \tfrac{1}{2}\,\mathrm{sech}^2(a_{\mathrm{m}})\,\frac{1}{\varepsilon_{\mathrm{t}}},$$

implying

$$\frac{\partial p_{\mathrm{s}}}{\partial \mathrm{sf}_{\mathrm{cf}}} = p_t f_{\mathrm{snow}}, \qquad \frac{\partial p_{\mathrm{s}}}{\partial T_{\mathrm{tr}}} = \mathrm{sf}_{\mathrm{cf}}\, p_t\, \frac{\partial f_{\mathrm{snow}}}{\partial T_{\mathrm{tr}}}, \qquad \frac{\partial p_{\mathrm{r}}}{\partial T_{\mathrm{tr}}} = -p_t\, \frac{\partial f_{\mathrm{snow}}}{\partial T_{\mathrm{tr}}}.$$

B2.4 2) Melt and refreezing.

Let $T_{\mathrm{m}} = T_t - T_{\mathrm{tr}}$ and define smooth positive and negative parts

$$T_{\mathrm{m}}^{+} = \mathrm{pos}_{\varepsilon_{\mathrm{t}}}(T_{\mathrm{m}}), \qquad T_{\mathrm{m}}^{-} = \mathrm{pos}_{\varepsilon_{\mathrm{t}}}(-T_{\mathrm{m}}).$$

Potential melt and refreezing are

$$p_{\mathrm{pmelt}} = f_{\mathrm{dd}}\, T_{\mathrm{m}}^{+}, \qquad p_{\mathrm{prfreez}} = f_{\mathrm{r}}\, f_{\mathrm{dd}}\, T_{\mathrm{m}}^{-},$$

and actual melt p_{amelt} $(\mathrm{mm/d})$ and refreezing p_{arfreez} $(\mathrm{mm/d})$ are limited by storage/inputs using smooth minima,

$$p_{\mathrm{amelt}} = \min_{\varepsilon_{\mathrm{s}}}(s_{\mathrm{we}}, p_{\mathrm{pmelt}}), \qquad p_{\mathrm{arfreez}} = \min_{\varepsilon_{\mathrm{s}}}(p_{\mathrm{r}}, p_{\mathrm{prfreez}}).$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B2.4 3) Liquid water reaching the soil.

The liquid input p_{liq} $(\mathrm{mm/d})$ to the soil is

$$p_{\mathrm{liq}} = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(w^{*}_{\mathrm{in}}).$$

where $w^{*}_{\mathrm{in}} = p_{\mathrm{r}} + p_{\mathrm{amelt}} - p_{\mathrm{arfreez}}$ $(\mathrm{mm/d})$.

B2.4 4) Soil evaporation and recharge.

Actual evapotranspiration e_{a} $(\mathrm{mm/d})$ is controlled by soil moisture s_{m} (mm)

$$e_{\mathrm{a}} = \min_{\varepsilon_{\mathrm{s}}}\left(\frac{s_{\mathrm{m}}}{\ell_{\mathrm{p}} f_{\mathrm{c}}}, 1\right),$$

and recharge is $r_{\mathrm{e}} = p_{\mathrm{liq}}\, u^{\beta}$ $(\mathrm{mm/d})$, where

$$u = \mathrm{clamp01}_{\varepsilon_{\mathrm{s}}}\left(\frac{s_{\mathrm{m}}}{f_{\mathrm{c}}}\right),$$

and, for numerical stability of the power and log evaluations, the implementation uses $u_{\mathrm{p}} = \max_{\varepsilon_{\mathrm{x}}}(u, \varepsilon_{\mathrm{x}})$ and evaluates u_p^{β} and $\log(u_{\mathrm{p}})$.

B2.4 5) Response routine (upper/lower zones).

Fast runoff is activated only above a threshold in the upper zone,

$$h = \mathrm{pos}_{\varepsilon_{\mathrm{s}}}(s_{\mathrm{uz}} - u_{\mathrm{zl}}), \qquad q_0 = k_0 h, \qquad q_1 = k_1 s_{\mathrm{uz}}, \qquad q_2 = k_2 s_{\mathrm{lz}}.$$

Percolation from upper to lower zone is limited by available storage,

$$q_{\mathrm{perc}} = \min_{\varepsilon_{\mathrm{s}}}(\mathrm{perc}, s_{\mathrm{uz}}).$$

Total discharge is $q = q_0 + q_1 + q_2$.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B2.5 State equations

The HBV ODE system (integrated in the raw state space) is

$$\frac{\mathrm{d}s^*_{\mathrm{we}}}{\mathrm{d}t} = p_{\mathrm{s}} + p_{\mathrm{arfreez}} - p_{\mathrm{amelt}}$$
$$\frac{\mathrm{d}s^*_{\mathrm{m}}}{\mathrm{d}t} = p_{\mathrm{liq}} - r_{\mathrm{e}} - e_{\mathrm{a}},$$
$$\frac{\mathrm{d}s^*_{\mathrm{uz}}}{\mathrm{d}t} = r_{\mathrm{e}} - q_0 - q_1 - q_{\mathrm{perc}},$$
$$\frac{\mathrm{d}s^*_{\mathrm{lz}}}{\mathrm{d}t} = q_{\mathrm{perc}} - q_2,$$
$$\frac{\mathrm{d}s_{\mathrm{q}}}{\mathrm{d}t} = q_0 + q_1 + q_2.$$

The modeled raw discharge $q(t)$ is obtained by temporal differencing of the accumulator state s_{q}.

B2.6 Continuous routing operator

The classical HBV implementation applies a triangular unit-hydrograph routing operator controlled by the integer parameter `maxbas`. To preserve differentiability, we replace `maxbas` by a continuous routing-width parameter $b_{\mathrm{rt}} > 0$ and apply routing as a post-processing step to the raw discharge time series. Let $q(t_i)$ denote the raw discharge at discrete times $\{t_i\}_{i=1}^{n}$ obtained from temporal differencing of the state s_{q} of the streamflow accumulation store. We define a causal triangular kernel $w_k(b_{\mathrm{rt}})$ with fixed support length $L_{\max}$ (time steps), normalized such that $\sum_{k=0}^{L_{\max}-1} w_k(b_{\mathrm{rt}}) = 1$, and compute routed discharge as the causal convolution

$$q_{\mathrm{rt}}(t_i) = \sum_{k=0}^{L_{\max}-1} w_k(b_{\mathrm{rt}})\, q(t_{i-k}).$$

The derivative of routed discharge with respect to b_{rt} follows directly as

$$\frac{\partial q_{\mathrm{rt}}(t_i)}{\partial b_{\mathrm{rt}}} = \sum_{k=0}^{L_{\max}-1} \frac{\partial w_k(b_{\mathrm{rt}})}{\partial b_{\mathrm{rt}}}\, q(t_{i-k}),$$

where $\partial w_k / \partial b_{\mathrm{rt}}$ is computed analytically from the smooth kernel definition and normalization.

B2.7 Augmented ODE system for analytic sensitivities

Let $\mathbf{S}(t) = \partial \mathbf{x}(t) / \partial \boldsymbol{\theta}^{\top} \in \mathbb{R}^{m \times d}$. Forward sensitivities satisfy

$$\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t} = \mathbf{J}_f(\mathbf{x})\,\mathbf{S} + \mathbf{J}_f(\boldsymbol{\theta}),$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

where $\mathbf{J}_f(\mathbf{x}) = \partial \mathbf{f}/\partial \mathbf{x}^\top \in \mathbb{R}^{m\times m}$ and $\mathbf{J}_f(\boldsymbol{\theta}) = \partial \mathbf{f}/\partial \boldsymbol{\theta}^\top \in \mathbb{R}^{m\times d}$.

B2.8 Jacobian of system dynamics with respect to states

In the clamped-storage space the Jacobian is sparse with nonzero entries

$$
\begin{aligned}
\left[\mathbf{J}_f(\mathbf{x})\right]_{(1,1)} &= -\frac{\partial p_{\mathrm{amelt}}}{\partial s_{\mathrm{we}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(2,1)} &= \frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}} - \frac{\partial r_{\mathrm{e}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(2,2)} &= -\frac{\partial r_{\mathrm{e}}}{\partial s_{\mathrm{m}}} - \frac{\partial e_{\mathrm{a}}}{\partial s_{\mathrm{m}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(3,1)} &= \frac{\partial r_{\mathrm{e}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial s_{\mathrm{we}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(3,2)} &= \frac{\partial r_{\mathrm{e}}}{\partial s_{\mathrm{m}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(3,3)} &= -\frac{\partial q_0}{\partial s_{\mathrm{uz}}} - \frac{\partial q_1}{\partial s_{\mathrm{uz}}} - \frac{\partial q_{\mathrm{perc}}}{\partial s_{\mathrm{uz}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(4,3)} &= \frac{\partial q_{\mathrm{perc}}}{\partial s_{\mathrm{uz}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(4,4)} &= -\frac{\partial q_2}{\partial s_{\mathrm{lz}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(5,3)} &= \frac{\partial q_0}{\partial s_{\mathrm{uz}}} + \frac{\partial q_1}{\partial s_{\mathrm{uz}}} \\
\left[\mathbf{J}_f(\mathbf{x})\right]_{(5,4)} &= \frac{\partial q_2}{\partial s_{\mathrm{lz}}}.
\end{aligned}
$$

B2.9 Chain-rule correction

Because the ODE integrates raw storages but evaluates fluxes using clamped storages, each Jacobian column associated with a storage state must be mapped back to the raw state via

$$
\begin{aligned}
\frac{\partial f_i}{\partial s_\bullet^*} &= \frac{\partial f_i}{\partial s_\bullet}\frac{\partial s_\bullet}{\partial s_\bullet^*} \\
\frac{\partial s_\bullet}{\partial s_\bullet^*} &= \frac{\partial \,\mathrm{pos}_{\varepsilon_s}}{\partial a}(s_\bullet^*),
\end{aligned}
$$

for $\bullet \in \{\mathrm{we}, \mathrm{m}, \mathrm{uz}, \mathrm{lz}\}$, exactly as in the MATLAB and C++ codes.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

B2.10 Jacobian of system dynamics with respect to parameters

With the parameter ordering given above, the nonzero entries of $\mathbf{J}_f(\boldsymbol{\theta})$ correspond directly to the implemented expressions:

Row 1: $f_1 = p_{\mathrm{s}} + p_{\mathrm{arfreez}} - p_{\mathrm{amelt}}$:

$$\frac{\partial f_1}{\partial T_{\mathrm{tr}}} = \frac{\partial p_{\mathrm{s}}}{\partial T_{\mathrm{tr}}} + \frac{\partial p_{\mathrm{arfreez}}}{\partial T_{\mathrm{tr}}} - \frac{\partial p_{\mathrm{amelt}}}{\partial T_{\mathrm{tr}}}$$
$$\frac{\partial f_1}{\partial f_{\mathrm{dd}}} = \frac{\partial p_{\mathrm{arfreez}}}{\partial f_{\mathrm{dd}}} - \frac{\partial p_{\mathrm{amelt}}}{\partial f_{\mathrm{dd}}}$$
$$\frac{\partial f_1}{\partial \mathrm{sf}_{\mathrm{cf}}} = \frac{\partial p_{\mathrm{s}}}{\partial \mathrm{sf}_{\mathrm{cf}}}$$
$$\frac{\partial f_1}{\partial f_{\mathrm{r}}} = \frac{\partial p_{\mathrm{arfreez}}}{\partial f_{\mathrm{r}}},$$

Row 2: $f_2 = p_{\mathrm{liq}} - r_{\mathrm{e}} - e_{\mathrm{a}}$:

$$\frac{\partial f_2}{\partial T_{\mathrm{tr}}} = \frac{\partial p_{\mathrm{liq}}}{\partial T_{\mathrm{tr}}} - \frac{\partial r_{\mathrm{e}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial T_{\mathrm{tr}}}$$
$$\frac{\partial f_2}{\partial f_{\mathrm{dd}}} = \frac{\partial p_{\mathrm{liq}}}{\partial f_{\mathrm{dd}}} - \frac{\partial r_{\mathrm{e}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial f_{\mathrm{dd}}}$$
$$\frac{\partial f_2}{\partial f_{\mathrm{r}}} = \frac{\partial p_{\mathrm{liq}}}{\partial f_{\mathrm{r}}} - \frac{\partial r_{\mathrm{e}}}{\partial p_{\mathrm{liq}}}\frac{\partial p_{\mathrm{liq}}}{\partial f_{\mathrm{r}}}$$
$$\frac{\partial f_2}{\partial f_{\mathrm{c}}} = -\frac{\partial r_{\mathrm{e}}}{\partial f_{\mathrm{c}}} - \frac{\partial e_{\mathrm{a}}}{\partial f_{\mathrm{c}}}$$
$$\frac{\partial f_2}{\partial \beta} = -\frac{\partial r_{\mathrm{e}}}{\partial \beta}$$
$$\frac{\partial f_2}{\partial \ell_{\mathrm{p}}} = -\frac{\partial e_{\mathrm{a}}}{\partial \ell_{\mathrm{p}}},$$

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

1270

Row 3: $f_3 = r_\mathrm{e} - q_0 - q_1 - q_\mathrm{perc}$:

$$\begin{aligned}
\frac{\partial f_3}{\partial T_\mathrm{tr}} &= \frac{\partial r_\mathrm{e}}{\partial p_\mathrm{liq}} \frac{\partial p_\mathrm{liq}}{\partial T_\mathrm{tr}} \\
\frac{\partial f_3}{\partial f_\mathrm{dd}} &= \frac{\partial r_\mathrm{e}}{\partial p_\mathrm{liq}} \frac{\partial p_\mathrm{liq}}{\partial f_\mathrm{dd}} \\
\frac{\partial f_3}{\partial f_\mathrm{r}} &= \frac{\partial r_\mathrm{e}}{\partial p_\mathrm{liq}} \frac{\partial p_\mathrm{liq}}{\partial f_\mathrm{r}} \\
\frac{\partial f_3}{\partial f_\mathrm{c}} &= \frac{\partial r_\mathrm{e}}{\partial f_\mathrm{c}} \\
\frac{\partial f_3}{\partial \beta} &= \frac{\partial r_\mathrm{e}}{\partial \beta} \\
\frac{\partial f_3}{\partial k_0} &= -\frac{\partial q_0}{\partial k_0} \\
\frac{\partial f_3}{\partial u_\mathrm{zl}} &= -\frac{\partial q_0}{\partial u_\mathrm{zl}}, \\
\frac{\partial f_3}{\partial k_1} &= -\frac{\partial q_1}{\partial k_1} \\
\frac{\partial f_3}{\partial \mathrm{perc}} &= -\frac{\partial q_\mathrm{perc}}{\partial \mathrm{perc}},
\end{aligned}$$

Row 4: $f_4 = q_\mathrm{perc} - q_2$:

$$\begin{aligned}
\frac{\partial f_4}{\partial k_2} &= -\frac{\partial q_2}{\partial k_2} \\
\frac{\partial f_4}{\partial \mathrm{perc}} &= \frac{\partial q_\mathrm{perc}}{\partial \mathrm{perc}},
\end{aligned}$$

Row 5: $f_5 = q_0 + q_1 + q_2$:

$$\begin{aligned}
\frac{\partial f_5}{\partial k_0} &= \frac{\partial q_0}{\partial k_0} \\
\frac{\partial f_5}{\partial u_\mathrm{zl}} &= \frac{\partial q_0}{\partial u_\mathrm{zl}} \\
\frac{\partial f_5}{\partial k_1} &= \frac{\partial q_1}{\partial k_1} \\
\frac{\partial f_5}{\partial k_2} &= \frac{\partial q_2}{\partial k_2}.
\end{aligned} \tag{B.4}$$

This completes the analytic specification of the augmented ODE system for the hbv model.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Acknowledgments

During the preparation of this work, the authors used GPT-5 (developed by OpenAI) to assist with mathematical derivations and language editing. All AI-generated content was carefully reviewed and edited by the authors, who take full responsibility for the final version of the manuscript.

Competing Interests

The authors declare no competing interests.

Software and Data

The conceptual watershed models are implemented in MATLAB, C++, and PYTHON. The software will be made publicly available at https://github.com/jaspervrugt/sagehydrology upon formal acceptance of this paper (Vrugt and Frame, 2026). All data used in this study is available through NCAR CAMELS: https://ral.ucar.edu/solutions/products/camels (Addor et al., 2017b).

Financial support

This research was supported by the Cooperative Institute for Research to Operations in Hydrology (CIROH) with funding under award NA22NWS4320003 from the NOAA Cooperative Institute Program. The statements, findings, conclusions, and recommendations are those of the author(s) and do not necessarily reflect the opinions of NOAA.

Author contribution

Conceptualization: JV and JF, Data curation: JV and JF, Formal analysis: JV and JF, Investigation: JV, Methodology: JV, Project administration, Resources: JV and JF, Software: JV, Supervision: JV, Validation: JV, Visualization: JV and JF, Writing (original draft preparation): JV, Writing (review and editing): JV and JF

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

References

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, et al. TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 265–283, 2016.

N. Addor, A. J. Newman, N. Mizukami, and M. P. Clark. Catchment attributes for large-sample studies. *Boulder, CO: UCAR/NCAR*, 2017a. https://doi.org/10.5065/D6G73C3Q. URL https://doi.org/10.5065/D6G73C3Q/.

N. Addor, A. J. Newman, N. Mizukami, and M. P. Clark. The CAMELS data set: catchment attributes and meteorology for large-sample studies. *Hydrology and Earth System Sciences*, 21(10):5293–5313, 2017b. https://doi.org/10.5194/hess-21-5293-2017. URL https://www.hydrol-earth-syst-sci.net/21/5293/2017/.

E. A. Anderson. National weather service river forecast system—snow accumulation and ablation model. Technical Report NWS HYDRO-17, NOAA, 1973.

R. Arsenault and F. P. Brissette. Continuous streamflow prediction in ungauged basins: The effects of equifinality and parameter set selection on uncertainty in regionalization approaches. *Water Resources Research*, 50(7):6135–6153, 2014.

H. E. Beck, A. I. Van Dijk, A. De Roo, E. Dutra, G. Fink, R. Orth, and J. Schellekens. Global evaluation of runoff from 10 state-of-the-art hydrological models. *Hydrology and Earth System Sciences*, 21(6):2881–2903, 2017.

S. Bergström. *Development and Application of a Conceptual Runoff Model for Scandinavian Catchments*. Phd thesis, SMHI, Norrköping, Sweden, 1976.

S. Bergström. The HBV model—its structure and applications. Technical Report 4, Swedish Meteorological and Hydrological Institute (SMHI), Norrköping, Sweden, 1992.

S. Bergström. The hbv model. In *Computer Models of Watershed Hydrology*, pages 443–476. Water Resources Publications, 1995.

K. Beven and J. Freer. Equifinality, data assimilation, and uncertainty estimation in mechanistic modelling of complex environmental systems using the GLUE methodology. *Journal of Hydrology*, 249(1):11–29, 2001. ISSN 0022-1694. https://doi.org/10.1016/S0022-1694(01)00421-8. URL http://www.sciencedirect.com/science/article/pii/S0022169401004218.

J. Bradbury, R. Frostig, P. Hawkins, M. Johnson, C. Leary, D. Maclaurin, R. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs. *Version 0.1*, 2018. URL: https://github.com/google/jax.

R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995. https://doi.org/10.1137/0916069.

D. G. Cacuci. Sensitivity theory for nonlinear systems. *Journal of Mathematical Physics*, 22(12):2794–2802, 1981. https://doi.org/10.1063/1.524385.

M. P. Clark, B. Nijssen, J. D. Lundquist, D. Kavetski, D. E. Rupp, R. A. Woods, J. E. Freer, E. D. Gutmann, A. W. Wood, L. D. Brekke, J. R. Arnold, D. J. Gochis, and R. M. Rasmussen. A unified approach for process-based hydrologic modeling: 1. modeling concept. *Water Resources Research*, 51(4):2498–2514, 2015. https://doi.org/https://doi.org/10.1002/2015WR017198. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2015WR017198.

L. Coron, V. Andréassian, C. Perrin, J. Lerat, J. Vaze, M. Bourqui, and F. Hendrickx. Crash testing hydrological models in contrasted climate conditions: An experiment on 216 australian catchments. *Water Resources Research*, 48(5), 2012. https://doi.org/10.1029/2011WR011721. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2011WR011721.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. https://doi.org/10.1007/BF02551274.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

Q. Duan, S. Sorooshian, and V. Gupta. Effective and efficient global optimization for conceptual rainfall-runoff models. *Water Resources Research*, 28(4):1015–1031, 1992. https://doi.org/10.1029/91WR02985. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/91WR02985.

D. Feng, J. Liu, K. Lawson, and C. Shen. Differentiable, learnable, regionalized process-based models with multiphysical outputs can approach state-of-the-art hydrologic prediction accuracy. *Water Resources Research*, 58(10):e2022WR032404, 2022. https://doi.org/10.1029/2022WR032404.

D. Feng, H. Beck, K. Lawson, and C. Shen. The suitability of differentiable, physics-informed machine learning hydrologic models for ungauged regions and climate change impact assessment. *Hydrology and Earth System Sciences*, 27:2357–2373, 2023. https://doi.org/10.5194/hess-27-2357-2023.

D. Frazier, D. Nott, C. Drovandi, and R. Kohn. Bayesian inference using synthetic likelihood: Asymptotics and adjustments. *Journal of the American Statistical Association*, 118(544):2821–2832, 2023. https://doi.org/10.1080/01621459.2022.2086132. URL https://www.tandfonline.com/doi/full/10.1080/01621459.2022.2086132.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

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

A. Griewank and A. Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, 2008.

H. V. Gupta, S. Sorooshian, and P. O. Yapo. Toward improved calibration of hydrologic models: Multiple and noncommensurable measures of information. *Water Resources Research*, 34(4):751–763, 1998. https://doi.org/10.1029/97WR03495. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/97WR03495.

H. V. Gupta, H. Kling, K. K. Yilmaz, and G. F. Martinez. Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. *Journal of Hydrology*, 377(1):80–91, 2009. ISSN 0022-1694. https://doi.org/10.1016/j.jhydrol.2009.08.003. URL http://www.sciencedirect.com/science/article/pii/S0022169409004843.

Y. He, A. Bárdossy, and E. Zehe. A review of regionalisation for continuous streamflow simulation. *Hydrology and Earth System Sciences*, 15: 3539–3553, 2011. https://doi.org/10.5194/hess-15-3539-2011. URL https://doi.org/10.5194/hess-15-3539-2011.

G. M. Hornberger and R. C. Spear. An approach to the preliminary analysis of environmental systems. *Journal of Environmental Management*, 12:7–18, 1981. https://doi.org/10.1016/0301-4797(81)90043-9.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. https://doi.org/10.1016/0893-6080(91)90009-T.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. https://doi.org/10.1016/0893-6080(89)90020-8.

P. Huber and E. Ronchetti. *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley, 2011. ISBN 9781118210338. URL https://books.google.nl/books?id=j1OhquR_j88C.

P. J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964. https://doi.org/10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

N. N. T. Huynh, P.-A. Garambois, F. Colleoni, and J. Monnier. A hybrid physics–ai approach using universal differential equations with state-dependent neural networks for learnable, regionalizable, spatially distributed hydrological modeling. *Geoscientific Model Development*, 19(3):1055–1074, 2026. https://doi.org/10.5194/gmd-19-1055-2026. URL https://gmd.copernicus.org/articles/19/1055/2026/.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

M. Innes, A. Edelman, K. Fischer, C. Rackauckas, E. Saba, V. B. Shah, and W. Tebbutt. A differentiable programming system to bridge machine learning and scientific computing, 2019. URL https://arxiv.org/abs/1907.07587.

D. Kavetski, G. Kuczera, and S. W. Franks. Semidistributed hydrological modeling: A "saturation path" perspective on TOPMODEL and VIC. *Water Resources Research*, 39(9), 2003. https://doi.org/10.1029/2003WR002122. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2003WR002122.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

H. Kling, M. Fuchs, and M. Paulin. Runoff conditions in the upper Danube basin under an ensemble of climate change scenarios. *Journal of Hydrology*, 424–425:264–277, 2012. https://doi.org/10.1016/j.jhydrol.2012.01.011.

W. J. M. Knoben, R. A. Woods, and J. E. Freer. A quantitative hydrological climate classification evaluated with independent streamflow data. *Water Resources Research*, 54(7):5088–5109, 2018. https://doi.org/10.1029/2018WR022913. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2018WR022913.

W. J. M. Knoben, J. E. Freer, K. J. A. Fowler, M. C. Peel, and R. A. Woods. Modular assessment of rainfall–runoff models toolbox (marrmot) v1.2: an open-source, extendable framework providing implementations of 46 conceptual hydrologic models as continuous state-space formulations. *Geoscientific Model Development*, 12(6):2463–2480, 2019. https://doi.org/10.5194/gmd-12-2463-2480.

F. Kratzert, D. Klotz, C. Brenner, K. Schulz, and M. Herrnegger. Rainfall–runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022, 2018. https://doi.org/10.5194/hess-22-6005-2018. URL https://hess.copernicus.org/articles/22/6005/2018/.

F. Kratzert, D. Klotz, M. Herrnegger, A. K. Sampson, S. Hochreiter, and G. S. Nearing. Toward improved predictions in ungauged basins: Exploiting the power of machine learning. *Water Resources Research*, 55(12):11344–11354, 2019a. https://doi.org/https://doi.org/10.1029/2019WR026065. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026065.

F. Kratzert, D. Klotz, G. Shalev, G. Klambauer, S. Hochreiter, and G. Nearing. Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. *Hydrology and Earth System Sciences*, 23(12):5089–5110, 2019b. https://doi.org/10.5194/hess-23-5089-2019. URL https://www.hydrol-earth-syst-sci.net/23/5089/2019/.

F. Kratzert, D. Klotz, G. Shalev, G. Klambauer, S. Hochreiter, and G. S. Nearing. Benchmarking a catchment-aware long short-term memory network (lstm) for large-scale hydrological modeling. *ArXiv*, abs/1907.08456, 2019c. URL https://api.semanticscholar.org/CorpusID:197935507.

X. Liang, D. P. Lettenmaier, E. F. Wood, and S. J. Burges. A simple hydrologically based model of land surface water and energy fluxes for general circulation models. *Journal of Geophysical Research: Atmospheres*, 99(D7):14415–14428, 1994. https://doi.org/https://doi.org/10.1029/94JD00483. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/94JD00483.

G. Lindström, B. Johansson, M. Persson, M. Gardelin, and S. Bergström. Development and test of the distributed HBV-96 hydrological model. *Journal of Hydrology*, 201(1–4):272–288, 1997. https://doi.org/10.1016/S0022-1694(97)00041-3.

J. Martinec, A. Rango, and R. Roberts. Snowmelt runoff model (srm) user's manual. Technical Report NASA Reference Publication 1100, NASA Goddard Space Flight Center, Washington, D.C., USA, 1992. Updated Edition, Version 3.2.

J. E. Matheson and R. L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):1087–1096, 1976. https://doi.org/10.1287/mnsc.22.10.1087. URL https://doi.org/10.1287/mnsc.22.10.1087.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

S. A. Mathias, C. Thébault, and A. Ireson. A theoretical appraisal of the gr4j rainfall-runoff modelling framework. *Journal of Hydrology*, 664:134393, 2026. ISSN 0022-1694. https://doi.org/https://doi.org/10.1016/j.jhydrol.2025.134393. URL https://www.sciencedirect.com/science/article/pii/S0022169425017330.

J. Nash and J. Sutcliffe. River flow forecasting through conceptual models part I - A discussion of principles. *Journal of Hydrology*, 10(3): 282–290, 1970. ISSN 0022-1694. https://doi.org/10.1016/0022-1694(70)90255-6. URL https://www.sciencedirect.com/science/article/pii/0022169470902556.

G. S. Nearing, F. Kratzert, A. K. Sampson, C. S. Pelissier, D. Klotz, J. M. Frame, C. Prieto, and H. V. Gupta. What role does hydrological science play in the age of machine learning? *Water Resources Research*, 57(3):e2020WR028091, 2021. https://doi.org/https://doi.org/10.1029/2020WR028091. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020WR028091. e2020WR028091 10.1029/2020WR028091.

A. J. Newman, K. Sampson, M. P. Clark, A. Bock, R. J. Viger, and D. Blodgett. A large-sample watershed-scale hydrometeorological dataset for the contiguous USA. *Boulder, CO: UCAR/NCAR*, 2014. https://doi.org/10.5065/D6MW2F4D. URL https://dx.doi.org/10.5065/D6MW2F4D/.

A. J. Newman, M. P. Clark, K. Sampson, A. Wood, L. E. Hay, A. Bock, R. J. Viger, D. Blodgett, L. Brekke, J. R. Arnold, T. Hopson, and Q. Duan. Development of a large-sample watershed-scale hydrometeorological data set for the contiguous USA: data set characteristics and assessment of regional variability in hydrologic model performance. *Hydrology and Earth System Sciences*, 19(1):209–223, 2015. https://doi.org/10.5194/hess-19-209-2015. URL https://www.hydrol-earth-syst-sci.net/19/209/2015/.

A. J. Newman, N. Mizukami, M. P. Clark, A. W. Wood, B. Nijssen, and G. Nearing. Benchmarking of a physically based hydrologic model. *Journal of Hydrometeorology*, 18(8):2215–2225, 2017. https://doi.org/10.1175/JHM-D-16-0284.1. URL https://doi.org/10.1175/JHM-D-16-0284.1.

L. Oudin, F. Hervieu, C. Michel, C. Perrin, V. Andréassian, F. Anctil, and C. Loumagne. Which potential evapotranspiration input for a lumped rainfall–runoff model? part 2—towards a simple and efficient potential evapotranspiration model for rainfall–runoff modelling. *Journal of Hydrology*, 303:290–306, 2005. https://doi.org/10.1016/j.jhydrol.2004.08.026.

L. Oudin et al. Spatial proximity, physical similarity, and multiple regression in hydrologic regionalization. *Water Resources Research*, 44: W08416, 2008. https://doi.org/10.1029/2007WR006716. Explores multiple regionalization strategies with GR4J and TOPMODEL.

J. Parajka, G. Blöschl, and R. Merz. Regional calibration of catchment models: Potential for ungauged catchments. *Water Resources Research*, 43(6), 2007. https://doi.org/https://doi.org/10.1029/2006WR005271. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2006WR005271.

A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

C. Perrin, C. Michel, and V. Andréassian. Improvement of a parsimonious model for streamflow simulation. *Journal of Hydrology*, 279(1-4): 275–289, 2003. https://doi.org/10.1016/S0022-1694(03)00225-7. GR4J four-parameter rainfall–runoff model description and performance assessment.

A. Perumal and R. Gunawan. Understanding dynamics using sensitivity analysis: caveat and solution. *BMC Systems Biology*, 5(41):1–19, 2011. https://doi.org/10.1186/1752-0509-5-41.

G. Peyré and M. Cuturi. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5–6):355–607, 2019. https://doi.org/10.1561/2200000073.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

F. Pianosi, K. Beven, J. Freer, J. W. Hall, J. Rougier, D. B. Stephenson, and T. Wagener. Sensitivity analysis of environmental models: A systematic review with practical workflow. *Environmental Modelling & Software*, 79:214–232, 2016. ISSN 1364-8152. https://doi.org/https://doi.org/10.1016/j.envsoft.2016.02.008. URL https://www.sciencedirect.com/science/article/pii/S1364815216300287.

C. Rackauckas, Z. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman. Universal differential equations for scientific machine learning. *arXiv:2001.04385 [cs, stat]*, 2020.

M. Raissi, P. Perdikaris, and G. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. https://doi.org/https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

T. Razavi and P. Coulibaly. Streamflow prediction in ungauged basins: Review of regionalization methods. *Journal of Hydrologic Engineering*, 18(8):958–975, 2013. https://doi.org/10.1061/(ASCE)HE.1943-5584.0000690. URL https://doi.org/10.1061/(ASCE)HE.1943-5584.0000690.

M. Reichstein, G. Camps-Valls, B. Stevens, et al. Deep learning and process understanding for data-driven earth system science. *Nature*, 566: 195–204, 2019. https://doi.org/10.1038/s41586-019-0912-1.

L. Samaniego, R. Kumar, and S. Attinger. Multiscale parameter regionalization of a grid-based hydrologic model at the mesoscale. *Water Resources Research*, 46(5), 2010. https://doi.org/https://doi.org/10.1029/2008WR007327. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2008WR007327.

J. Seibert. Multi-criteria calibration of a conceptual runoff model using a genetic algorithm. *Hydrology and Earth System Sciences*, 4(2): 215–224, 2000. https://doi.org/10.5194/hess-4-215-2000. URL https://hess.copernicus.org/articles/4/215/2000/.

J. Seibert and M. J. P. Vis. Teaching hydrological modeling with a user-friendly catchment-runoff-model software package. *Hydrology and Earth System Sciences*, 16(9):3315–3325, 2012. https://doi.org/10.5194/hess-16-3315-2012. URL https://hess.copernicus.org/articles/16/3315/2012/.

C. Shen, A. P. Appling, P. Gentine, T. Bandai, H. Gupta, A. Tartakovsky, M. Baity-Jesi, F. Fenicia, D. Kifer, L. Li, et al. Differentiable modelling to unify machine learning and physical models for geosciences. *Nature Reviews Earth & Environment*, 4(8):552–567, 2023.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1): 267–288, 1996.

J. A. Vrugt. Markov chain Monte Carlo simulation using the DREAM software package: Theory, concepts, and matlab implementation. *Environmental Modelling & Software*, 75:273–316, 2016. ISSN 1364-8152. https://doi.org/10.1016/j.envsoft.2015.08.013. URL http://www.sciencedirect.com/science/article/pii/S1364815215300396.

J. A. Vrugt. Distribution-based model evaluation and diagnostics: Elicitability, propriety, and scoring rules for hydrograph functionals. *Water Resources Research*, 60(6):e2023WR036710, 2024. https://doi.org/10.1029/2023WR036710. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2023WR036710. e2023WR036710 2023WR036710.

J. A. Vrugt and C. G. H. Diks. The learning rate is not a constant: Sandwich-adjusted Markov chain Monte Carlo simulation. *Entropy*, 27(10), 2025. ISSN 1099-4300. https://doi.org/10.3390/e27100999. URL https://www.mdpi.com/1099-4300/27/10/999.

J. A. Vrugt and J. M. Frame. Sagehydrology: Software for hydrologic model calibration and analysis. https://doi.org/10.5281/zenodo.18488836, 2026. GitHub repository: https://github.com/jaspervrugt/SAGEhydrology/.

J. A. Vrugt and Y. Gao. On the three-parameter infiltration equation of parlange et al. (1982): Numerical solution, experimental design, and parameter estimation. *Vadose Zone Journal*, 21(1):e20167, 2022. https://doi.org/https://doi.org/10.1002/vzj2.20167. URL https://acsess.onlinelibrary.wiley.com/doi/abs/10.1002/vzj2.20167.

https://doi.org/10.5194/egusphere-2026-693
Preprint. Discussion started: 18 February 2026

J. A. Vrugt, H. V. Gupta, W. Bouten, and S. Sorooshian. A shuffled complex evolution Metropolis algorithm for optimization and uncertainty assessment of hydrologic model parameters. *Water Resources Research*, 39(8), 2003. https://doi.org/10.1029/2002WR001642. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2002WR001642.

J. A. Vrugt, C. G. H. Diks, R. de Punder, and P. Grünwald. A sandwich with water: Bayesian & frequentist uncertainty quantification under model misspecification. *ARC Geophysical Research*, 16(1), 2025. https://doi.org/10.5149/ARC-GR.1824. URL https://janeway.uncpress.org/ARC-GR/article/id/1824/.

J. A. Vrugt, J. M. Frame, and E. Bollman. Reclaiming first principles: A differentiable framework for conceptual hydrologic models. *Water Resources Research*, 2026. Manuscript under review.

T. Wagener, N. McIntyre, M. J. Lees, H. S. Wheater, and H. V. Gupta. Towards reduced uncertainty in conceptual rainfall-runoff modelling: dynamic identifiability analysis. *Hydrological Processes*, 17(2):455–476, 2003. https://doi.org/10.1002/hyp.1135. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/hyp.1135.

E. Walter and L. Pronzato. *Identification of Parametric Models*. Springer, 1997. https://doi.org/10.1007/978-1-4612-2020-6.

Y.-H. Wang and H. V. Gupta. Towards interpretable physical-conceptual catchment-scale hydrological modeling using the mass-conserving-perceptron. *Water Resources Research*, 60(10):e2024WR037224, 2024. https://doi.org/https://doi.org/10.1029/2024WR037224. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2024WR037224. e2024WR037224 2024WR037224.