

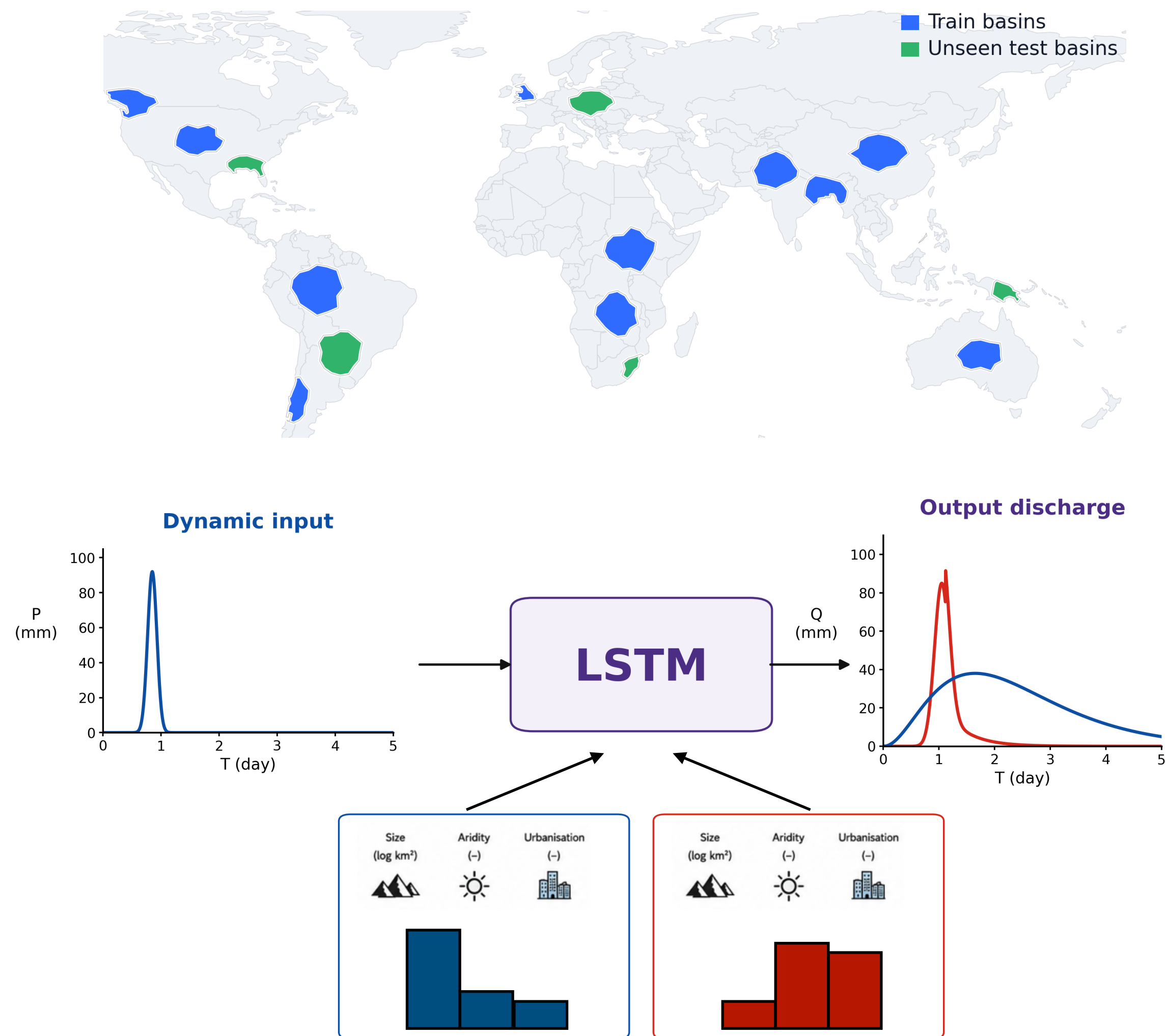
Differentiable river routing for end-to-end learning of hydrological processes

36th Hydraulic Engineering Online Continuous Lecture Series

Tristan Hascoet, July 6th 2026

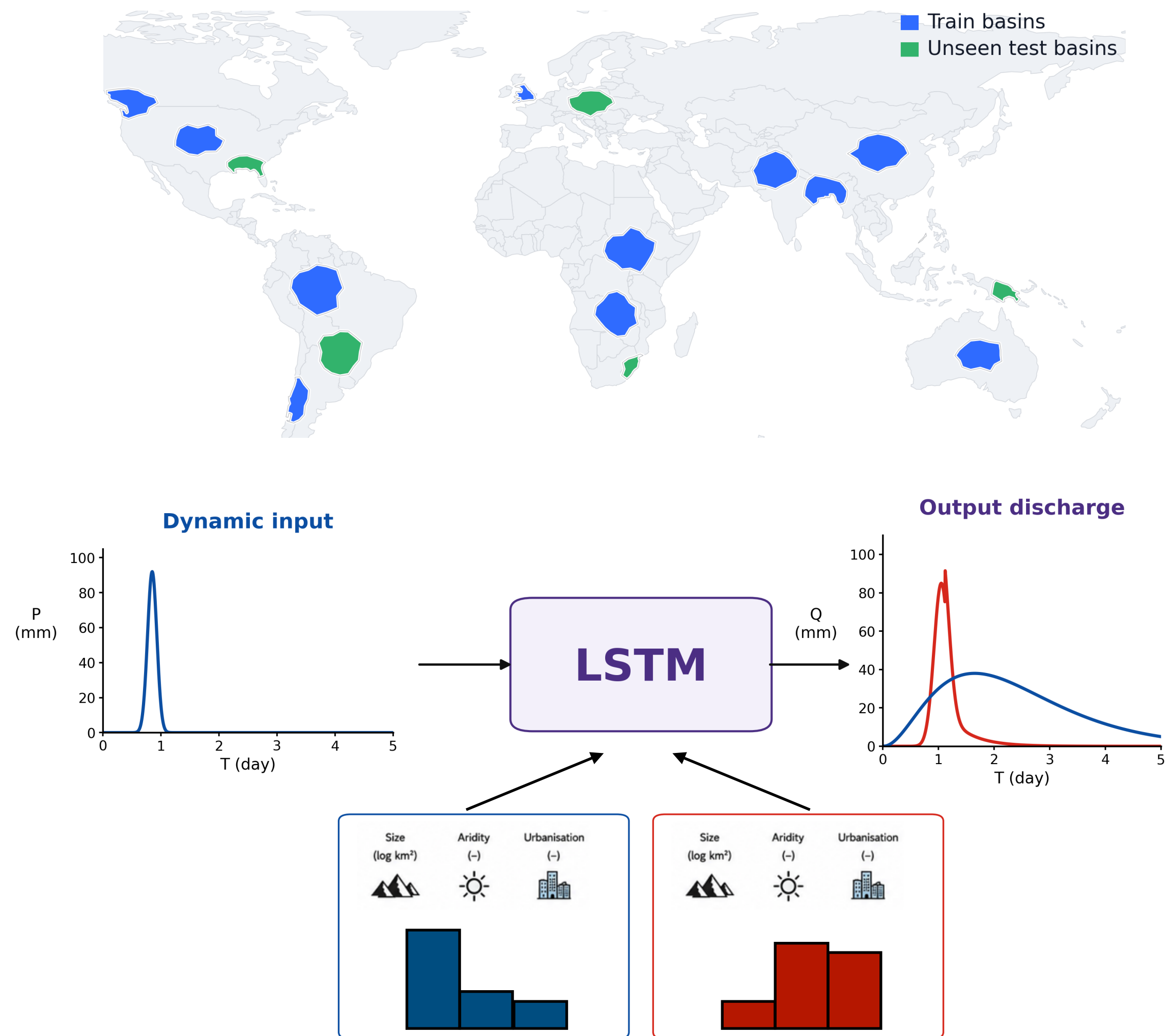


Context / Positioning

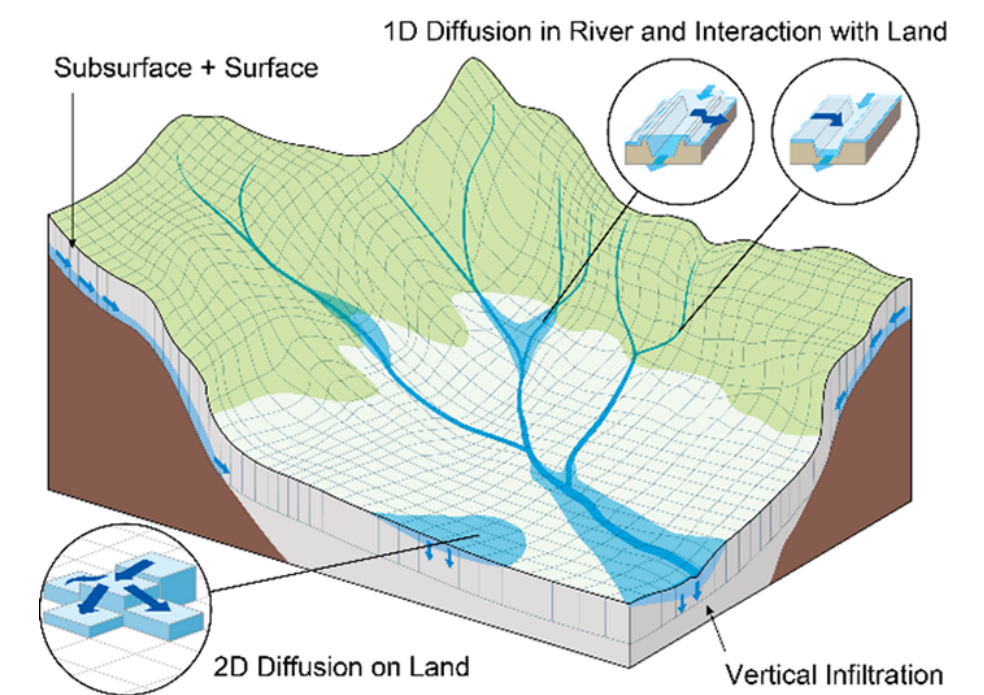


- Deep Learning (LSTM) have been quite successful in lumped modeling at global scale
- Trained on large sample datasets:
 - GRDC, Caravan > 10,000 catchments
- Demonstrated high accuracy on unseen test regions (ungaged basins)
- Support services like Google Flood
- High predictive accuracy
- Hard interpretation
- No physical consistency
- No spatial structure
- Fragile to generalize in a changing climate

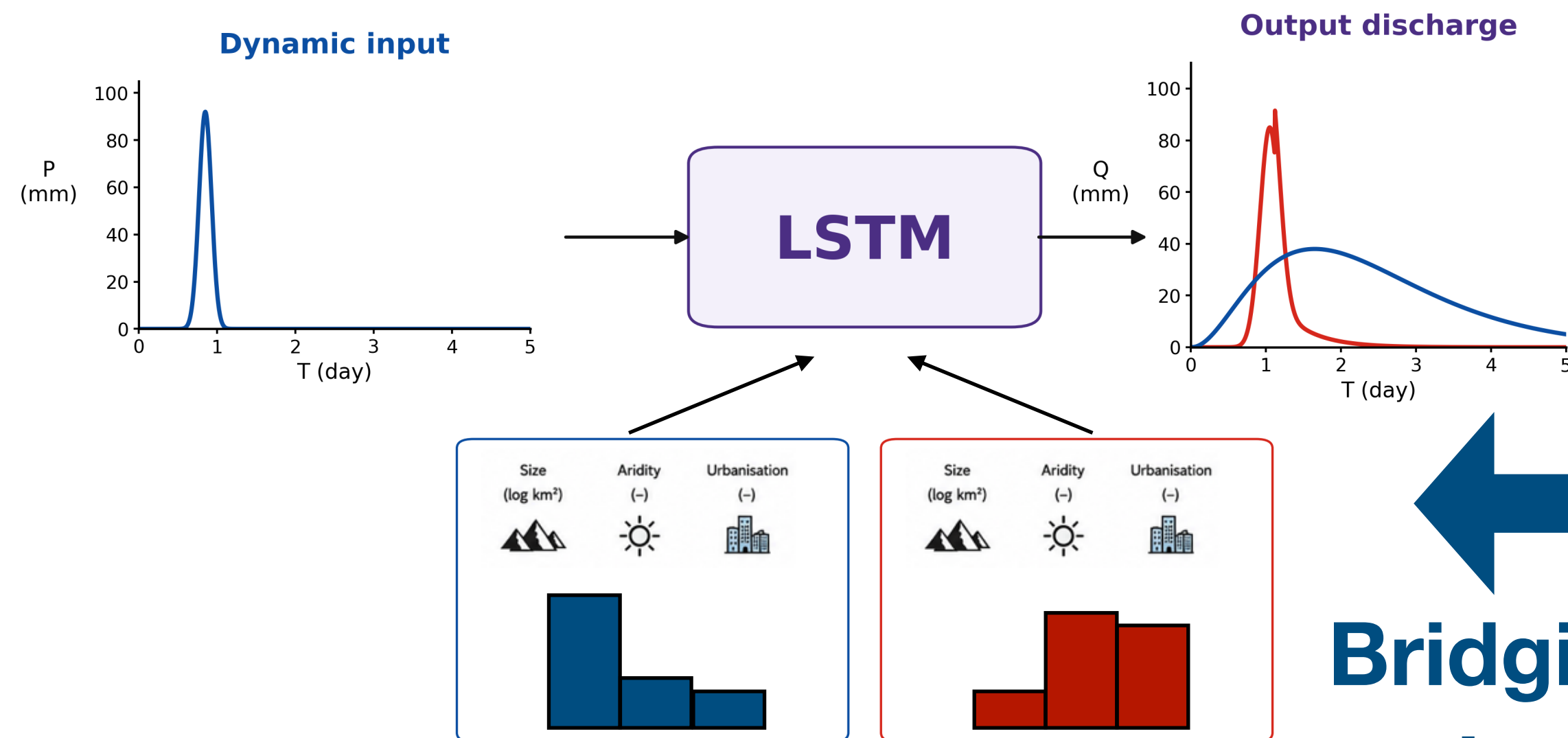
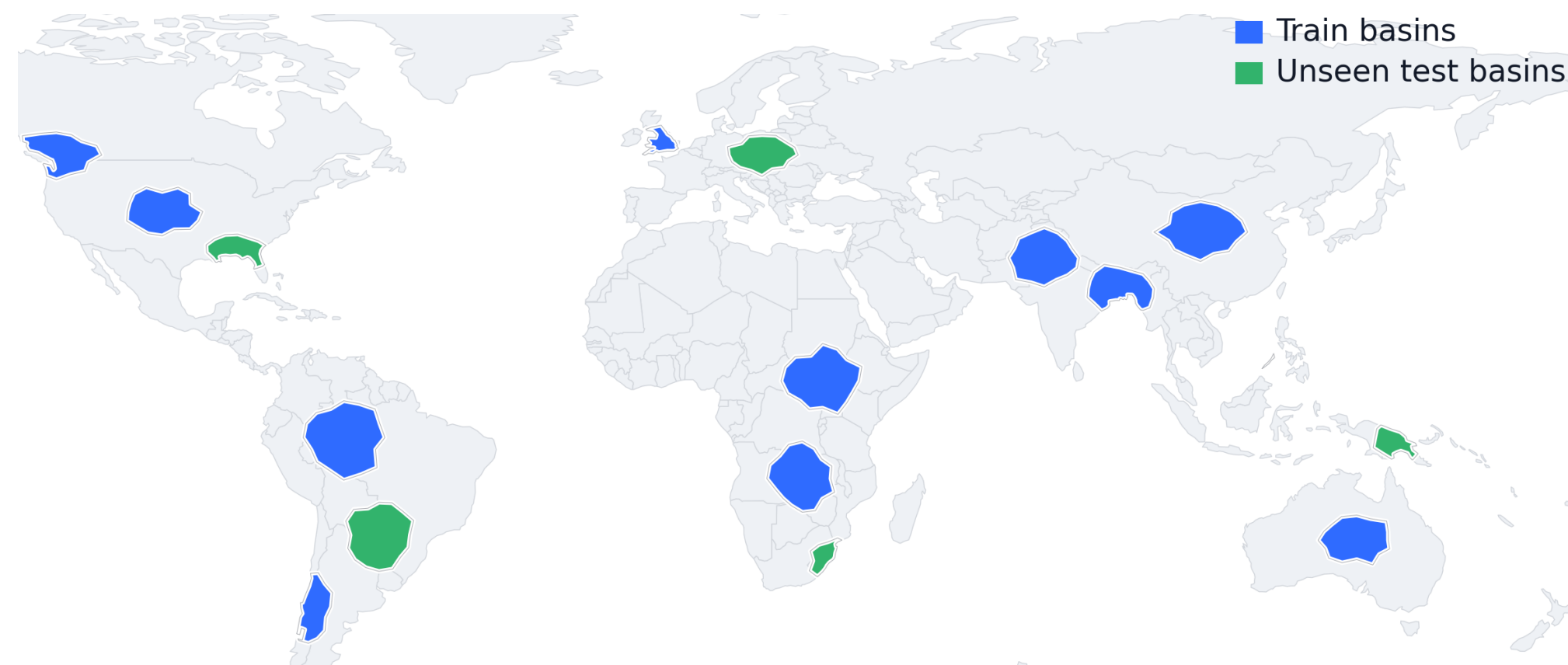
Context / Positioning



- Long history of successful physical models
 - RRI, Matsiro/CamaFlood, VIC, LISFLOOD, etc.
 - Supporting a variety of water services
- Spatially distributed
- Physically consistent and interpretable processes
- Explicitly account for human interventions
- Robust to non-stationarity of climate change
- Harder to fit to observations -> lower accuracy

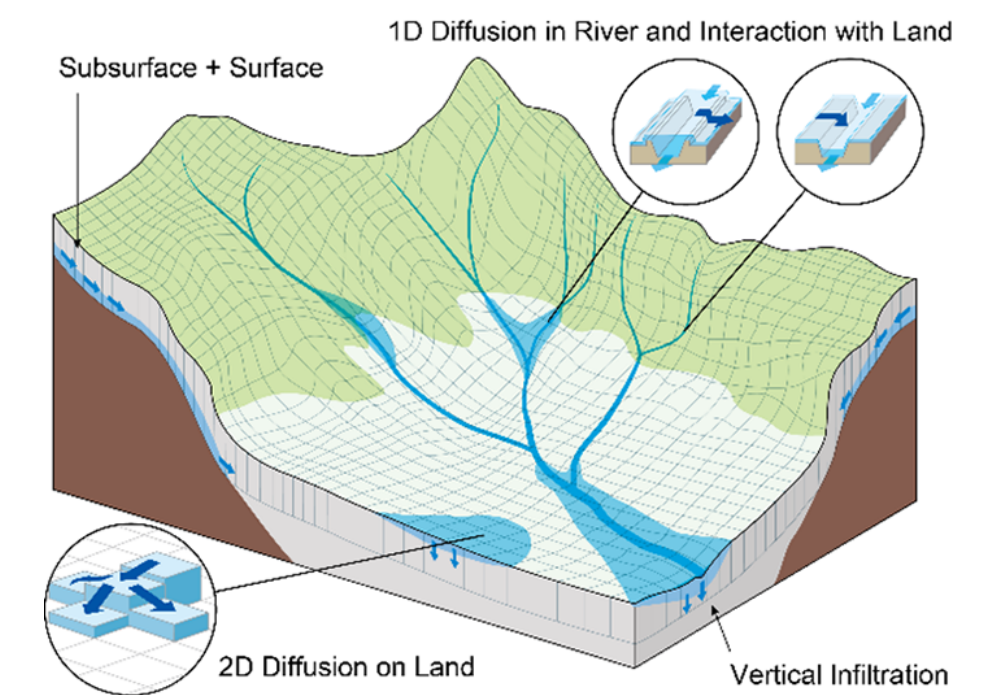


Context / Positioning



- Long history of successful physical models
 - RRI, Matsiro/CamaFlood, VIC, LISFLOOD, etc.
 - Supporting a variety of water services

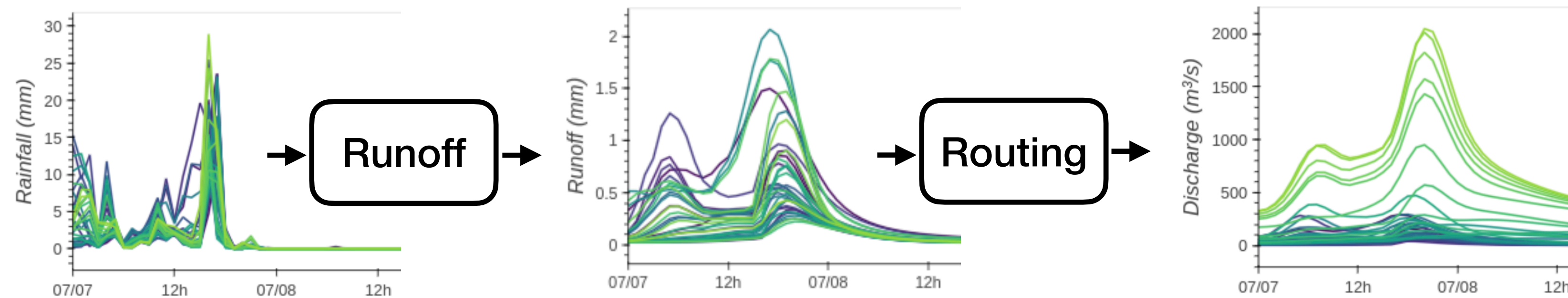
- Spatially distributed
- Physically consistent and interpretable processes
- Explicitly account for human interventions
- Robust to non-stationarity of climate change
- Harder to fit to observations -> lower accuracy



**Bridging the gap between
these two paradigms**

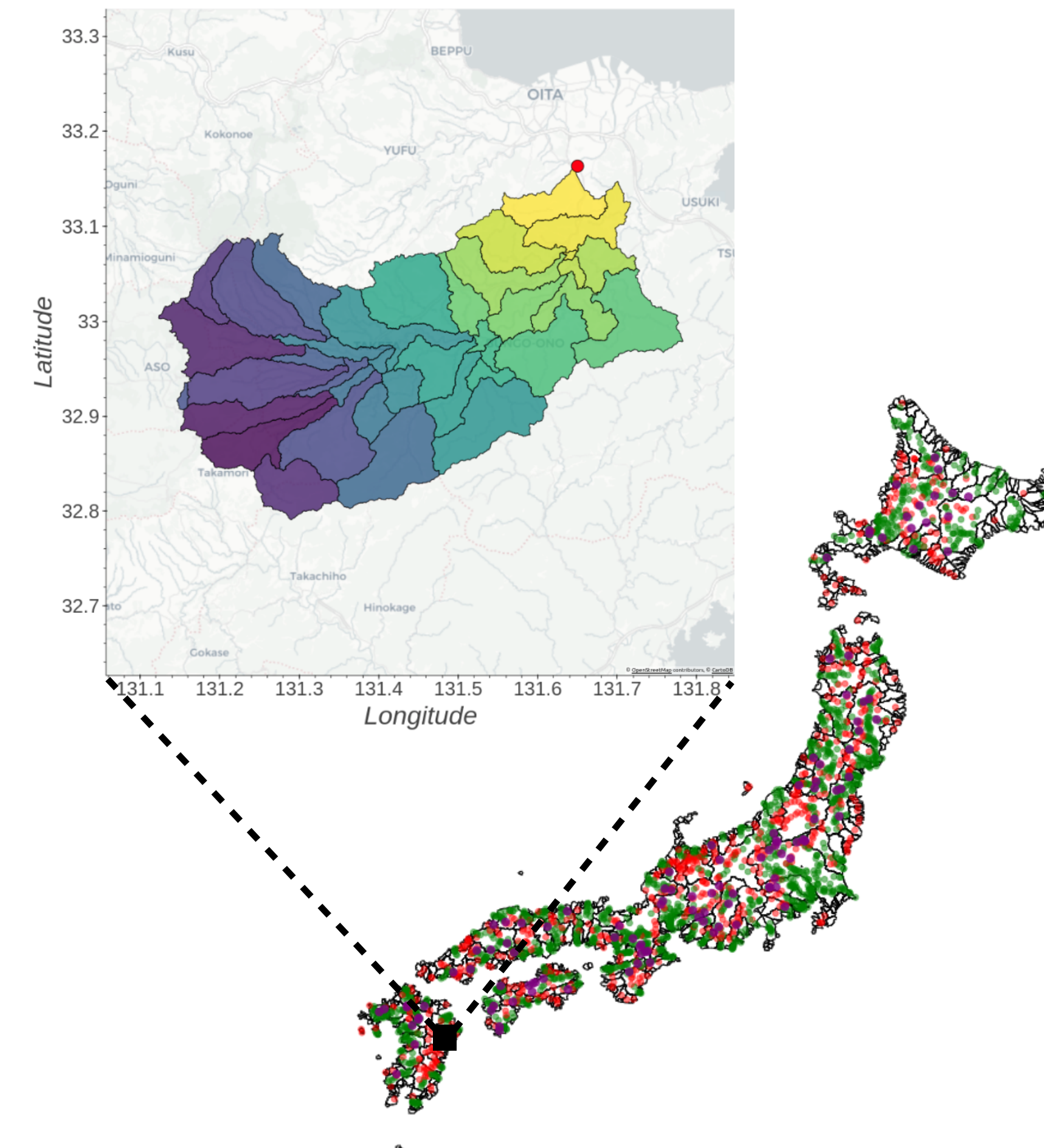
Research Question

Can **end-to-end learning** from **sparse and noisy** discharge observations resolve the **equifinality** of **spatially distributed** river discharge prediction **ML models**?



What happens when we learn (runoff + routing) ML models?

- Accurate predictions?
- Recover the “true” dynamics? (Identify consistent parameters)



Outline

- 1. Differentiable routing (DiffRoute) **enables** end-to-end learning (E2EL)
 - *“Differentiable river routing for end-to-end learning of hydrological processes”, AGU JGR Machine Learning and Computations, 2026*
- 2. E2EL produces **high accuracy** predictions with **physically consistent spatial distribution**
 - *“Leveraging Japan's streamflow records for end-to-end learning of hydrological simulations at national scale”, Water Resource Research, 2026, (preprint)*
- 3. E2EL allows to **identify physically meaningful** dynamic parameters
 - *“End-to-end learning from sparse and noisy observations mostly resolves the equifinality of distributed river-discharge models over Japan”, ECMWF workshop, in preparation*

Outline

- 1. Differentiable routing (DiffRoute) **enables** end-to-end learning (E2EL)
 - *“Differentiable river routing for end-to-end learning of hydrological processes”, AGU JGR Machine Learning and Computations, 2026*
- 2. E2EL produces **high accuracy** predictions with **physically consistent spatial distribution**
 - *“Leveraging Japan's streamflow records for end-to-end learning of hydrological simulations at national scale”, Water Resource Research, 2026, (preprint)*
- 3. E2EL allows to **identify physically meaningful** dynamic parameters
 - *“End-to-end learning from sparse and noisy observations mostly resolves the equifinality of distributed river-discharge models over Japan”, ECMWF workshop, in preparation*

DiffRoute

- Idea: Computations of **LTI Routing** models are equivalent to **Conv1D** layer in Deep Learning
 - Bridge between Deep Learning and (old) hydrological models
- Implementation: Scaling computations to large problems on GPU is hard:
 - Provide algorithmic and low-level GPU **optimization**
 - Package into an efficient and **easy-to-use library**
- Results
 - **Speed**: *Global climatic scale simulation in 10s*
 - **Generality**: *Compute exactly any linearization of physical equations*
 - **Differentiability**: *Enables stable learning across many use cases*

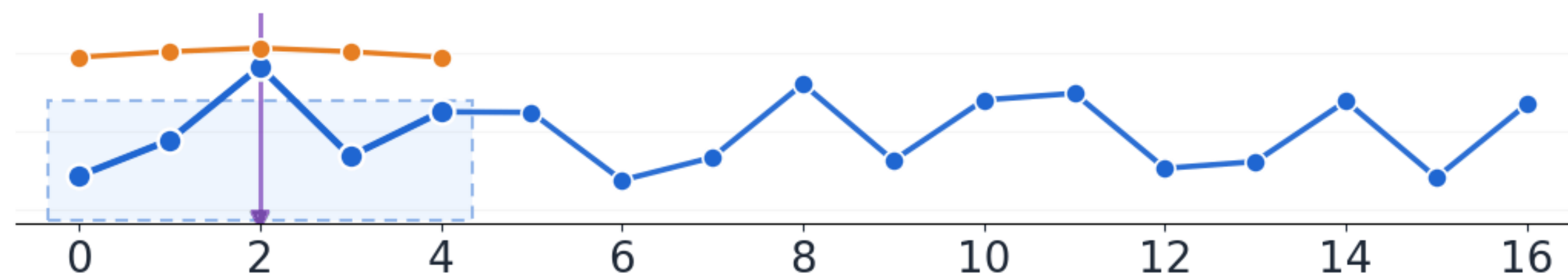
DiffRoute - convolution

$$y(t) = (x * h)(t) = \sum_{k=-\infty}^{\infty} h(k) x(t + k)$$

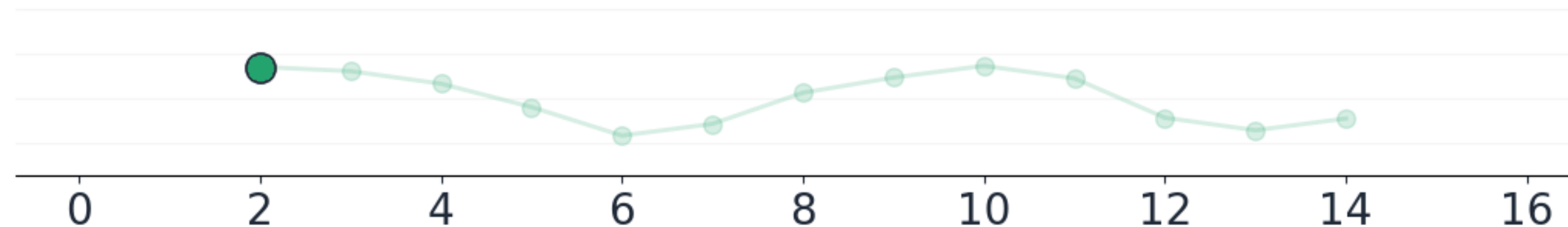
filter $h(k)$



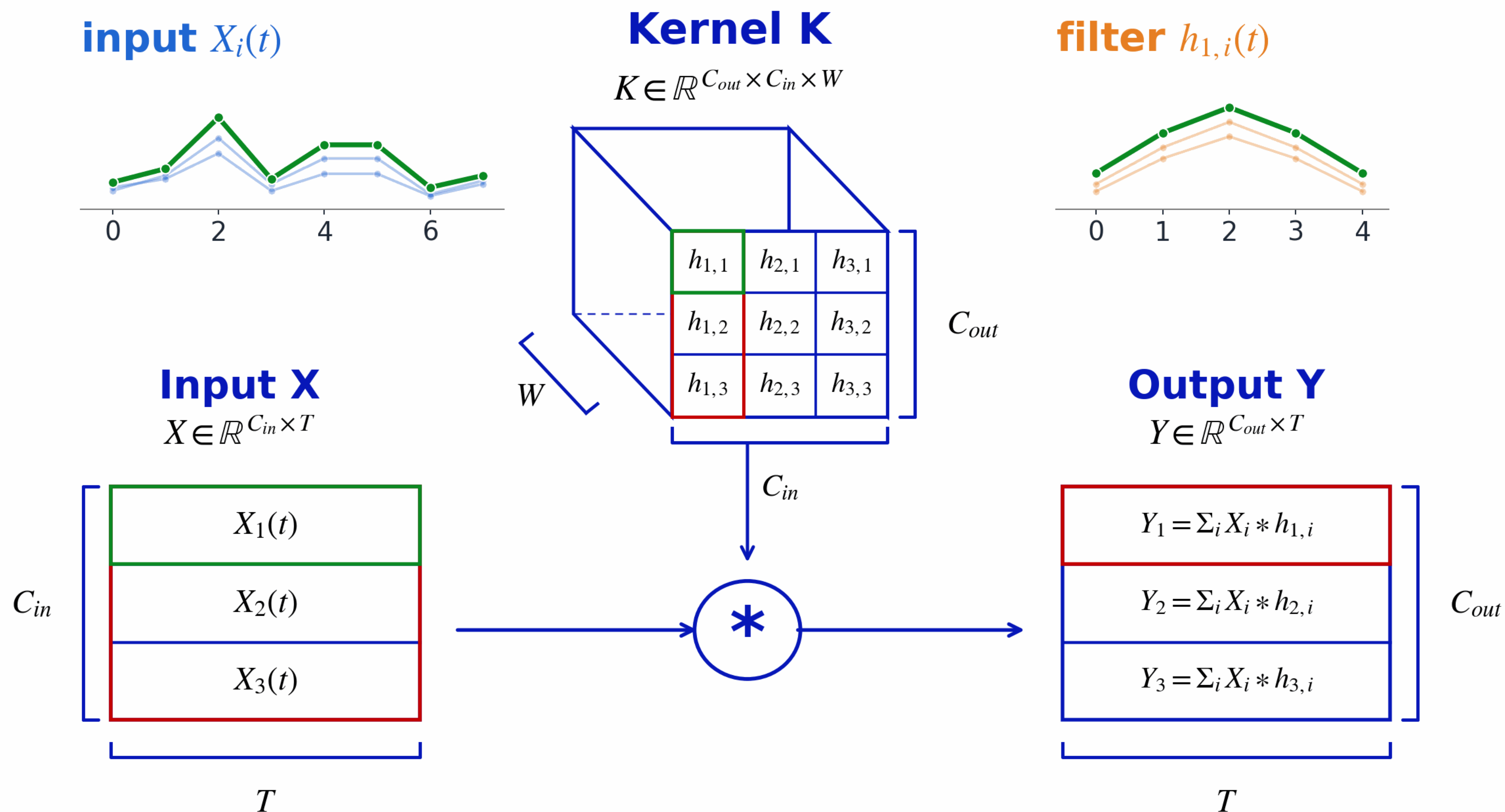
input $x(t)$



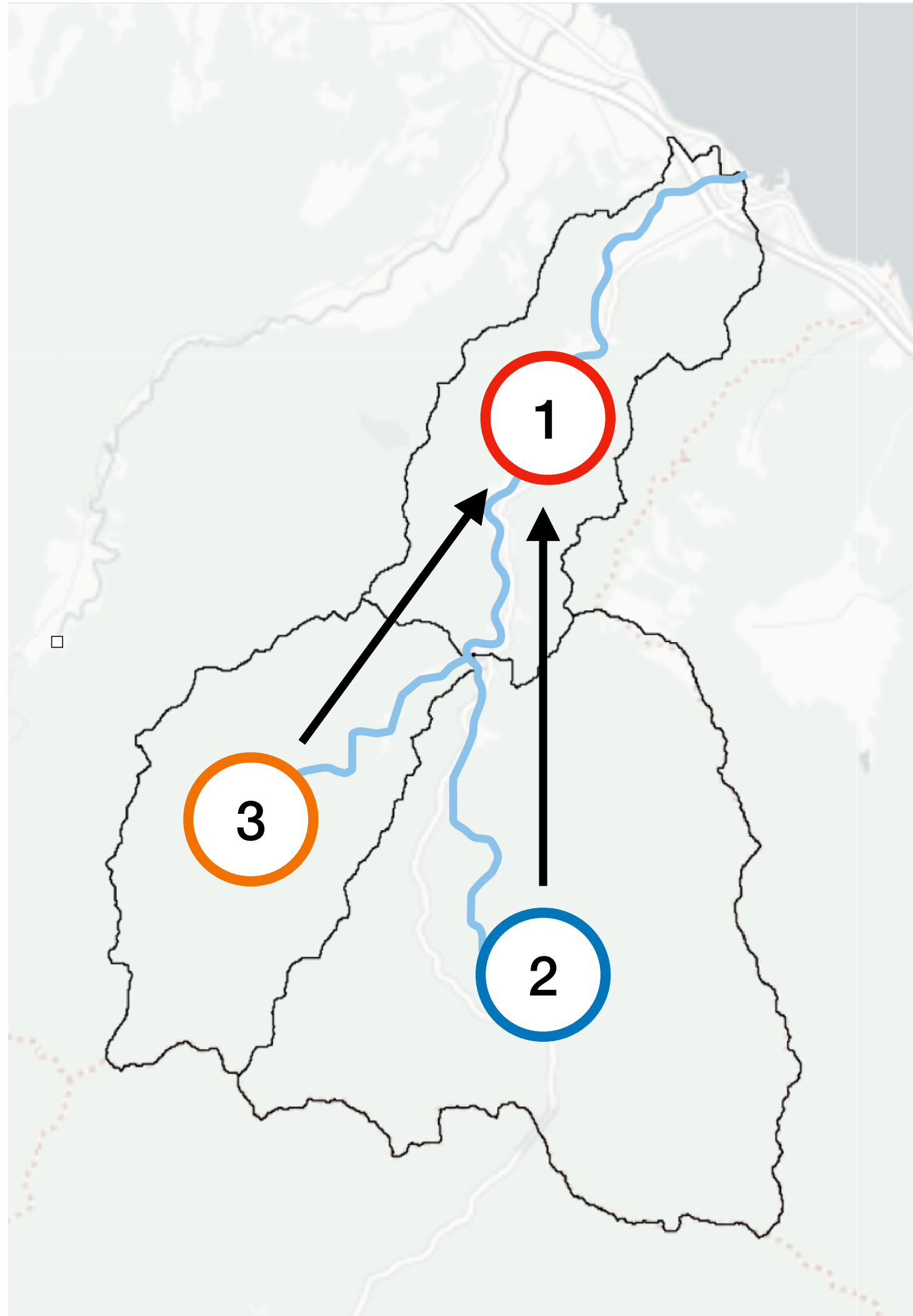
output $y(t)$



DiffRoute - Conv1d layer (Deep Learning)



DiffRoute - LTI Routing



LTI routing example: linear storage model

$$I_c(t) = \sum_{i \in up(c)} Q_i(t) + R_c(t) \quad (m^3/s)$$

↑ Total Inflow ↑ Discharge (Upstream inflows) ↑ Runoff (lateral inflows)

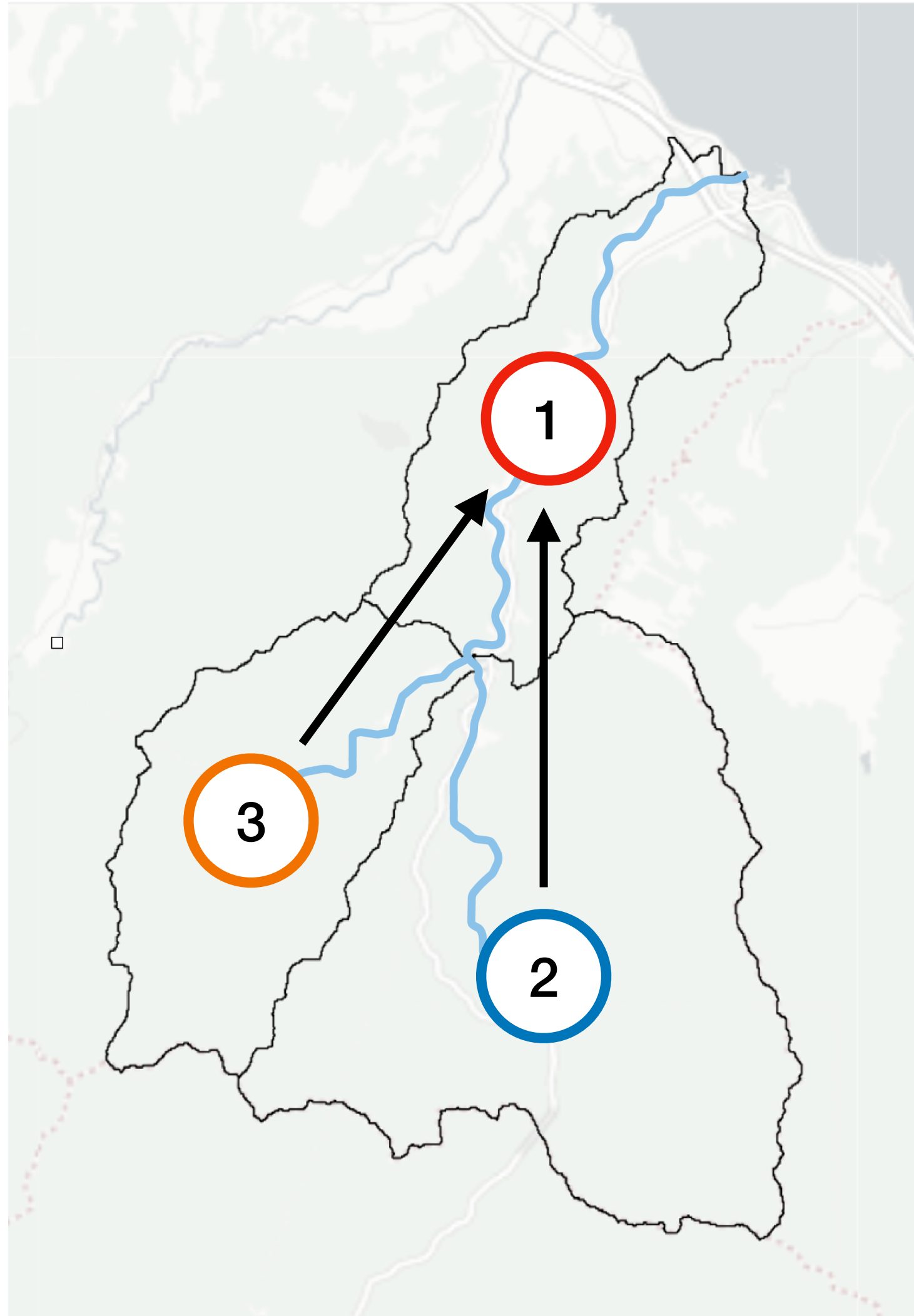
$$S_c(t) = S_c(t - 1) + \Delta t (I_c(t) - Q_c(t))$$

↑ Storage (m^3)

$$Q_c(t) = \frac{1}{\tau} S_c(t)$$

↑ Residence time (s)

DiffRoute - Demonstration



Demonstration of LTI Routing = Conv1D equivalence

- **Time:** linear recurrence = convolution

$$Q_c(t) = \frac{1}{\tau} S_c(t)$$

$$S_c(t) = S_c(t-1) + \Delta t (I_c(t) - Q_c(t))$$

Substitute **S** for **Q**

$$\tau Q_c(t) = \tau Q_c(t-1) + \Delta t (I_c(t) - Q_c(t))$$

factorize with $x = \frac{\Delta t}{\tau + \Delta t}$

$$Q_c(t) = (1-x)Q_c(t-1) + xI_c(t).$$

$$= (1-x)^2 Q_c(t-2) + x(1-x)I_c(t-1) + xI_c(t)$$

$$= (1-x)^3 Q_c(t-3) + x(1-x)^2 I_c(t-2) + x(1-x)I_c(t-1) + xI_c(t)$$

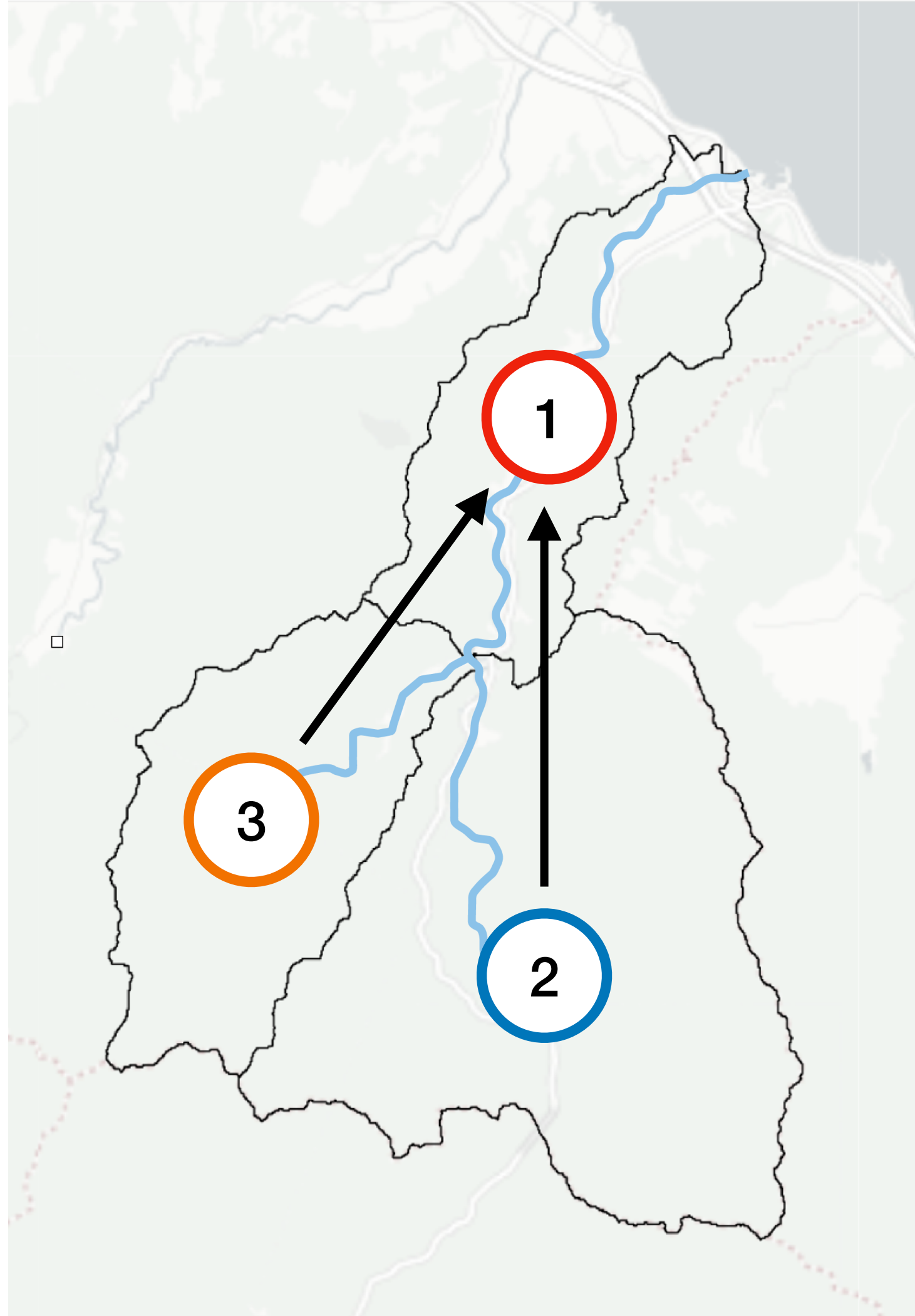
$$Q_c(t) = \sum_{k=0}^t x(1-x)^k I_c(t-k)$$

$$h_c(t) = x(1-x)^t$$

$$Q_c(t) = (I_c * h_c)(t)$$

Impulse Response Function (IRF)

DiffRoute - Idea



Demonstration of LTI Routing = Conv1D equivalence

- **Time:** linear recurrence = convolution
- **Space:** Superposition principle of LTI systems

$$Q_1(t) = \overbrace{(Q_2(t) + Q_3(t) + R_1(t))}^{I_1(t)} * h_1(t)$$

$$Q_1(t) = (I_2(t) * h_2(t) + I_3(t) * h_3(t) + R_1(t)) * h_1(t)$$

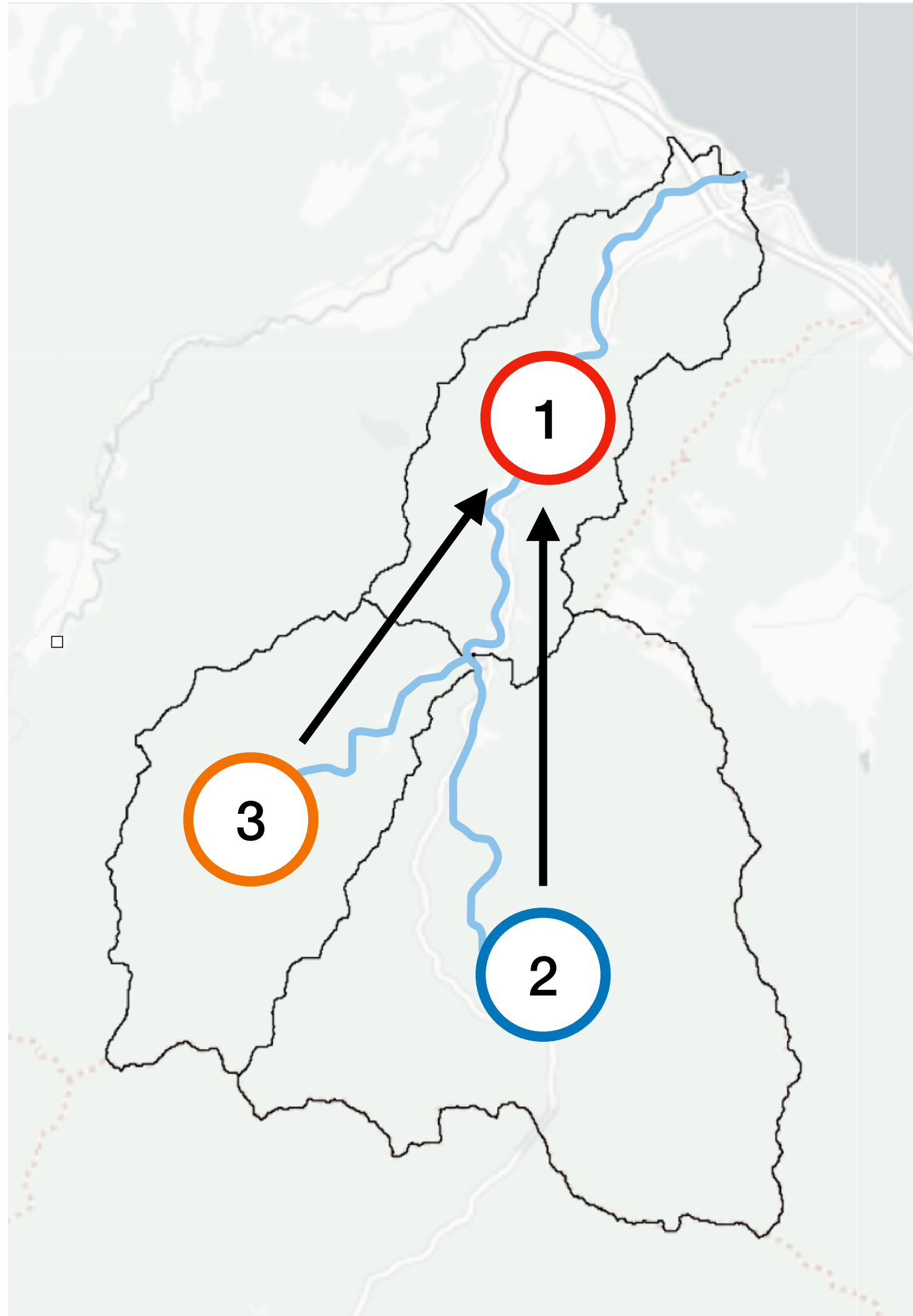
$$Q_1(t) = (R_2(t) * h_2(t) + R_3(t) * h_3(t) + R_1(t)) * h_1(t)$$

$$Q_1(t) = (R_2(t) * h_2(t) * h_1(t)) + (R_3(t) * h_3(t) * h_1(t)) + (R_1(t) * h_1(t))$$

$$Q_c(t) = \sum_{i \in \text{anc}(c)} R_i(t) * \underbrace{h_{i \rightarrow c}(t)}_{\text{Path-transverse IRF to channel 1}}$$

Path-transverse IRF to channel 1

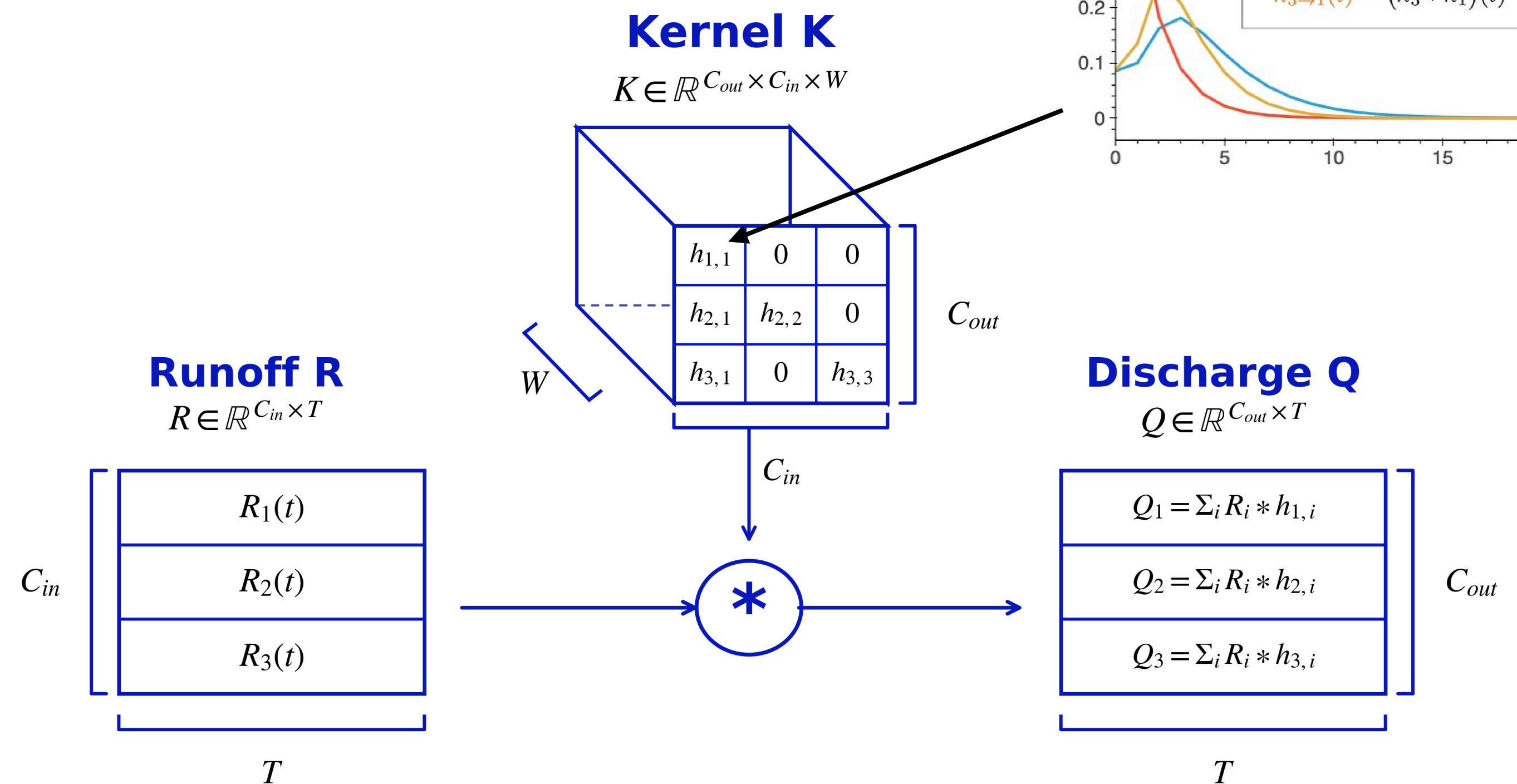
DiffRoute - Idea



Demonstration of LTI Routing = Conv1D equivalence

- **Time:** linear recurrence = convolution
- **Space:** Superposition principle of LTI systems

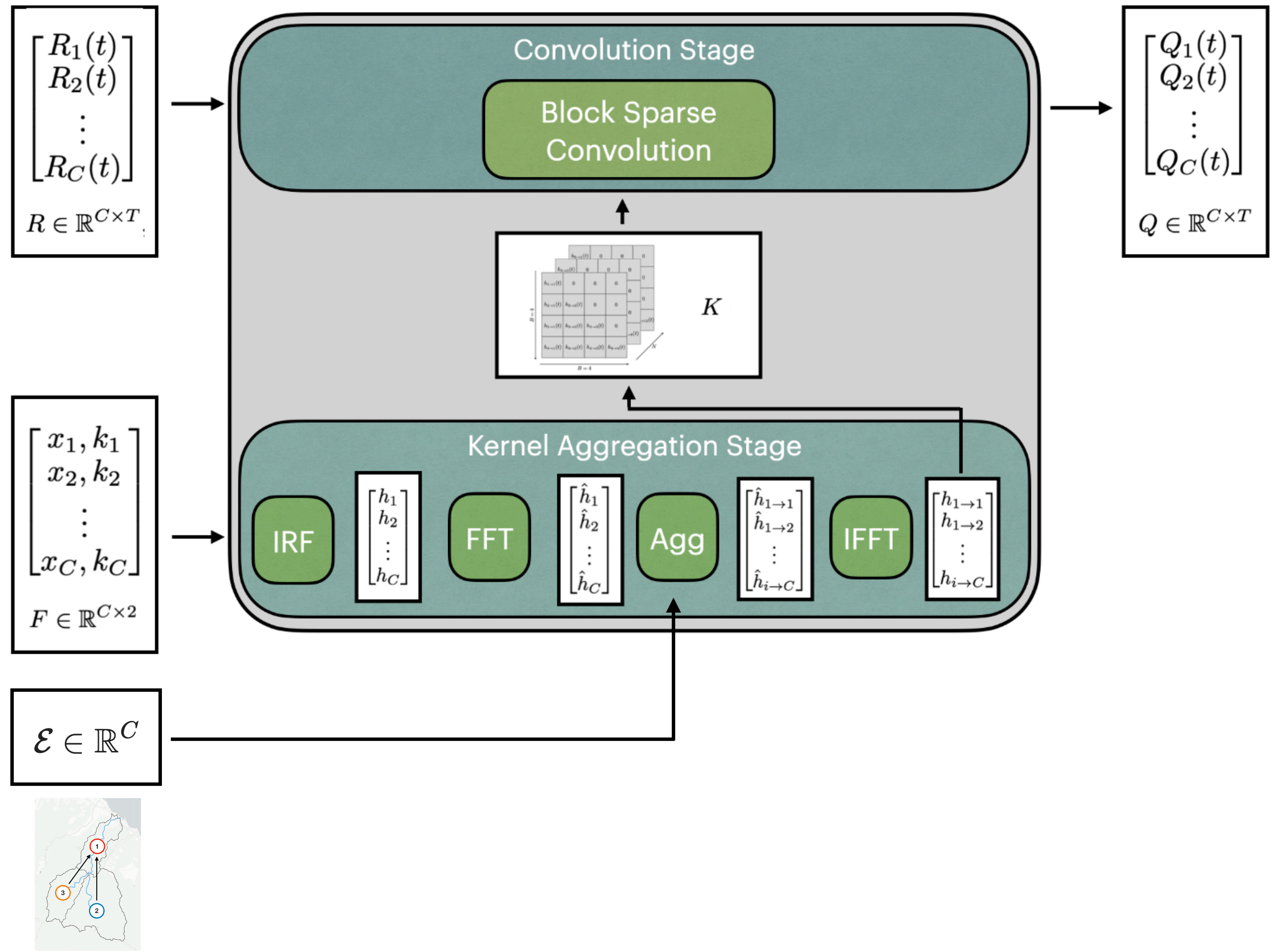
$$Q_c(t) = \sum_{i \in \text{anc}(c)} R_i(t) * h_{i \rightarrow c}(t)$$



DiffRoute - Implementation

The computations happen in 2 stages:

- Kernel aggregation stage:
 - **Convert** parameters -> IRFs
 - Aggregate path-IRF in **log space**
 - **K** describes the full dynamic
- Routing convolution stage:
 - $Q = \text{conv1d}(R, K)$



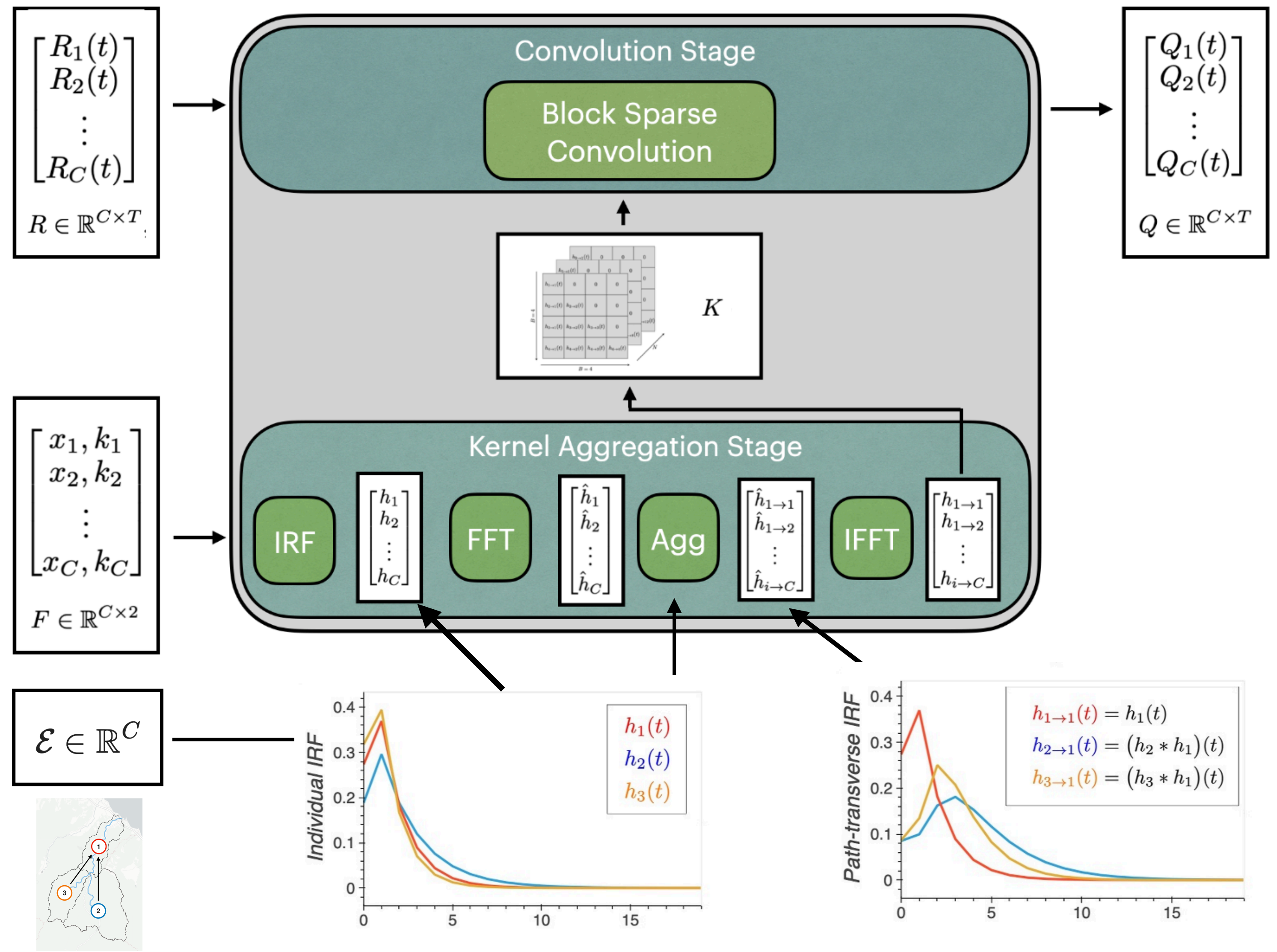
DiffRoute - Implementation

The computations happen in 2 stages:

- Kernel aggregation stage:
 - **Convert** parameters -> IRFs
 - Aggregate path-IRF in **log space**
 - **K** describes the full dynamic
- Routing convolution stage:
 - $Q = \text{conv1d}(R, K)$

Result:

- **Generality**: Only IRF formula change
- **Speed**: Extremely parallel for GPUs
- **Stability**: log-space aggregation



DiffRoute - Implementation

Naive implementation does not scale

Compute complexity: $C^2 \times W \times T$

Memory complexity: $C^2 \times W$

Global climatic simulations (Naive)

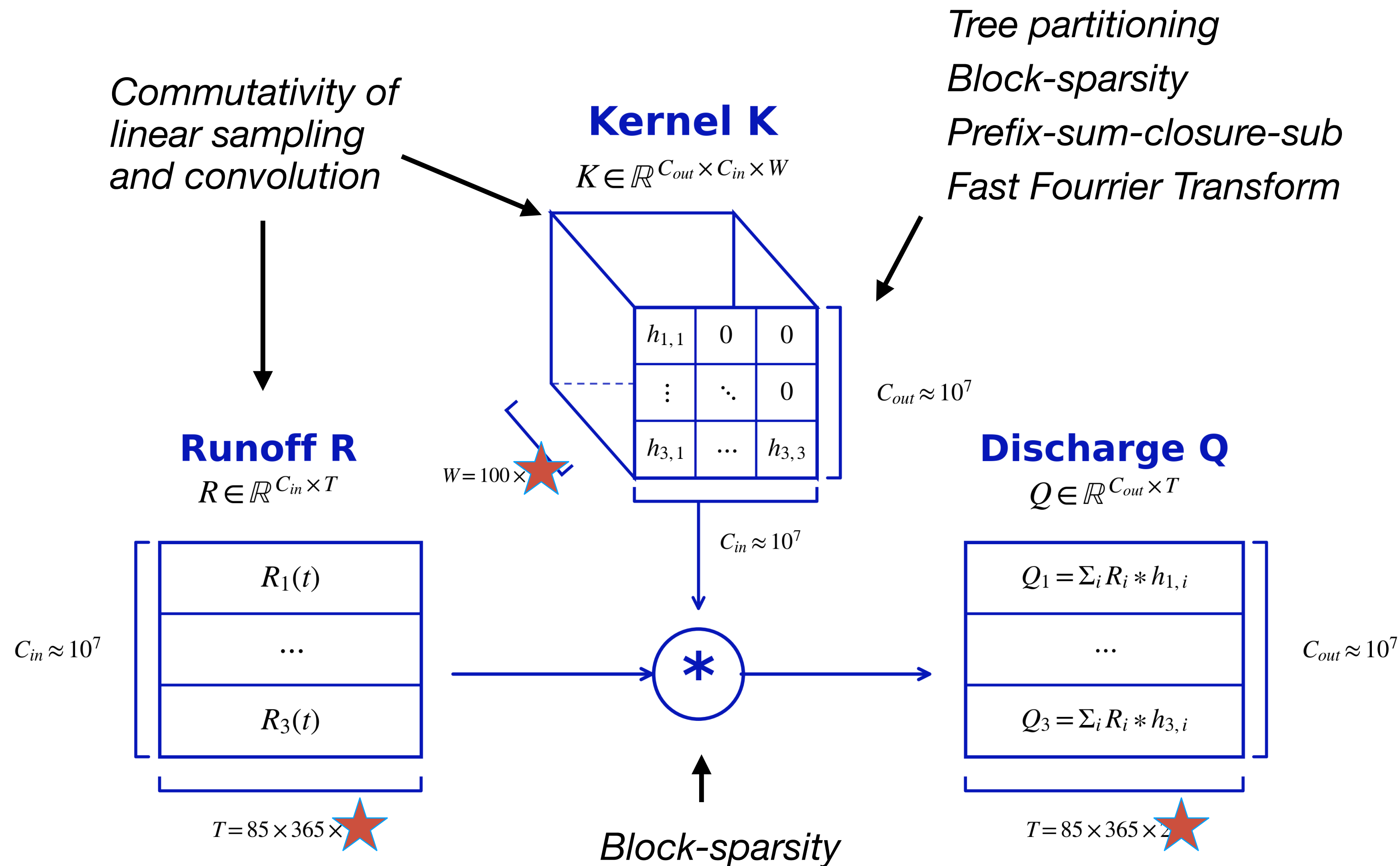
Compute: ~**10⁶ PetaFlop**

Memory: ~**1PetaBytes**

Global climatic simulations (DiffRoute)

Compute < **1 PetaFlop**

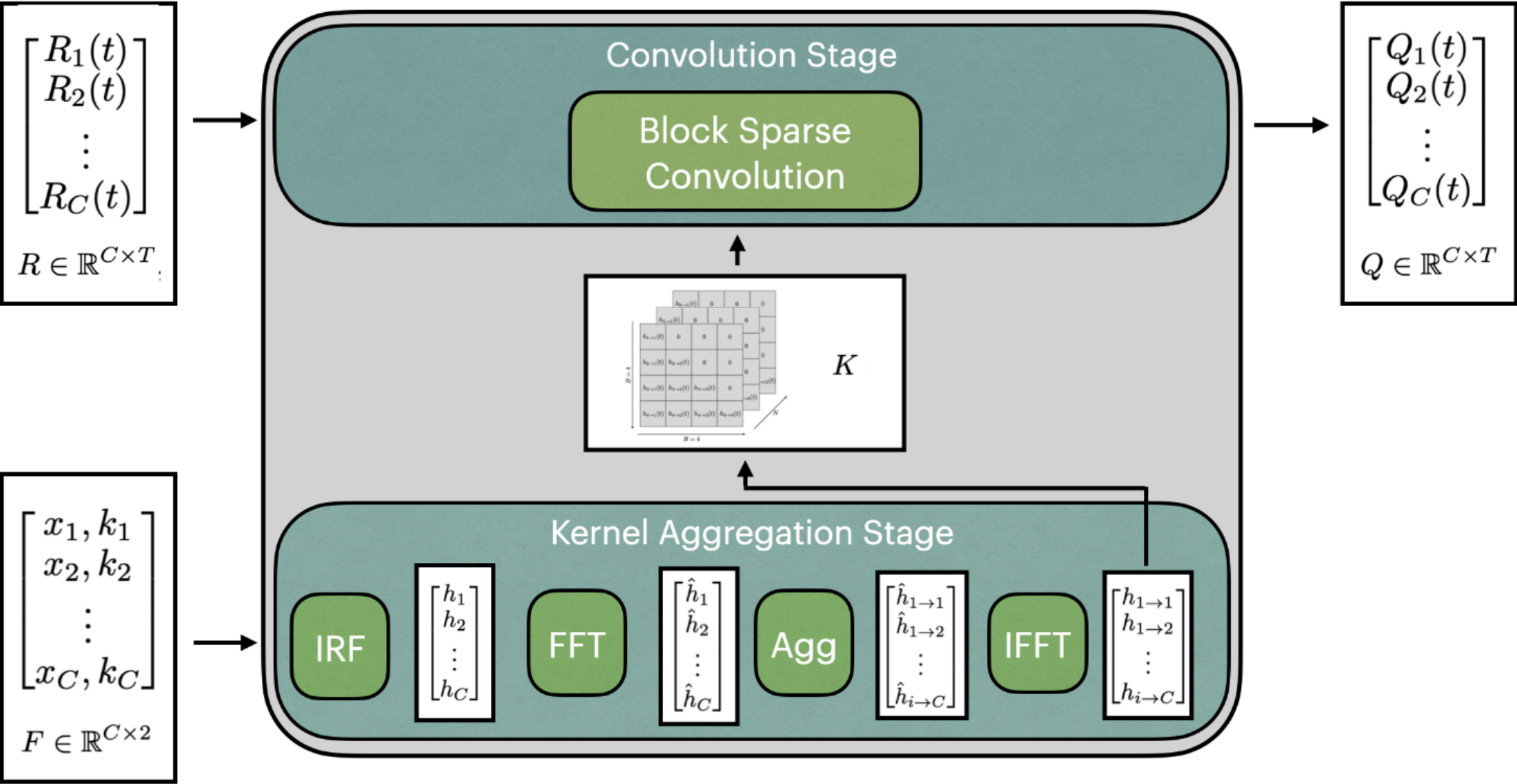
Memory < **20GB**



DiffRoute - Implementation

Generality: This approach allows to scale any LTI river routing equation to arbitrary large problems

Model	Parameter F	IRF Formula h
Pure Lag	Δt : lag (time steps)	$h(t) = \max\{1 - t - \Delta t , 0\}$
Linear Reservoir	τ : residence time dt : model time step	$h(t) = x(1 - x)^t$, with $x = \frac{1}{1 + \frac{\tau}{dt}}$
Nash Cascade	τ : residence time n : number of reservoirs dt : model time step	$h(t) = \binom{t+n-1}{t} x^n (1 - x)^t$ with $x = \frac{1}{1 + \frac{\tau}{dt}}$.
Muskingum	x : weighting coefficient k : storage time constant dt : model time step	$h(t) = \begin{cases} C_0, & t = 0, \\ (C_1 + C_2 C_0) C_2^{t-1}, & t \geq 1, \end{cases}$ with $C_0 = \frac{\frac{dt}{2} - kx}{D_c}$, $C_1 = \frac{\frac{dt}{2} + kx}{D_c}$, $C_2 = \frac{k(1-x) - \frac{dt}{2}}{D_c}$, and $D_c = k(1-x) + \frac{dt}{2}$.
Linear Diffusive Wave (Hayami)	L : channel length D : hydraulic diffusivity c : wave celerity	$h(t) = \frac{L}{2\sqrt{4\pi D t^3}} \exp\left(-\frac{(L - ct)^2}{4Dt}\right)$



DiffRoute - Implementation

Generality: This approach allows to scale any LTI river routing equation to arbitrary large problems

Ease of use: We make it trivial to implement the linearization of any routing equation

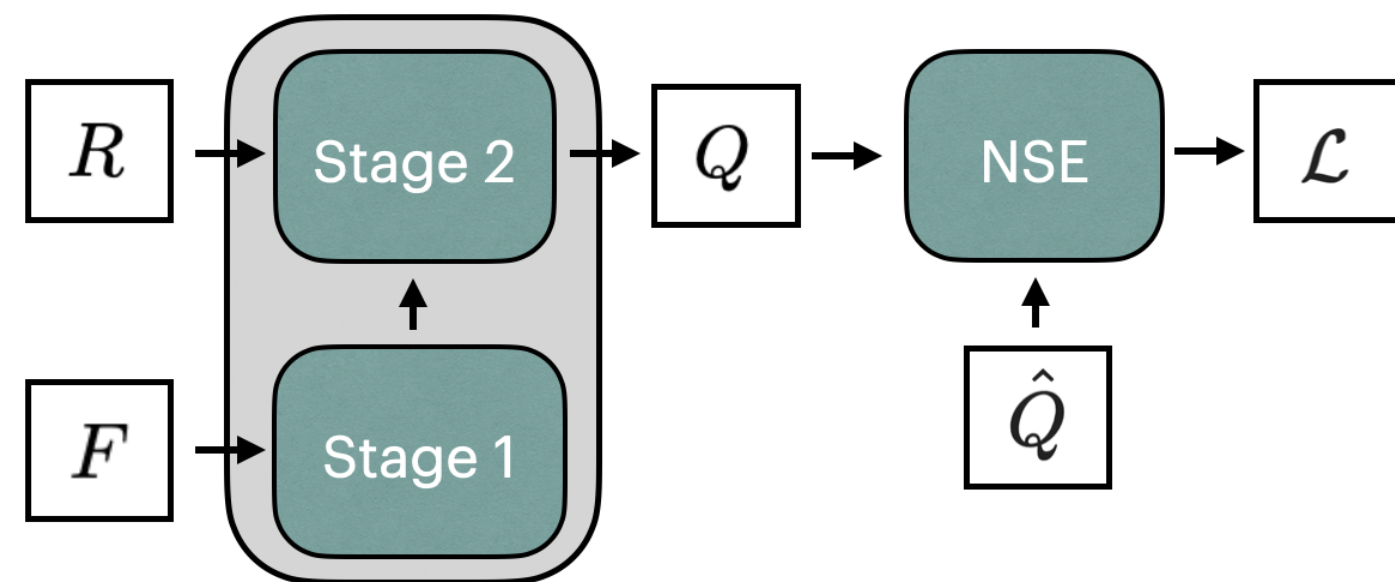
Model	Parameter F	IRF Formula h
Pure Lag	Δt : lag (time steps)	$h(t) = \max\{1 - t - \Delta t , 0\}$
Linear Reservoir	τ : residence time dt : model time step	$h(t) = x(1 - x)^t$, with $x = \frac{1}{1 + \frac{\tau}{dt}}$
Nash Cascade	τ : residence time n : number of reservoirs dt : model time step	$h(t) = \binom{t+n-1}{t} x^n (1 - x)^t$ with $x = \frac{1}{1 + \frac{\tau}{dt}}$.
Muskingum	x : weighting coefficient k : storage time constant dt : model time step	$h(t) = \begin{cases} C_0, & t = 0, \\ (C_1 + C_2 C_0) C_2^{t-1}, & t \geq 1, \end{cases}$ with $C_0 = \frac{\frac{dt}{2} - kx}{D_c}$, $C_1 = \frac{\frac{dt}{2} + kx}{D_c}$, $C_2 = \frac{k(1 - x) - \frac{dt}{2}}{D_c}$, and $D_c = k(1 - x) + \frac{dt}{2}$.
Linear Diffusive Wave (Hayami)	L : channel length D : hydraulic diffusivity c : wave celerity	$h(t) = \frac{L}{2\sqrt{4\pi Dt^3}} \exp\left(-\frac{(L - ct)^2}{4Dt}\right)$

```
def linear_reservoir_irf(param, W, dt):
    taus = param
    t = torch.arange(W)
    x = 1 / (1 + taus[:,None] / dt)
    h = x * (1 - x)**t
    return h

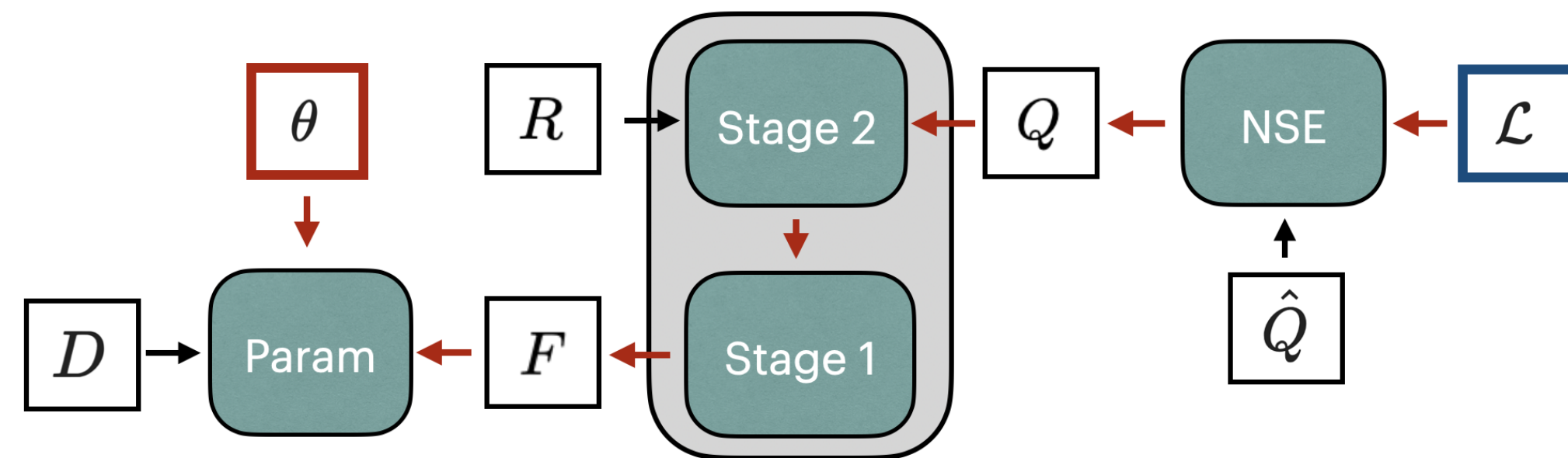
def muskingum_irf(param, W, dt):
    x,k= param
    t = torch.arange(W-1)
    D = k * (1 - x) + dt / 2
    C0 = (dt / 2 - k * x) / D
    C1 = (dt / 2 + k * x) / D
    C2 = (k * (1 - x) - dt / 2) / D
    h = ((C1 + C2*C0)[:,None] \
         * (C2[:,None]**t ))
    h = torch.cat([C0[:,None], h], dim=1)
    return h
```

DiffRoute - Implementation

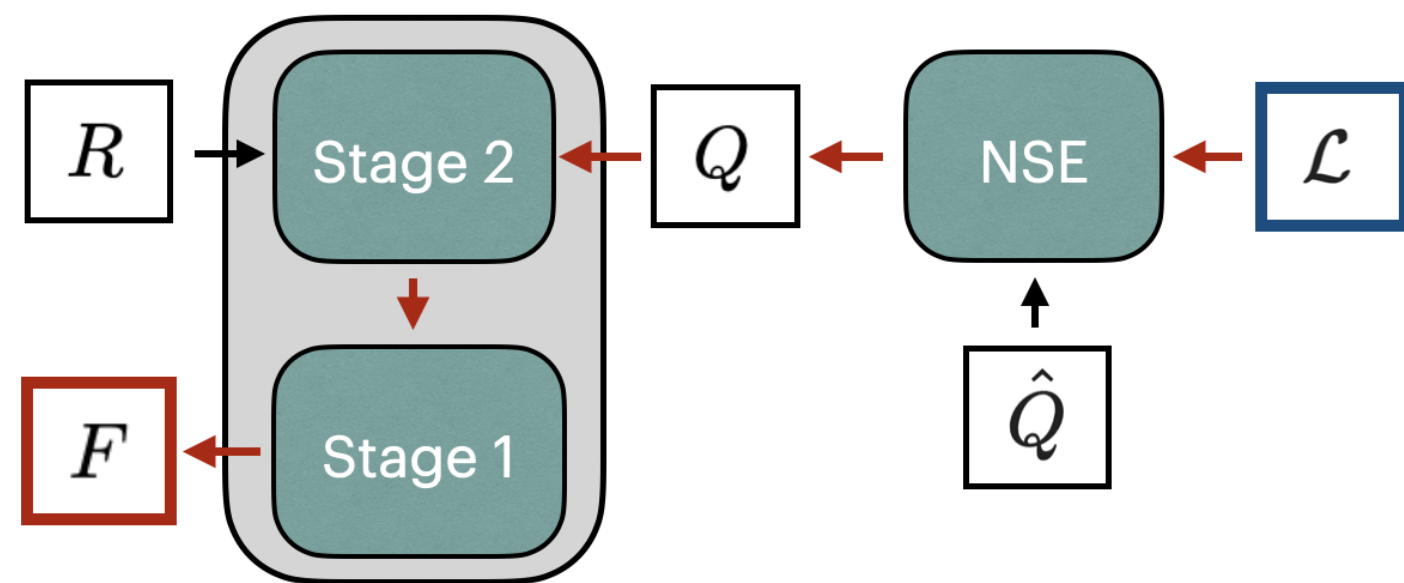
Differentiability: DiffRoute shows *stable* and *efficient* differentiation



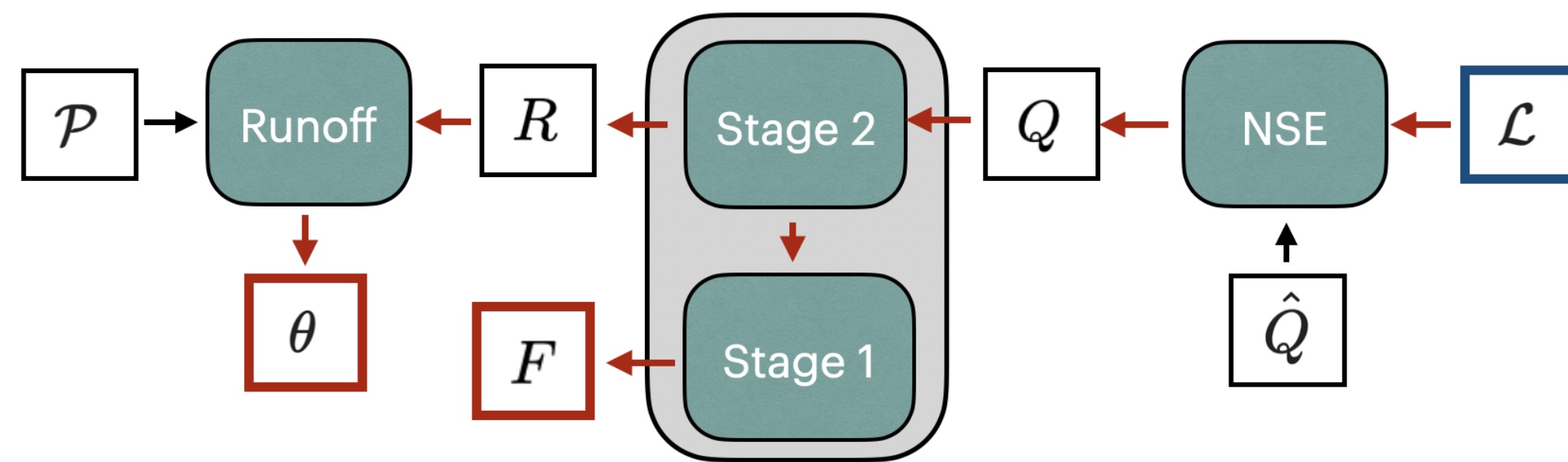
(a) Forward simulation



(b) Learning routing-physics relations



(c) Calibration of routing parameters

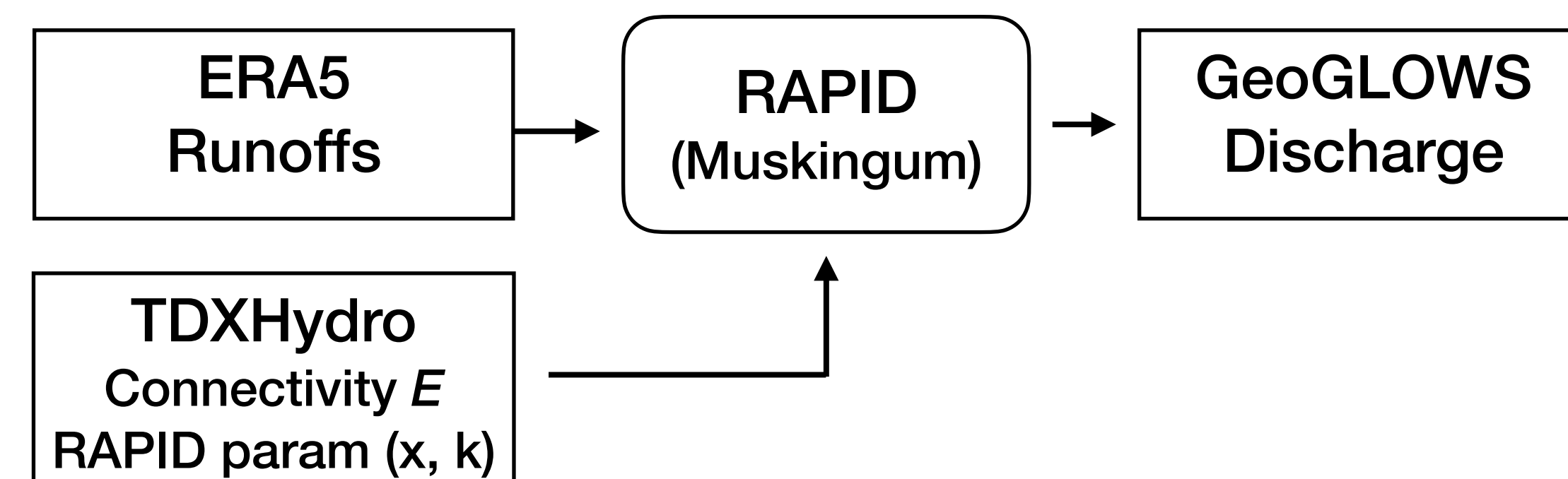
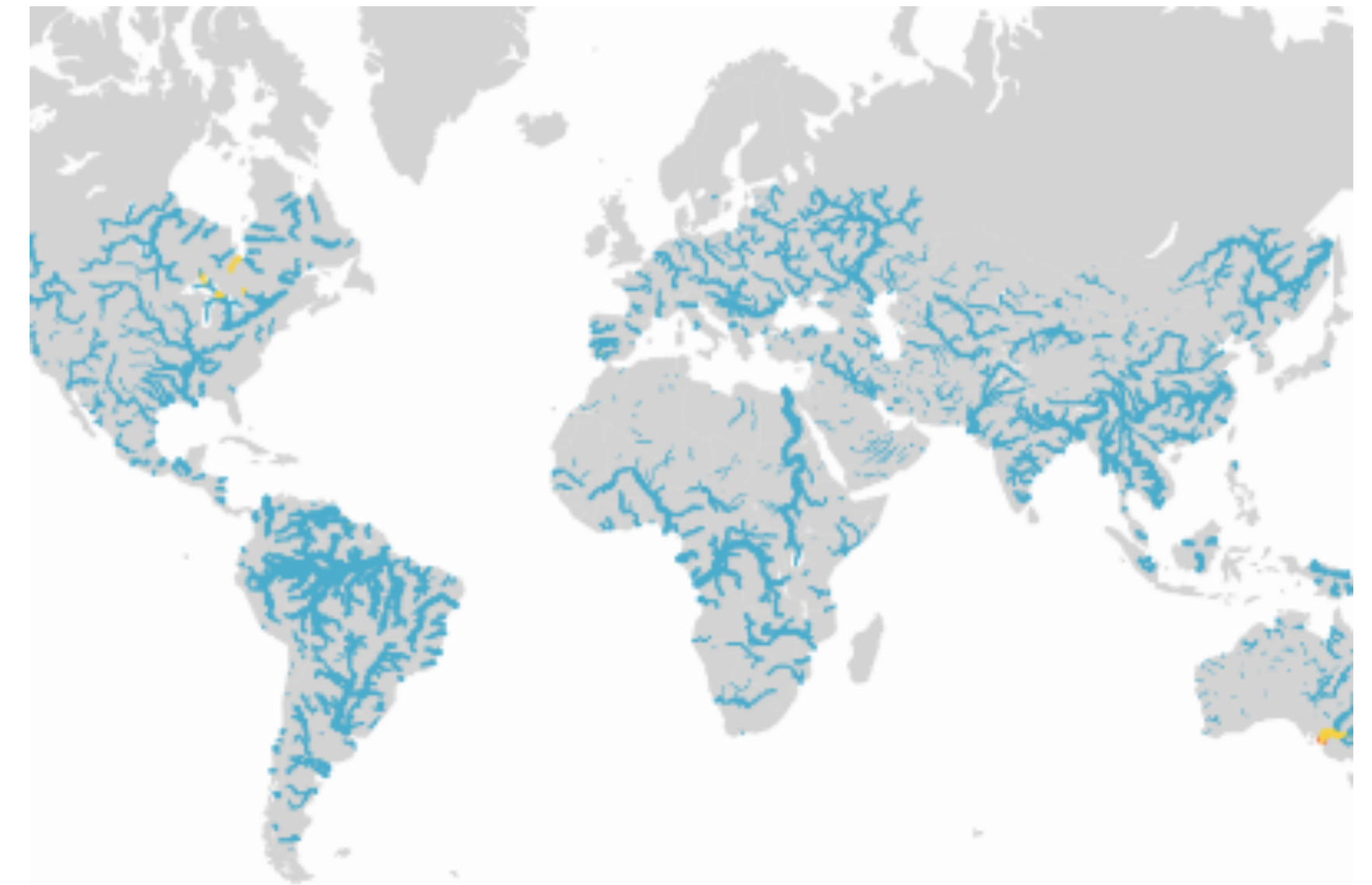


(d) End-to-end learning of runoff and routing

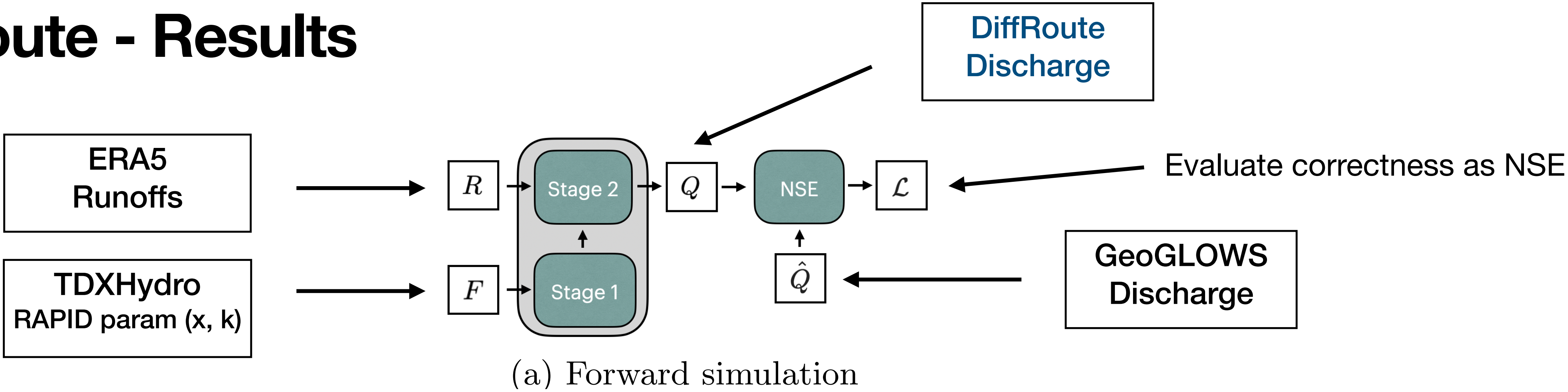
DiffRoute - Results

Global climatic simulation: GeoGLOWS V2 simulation

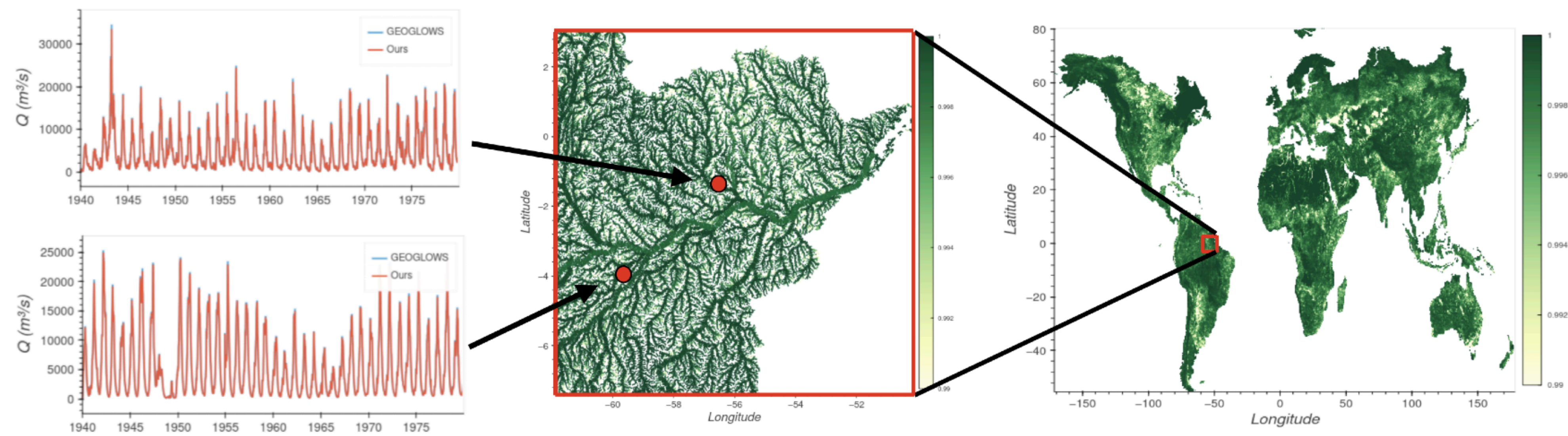
- Service of the **GEO** (Group on Earth Observations)
- River network: **7 million river channels** (TDX-Hydro 12m DEM)
- Channel routing: **RAPID model** (Muskingum, **hourly** routing)
- Forecast: using ECMWF IFS
- Retrospective simulation: Route **85 years of daily** ERA5 runoffs from 1 January 1940 to present, updated weekly
- Simulation is **fully open** (output and parameters)
- **Heavy infrastructure**: Run at ECMWF supercomputing facility in Bologna. Forecast and Analysis data served through AWS



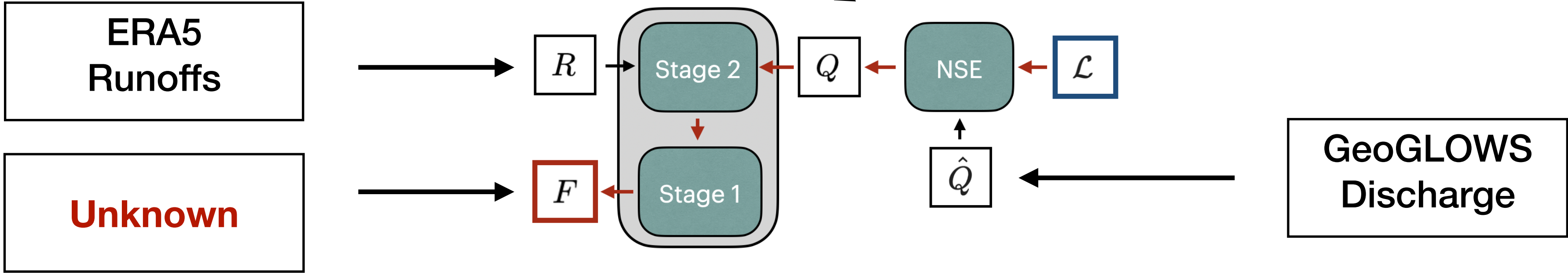
DiffRoute - Results



We reproduce GeoGLOWs V2 **global climatic scale** simulation within daily NSE of .9994 (85 years of runoff routed at hourly resolution through ~7M river channels) **in 10s on a single GPU**



DiffRoute - Results



(c) Calibration of routing parameters



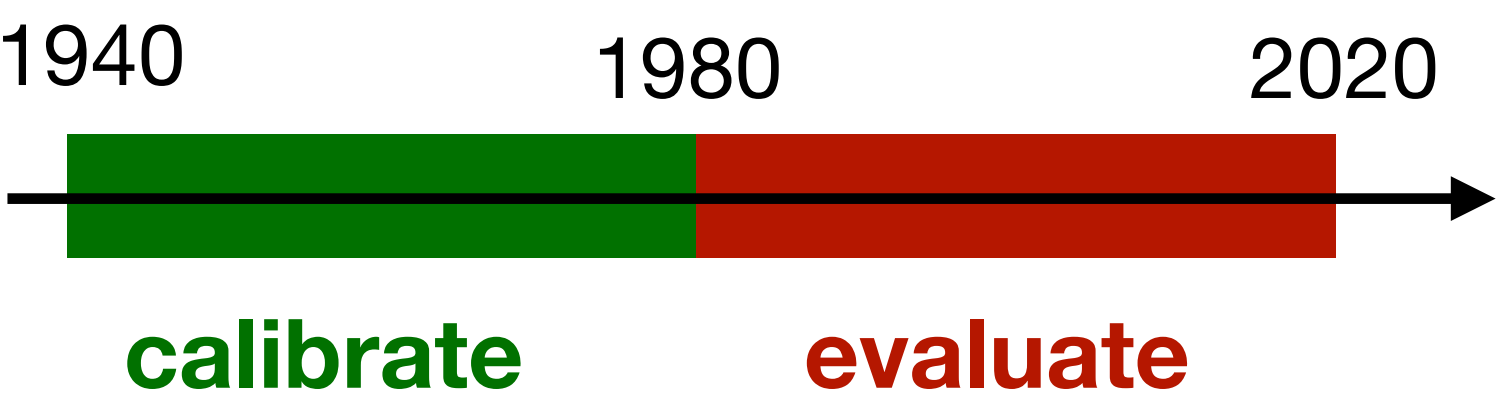
Calibrate parameters F by minimizing the NSE with gradient descents

$$F^* = \min_F \mathcal{L}(Q, \hat{Q}) \qquad F_{k+1} = F_k + \mu \frac{\delta \mathcal{L}}{\delta F_k}$$

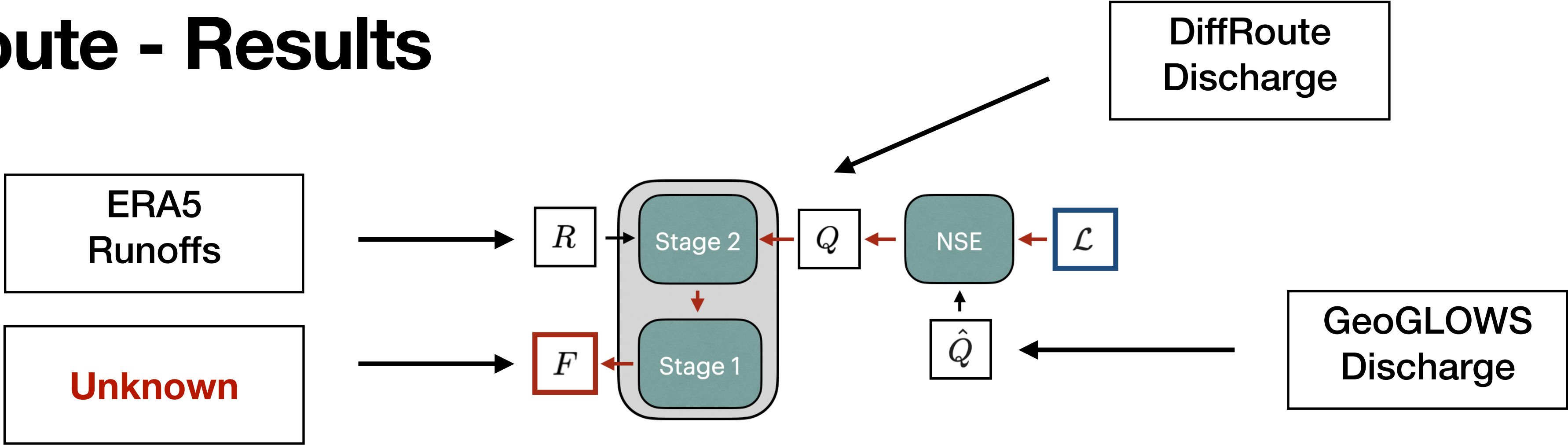
Gradients are computed by automatic differentiation

$$\frac{\delta \mathcal{L}}{\delta F} = \underbrace{\frac{\delta \mathcal{L}}{\delta Q}}_{\text{Output gradients}} \times \underbrace{\frac{\delta Q}{\delta K}}_{\text{Stage ii}} \times \underbrace{\frac{\delta K}{\delta F}}_{\text{Stage i}}$$

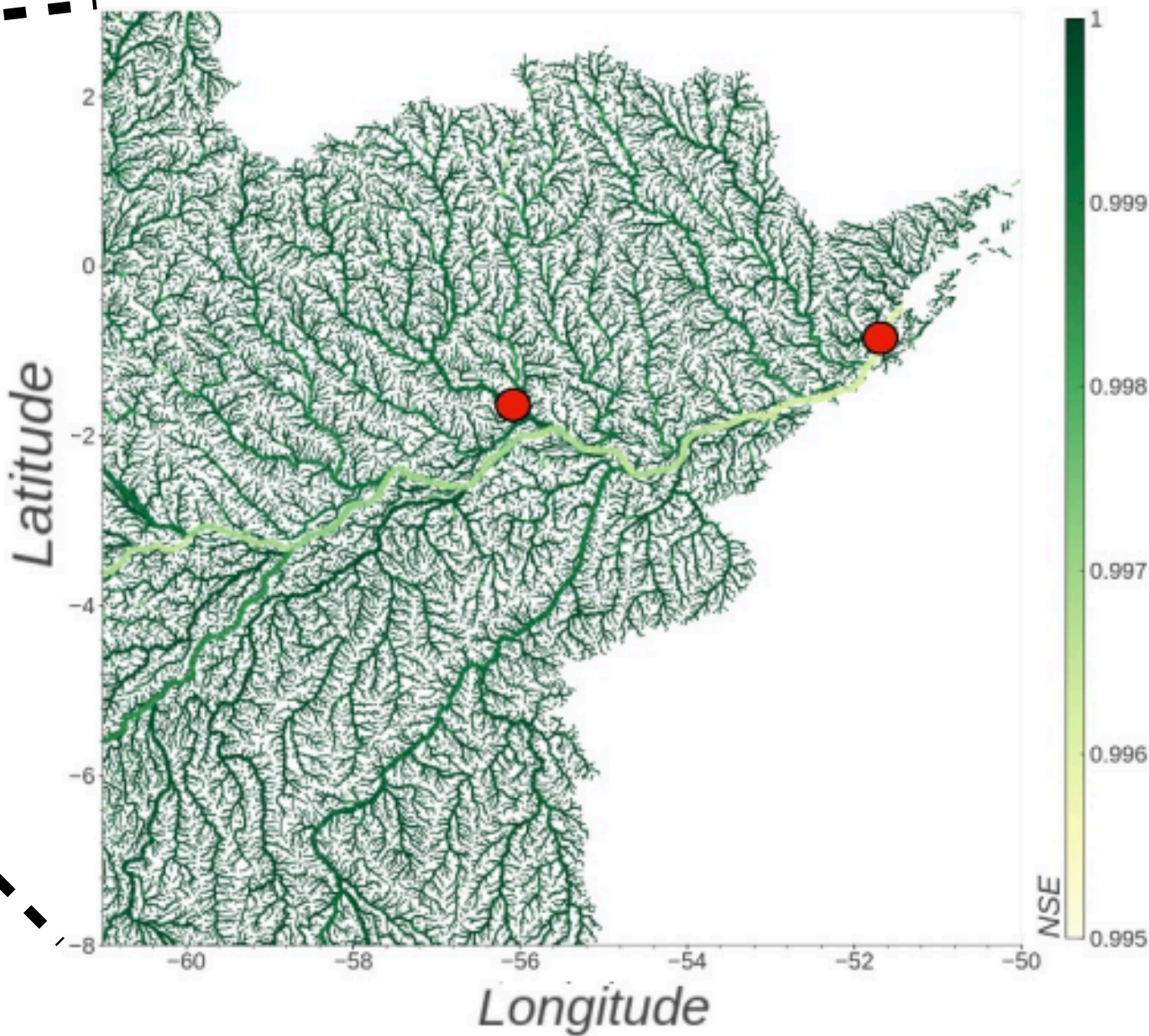
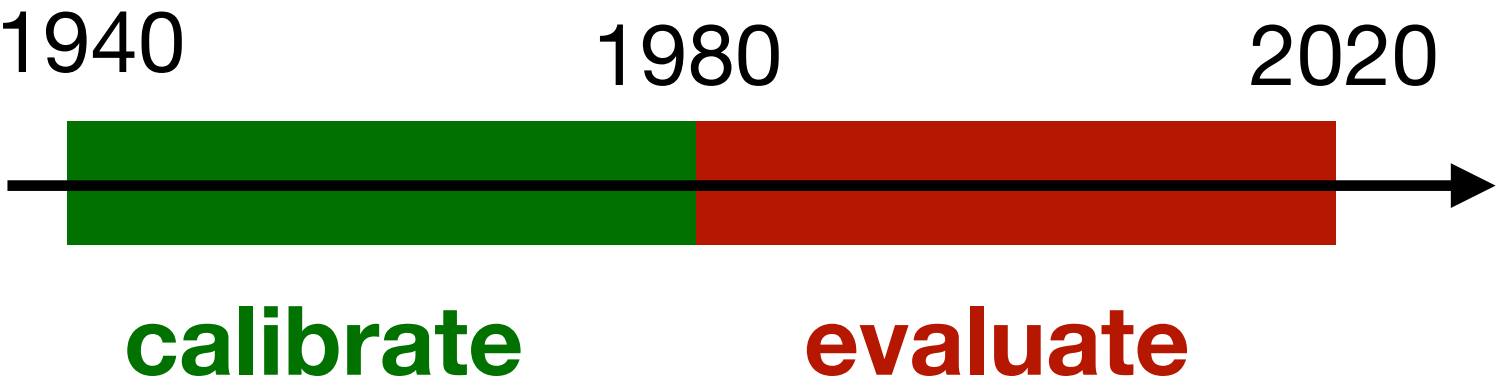
Calibrate on the past and evaluate on the future



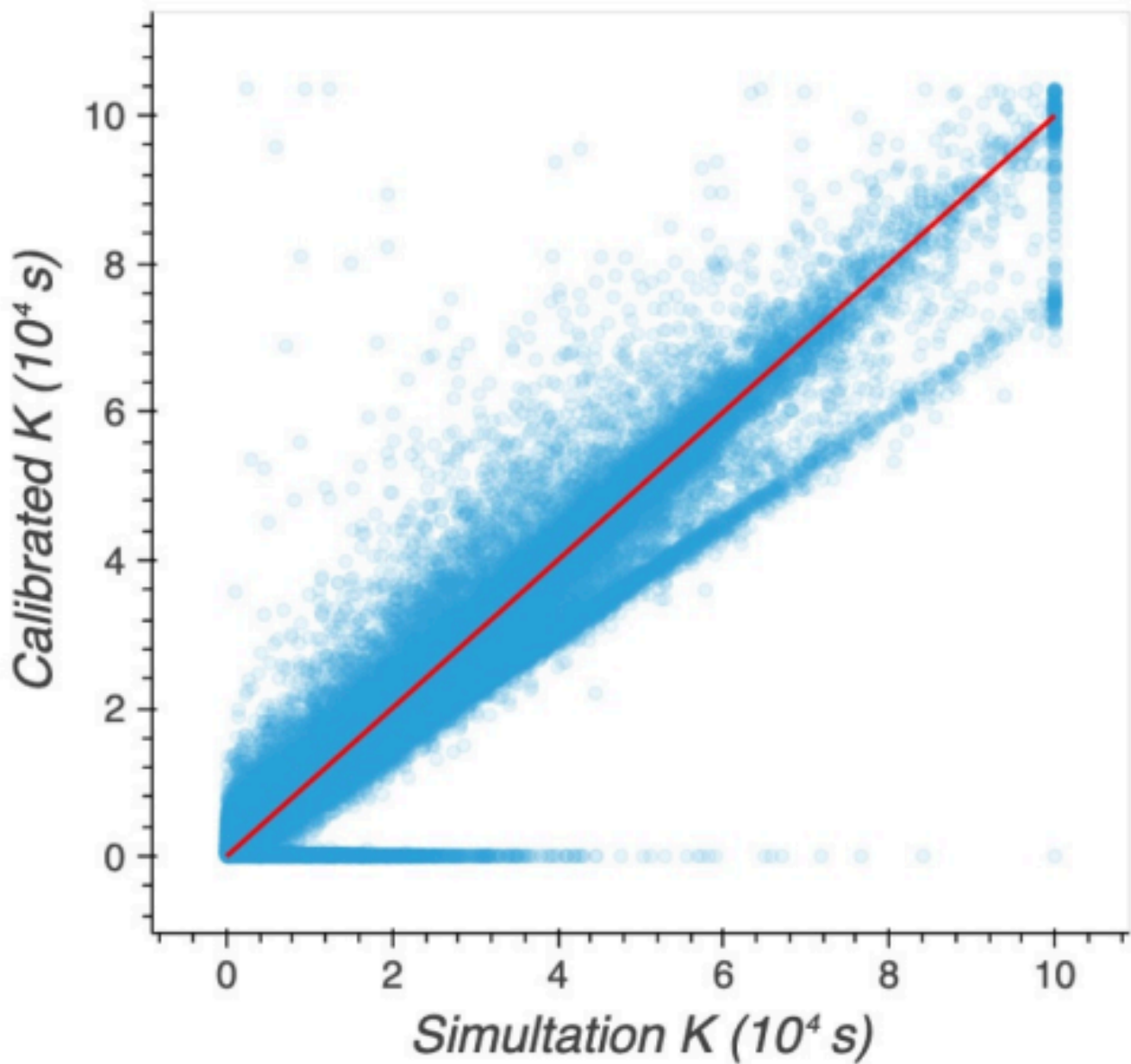
DiffRoute - Results



(c) Calibration of routing parameters

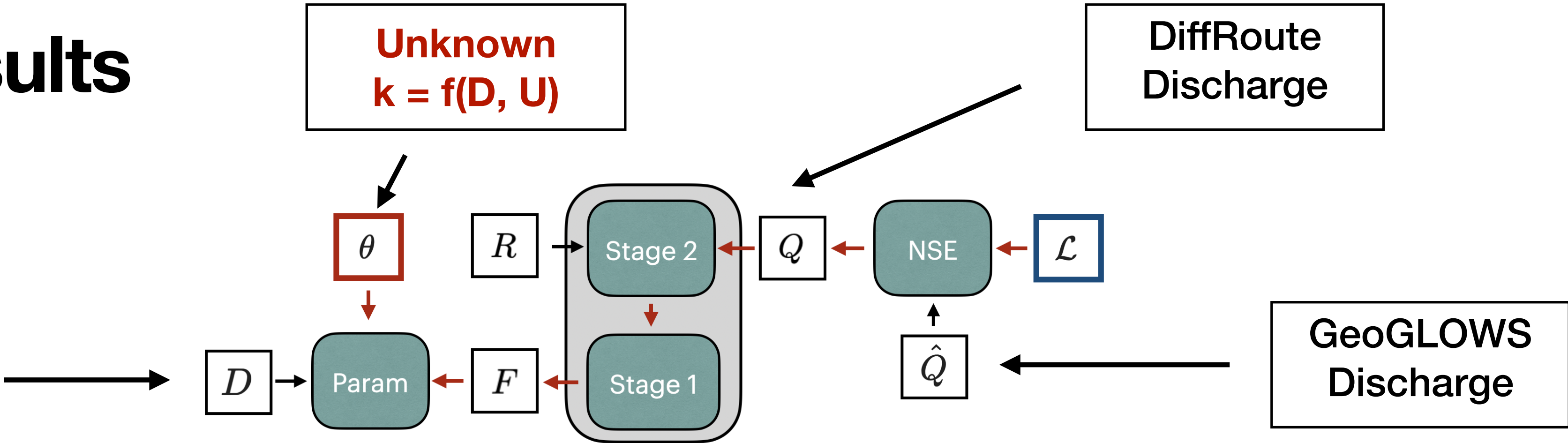
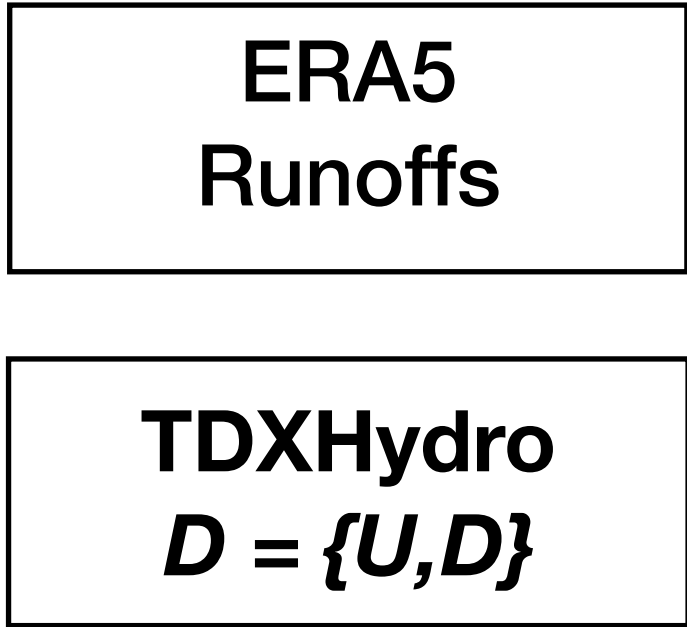


(a) NSE Map over the Amazon

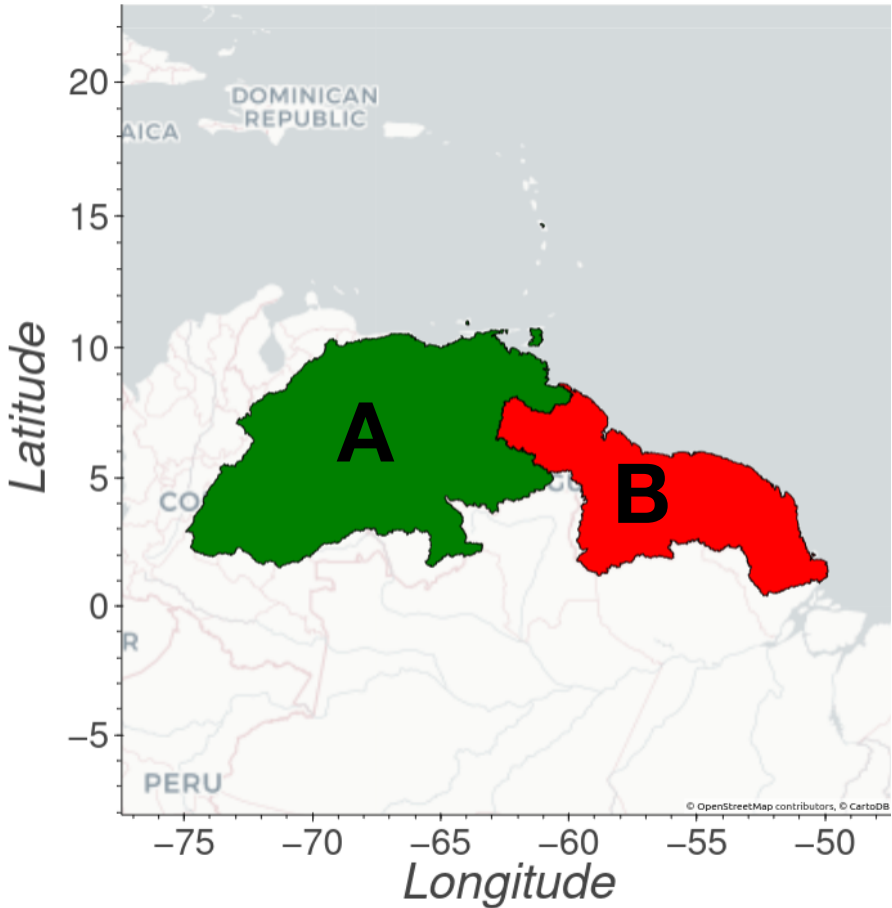


(b) K Parameter scatter plot

DiffRoute - Results



(b) Learning routing–physics relations



Geoglows parameterizes k as a function of channel D and U

$$k = \frac{D}{\min(10^{-3}, e^{(0.16842 \times \ln(U - 4.68))})}$$

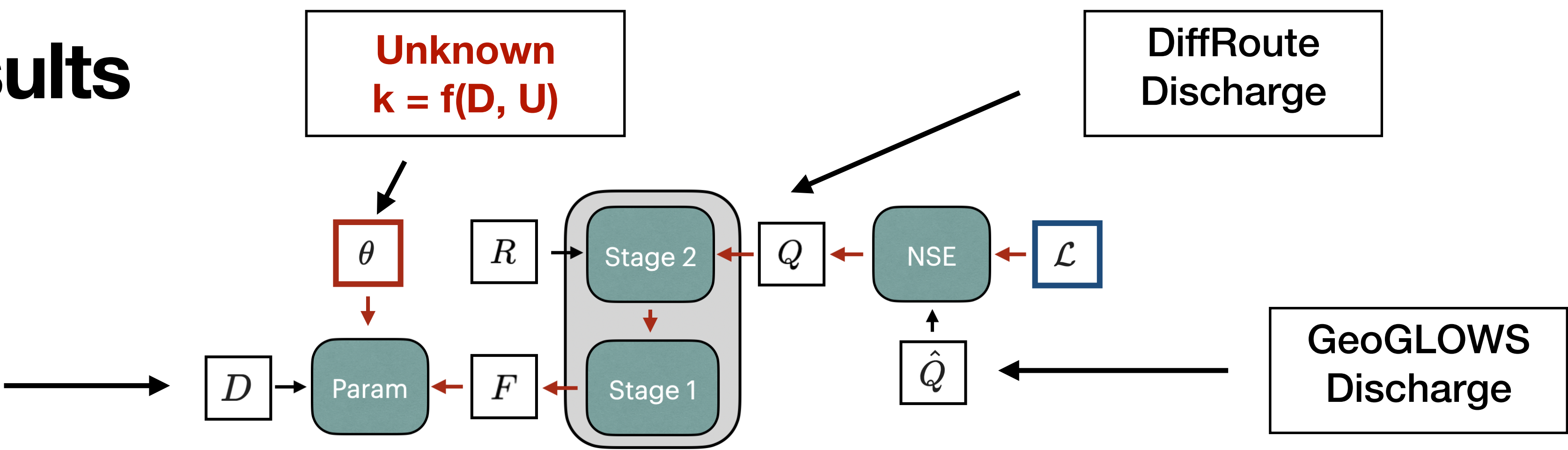
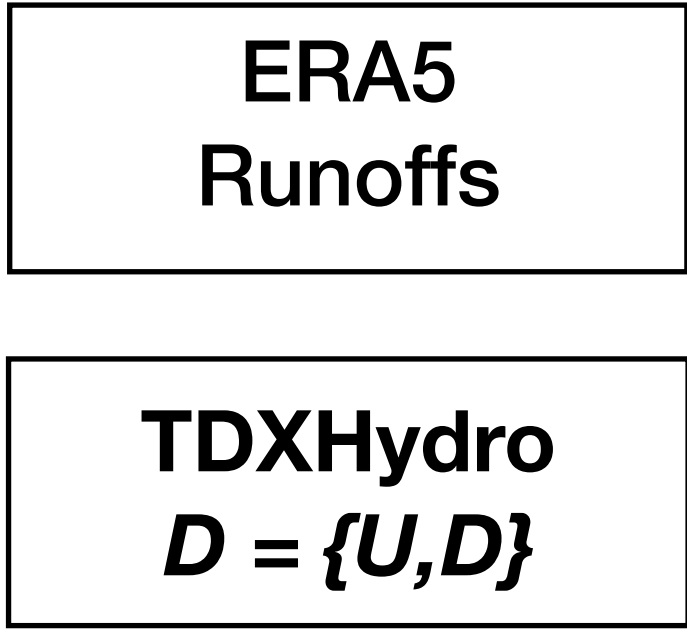
We parameterize this relation with a neural network and train it by regressing the ground truth discharge Q

$$k = f_{\theta}(U, D) \qquad \theta^* = \min_{\theta} \mathcal{L}(Q, \hat{Q})$$

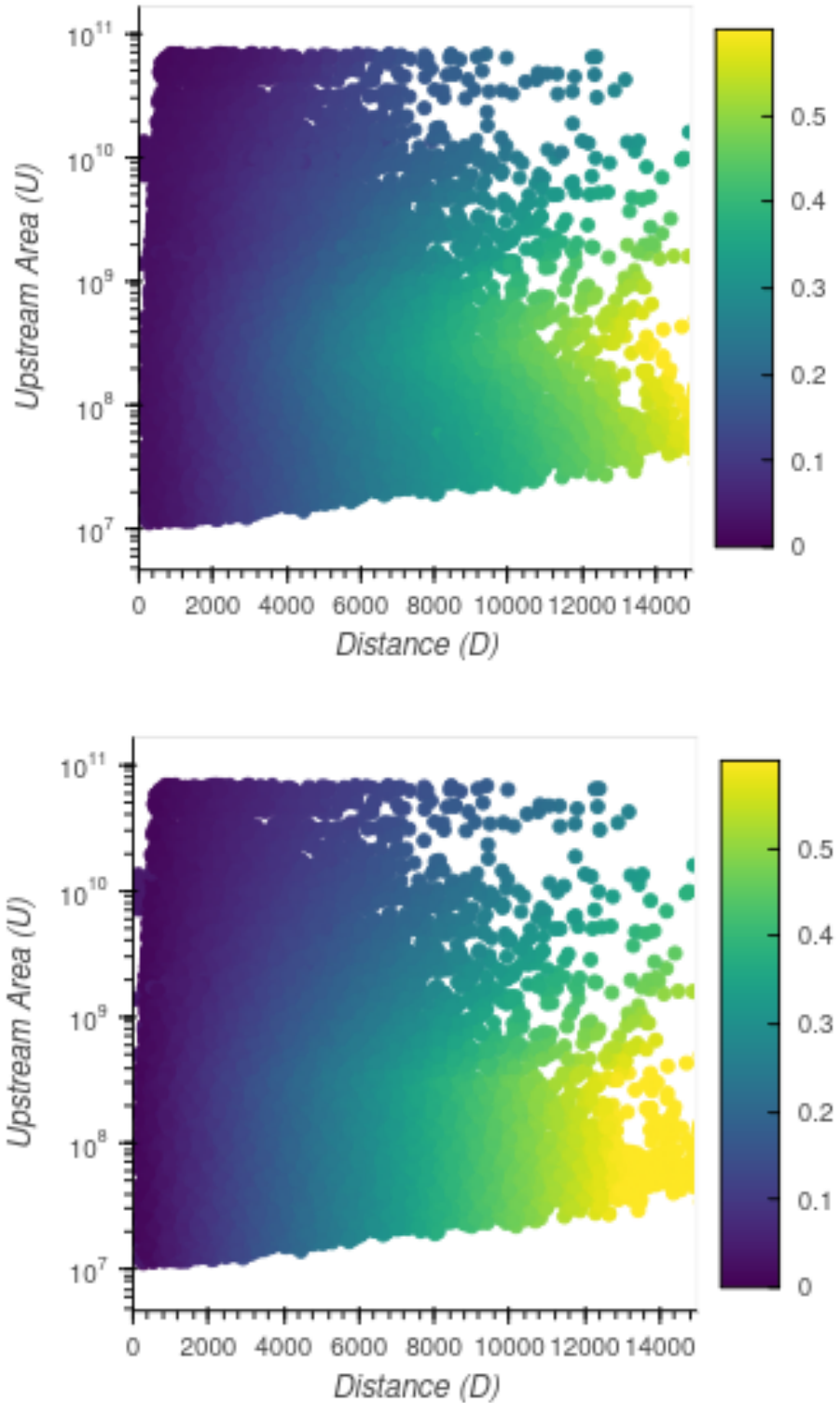
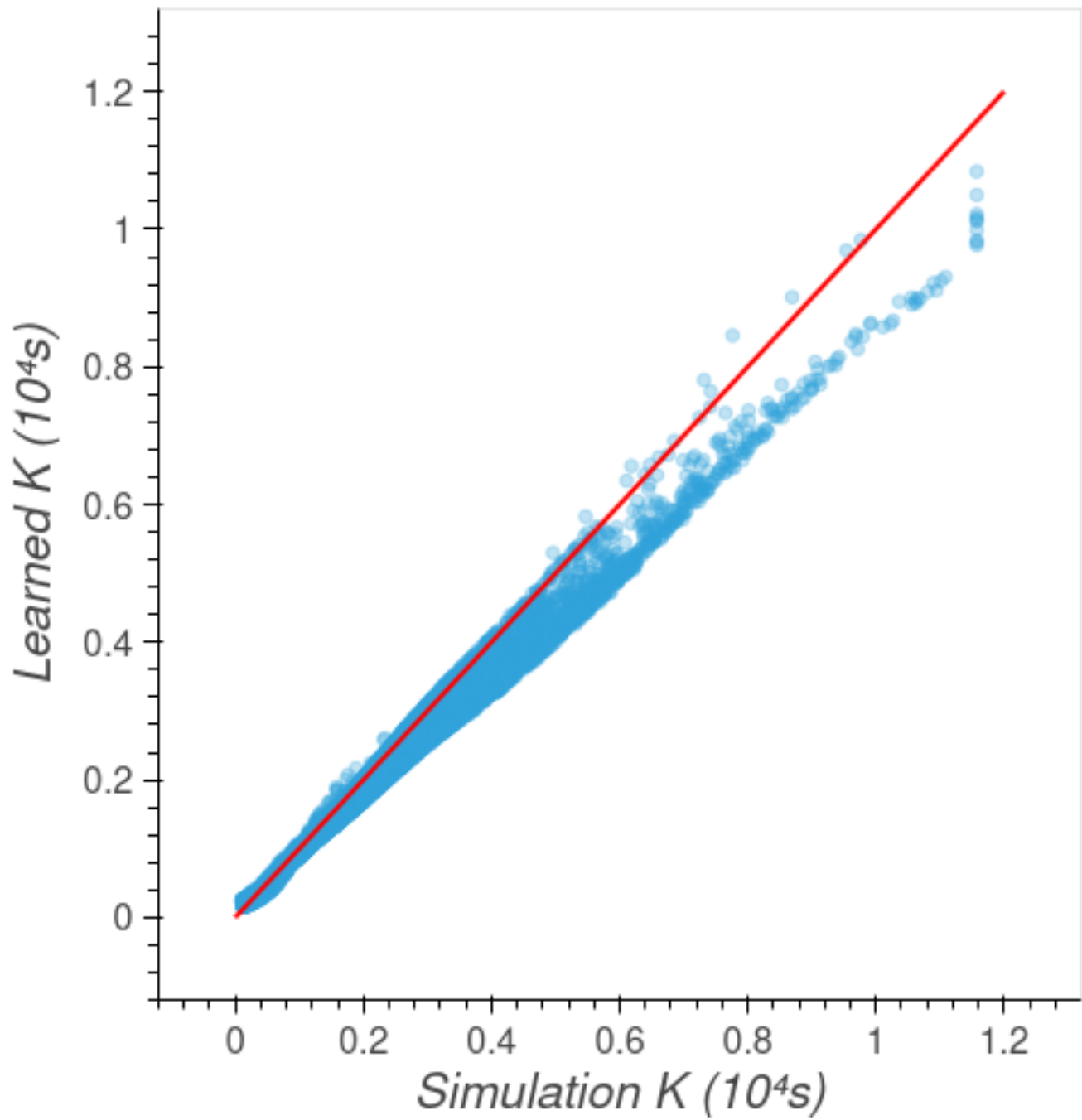
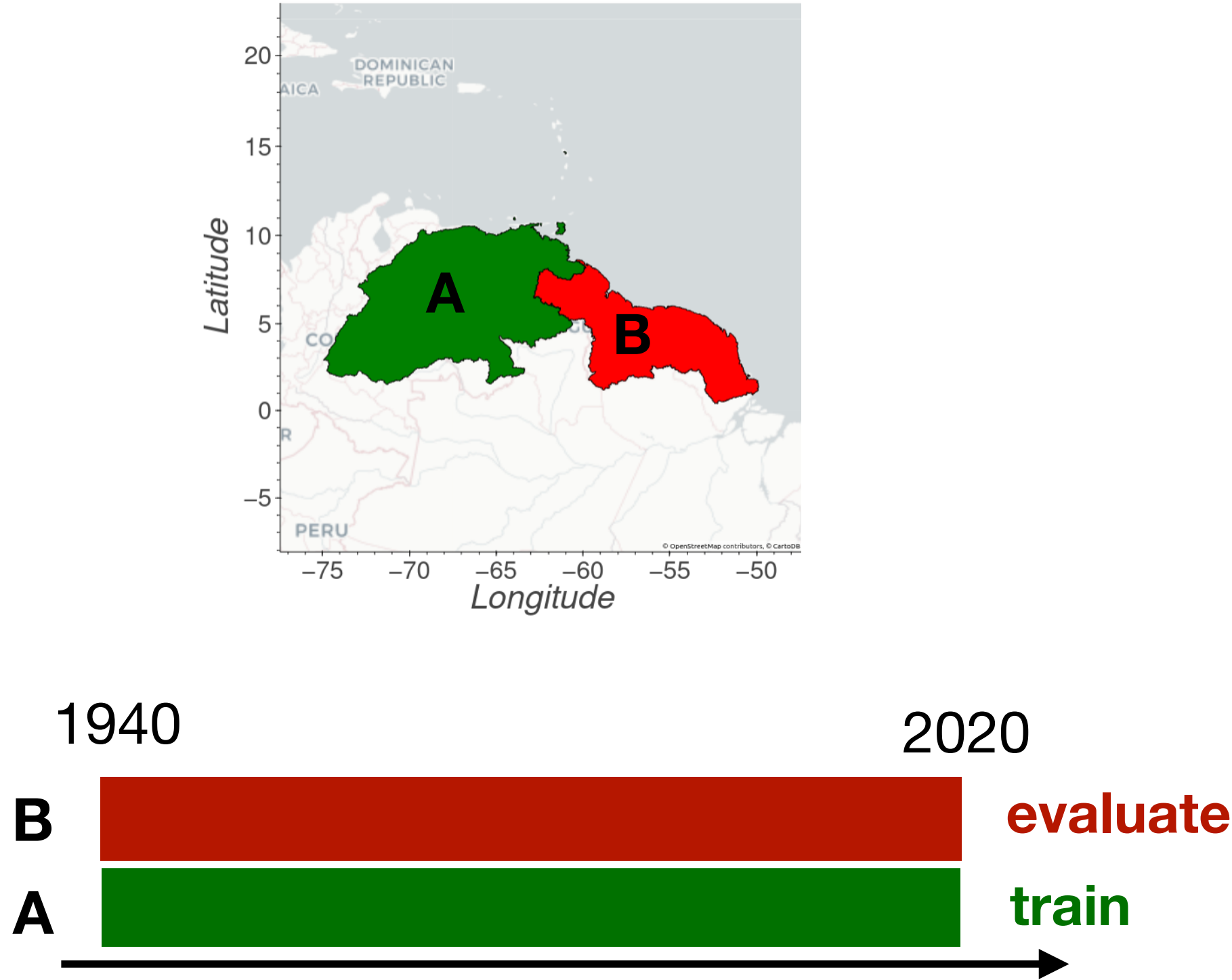


This formulation generalizes in space: the relation f learned on basin A should generalize on any basin B

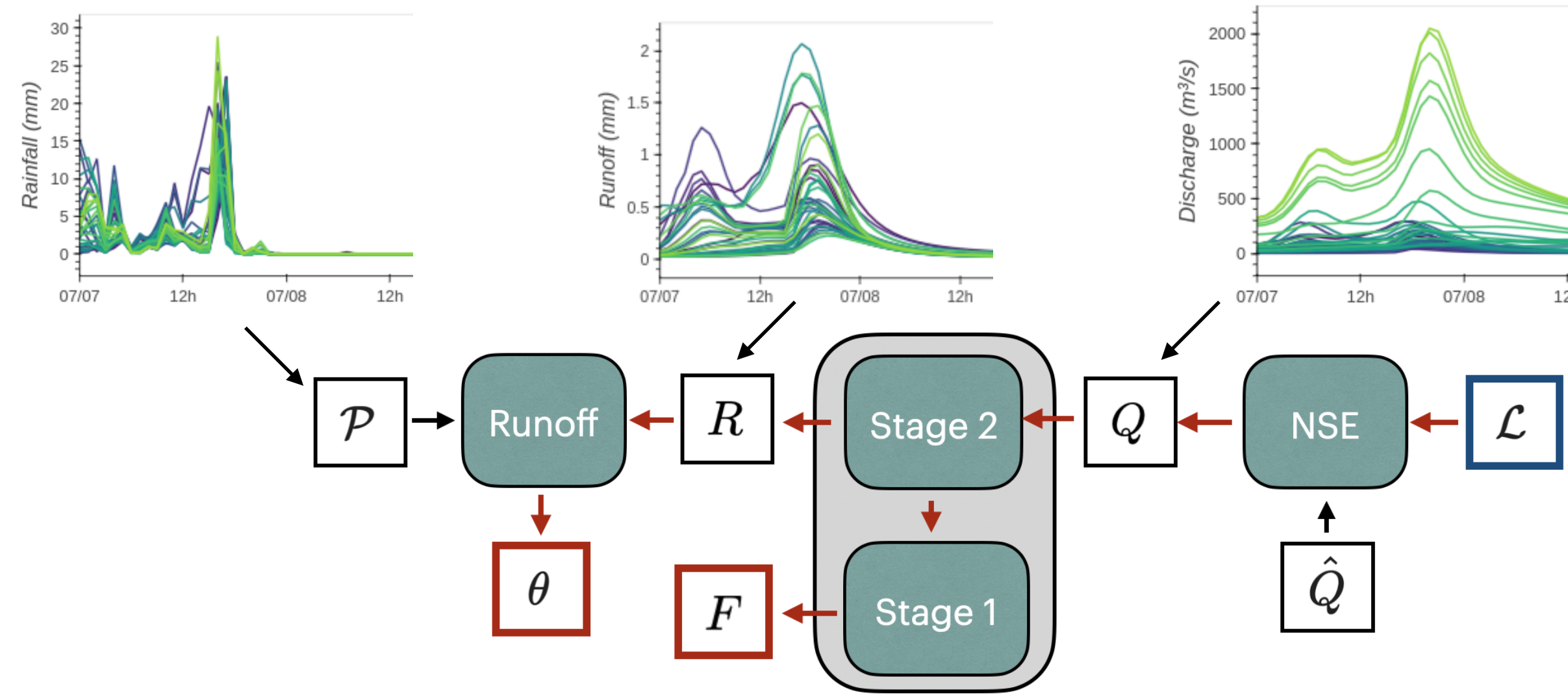
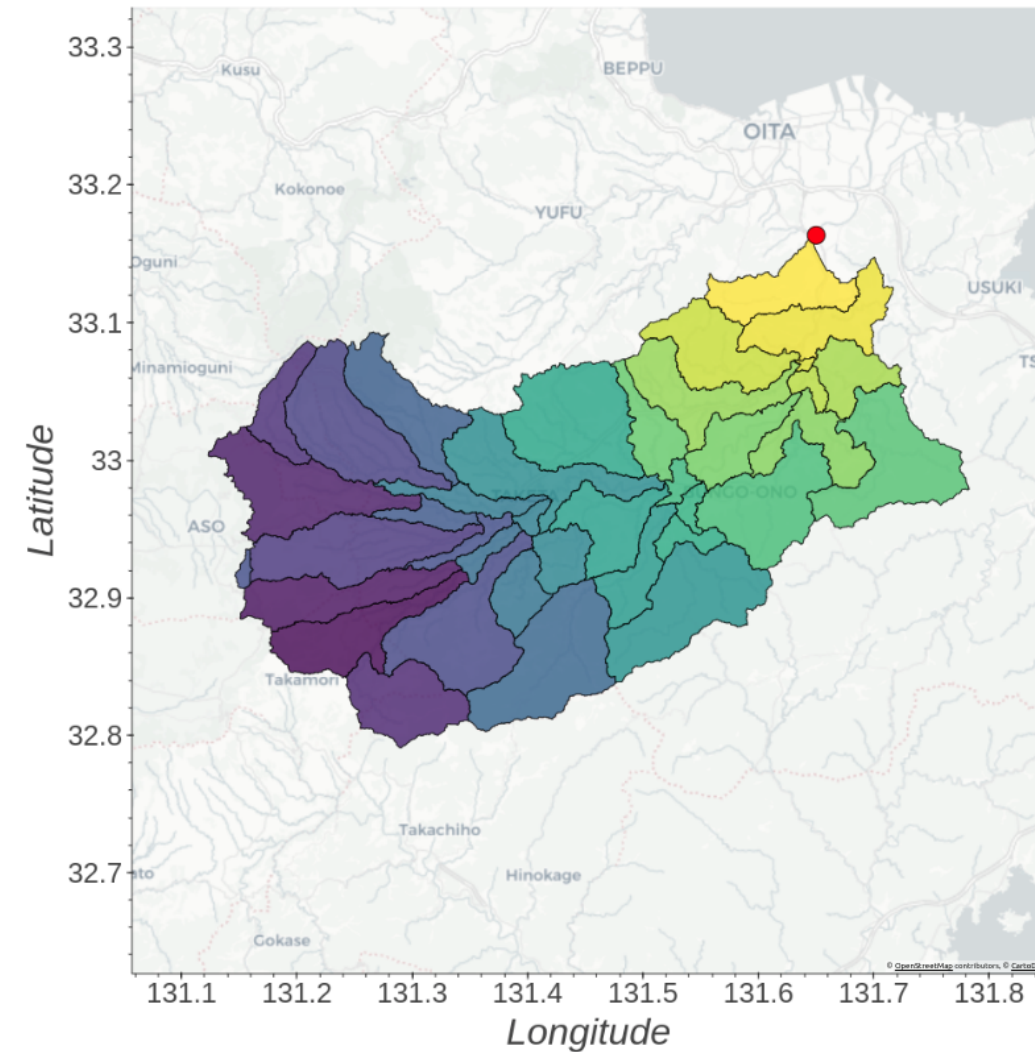
DiffRoute - Results



(b) Learning routing–physics relations



DiffRoute - Results

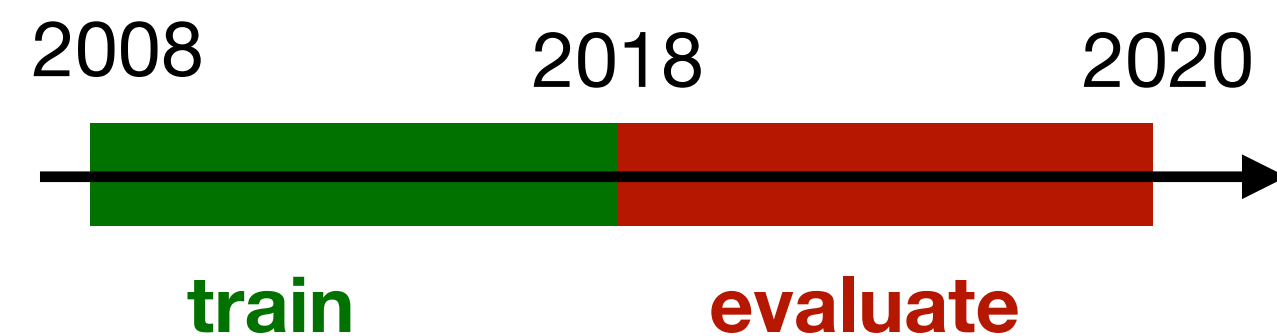


(d) End-to-end learning of runoff and routing

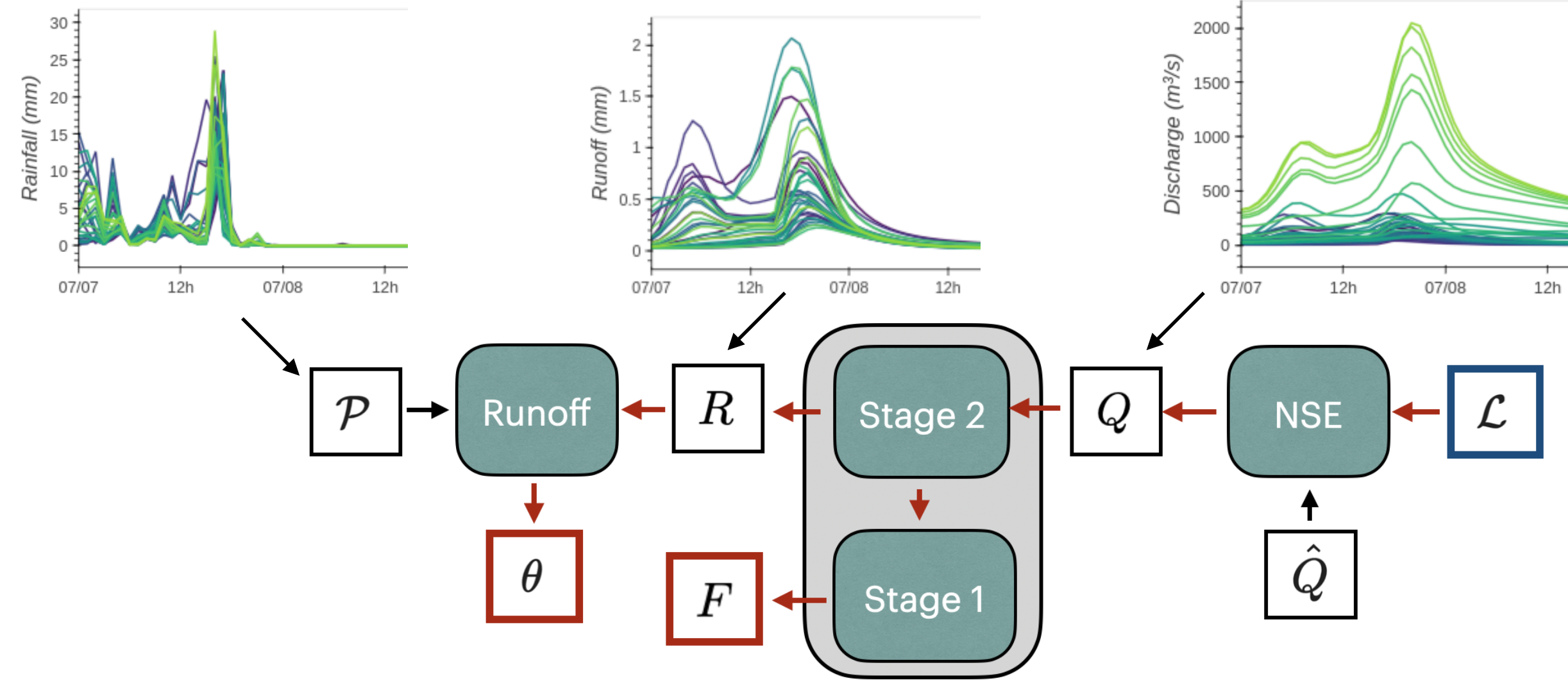
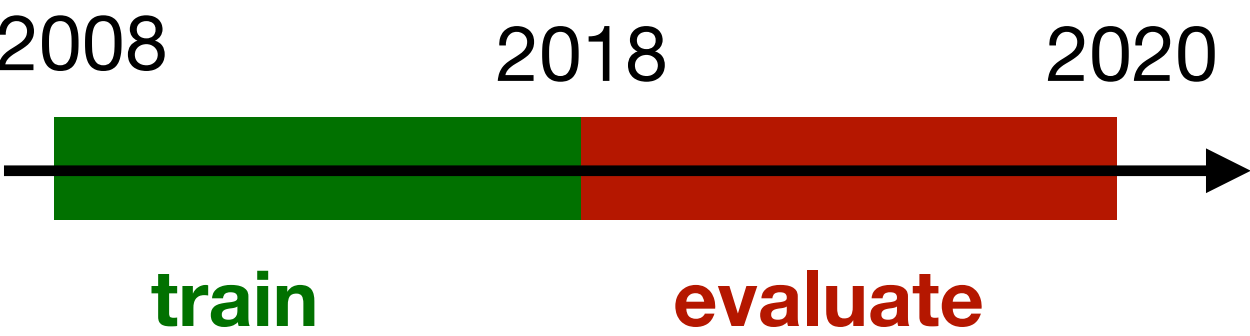
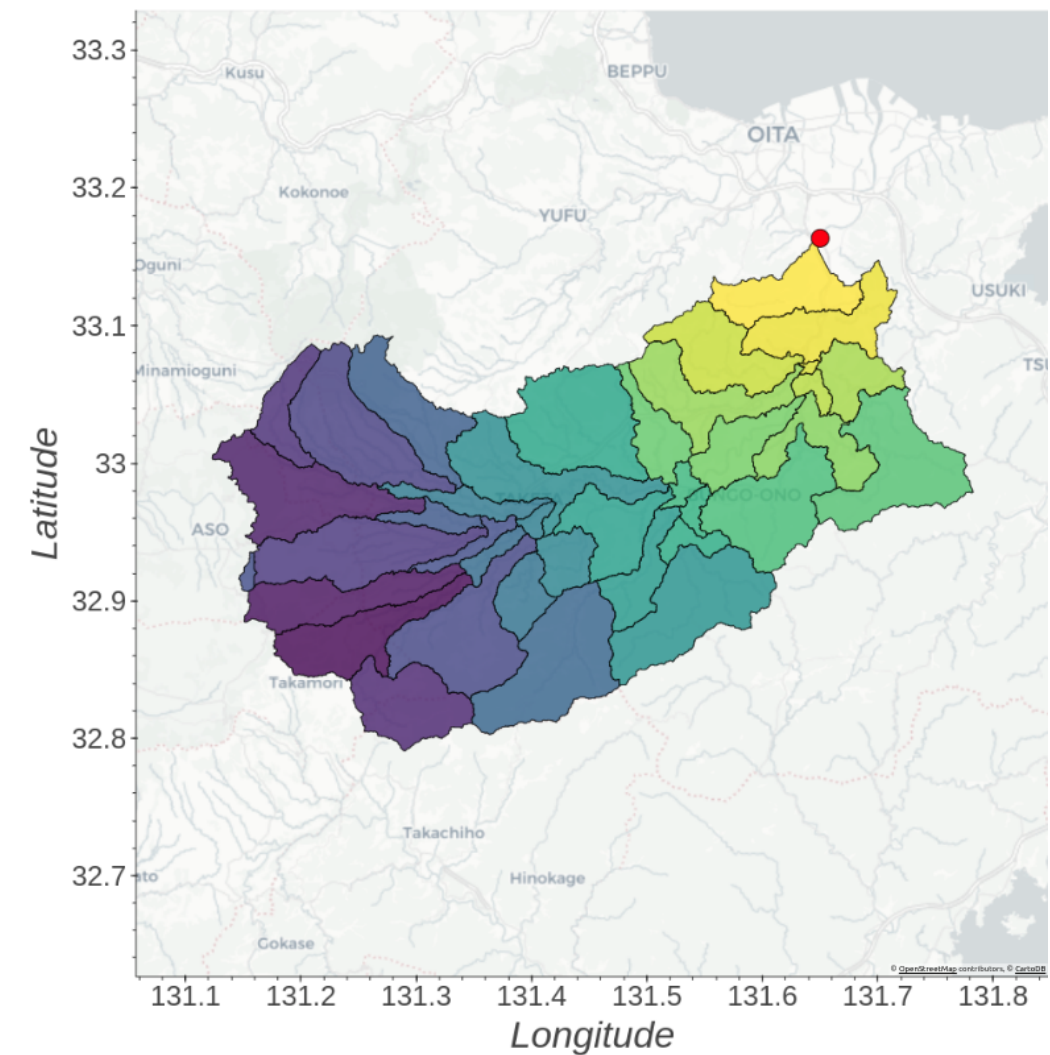
E2EL: Jointly learn Runoff (LSTM) + routing (DiffRoute model) from downstream obs

Compare E2EL to two approaches:

- A Lumped LSTM (gold standard)
- A Two-step approach in which:
 - The runoff LSTM is trained in a lumped at basin scale
 - The runoff is applied at catchment scale to learn the routing (DiffRoute)



DiffRoute - Results

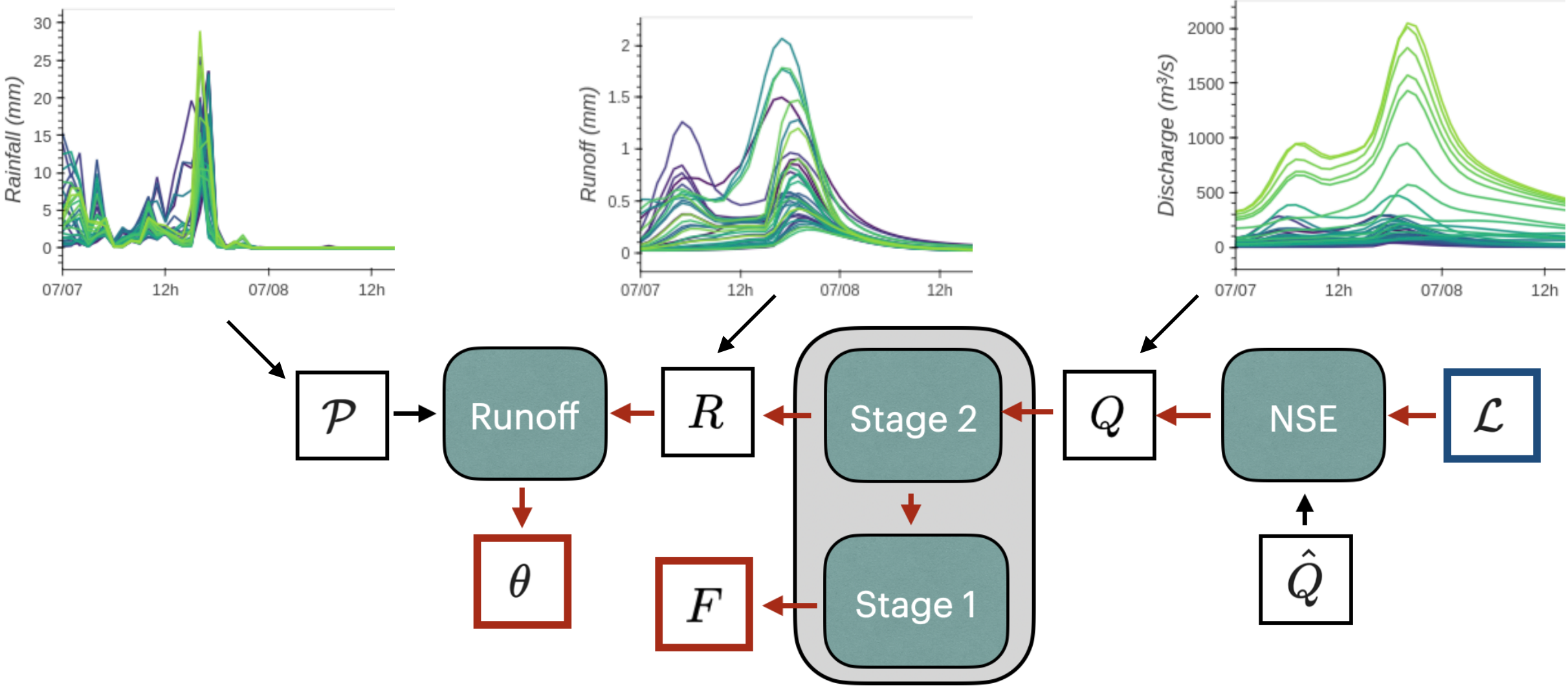
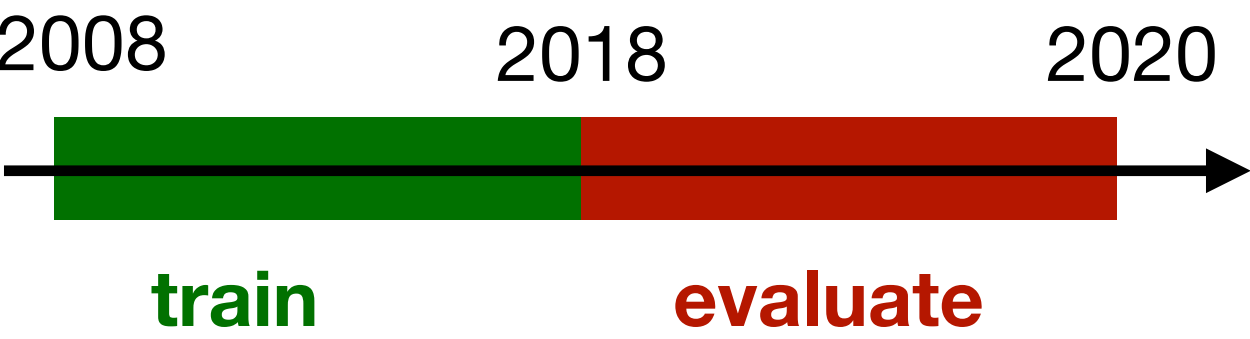
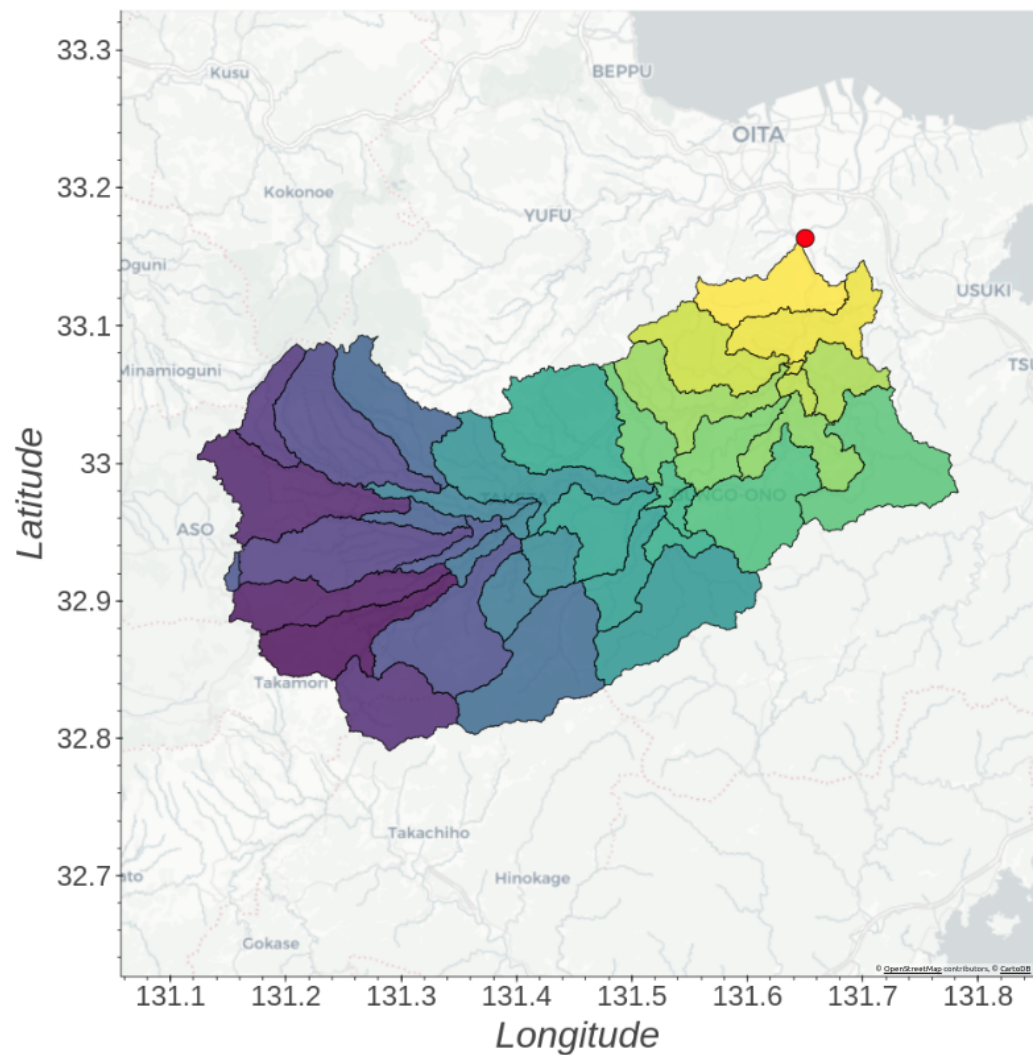


(d) End-to-end learning of runoff and routing

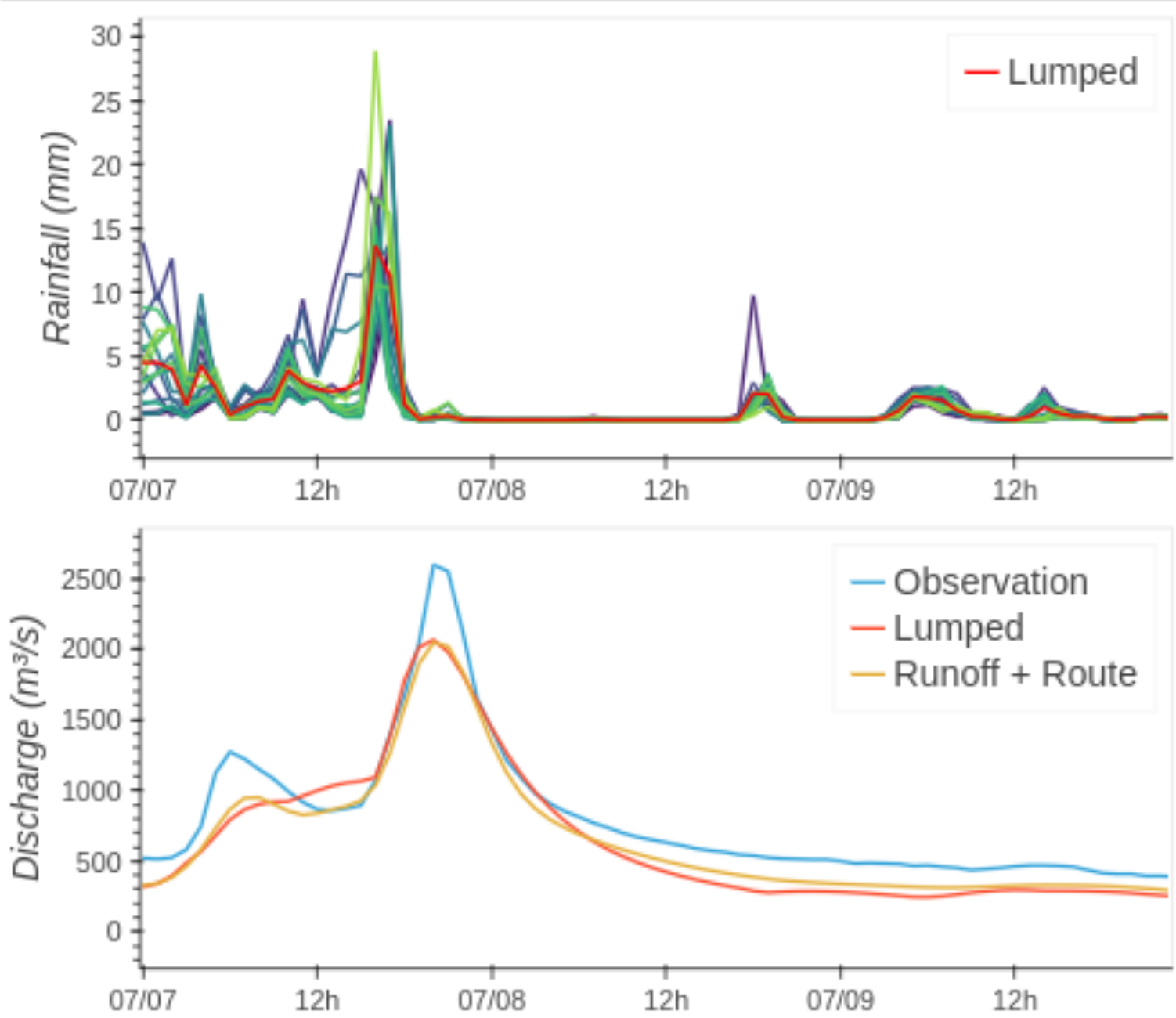
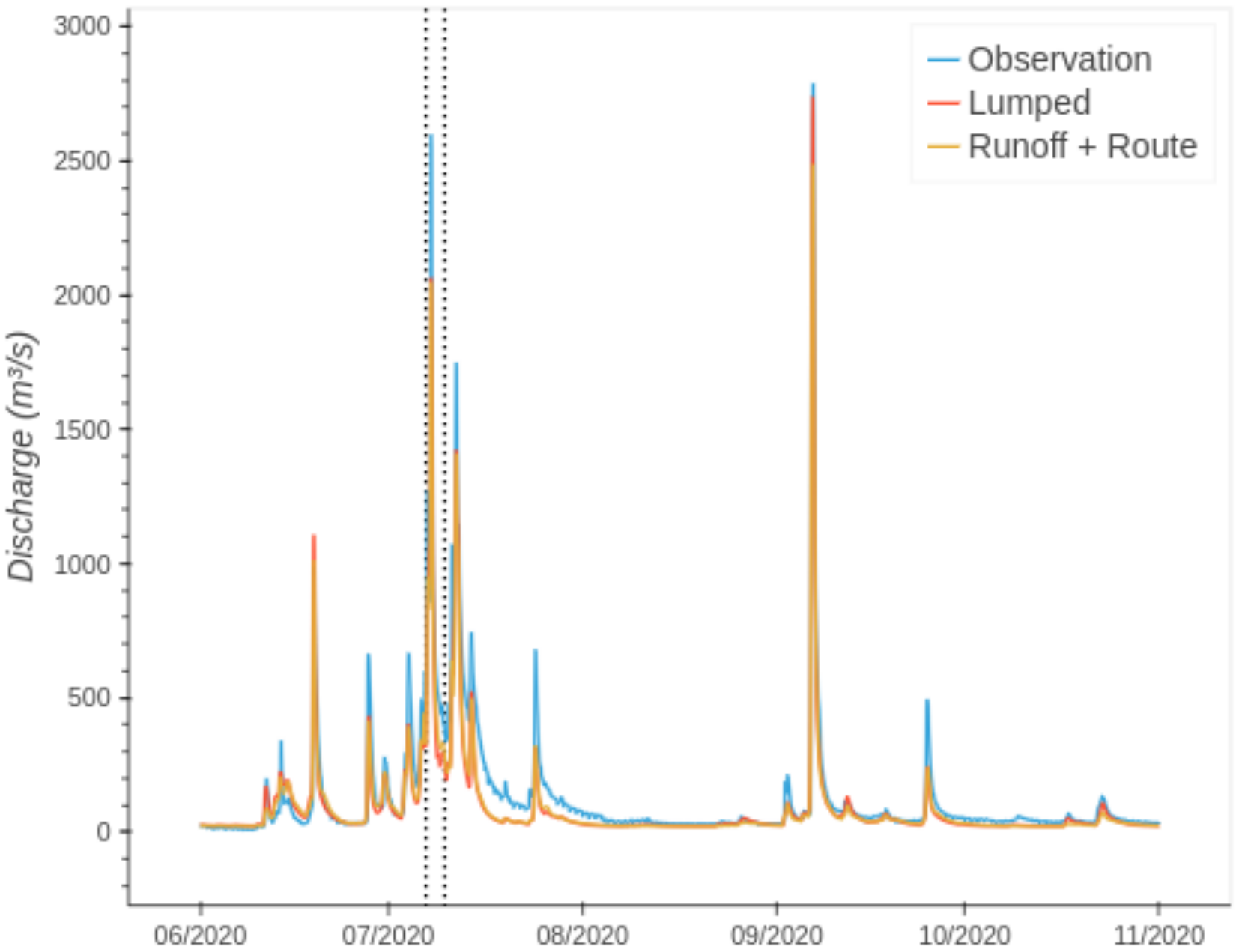
Routing Scheme	Lumped	End-to-end	Two-steps
Pure Lag		0.905 ± 0.005	0.888 ± 0.004
Linear Reservoir		0.921 ± 0.003	0.893 ± 0.004
Muskingum	0.896 ± 0.002	0.910 ± 0.005	0.893 ± 0.004
Nash Cascade		0.919 ± 0.008	0.892 ± 0.004
Linear Diffusive Wave (Hayami)		0.923 ± 0.003	0.874 ± 0.005

NSE Results for different approaches

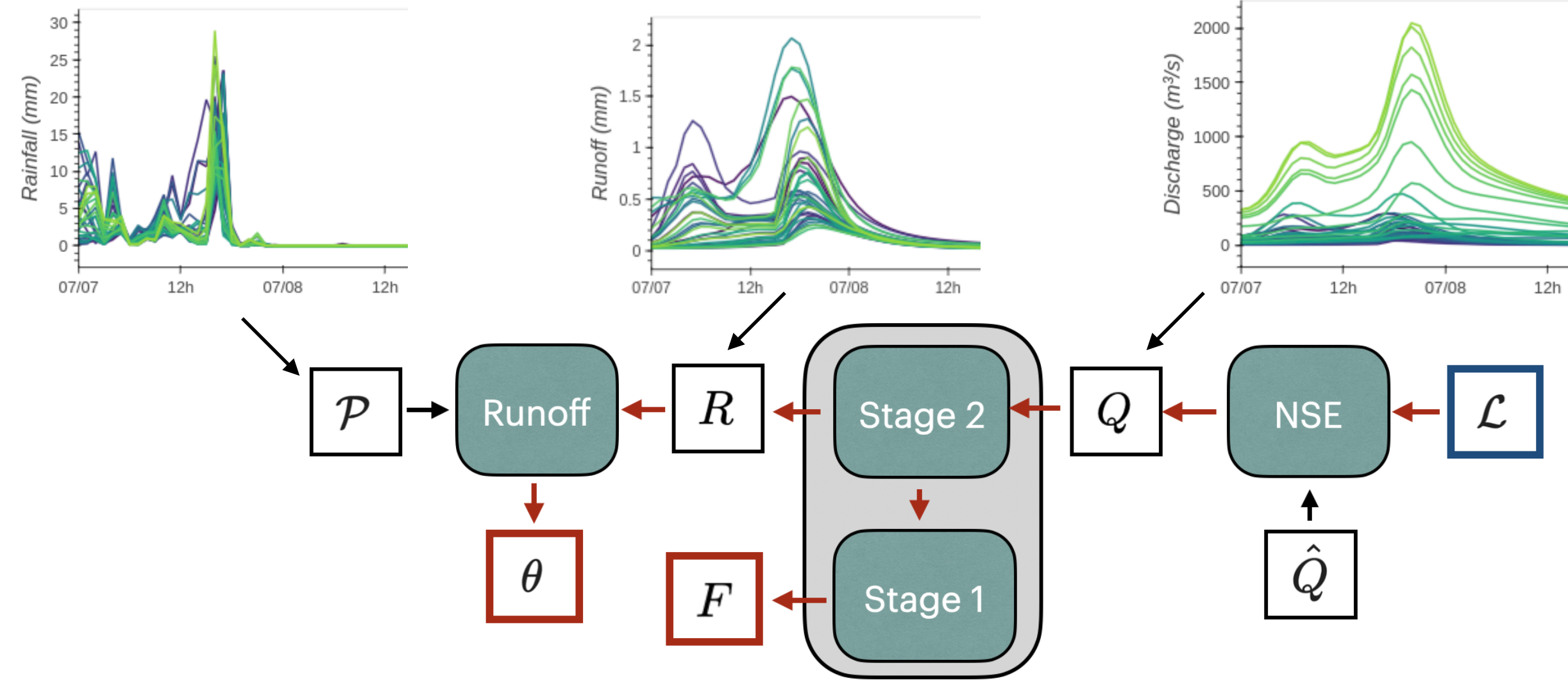
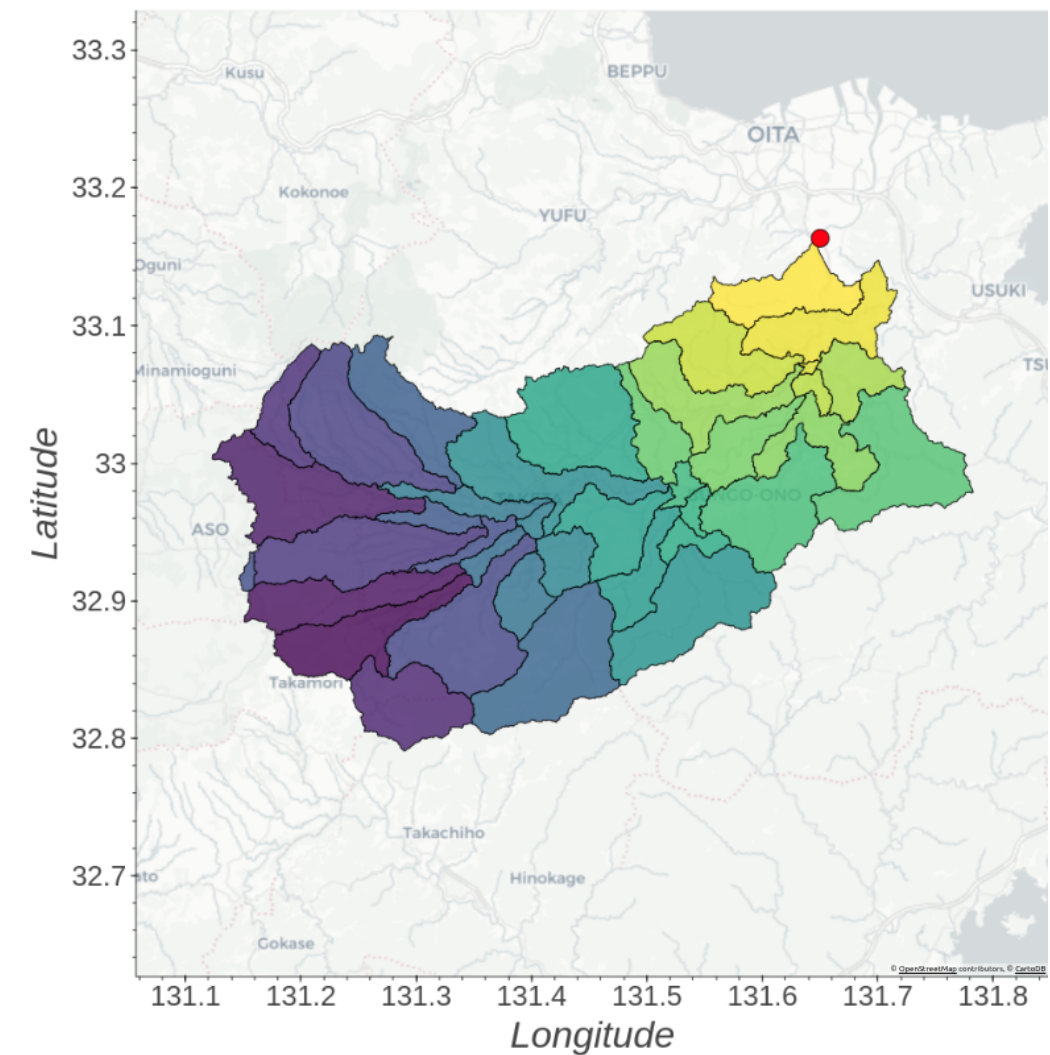
DiffRoute - Results



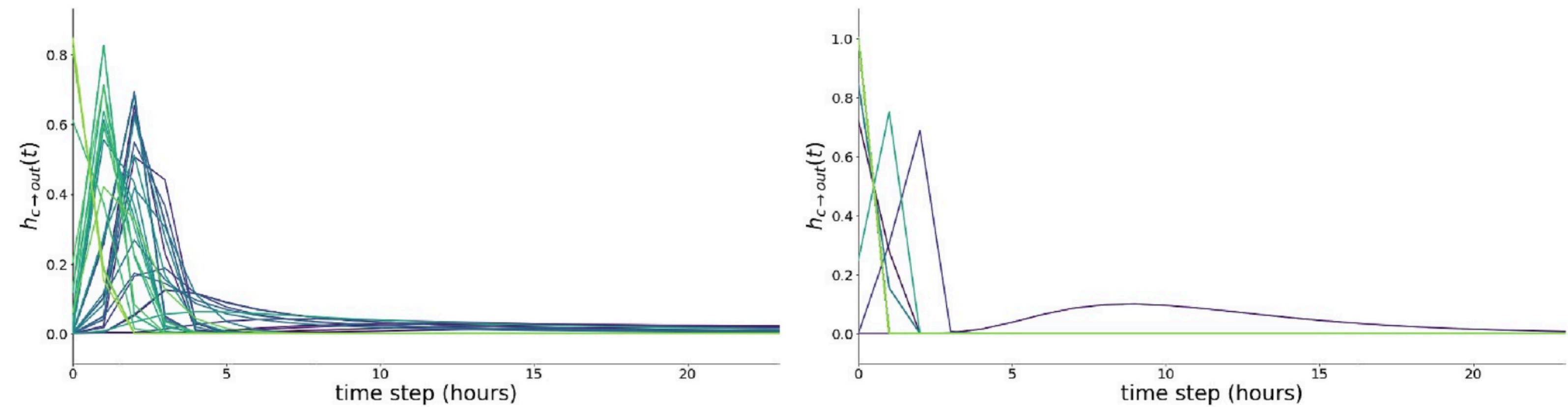
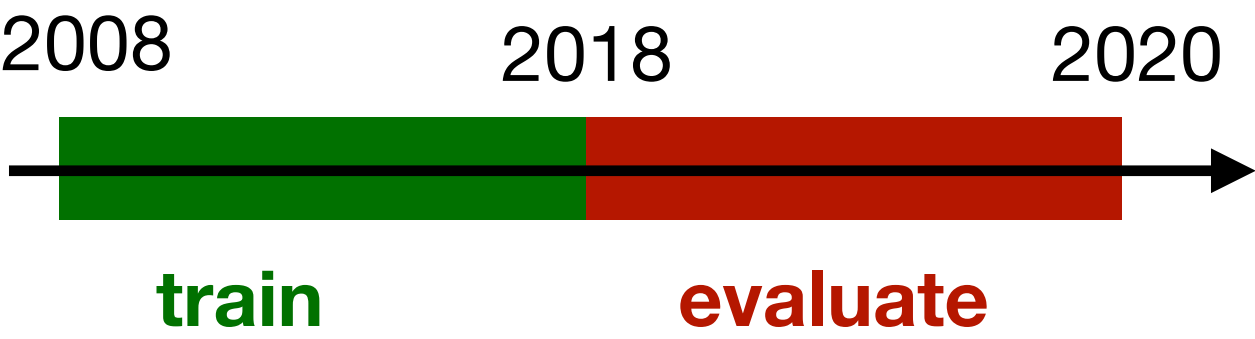
(d) End-to-end learning of runoff and routing



DiffRoute - Results



(d) End-to-end learning of runoff and routing



DiffRoute - Recap

- Idea: Computations of *LTI Routing* models are equivalent to *Conv1D* layer in Deep Learning
 - Bridge modern DL and (old) hydrological models
- Results
 - **Speed**: *Global climatic scale simulation in 10s*
 - **Generality**: *Works for the linearization of any physical model*
 - **Differentiability**: *Enables stable learning across many use cases (including E2EL)*
- Outputs
 - **DiffRoute**: <https://github.com/TristHas/DiffRoute>
 - **DiffHydro**: <https://github.com/TristHas/DiffHydro>
 - **Paper**: “*Differentiable River Routing for End-to-End Learning of Hydrological Processes*”, AGU JGR Machine Learning and Computations, 2026

DiffRoute - Recap



DiffRoute Overview

- > `diffroute` provides a differentiable and GPU-accelerated formulation of classical Linear Time Invariant (LTI) river routing models. Every component is written in PyTorch, which keeps routing differentiable and easy to integrate in deep learning workflows.
- > For advanced use-cases, we recommend using `diffroute` through the companion module `diffhydro`, which layers additional data structures and utilities to integrate DiffRoute into more complex differentiable hydrological pipelines, with minimal dependency. However, DiffRoute can also be used as a standalone component as showcased in this documentation.

Please note that `diffroute` is in an early development stage and thus subject to changes. Nevertheless, the explanations included in this documentation are limited to API and structural components that are unlikely to change in the short to middle term.

DiffRoute - Caution

- **Wrong tool** for non-linear hydraulic
 - Flood and inundation dynamics
 - Any complex hydraulics modeling
- The **right tool**
 - Research on end-to-end learning of hydrological process
 - Theoretical study of ML parameter identifiability
 - For applications requiring extreme computation speed (counterfactual, etc.)
 - Water resource management modeling over short/steep rivers (Japan)
 - For complex hydraulic model surrogate research
 - Use the fast operator as a building block of non-linear models

Outline

- 1. Differentiable routing (DiffRoute) **enables** end-to-end learning (E2EL)
 - *“Differentiable river routing for end-to-end learning of hydrological processes”, AGU JGR Machine Learning and Computations, 2026*
- 2. E2EL produces **high accuracy** predictions with **physically consistent spatial distribution**
 - *“Leveraging Japan's streamflow records for end-to-end learning of hydrological simulations at national scale”, Water Resource Research, 2026, (preprint)*
- 3. E2EL allows to **identify physically meaningful** dynamic parameters
 - *“End-to-end learning from sparse and noisy observations mostly resolves the equifinality of distributed river-discharge models over Japan”, ECMWF workshop, in preparation*

Overview

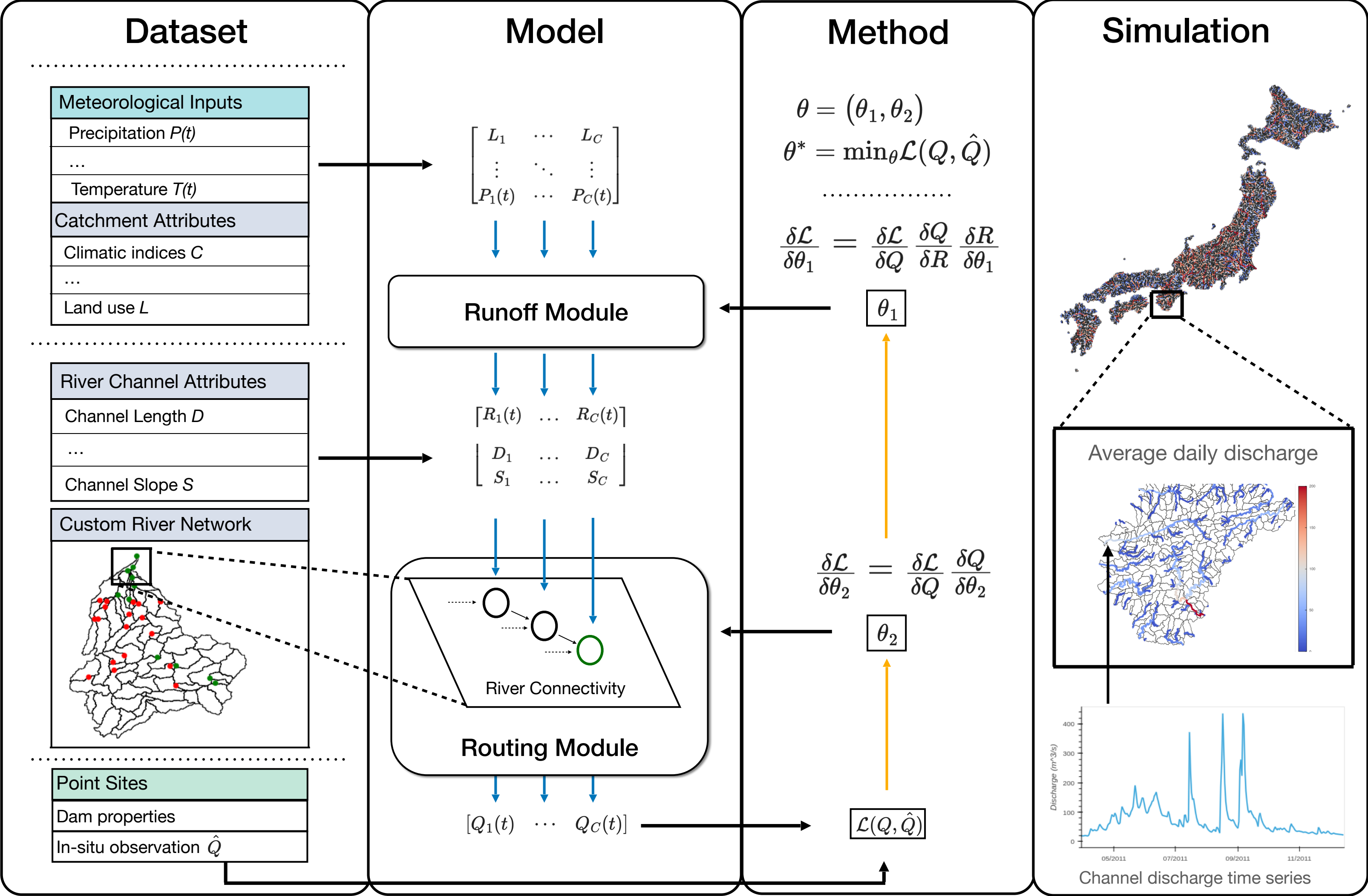
Present a national scale river discharge **simulation** shows both **high accuracy** and physically consistent **spatial distribution**. This simulation relies on three core contributions.

- A national scale hydro-meteorological Dataset that aggregates high accuracy regional data sources over a custom river network that aligns catchments to in-situ measurements
- A differentiable hydrological Model combining LSTM Runoff + DiffRoute routing
- **Method:** End-to-end training of the differentiable model from river discharge observations

Results

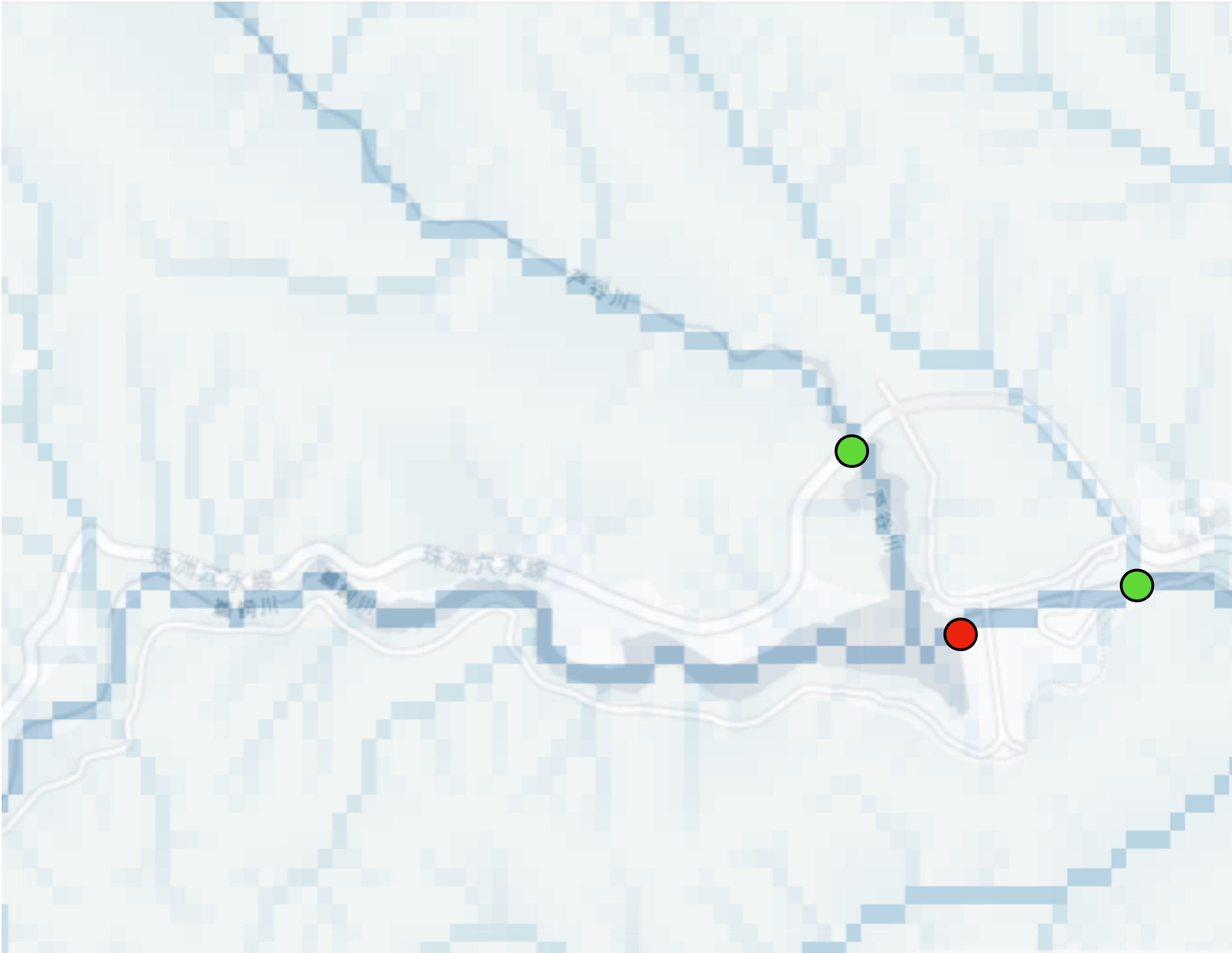
- Release simulation, dataset, model and experiments
- Demonstration that E2EL at national scale works and highlight future research directions

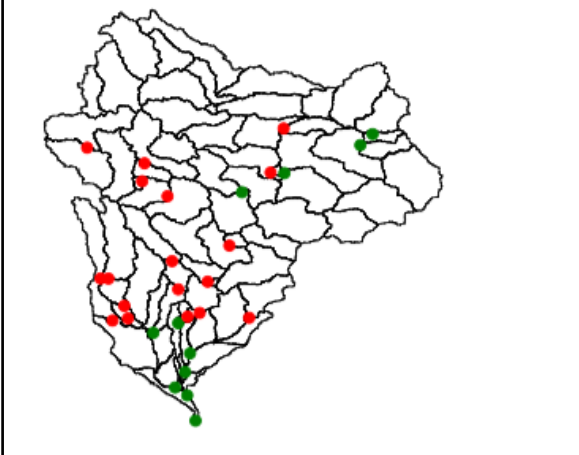
National scale simulation



National scale simulation - dataset

The location of interest points is **mapped onto digital elevation model** (JFlowDir)

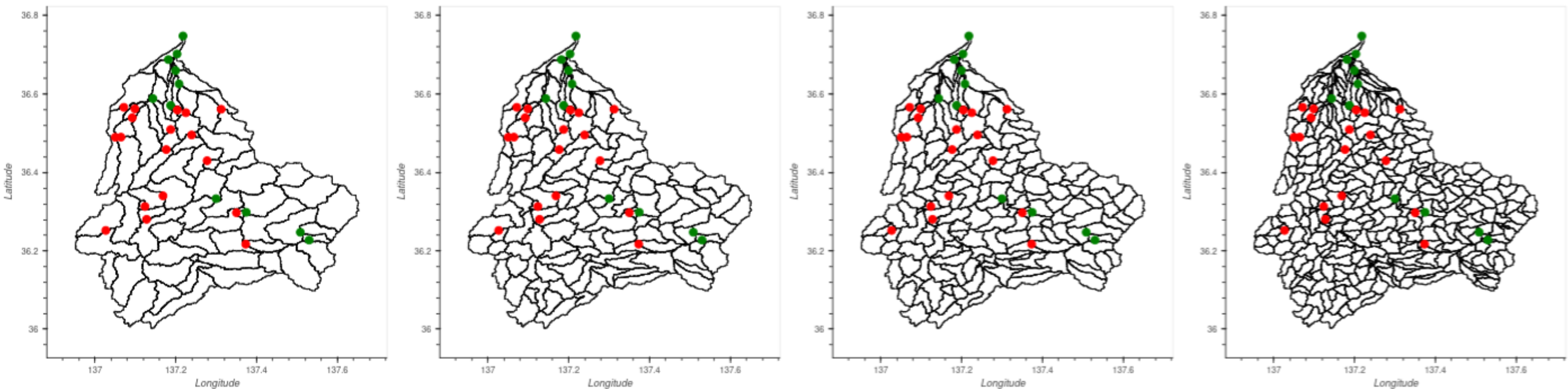


Data
Meteorological Inputs
Precipitation $P(t)$
...
Temperature $T(t)$
Catchment Attributes
Climatic indices C
...
Land use L
River Channel Attributes
Channel Length D
Channel Slope S
Custom River Network

Streamflow
In-situ observation $\hat{Q}$

National scale simulation - dataset

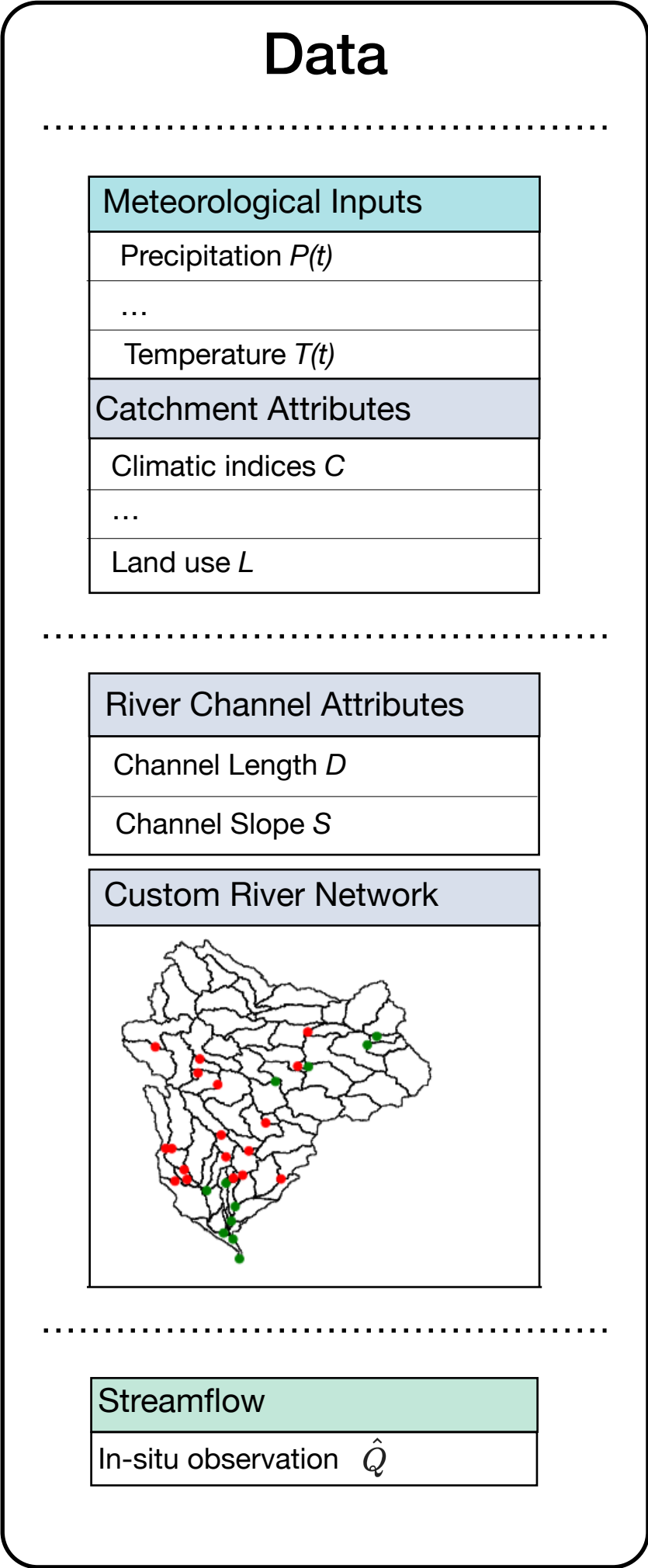
The location of interest points is **mapped onto digital elevation model** (JFlowDir)

- **Exact** drainage area of observations and channel distance between observations
- **Disambiguates** upstream-downstream relations between points
- To do so **at any scale**



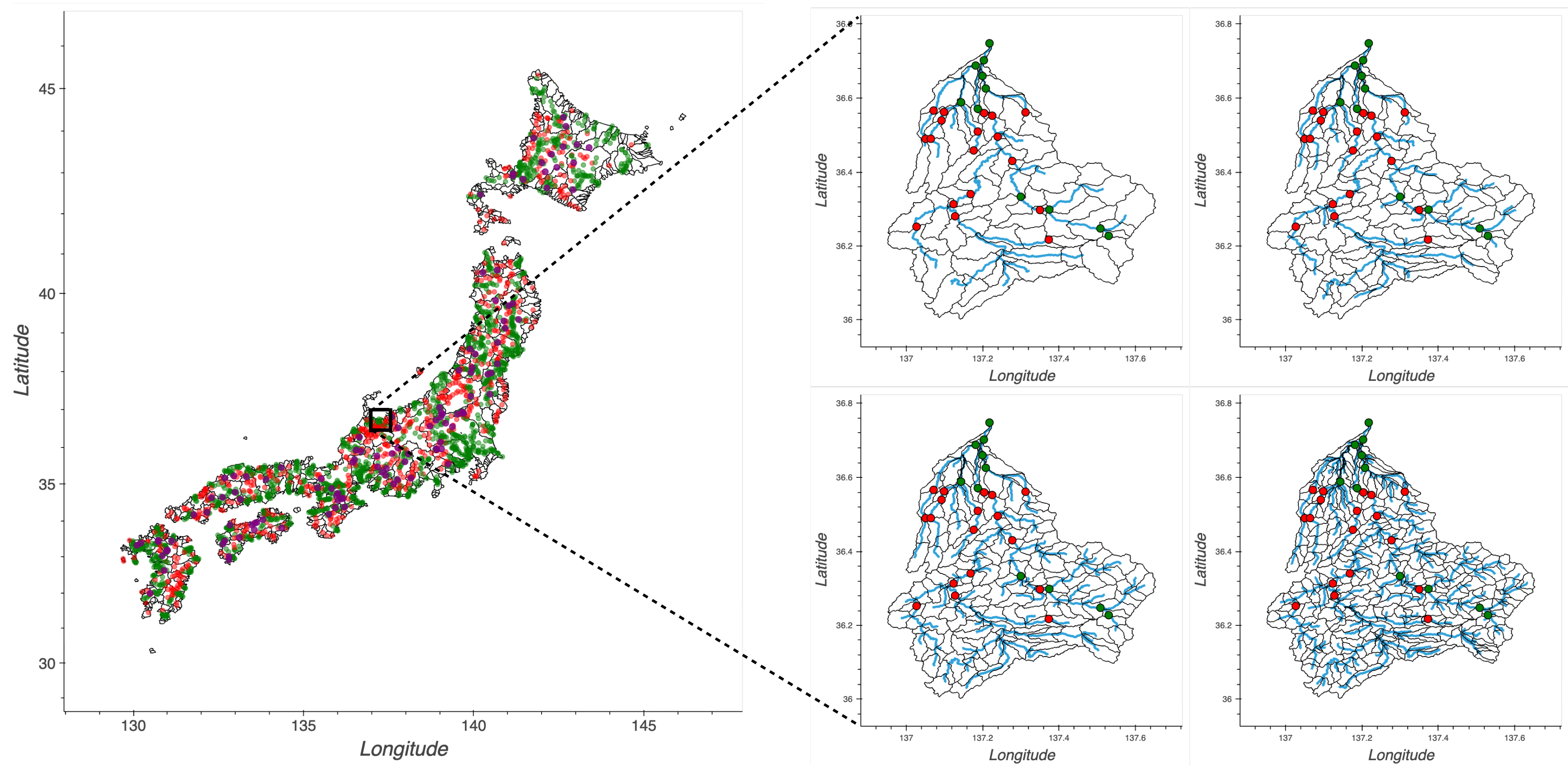
Low spatial variability
Computationally cheap
More constrained

High spatial variability
Computationally expensive
Less constrained



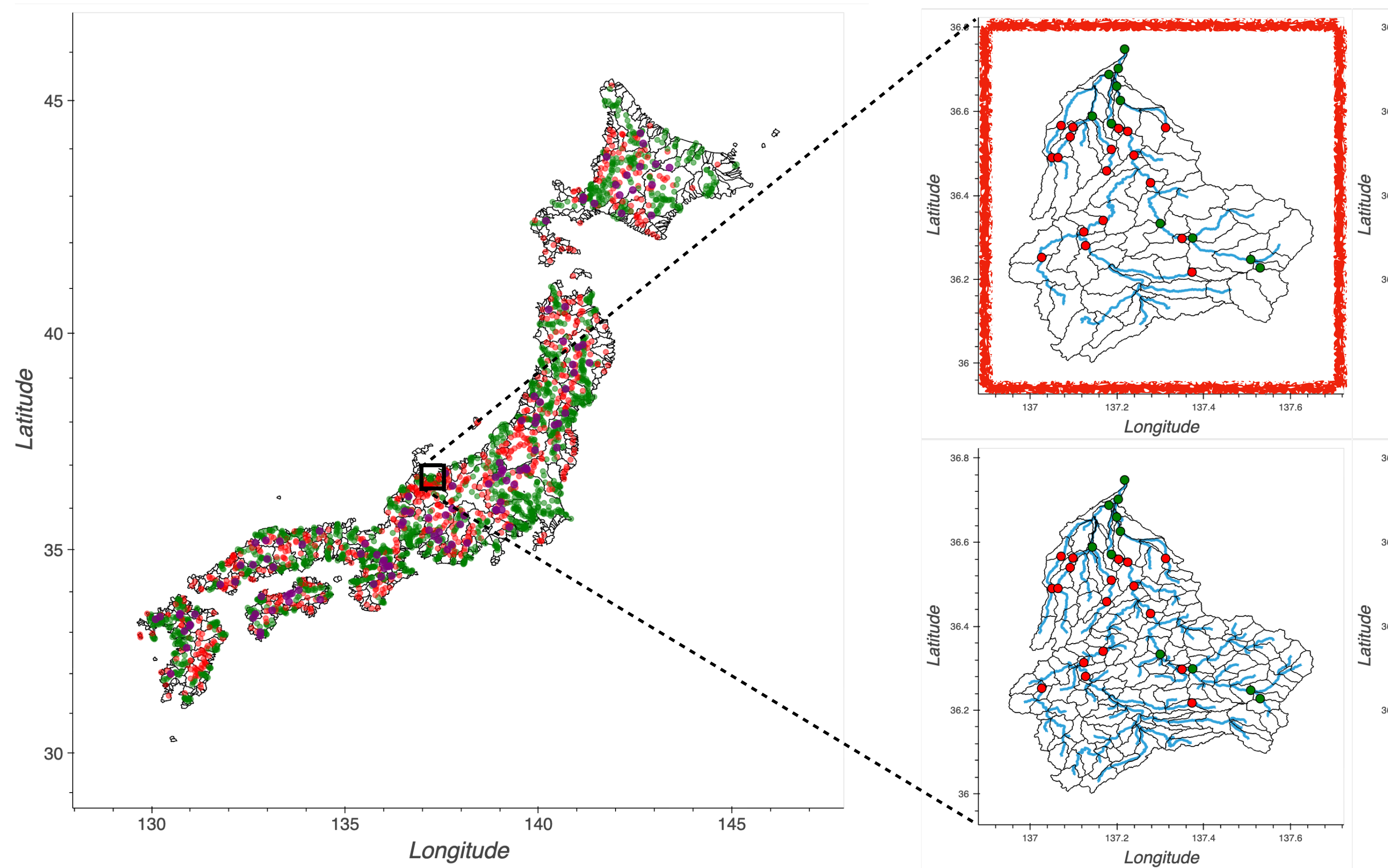
National scale simulation - dataset

We map all MIT gauges and JDA dam locations at national scale and segment networks of different sizes



National scale simulation - dataset

We map all MIT gauges and JDA dam locations at national scale and segment networks of different sizes



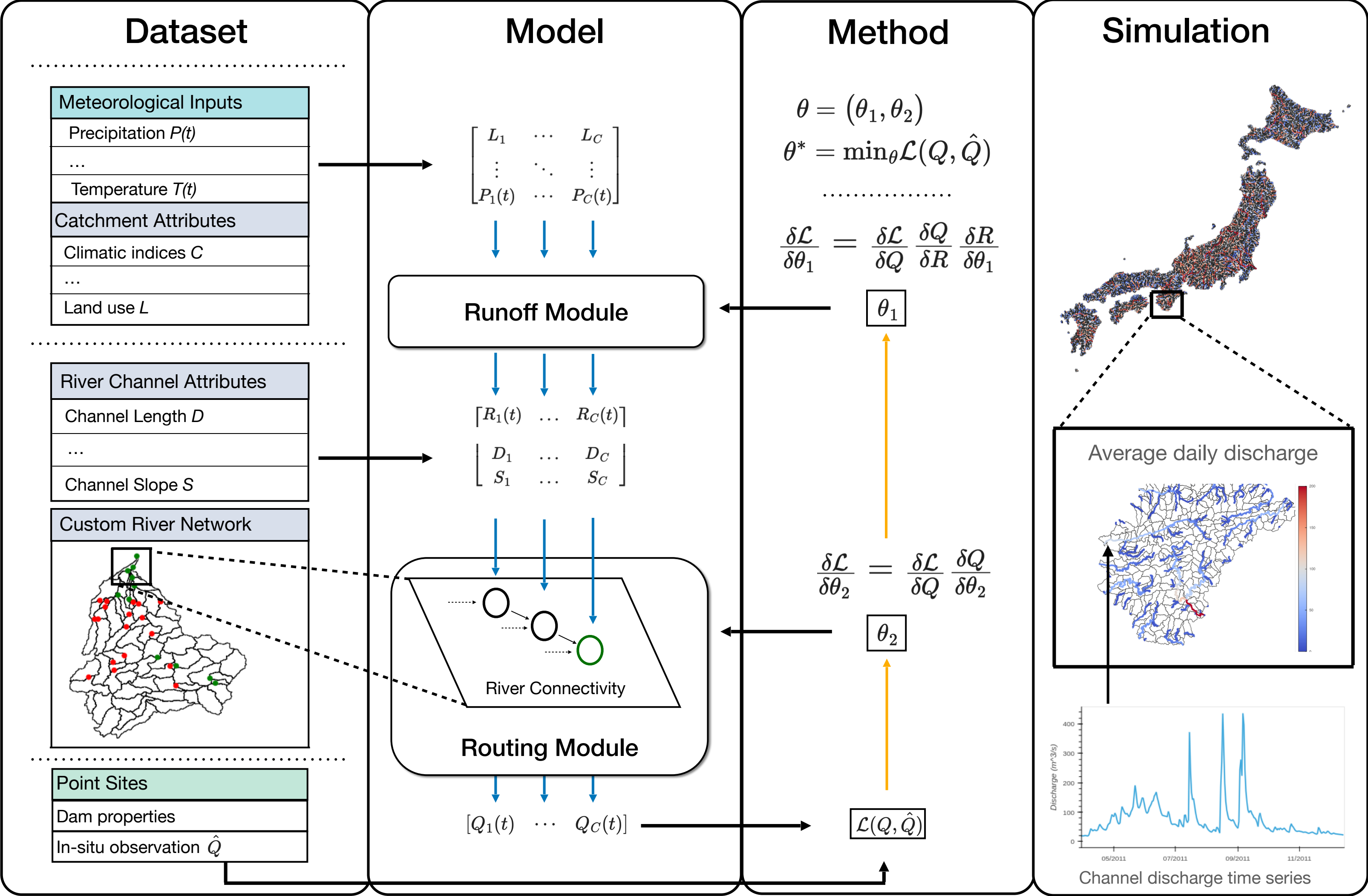
Category	Count
Total catchments	8,893
Total Gauged	1,768
With stage only	508
With Calibrated discharge	1,260
→ Pass Quality Controls	915
→ Training sites (no upstream dams)	314
Total Dammed	852
→ Monitored	116
→ Unmonitored	736
Other catchments	6,273

National scale simulation - dataset

We aggregate *high accuracy regional products* onto the river network catchments

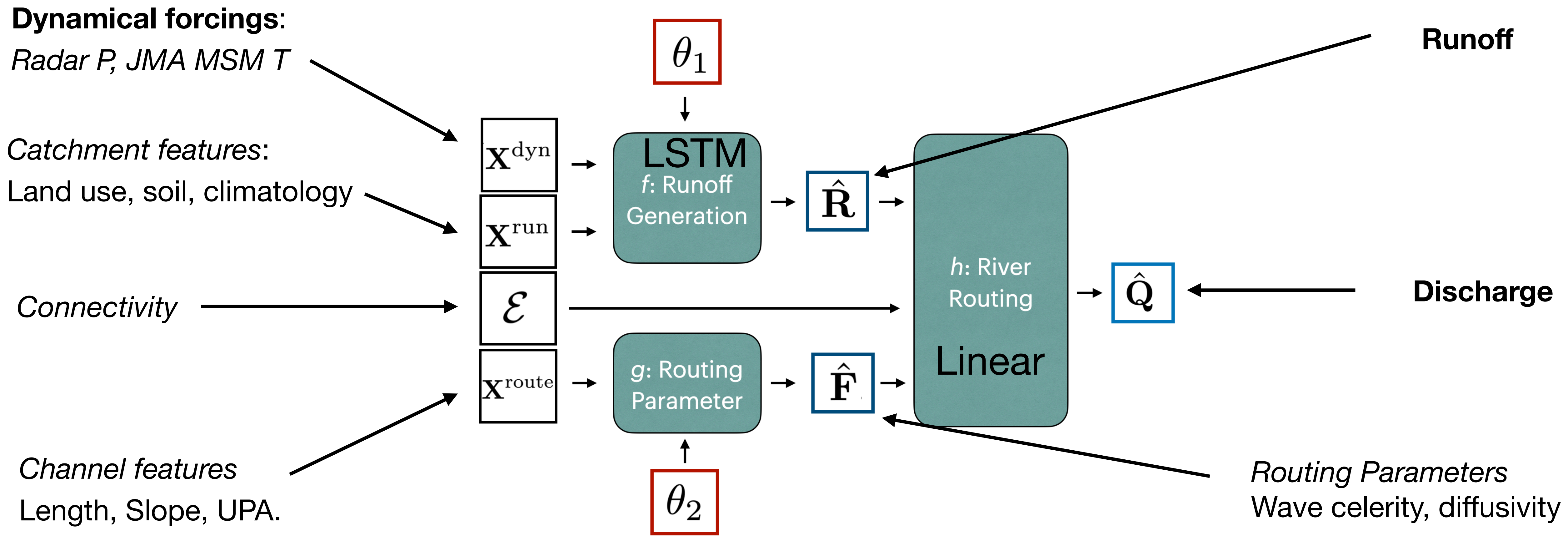
Provider	Variable Types	Resolutions	Availability	Reference
	Observational streamflow and infrastructure			
MLIT	<i>In situ</i> stage / Q	-/-	1950–2024	(MLIT, 2024)
MLIT	Dam inflows/outflows and volumes	-/-	1950–2024	(MLIT, 2024)
Japanese Dam Assoc.	Reservoir dimensions	-/-	-	(Japan Dam Association, 2024)
	Dynamic meteorological forcings			
U. Tokyo	Interpolated <i>in situ</i> P	1d / 0.01°	1926–2020	(Hatono et al., 2022)
JMA	Radar P	10 min. / 250m	2006–2024	(Nagata, 2011)
JMA	Gauge adjusted Radar P	30 min. / 250m	2005–2021	(Nagata, 2011)
JMA	MSM analysis surface variables	1h / 5km	2008–2024	(Saito et al., 2006)
ECMWF	ERA5-land reanalysis variables	1d / 0.25°	1979–present	(Hersbach et al., 2020)
	Static catchment and channel attributes			
JAXA EORC	LULC classes	- / 10m	NA	(Hirayama et al., 2022)
Geospatial Info. Auth.	J-FlwDir UPA, elv, fdir	- / 30m	NA	(Yamazaki et al., 2020)
HydroATLAS	Hydro-environmental attr.	- / ~500m	NA	(Linke et al., 2019)
	Comparison baselines			
GRADES-DL	LSTM+RAPID Q	1d / 0.25°	1979–2024	(Y. Yang et al., 2023)
TE	MATSIRO+CaMa-Flood Q	1h / 1 arcmin	2007–2024	(Hibino & Yamamoto, 2018)

National scale experiments

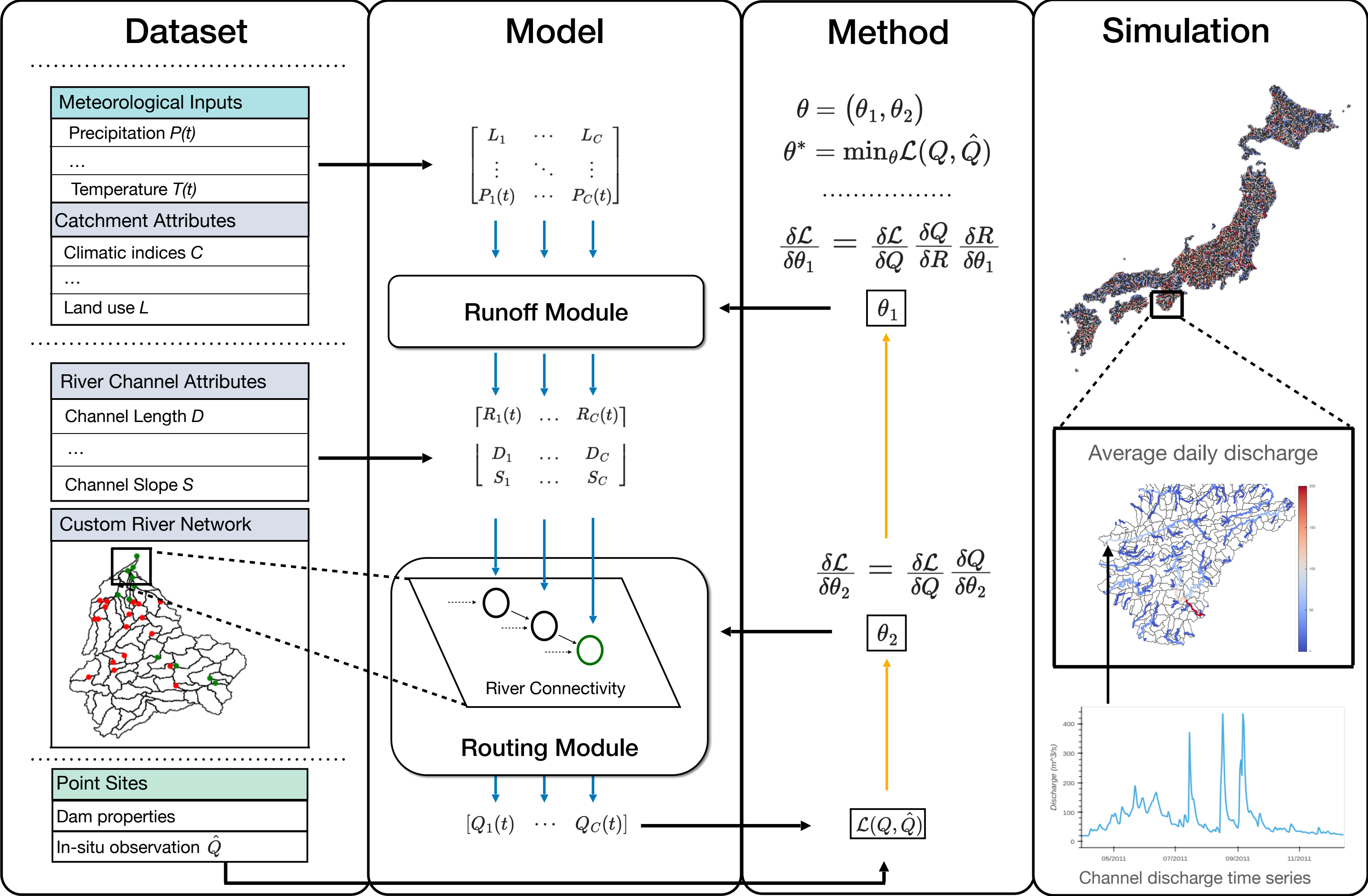


National scale experiments - Model

$$\mathcal{F} : \left(\mathbf{X}^{\text{dyn}}, \mathbf{X}^{\text{run}}, \mathbf{X}^{\text{route}}, \mathcal{E} \right) \rightarrow \left(\hat{\mathbf{Q}}, \hat{\mathbf{R}}, \hat{\mathbf{F}} \right)$$



National scale simulation



National scale simulation - Method

We jointly learn the runoff and routing module from the sparse supervision of monitored catchments

$$\begin{aligned}\hat{\mathbf{Q}} &= \mathcal{F}(\mathbf{X}^{\text{dyn}}, \mathbf{X}^{\text{run}}, \mathbf{X}^{\text{route}}, \mathcal{E}), & \in \mathbb{R}^{C \times T} & \xleftarrow{\hspace{1cm}} & 8,893 \text{ catchments} \\ \hat{\mathbf{Q}}' &= H\hat{\mathbf{Q}}, & \in \mathbb{R}^{C' \times T} & \xleftarrow{\hspace{1cm}} & 314 \text{ monitored catchments} \\ \mathcal{L} &= \text{NSE}(\hat{\mathbf{Q}}', \mathbf{Q})\end{aligned}$$

The loss is minimized by gradient descent. Parameter gradient are computed by automatic differentiation

$$(\theta_1^*, \theta_2^*) = \arg \min_{\theta_1, \theta_2} \mathcal{L}(\theta_1, \theta_2)$$

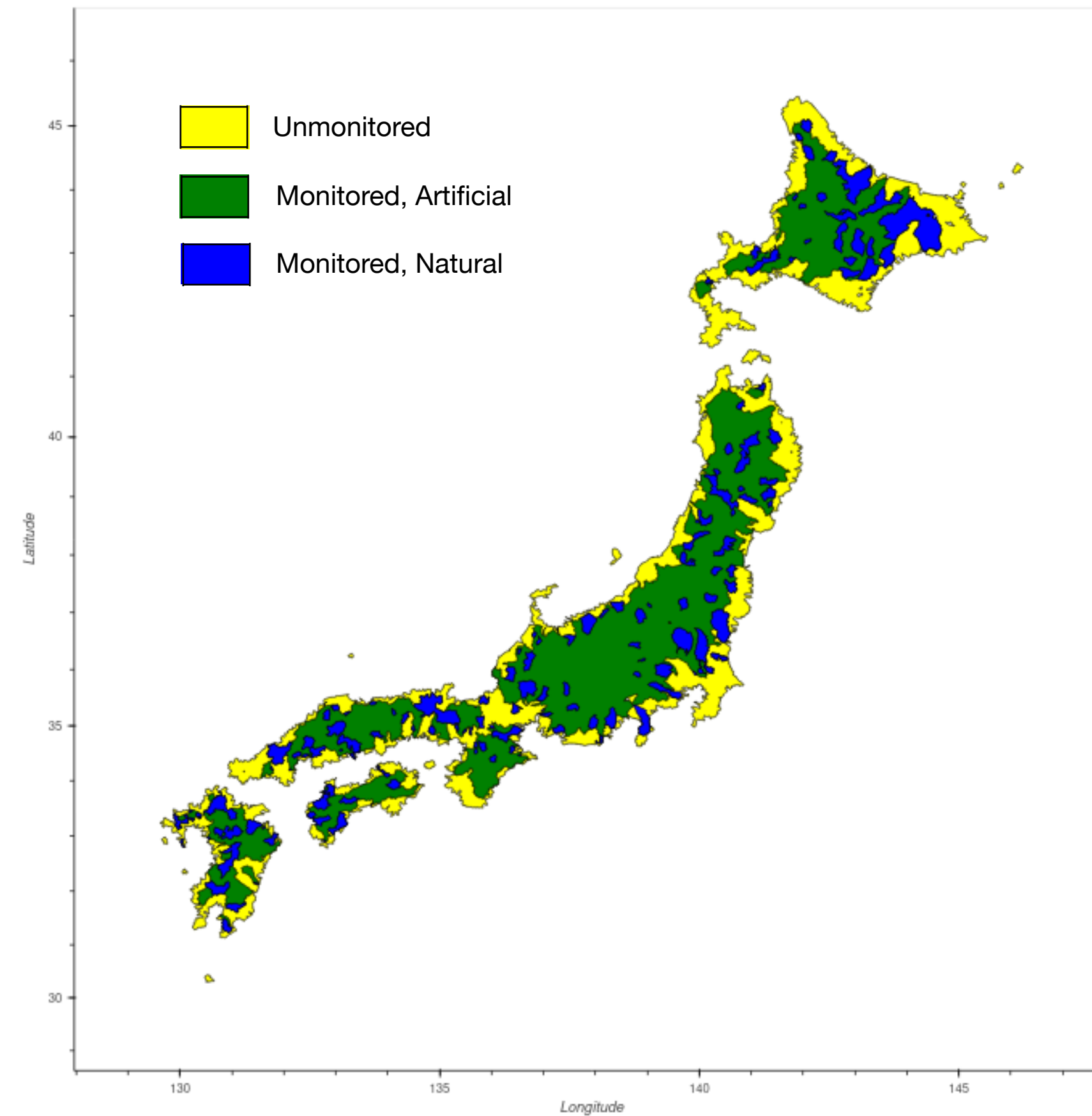
$$\theta_1^{(k+1)} = \theta_1^{(k)} - \eta \frac{\partial \mathcal{L}}{\partial \theta_1}, \quad \theta_2^{(k+1)} = \theta_2^{(k)} - \eta \frac{\partial \mathcal{L}}{\partial \theta_2}$$

Although the supervision is sparse, it informs the full domain because the runoff and routing models are shared among all catchments

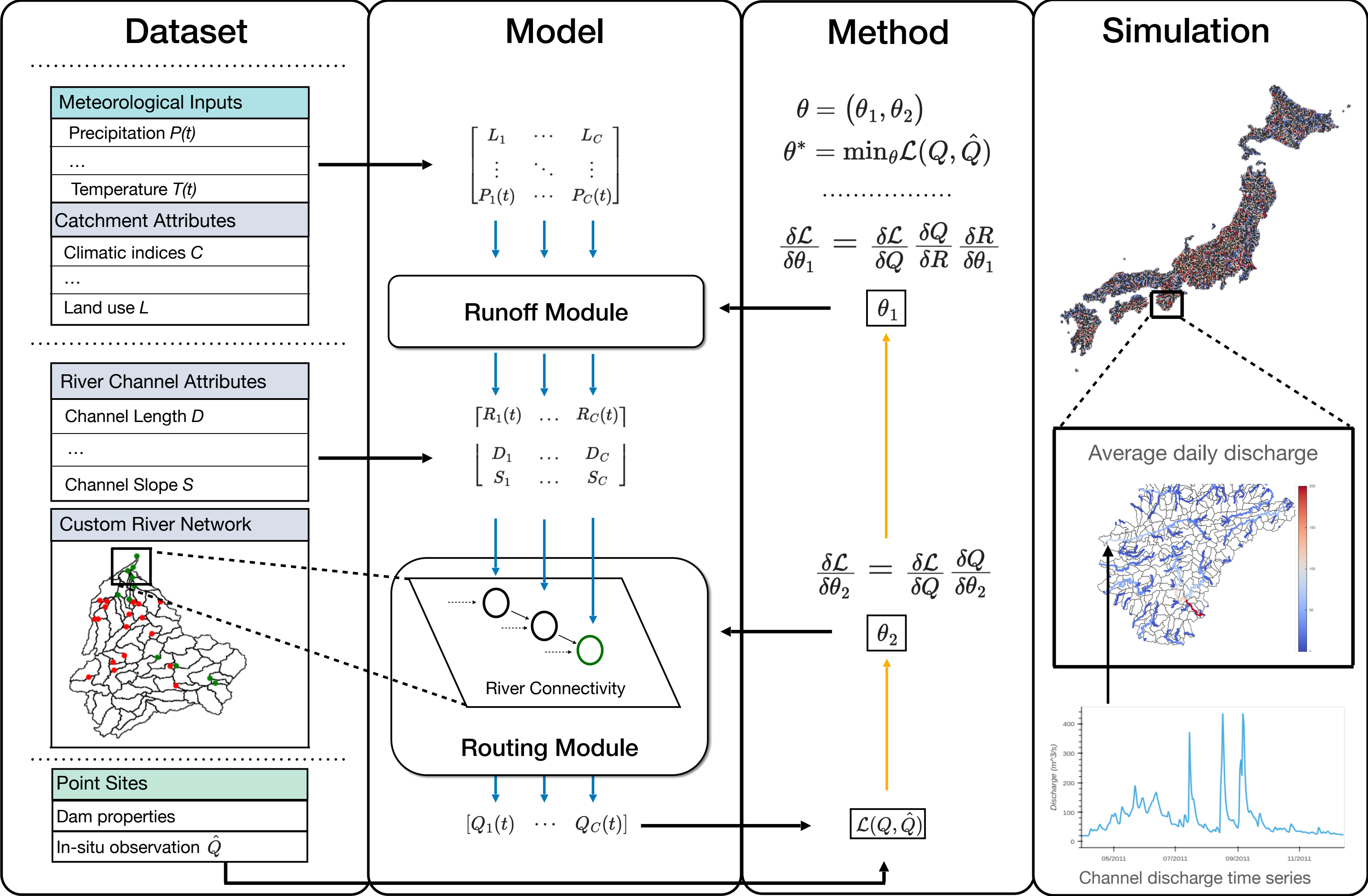
National scale simulation - Method

Training: We train the model on a carefully selected subset of 314 gauges to avoid biases due to dams, etc.

Evaluation: We evaluate the model on all monitored catchments (914 gauges)



National scale simulation



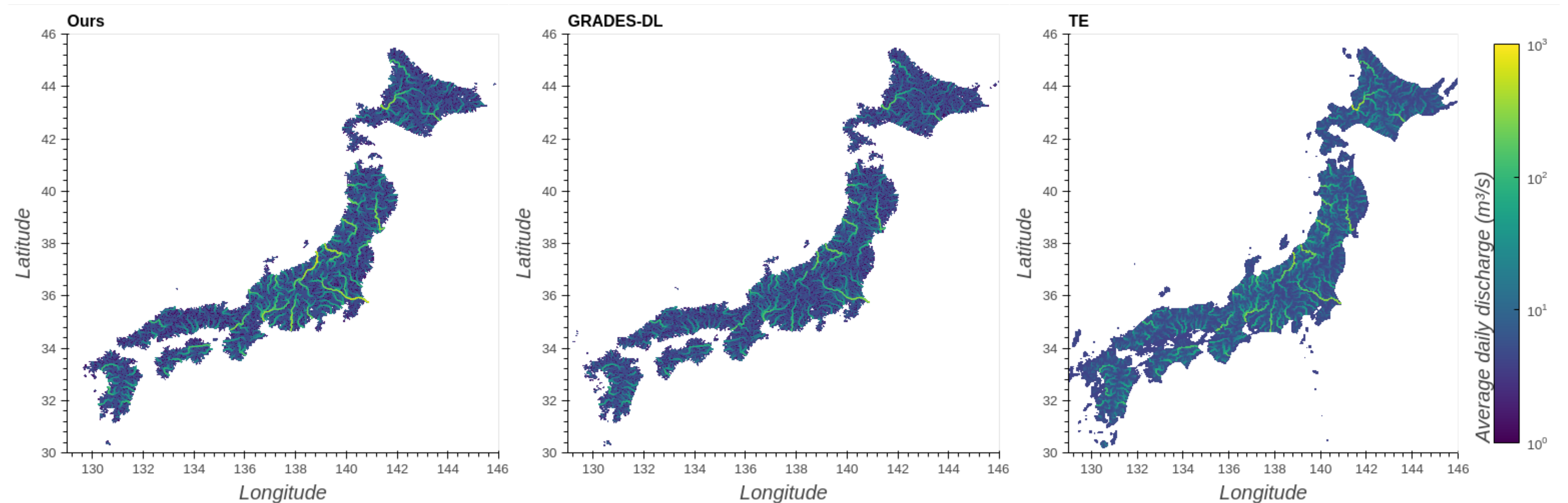
National scale simulation

- **Simulated variables:** Discharges (m^3/s), Runoff (mm), celerity and diffusivity
- **Temporal resolution:** Daily
- **Temporal range:** January 1st 2009 to December 31st 2021
- **Spatial resolution:** 40 km^2 average catchment area (total 8,893 catchments)
- **Spatial range:** Honshu, Hokkaido, Kyushu, Shikoku
- **Evaluation setting:** Report median NSE on ungauged basins (915 location sites)

National scale experiments - Results

Compare the accuracy of the proposed simulation against observations with two existing products:

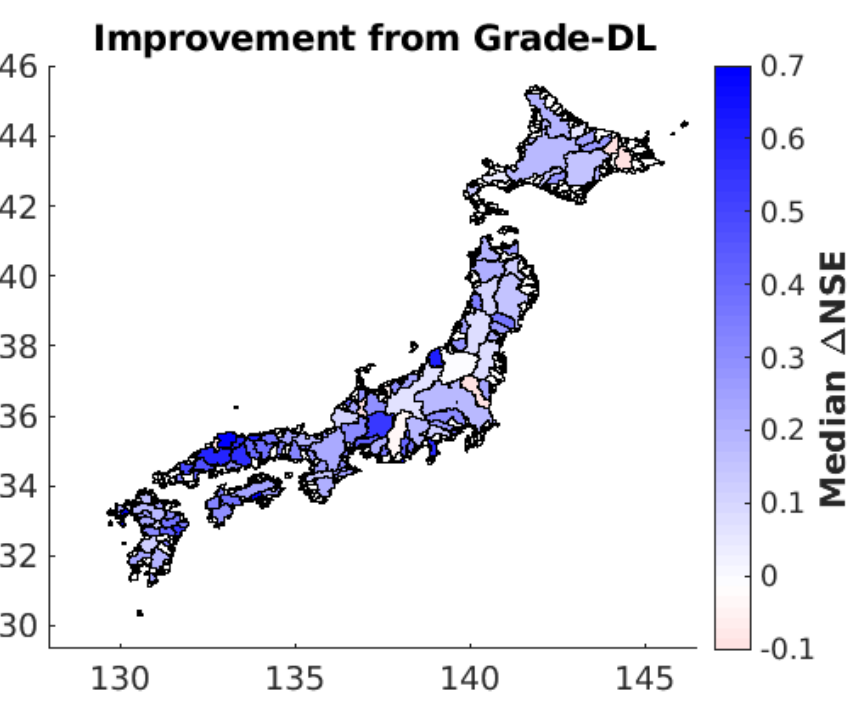
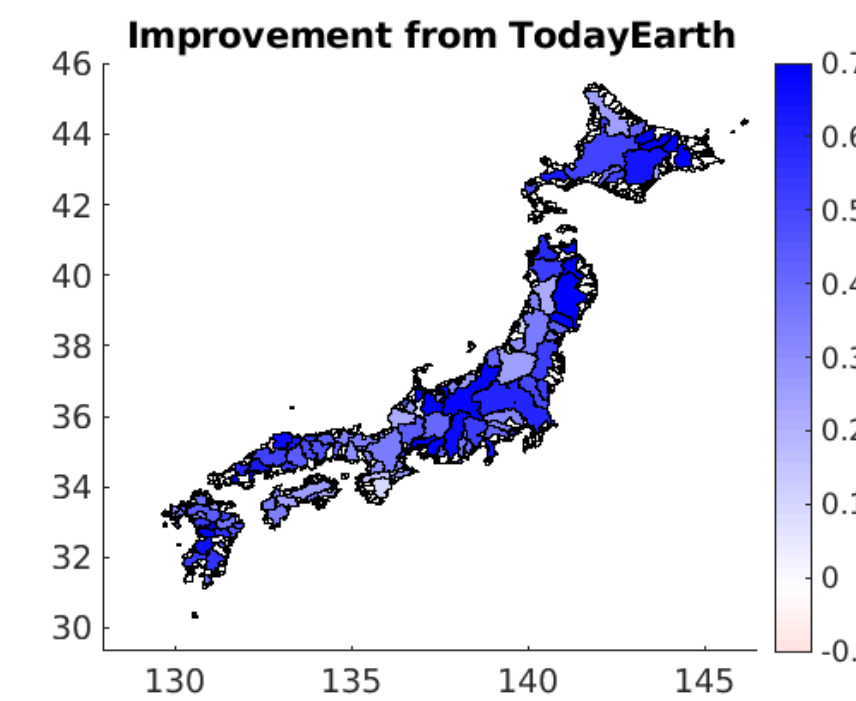
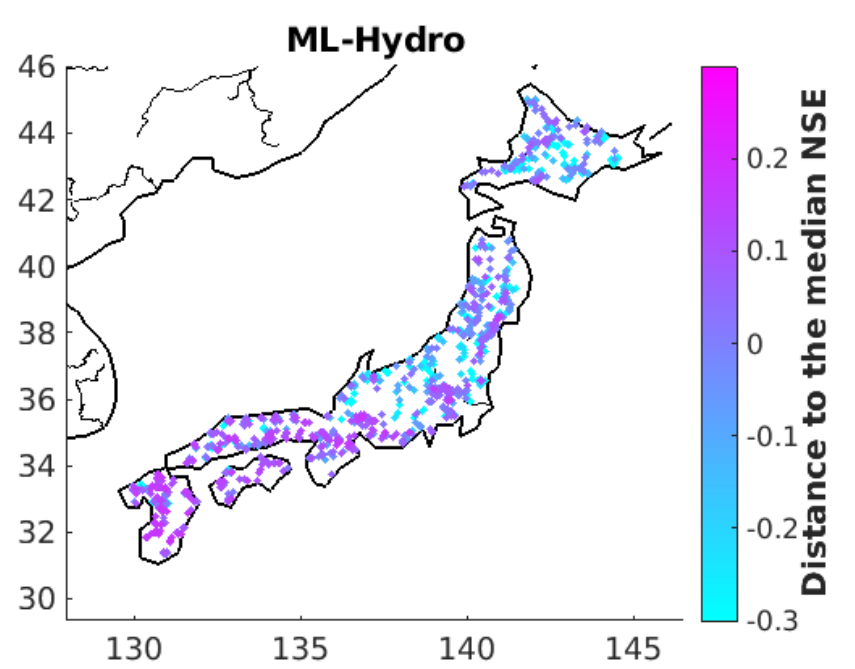
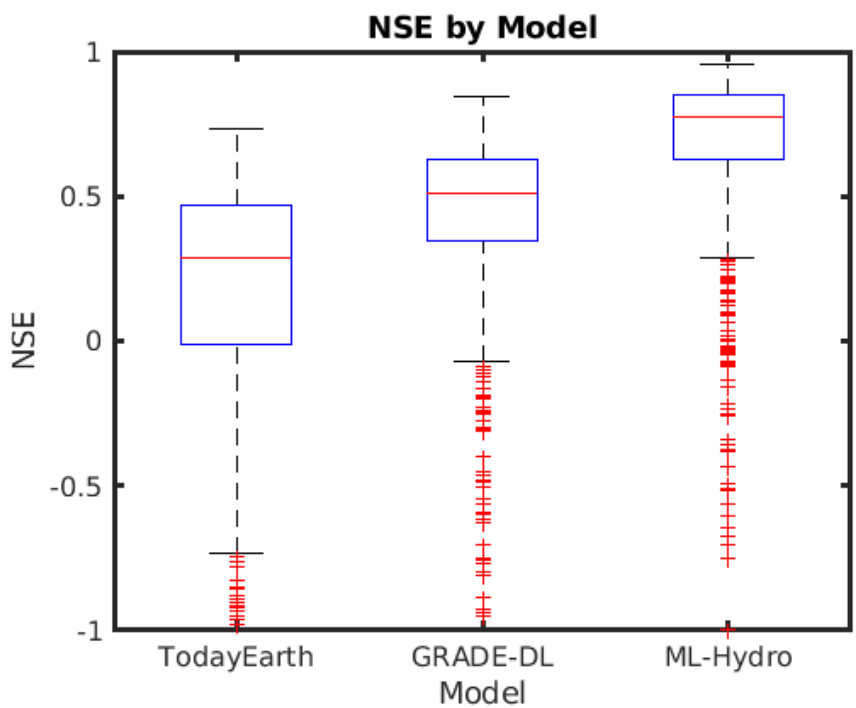
- **Proposed simulation:** E2EL (LSTM + DiffRoute) with regional high quality forcings (JMA-MSM, gauge adjusted radar)
- **Grades-DL:** ML Runoff (LSTM) + physical routing (RAPID) with global forcings (MSWEP, ERA5)
- **Today Earth** (Japan): Physical models (Matsiro + CamaFlood) near real-time forcings (amedas, JMA MSM)



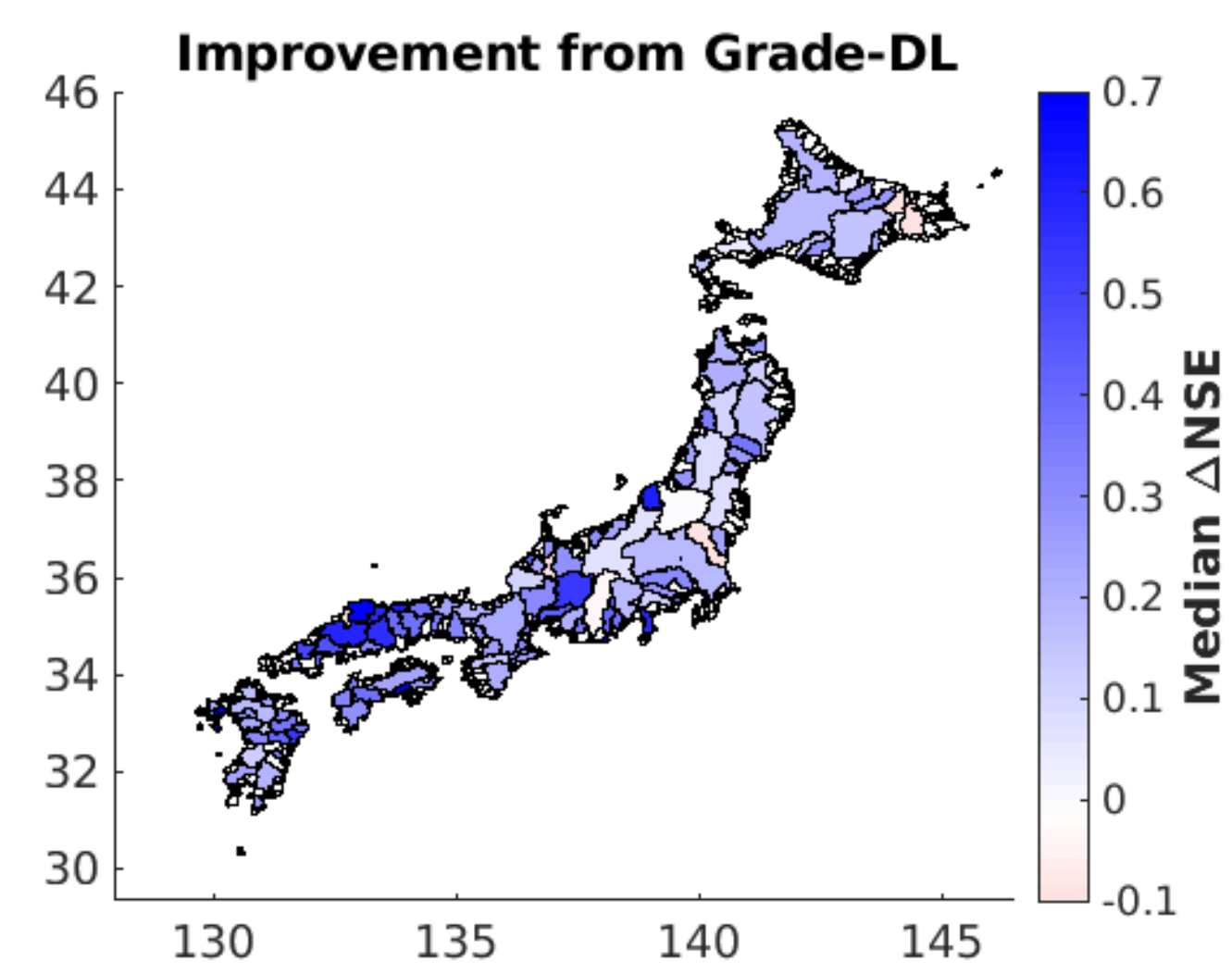
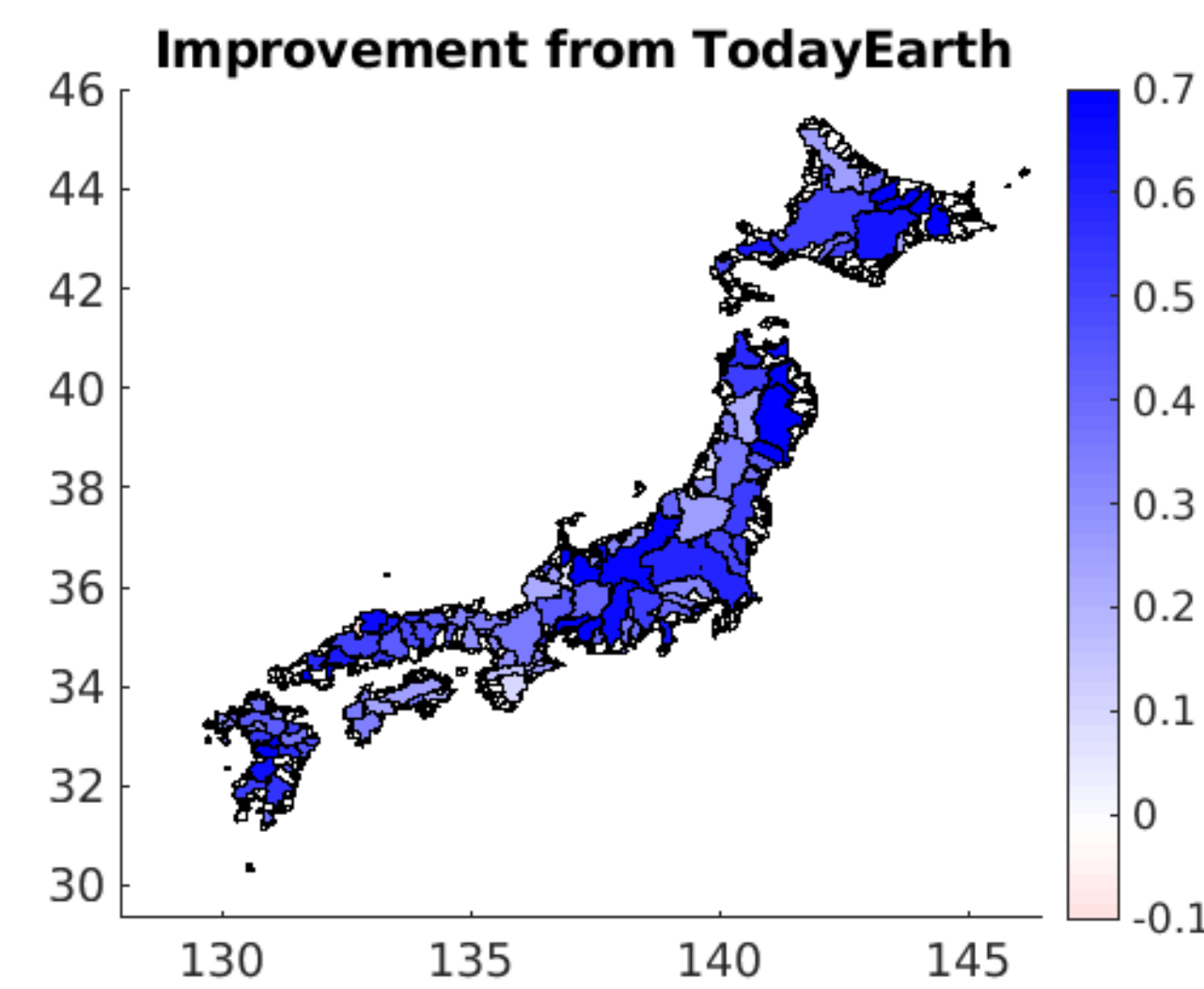
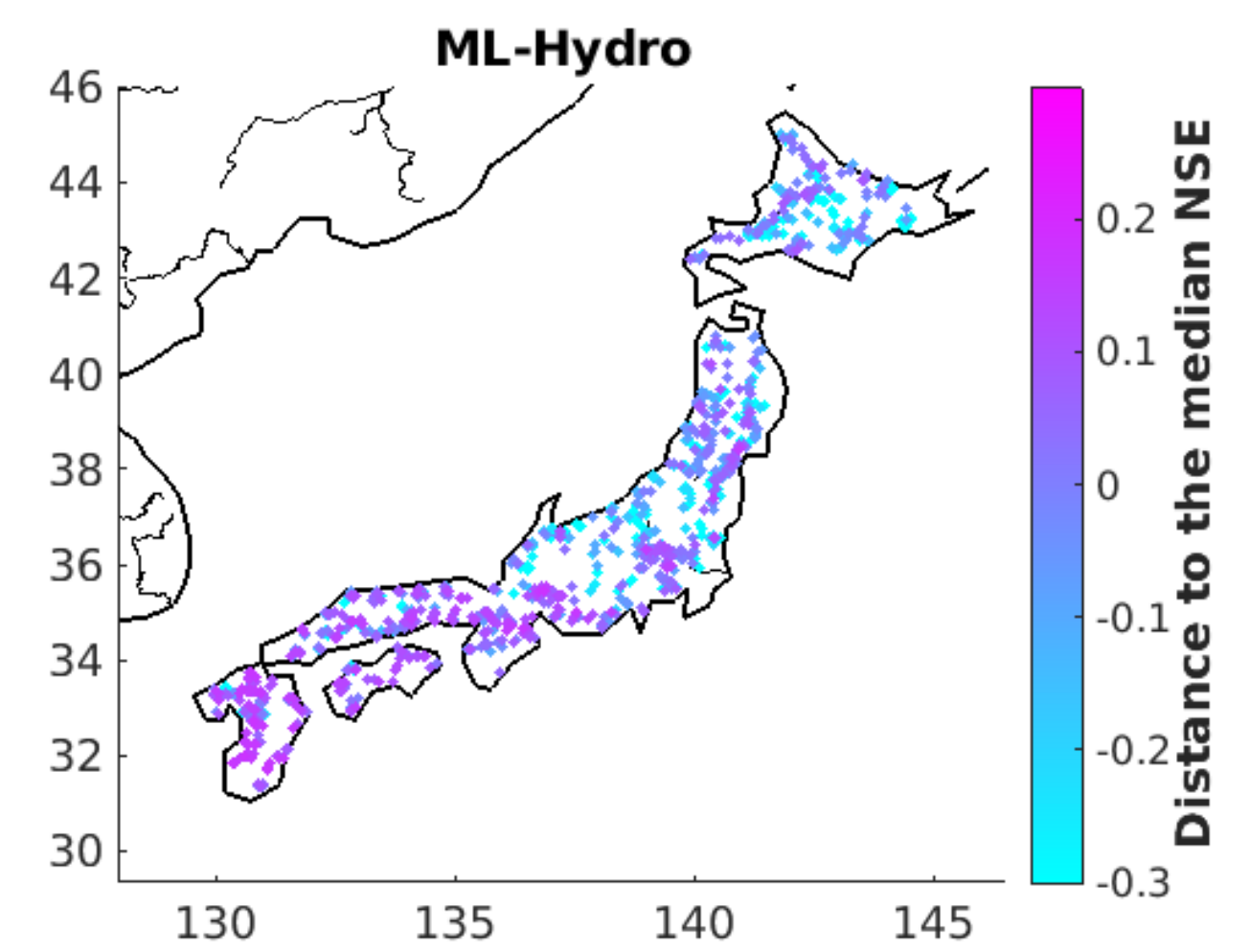
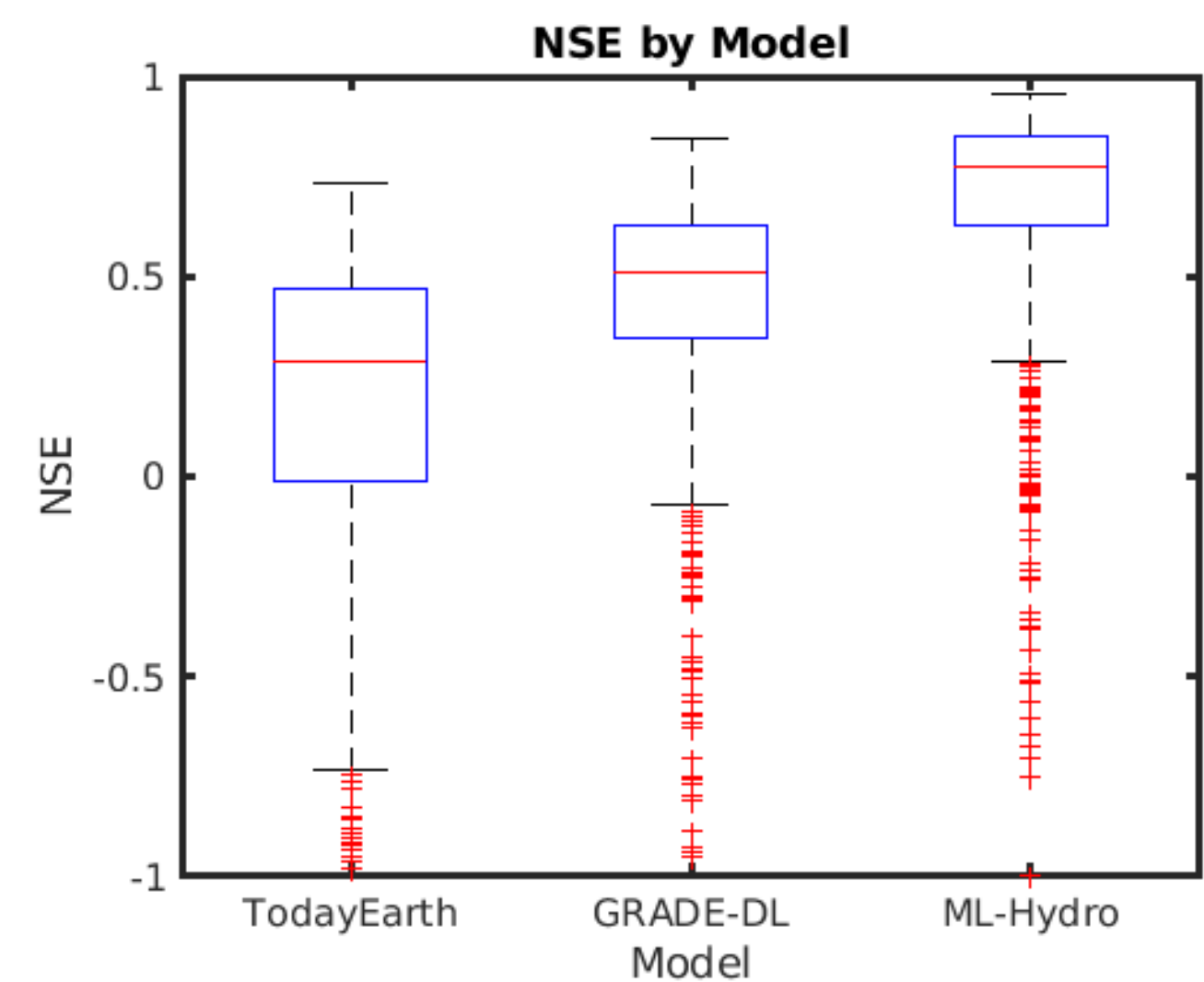
National scale experiments - Results

We compare our model results to **TodayEarth** and **GradesDL** river discharge outputs on our proposed dataset.

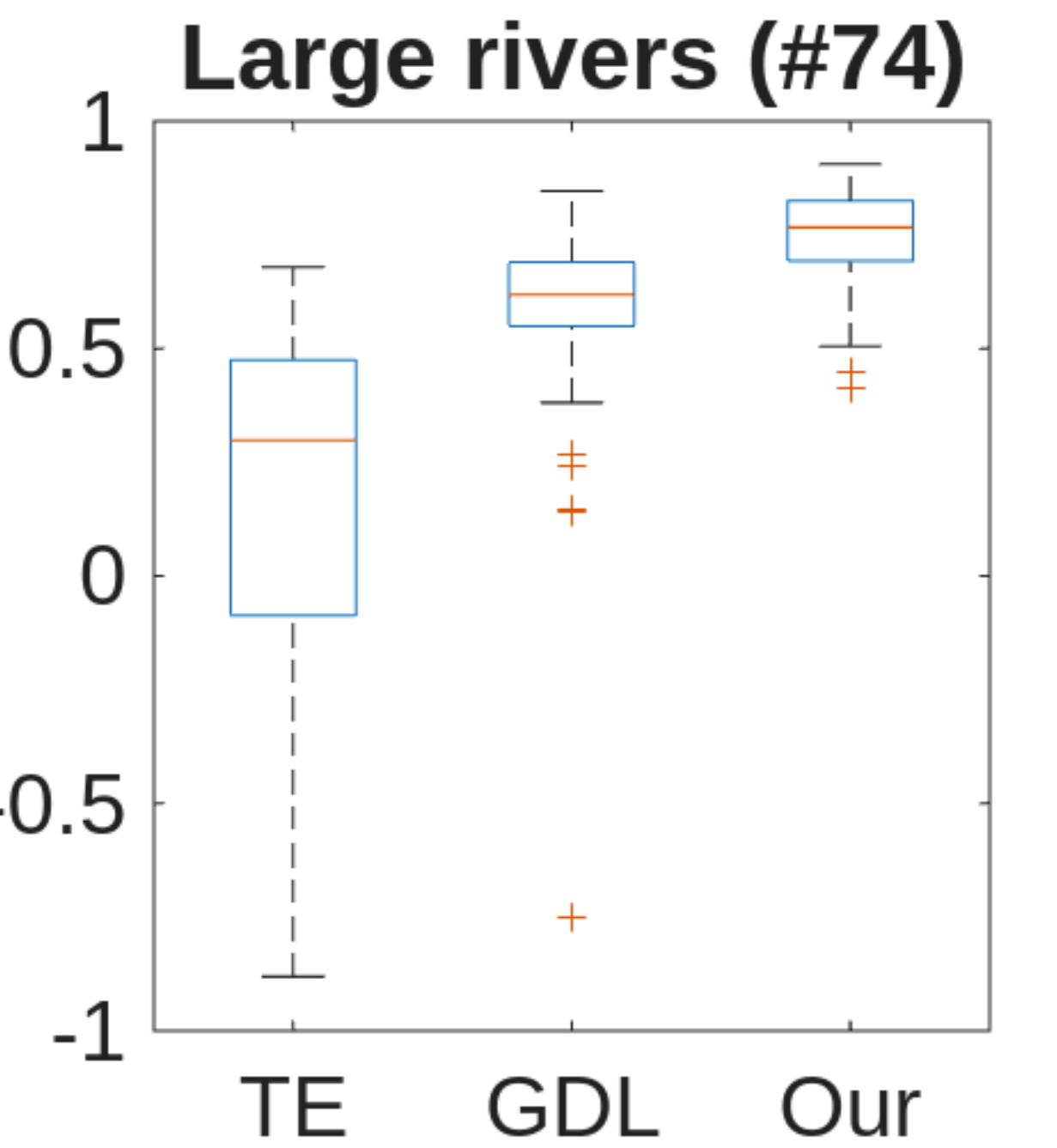
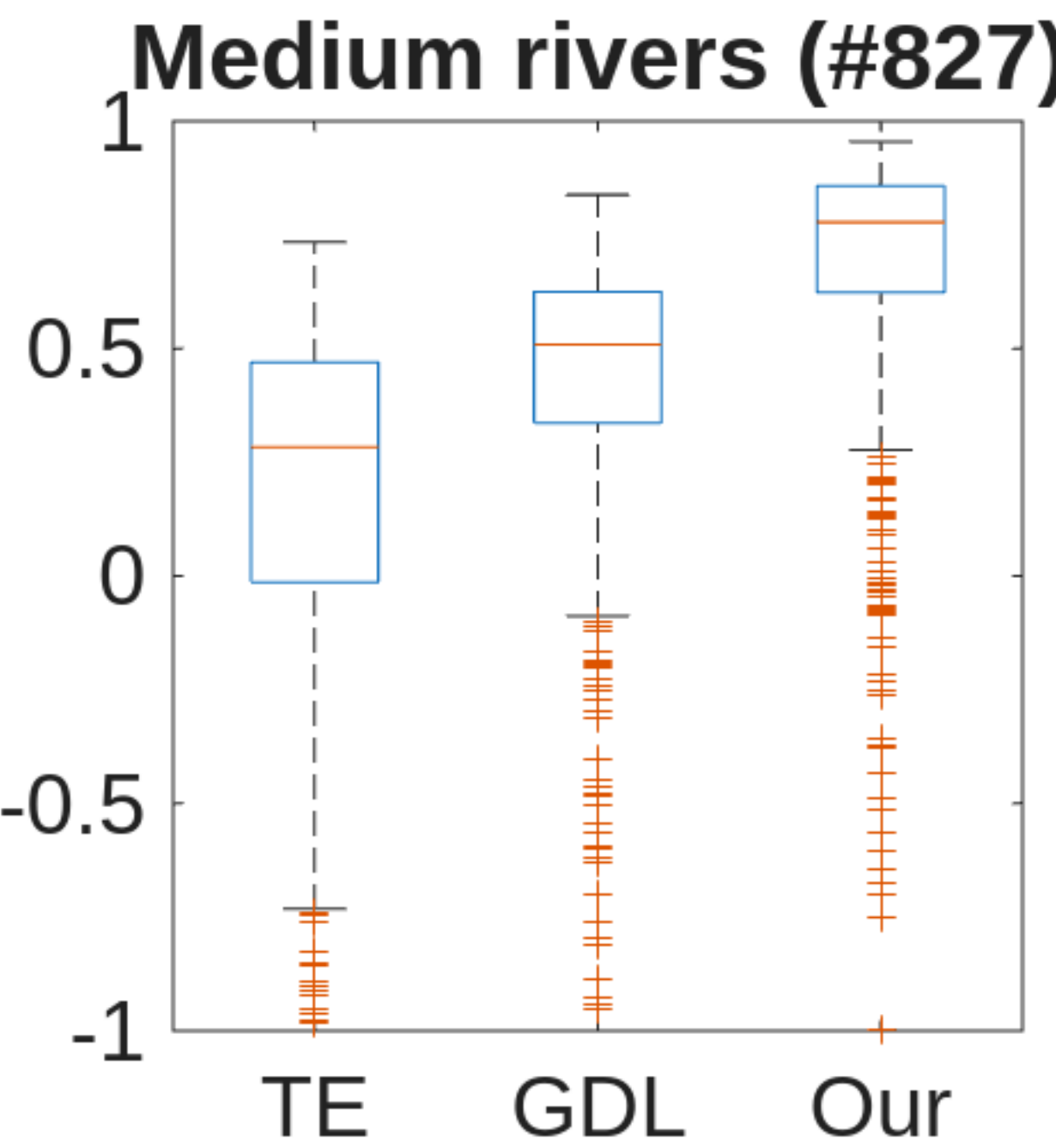
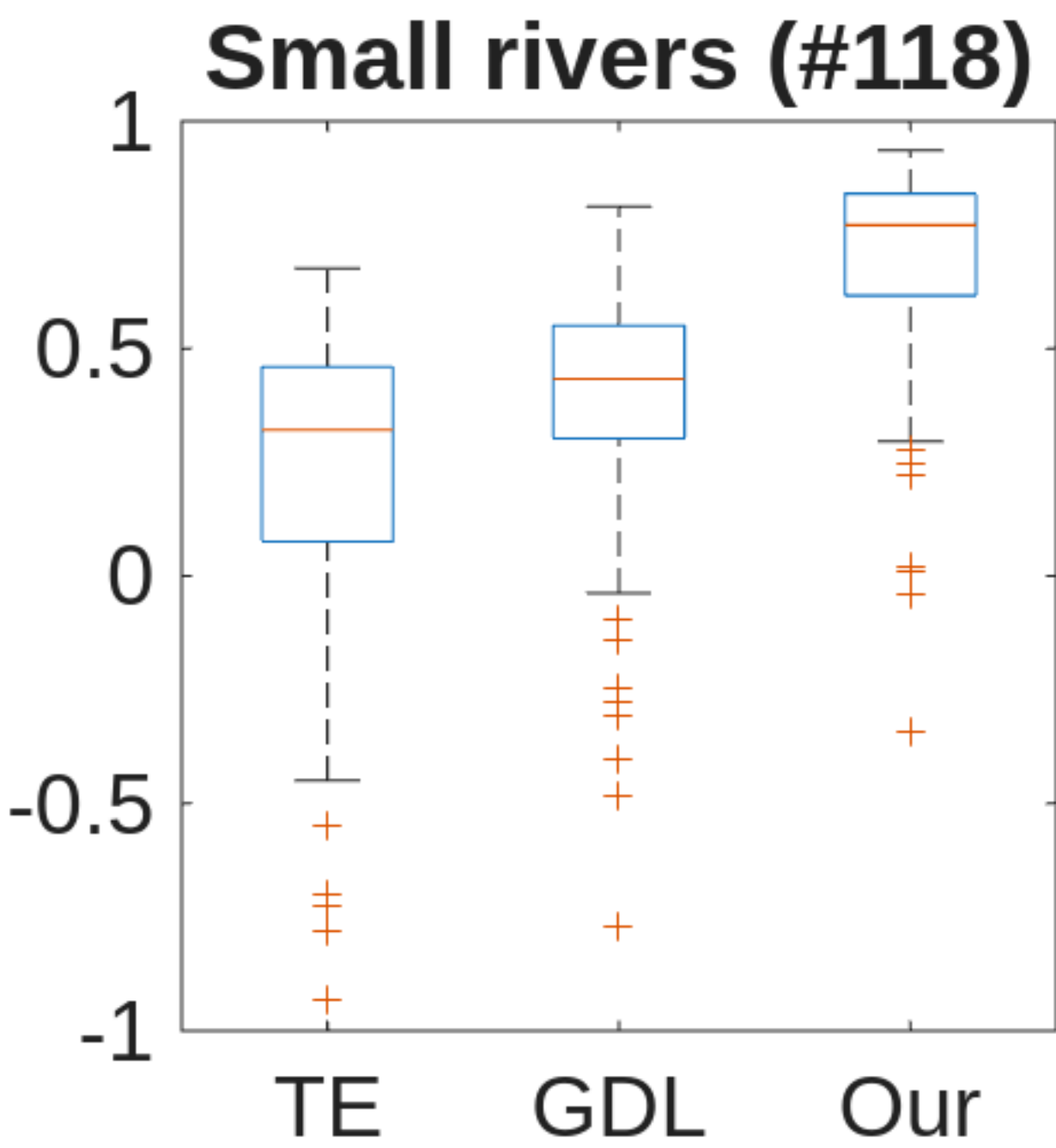
Models	NSE	KGE'				nRMSD
		α	γ	β	KGE'	
TE	0.29	0.71	1.58	0.62	0.21	1.35
GRADES-DL	0.51	0.78	0.81	0.91	0.57	1.07
Ours	0.80	0.93	0.93	1.06	0.73	0.69



National scale experiments - Results

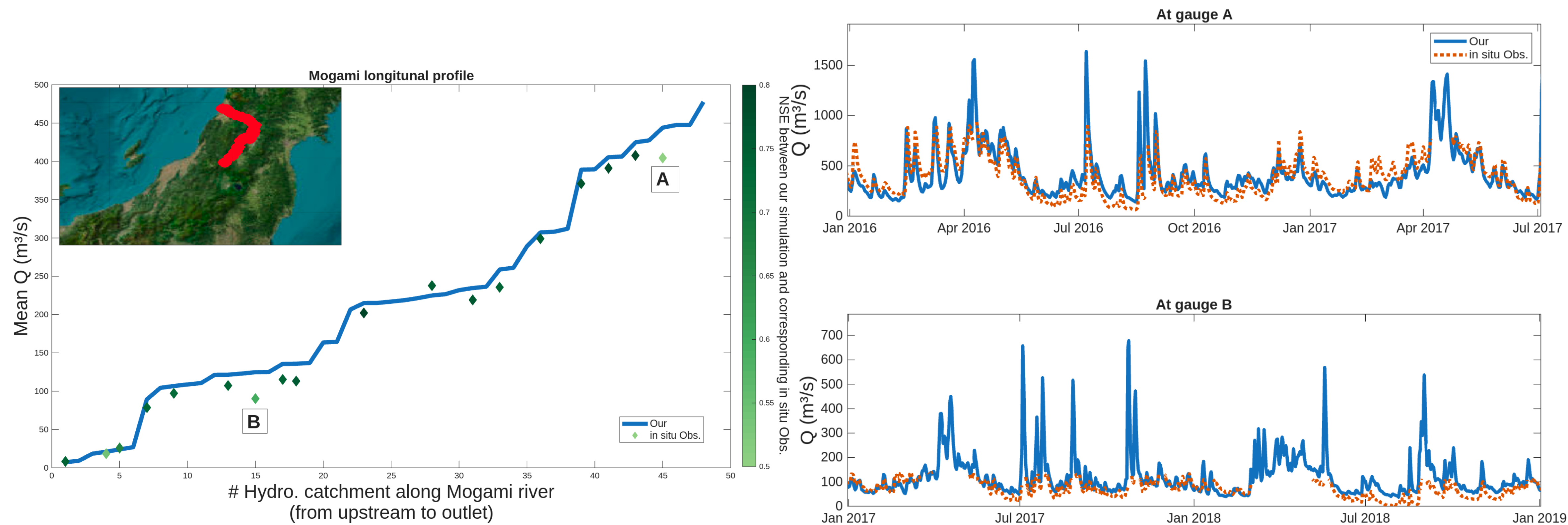


National scale experiments - Results



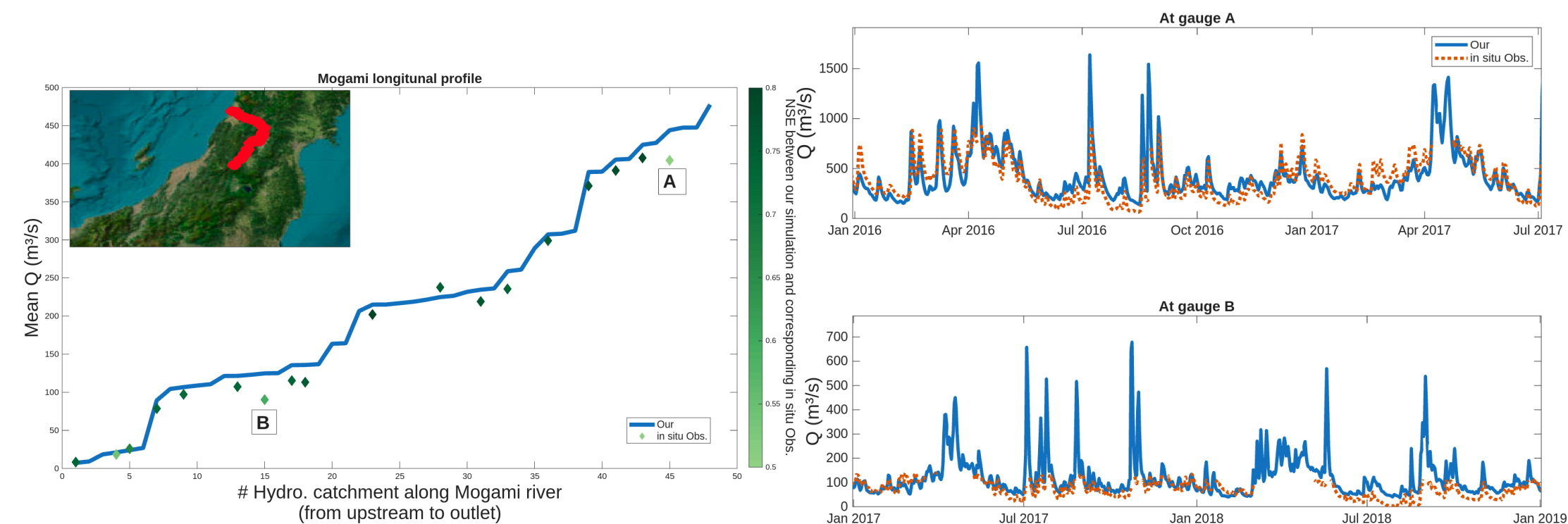
National scale experiments - Results

The simulation achieves both high accuracy and water mass conservation



National scale experiments - Results

The simulation achieves both high accuracy and water mass conservation



High accuracy + spatial consistency allows to detect incomplete observations.

Can be used as a diagnostic tool for the observation network

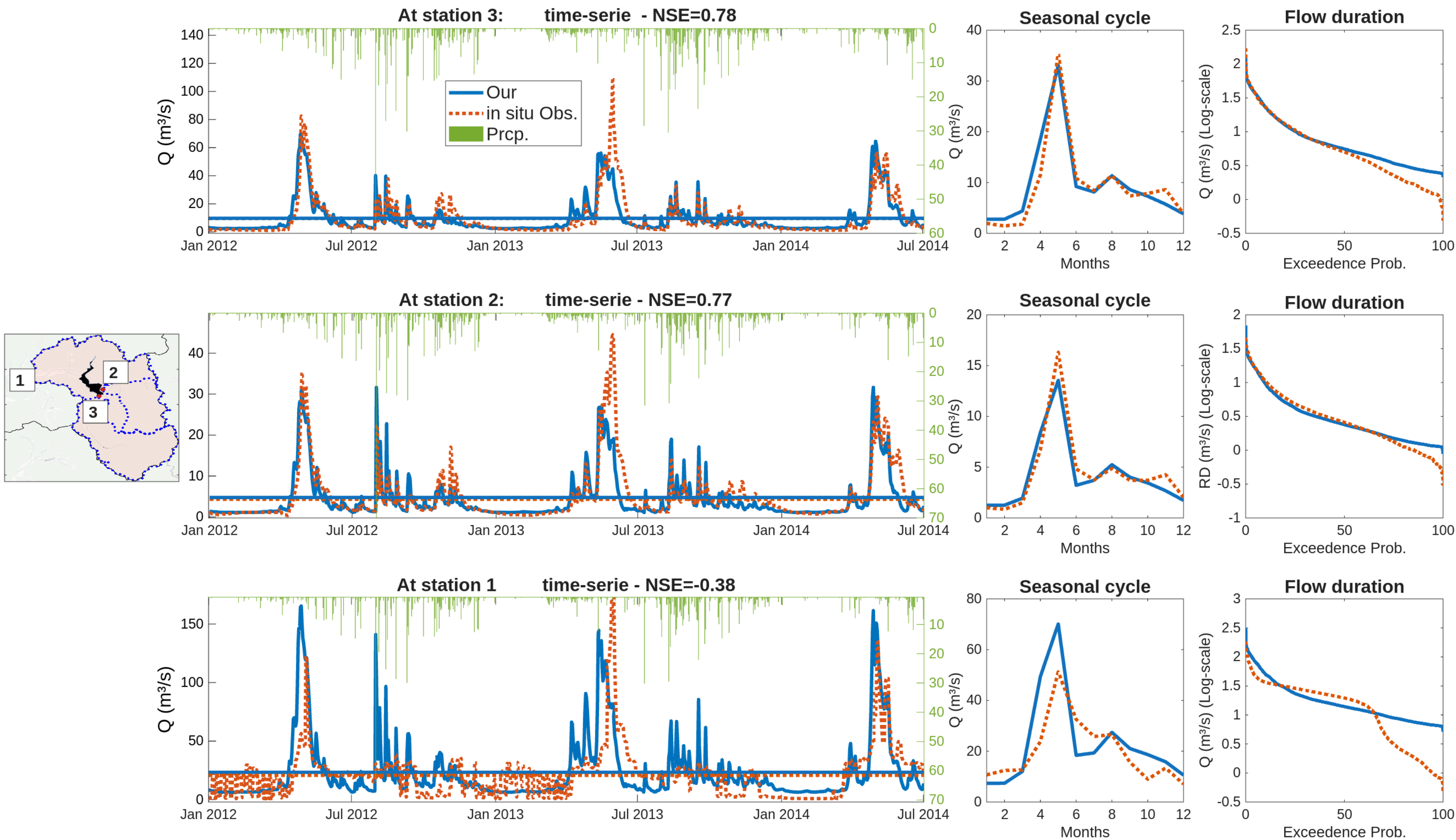
May be used to extrapolate the range of existing rating curves

National scale experiments - Error Regimes

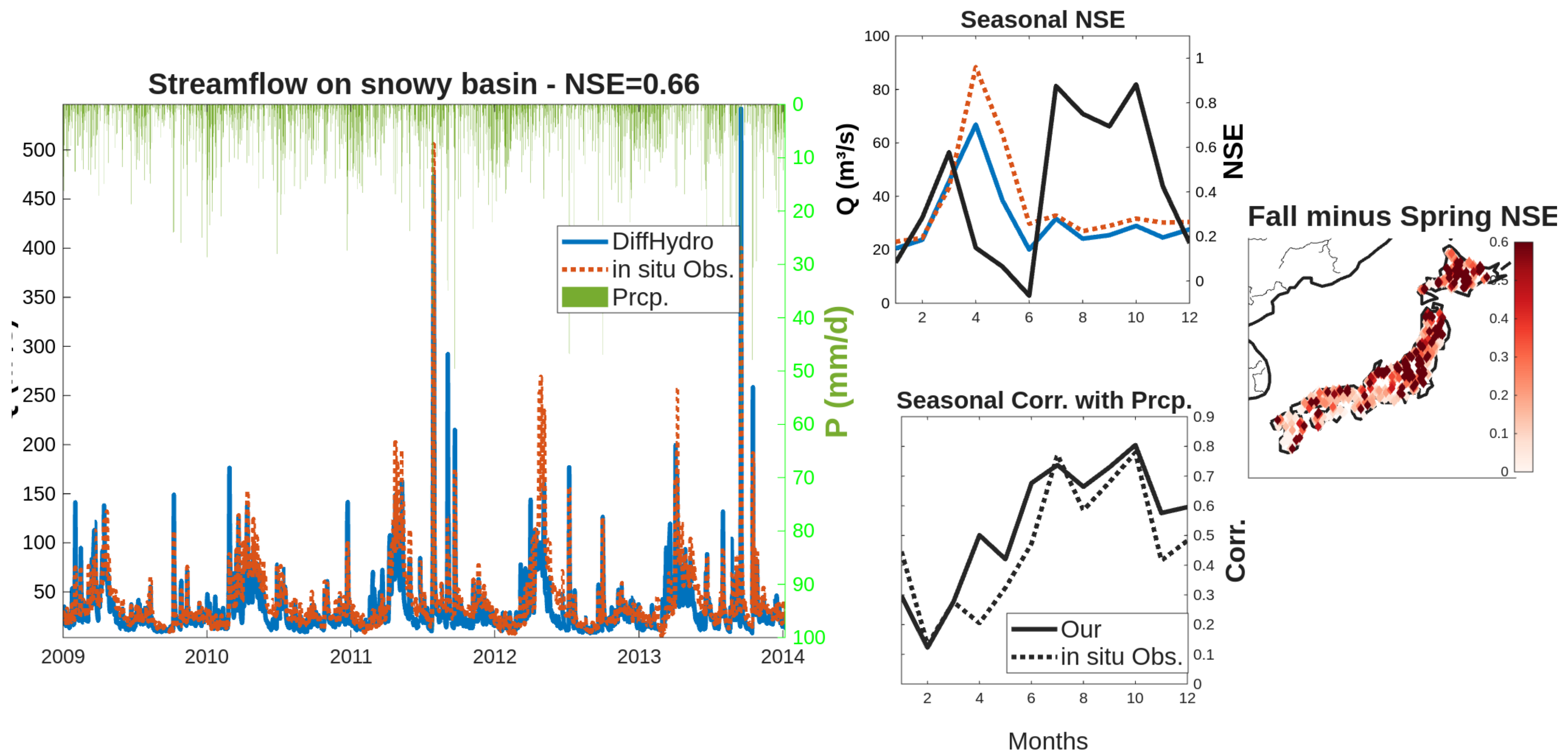
Remaining errors focus on three main components:

- **Dam impact:** Among the 147 gauges with $NSE < 0.3$, we estimate roughly 100 are due to dam operation impacts
- Surface: **Snow melt** (and wetlands) is poorly modeled, leading to clear degradation of accuracy in spring season
- **Unmapped infrastructure:** The river network is an idealized representation of the river structure lacking important artificial and natural waterways impacting streamflows

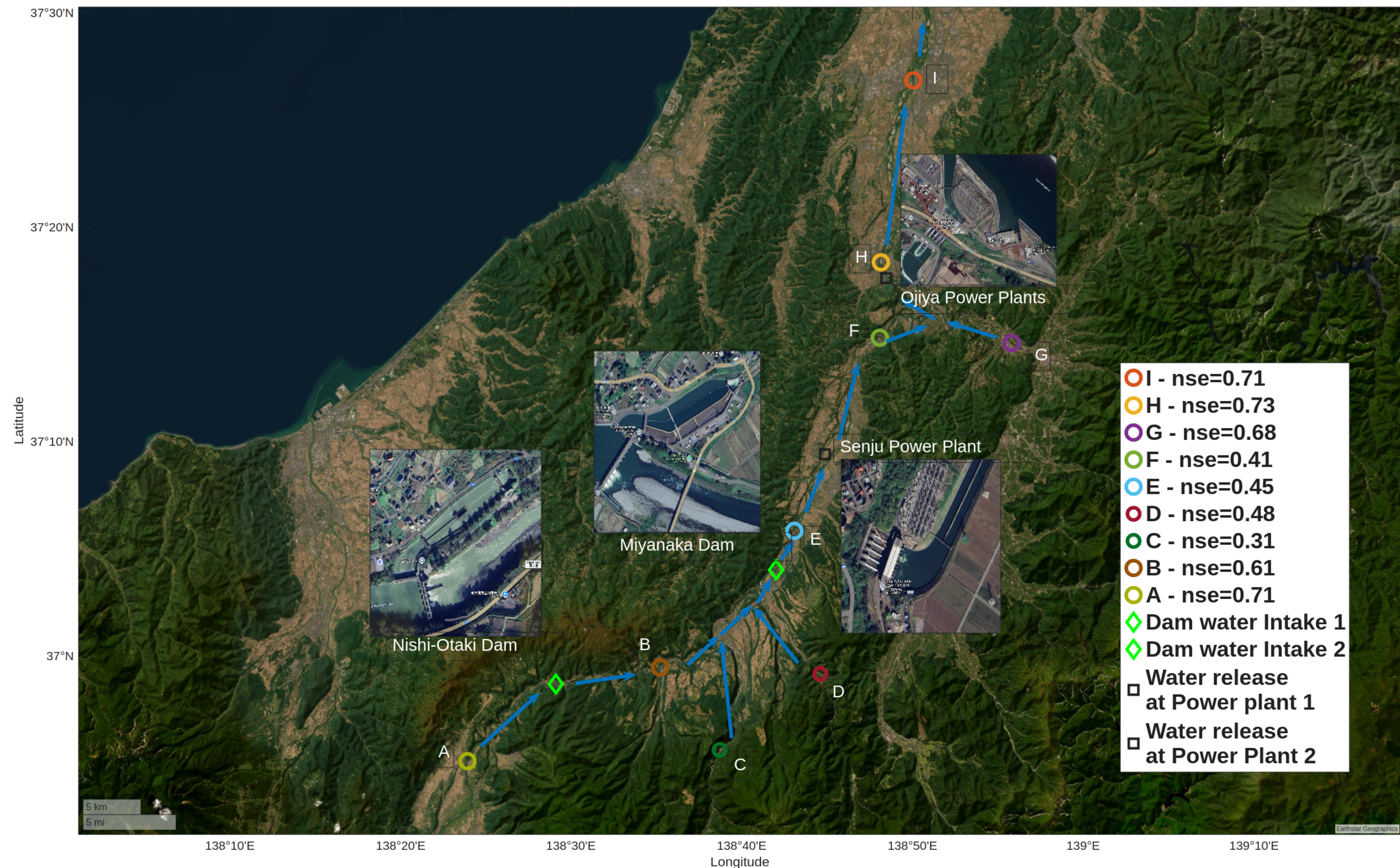
National scale experiments - Dam Impact



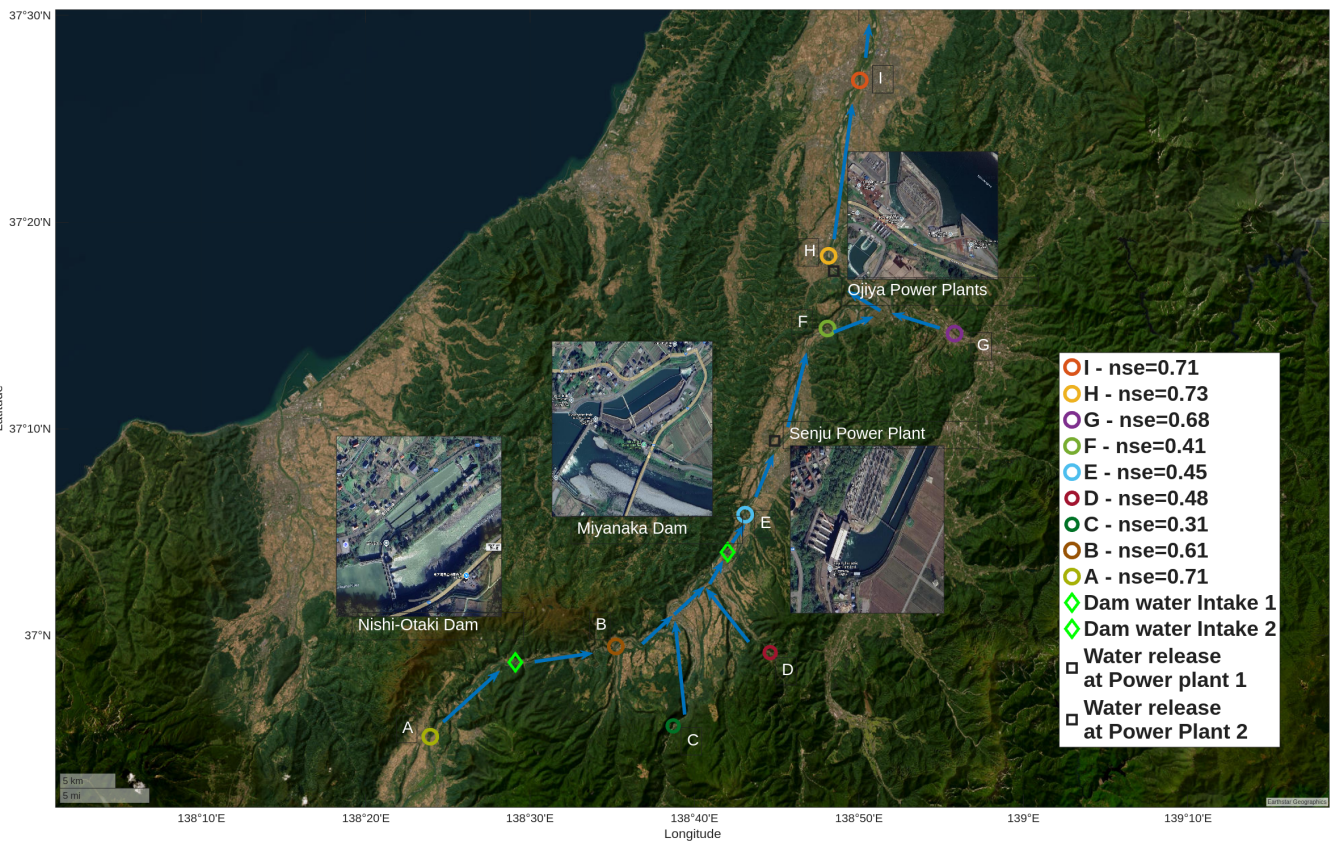
National scale experiments - Snow Melt



National scale experiments - Unmapped waterways

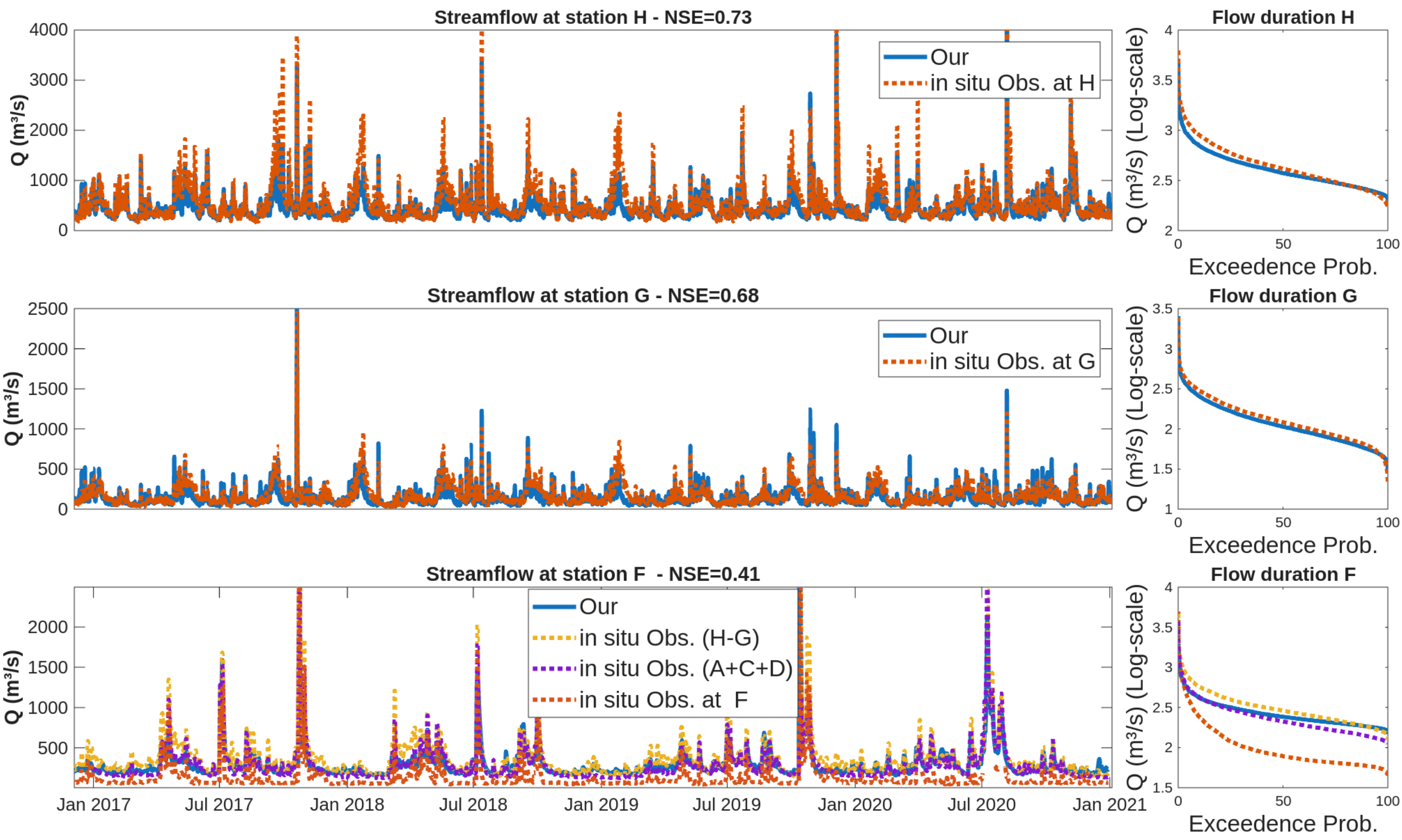


National scale experiments - Unmapped waterways



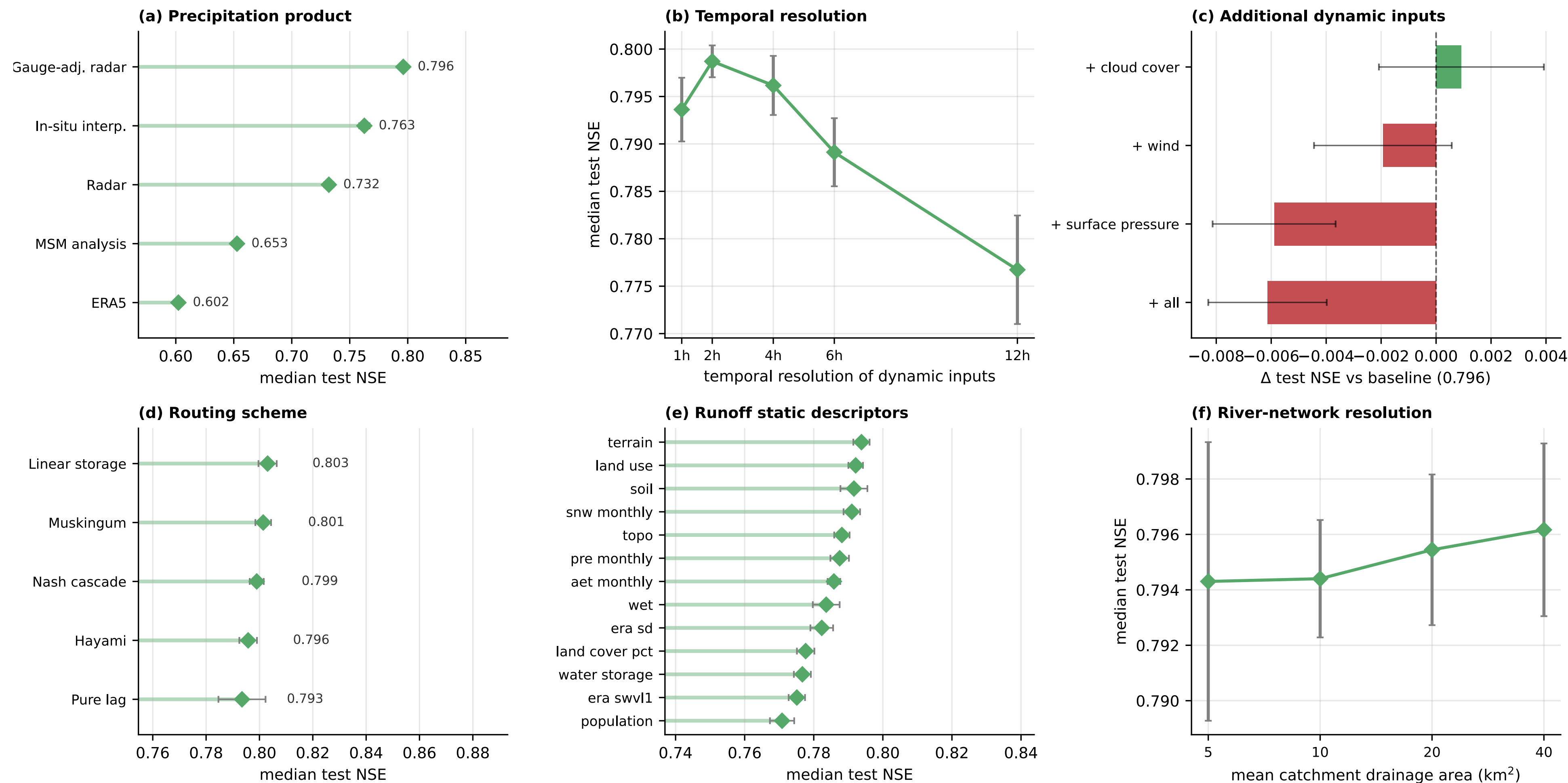
Complexity of real infrastructure:
unmapped artificial waterways impact
natural rivers streamflow

High accuracy + spatial consistency
allows us to **detect** and **diagnose** these
impacts (sometimes)



National scale experiments - Ablation study

Sensitivity of river-discharge accuracy to model and input factors (median NSE over ungauged test stations)



National scale experiments - Recap

This approach consists in *jointly designing* the **dataset**, **model** and **method** to produce a river discharge **simulation** with *high accuracy* and *physical consistency*

It has *important limitations*: dams, snow melt and missing water ways

We would appreciate any feedback and advice to help us improve missing processes

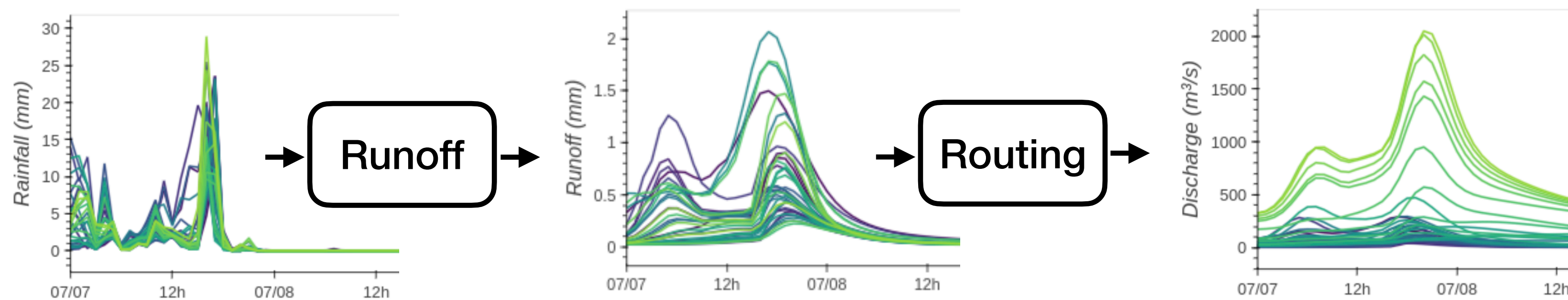
To the best of our knowledge is was the first demonstration of successful national-scale end-to-end learning of physically consistent hydrological processes

Outline

- 1. Differentiable routing (DiffRoute) **enables** end-to-end learning (E2EL)
 - *“Differentiable river routing for end-to-end learning of hydrological processes”, AGU JGR Machine Learning and Computations, 2026*
- 2. E2EL produces **high accuracy** predictions with **physically consistent spatial distribution**
 - *“Leveraging Japan's streamflow records for end-to-end learning of hydrological simulations at national scale”, Water Resource Research, 2026, (preprint)*
- 3. E2EL allows to **identify physically meaningful** dynamic parameters
 - *“End-to-end learning from sparse and noisy observations mostly resolves the equifinality of distributed river-discharge models over Japan”, ECMWF workshop, in preparation*

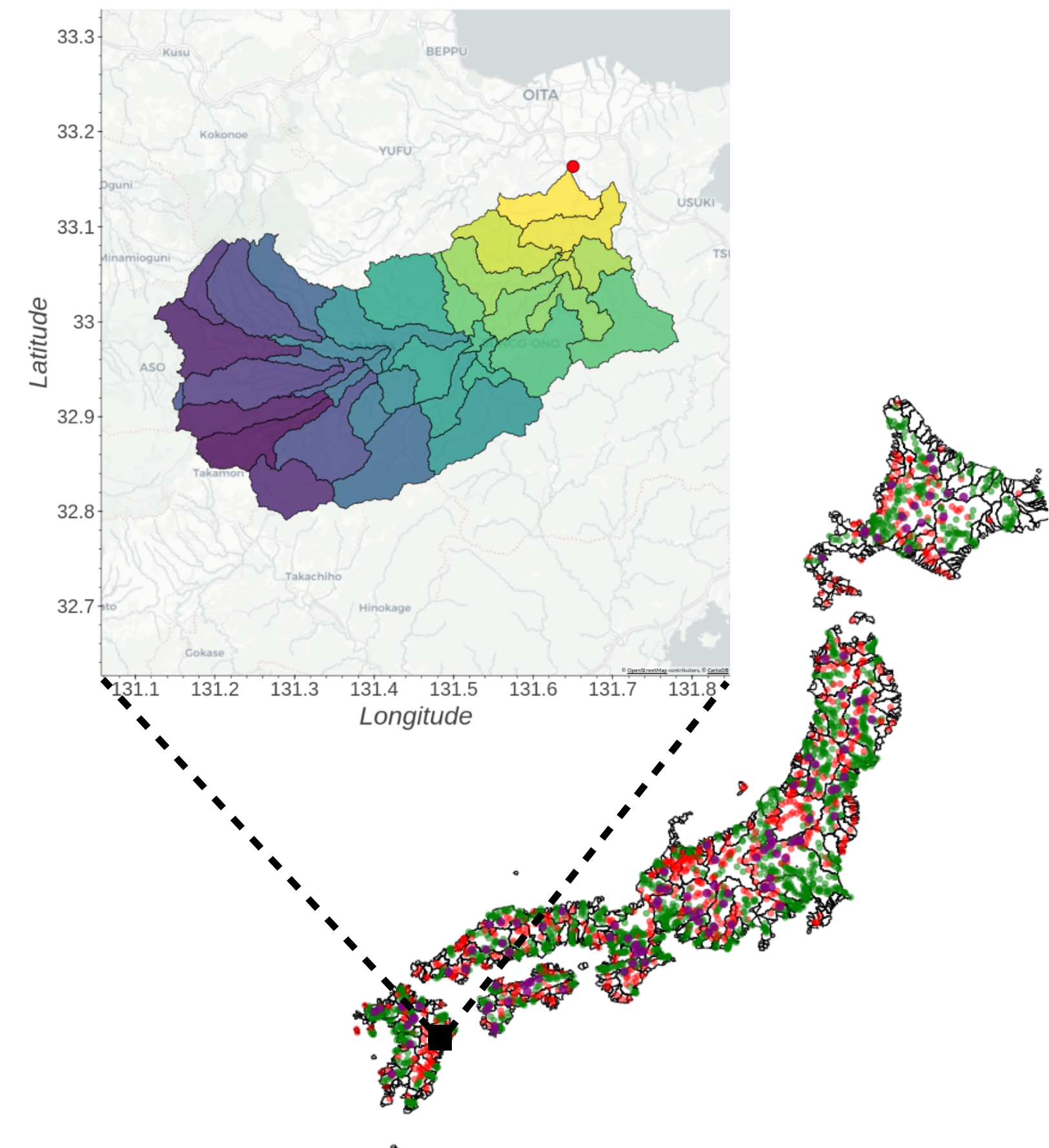
Scientific Question

Can **end-to-end learning** from **sparse and noisy** downstream streamflow observations resolve the **equifinality** of **spatially distributed** river discharge prediction models?



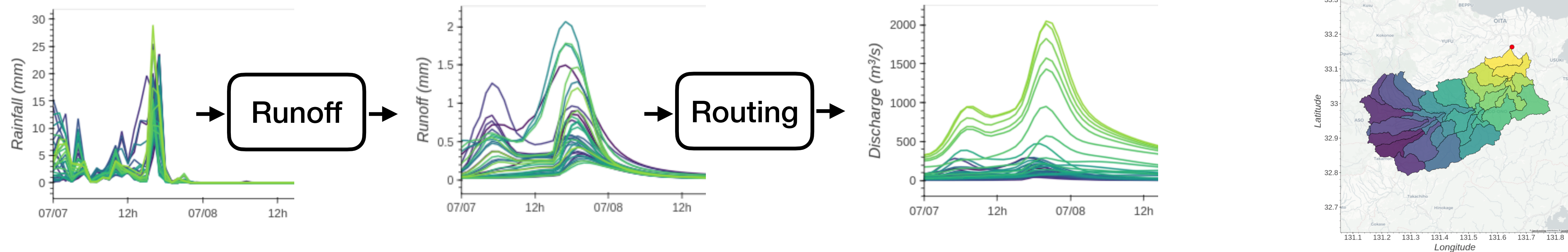
Equifinality: Different upstream parameters values result in similar downstream outputs.

- We showed that the simulation provide high accuracy
- But does it learn the “correct” physical dynamics?



Equifinality

We argue that we only need to show that the routing dynamics is “correct”



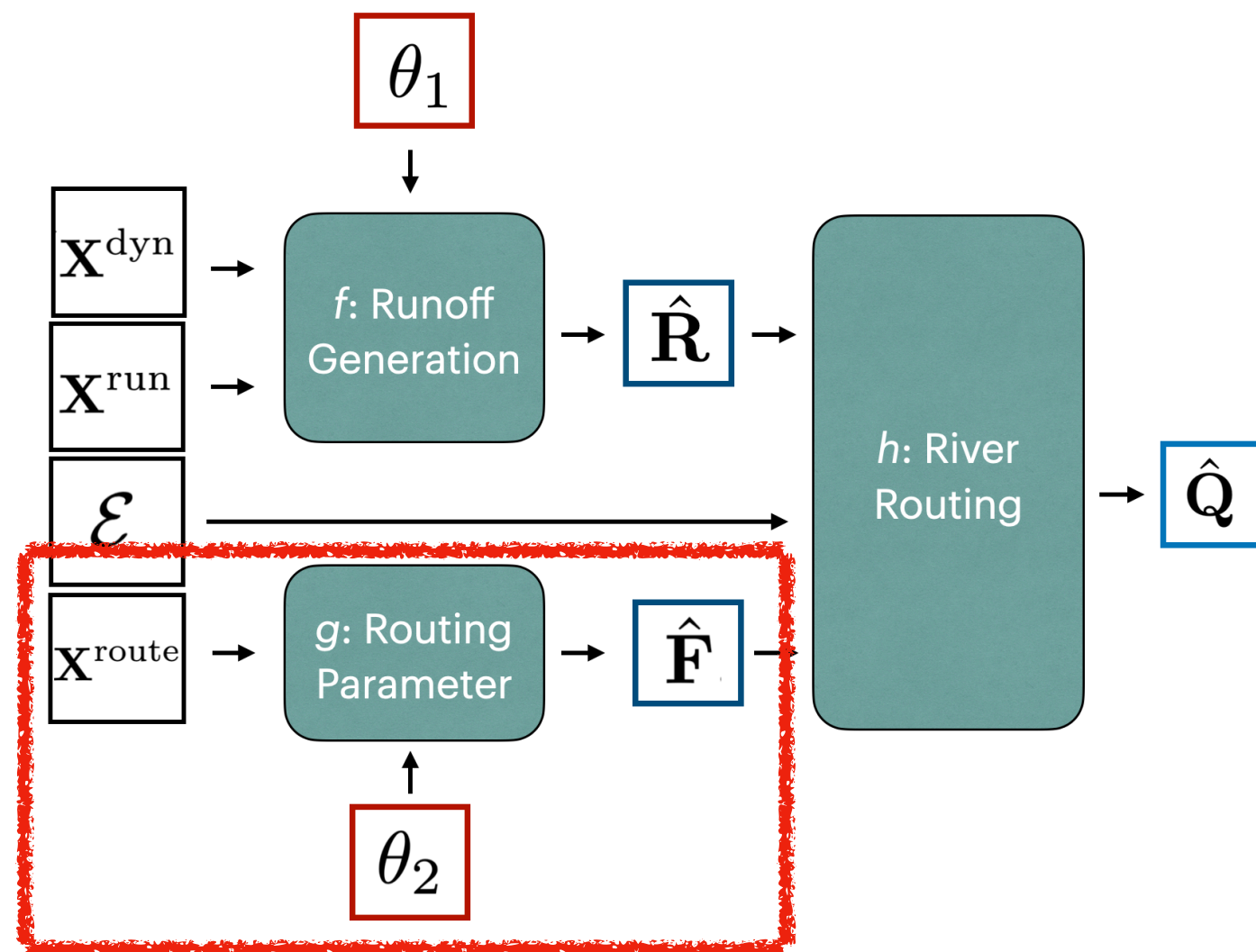
The **runoff** is an unconstrained blackbox: can represent any function, hard to interpret

LTI **routing** is *mass-conserving, advection + diffusion, interpretable* parameters

E2EL: runoff module anticipates the routing downstream advection and diffusivity

Full dynamic correct (high accuracy) routing correctly -> runoff correct

Methodology



How can we tell if routing parameters are correctly identified?

If an unconstrained parameter function converges to a **low variance** solution conform to prior **physical expectations**.

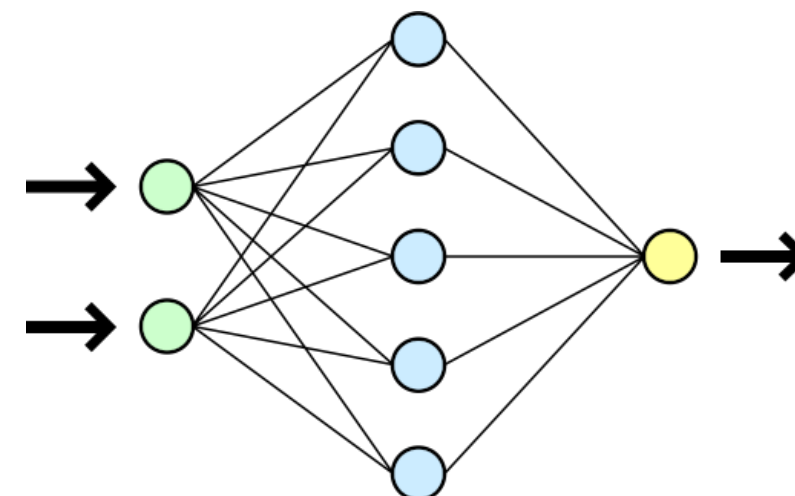
Variance: we train the E2EL many times across diverse: temporal resolution, spatial resolution and random initial conditions

Physics expectation: residence time to be linear with channel length, power law between slope and celerity, etc.

Channel features

- Distance
- Slope
- UPA

Unconstrained model



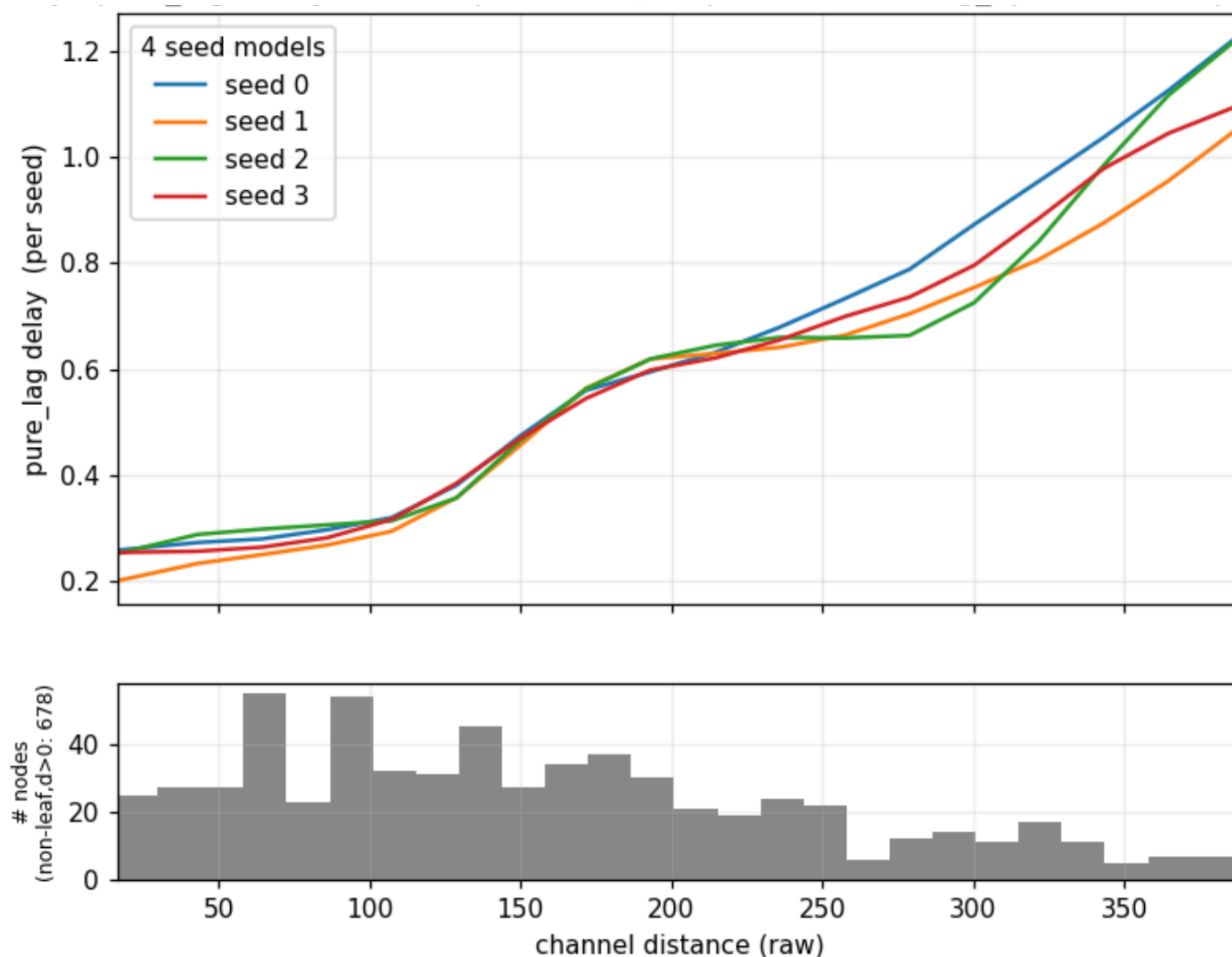
Routing parameters

Muskingum k, x
 Hayami c, D
 Pure lag delay
 Linear storage τ

Methodology

Example: pure lag delay as a function of channel length, slope and UPA, at hourly resolution

$$\Delta = f(\text{length}, \text{slope}, \text{UPA})$$



Physical expectation:

- Partial spearman correlation: linear approximation of $d = f(L)$ after factoring out the effect of slope and UPA
- Symbolic regression: use ML to infer the mathematical formula that best matches the learned function

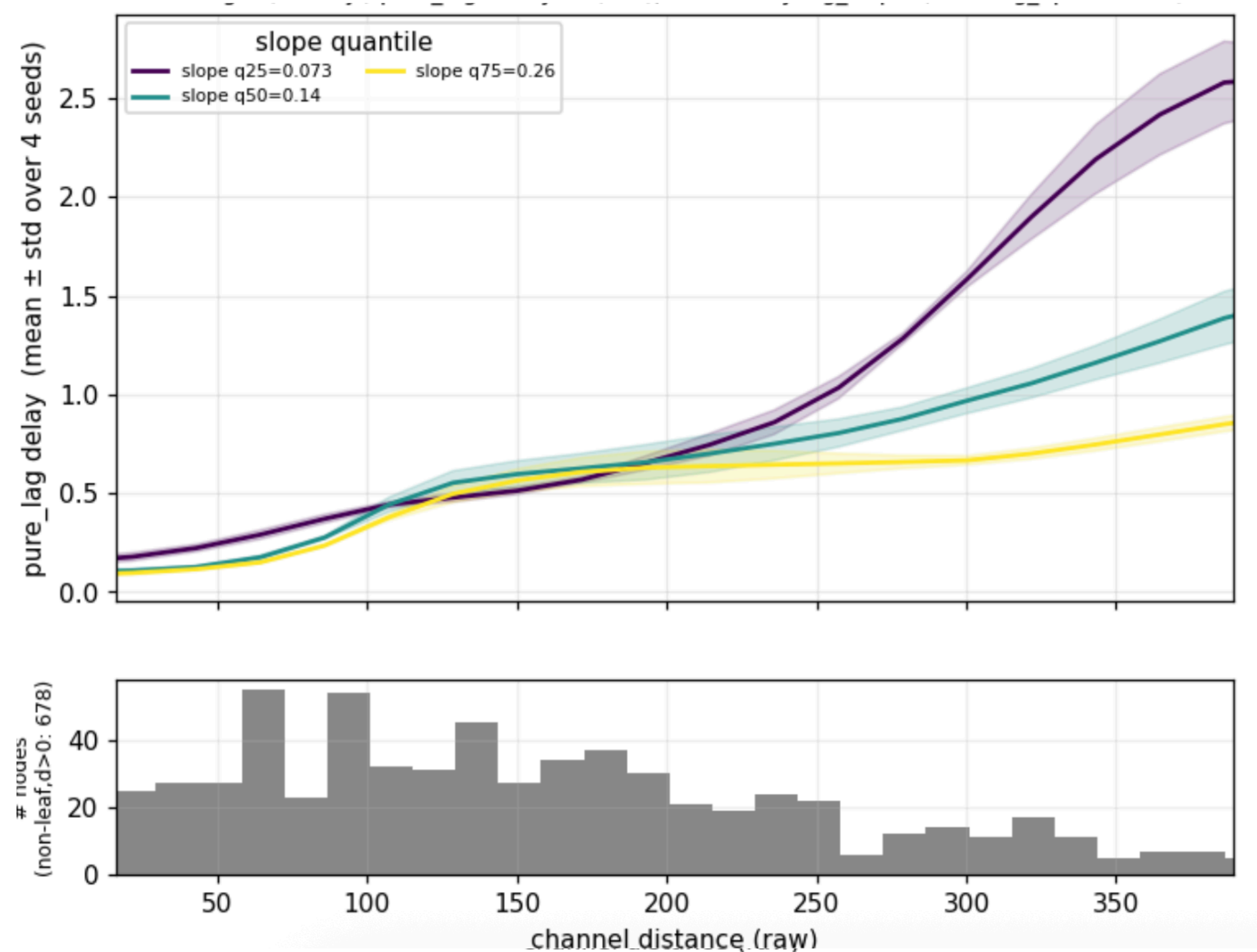
Variance Analysis:

- Use the mean Lin's Concordance Correlation Coefficient (CCC) between over all pairs of runs.
- 1 -> perfect reproducibility, no variance.
- 0 -> random results

Methodology

Example: pure lag delay as a function of channel length, slope and UPA, at hourly resolution

$$\Delta = f(\text{length}, \text{slope}, \text{UPA})$$



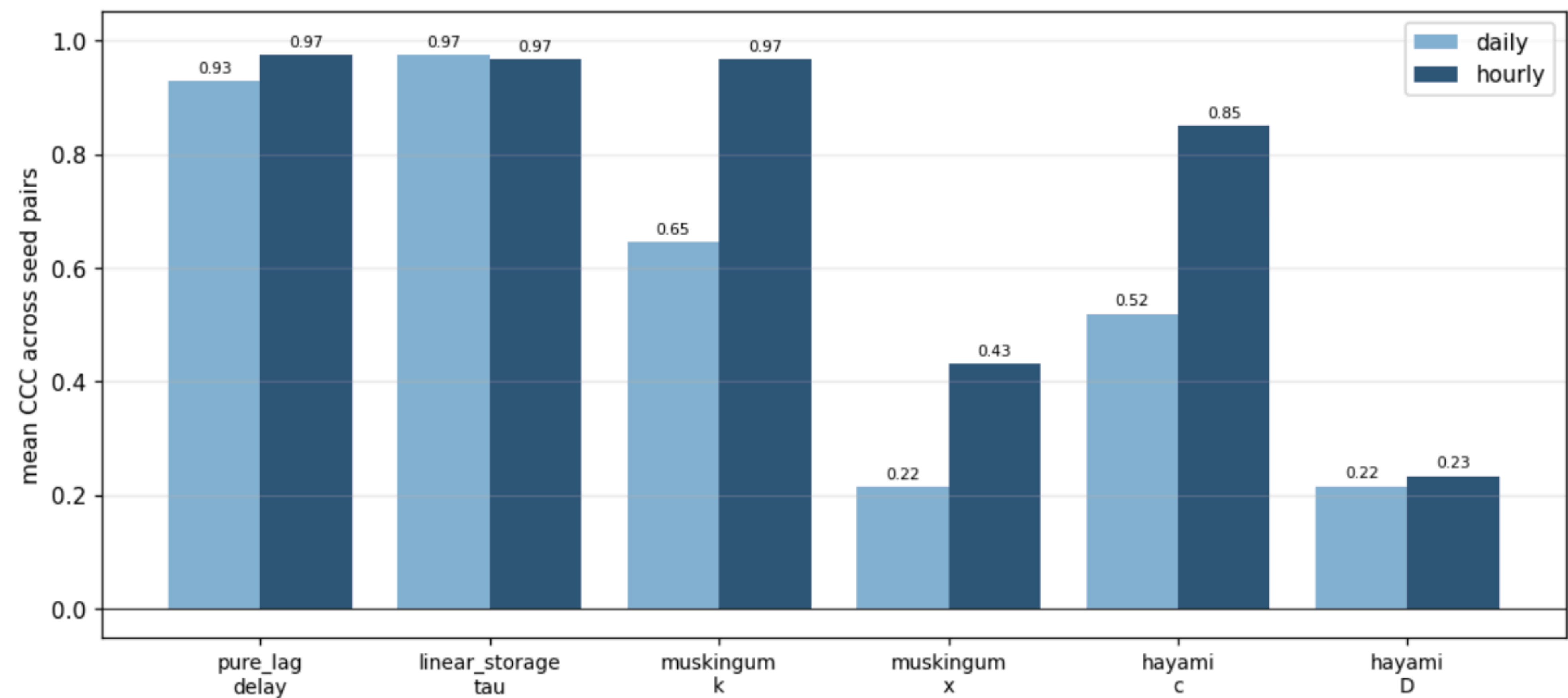
Physical expectation:

- Partial spearman correlation: linear approximation of $d = f(L)$ after factoring out the effect of slope and UPA
- Symbolic regression: use ML to infer the mathematical formula that best matches the learned function

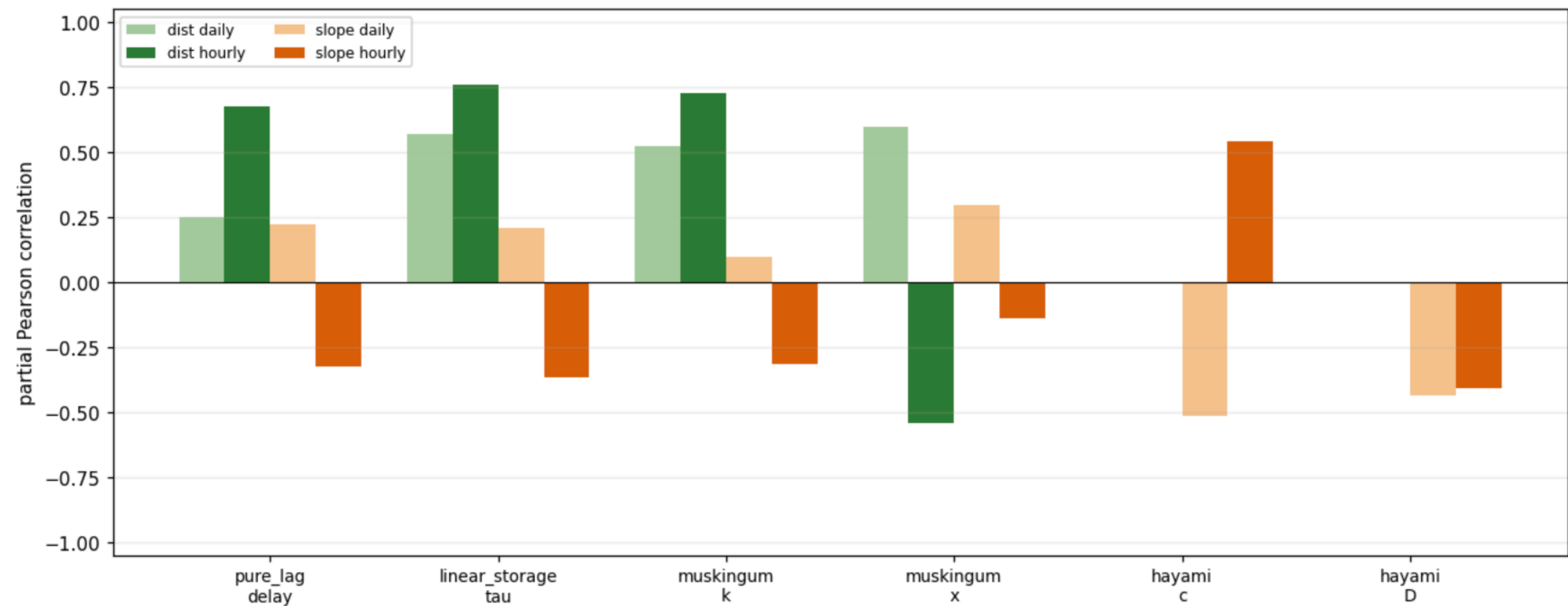
Variance Analysis:

- Use the mean Lin's Concordance Correlation Coefficient (CCC) between over all pairs of runs.
- 1 -> perfect reproducibility, no variance.
- 0 -> random results

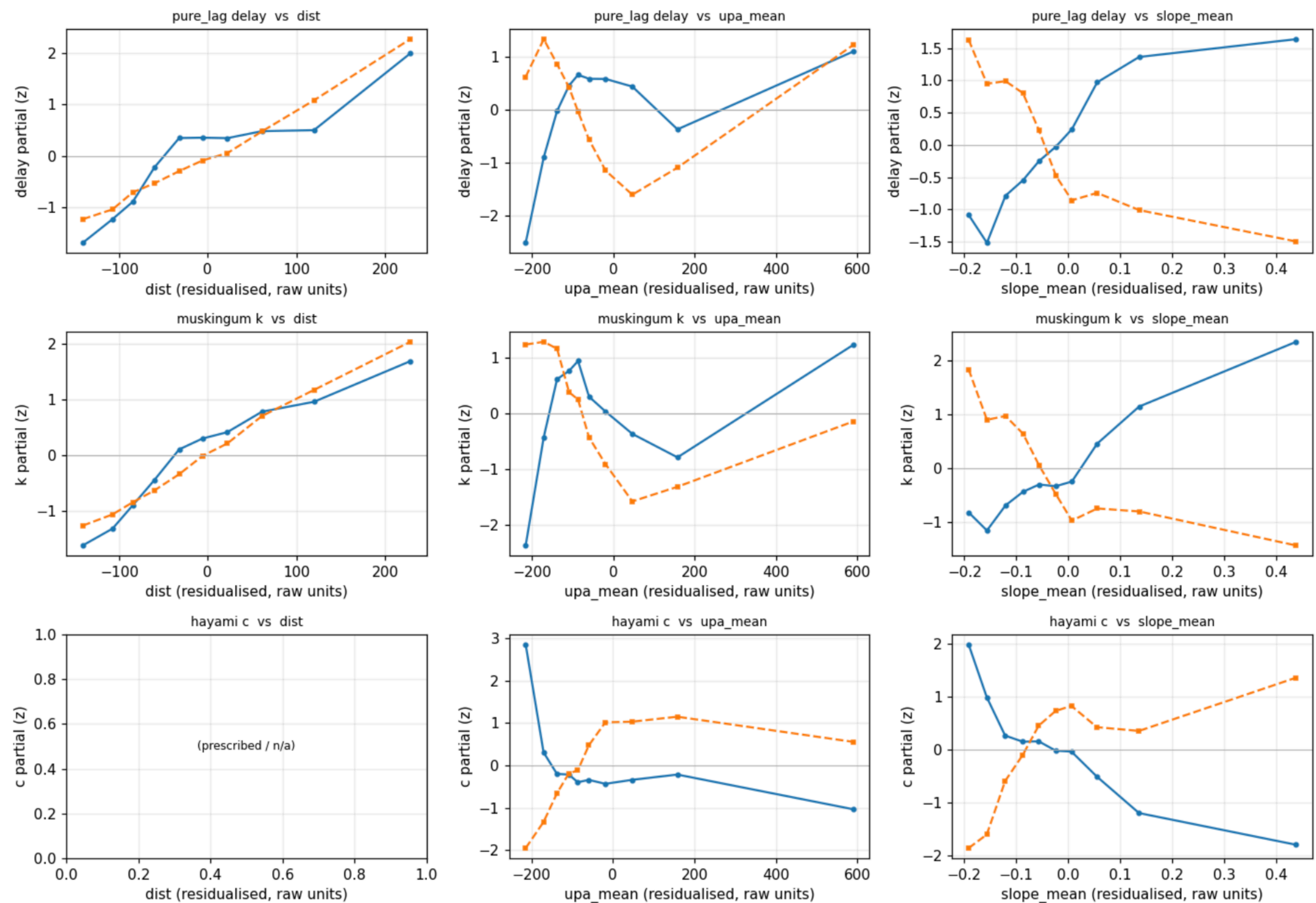
Results - Variance



Results - Physical expectation - Partial Pearson



Results - Physical expectation - Pearson



Results - Physical expectation

Symbolic Regression (SR) finds the mathematical formula that best explains samples drawn from a given function

For a Hayami routing $c = f(S, U)$ Symbolic Regression converges to a clean power law:

$$\log c = 2.0539 + 0.1926 \log U + 0.2590 \log S$$

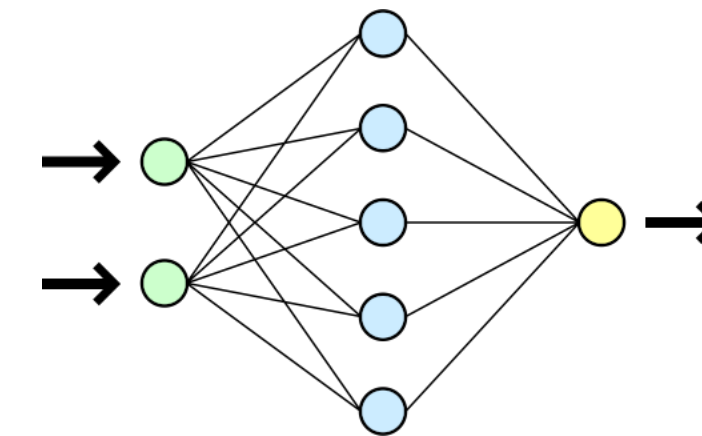
Similar results are observed for most function relation f

Symbolic regression converges close to power laws

Channel features

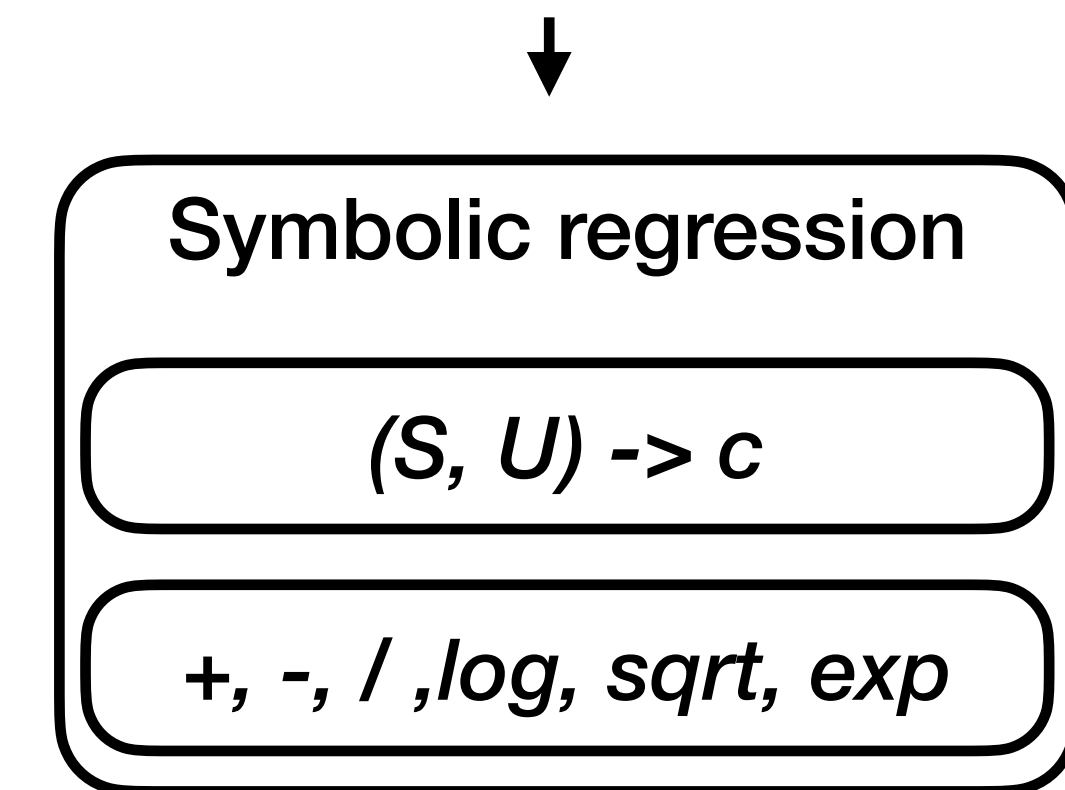
- Distance
- Slope
- UPA

Unconstrained model



Routing parameters

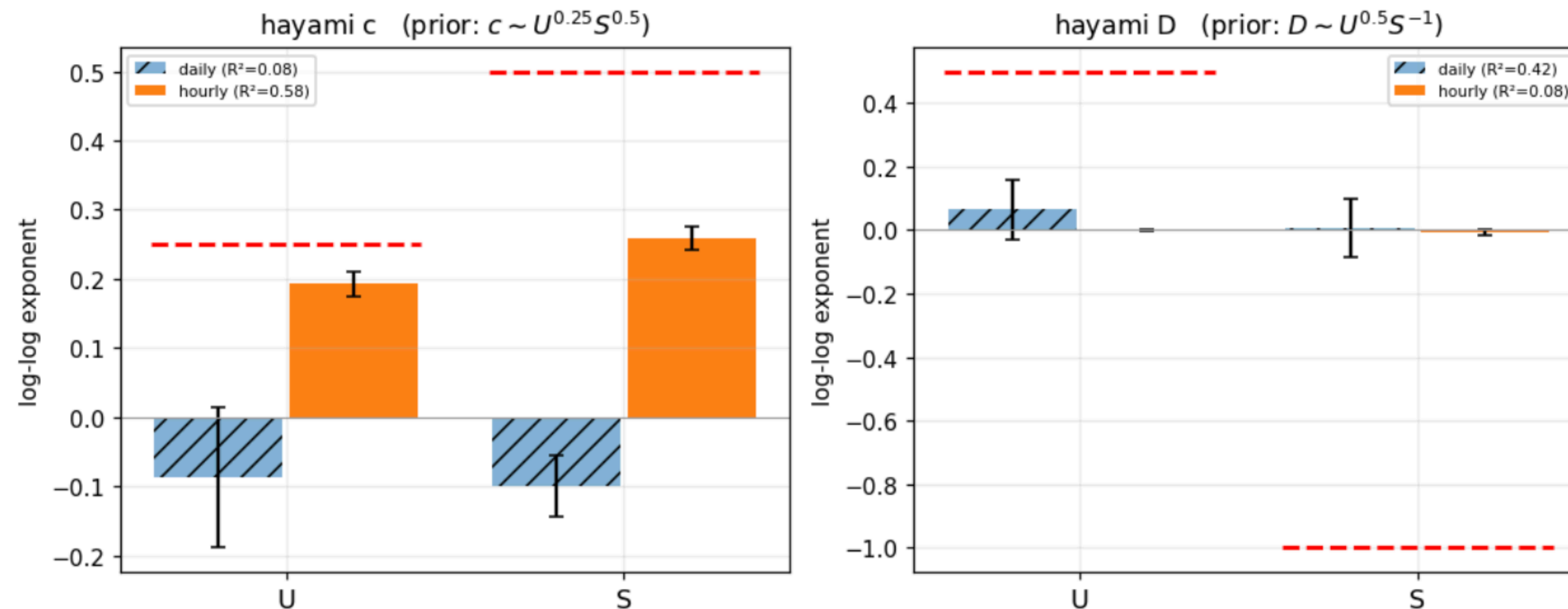
Muskingum k, x
Hayami c, D
Pure lag delay
Linear storage τ



$$c = 5.798 A^{0.1926} S^{0.2590}$$

Results - Physical expectation

How do the learned power laws compare to prior physical expectations?

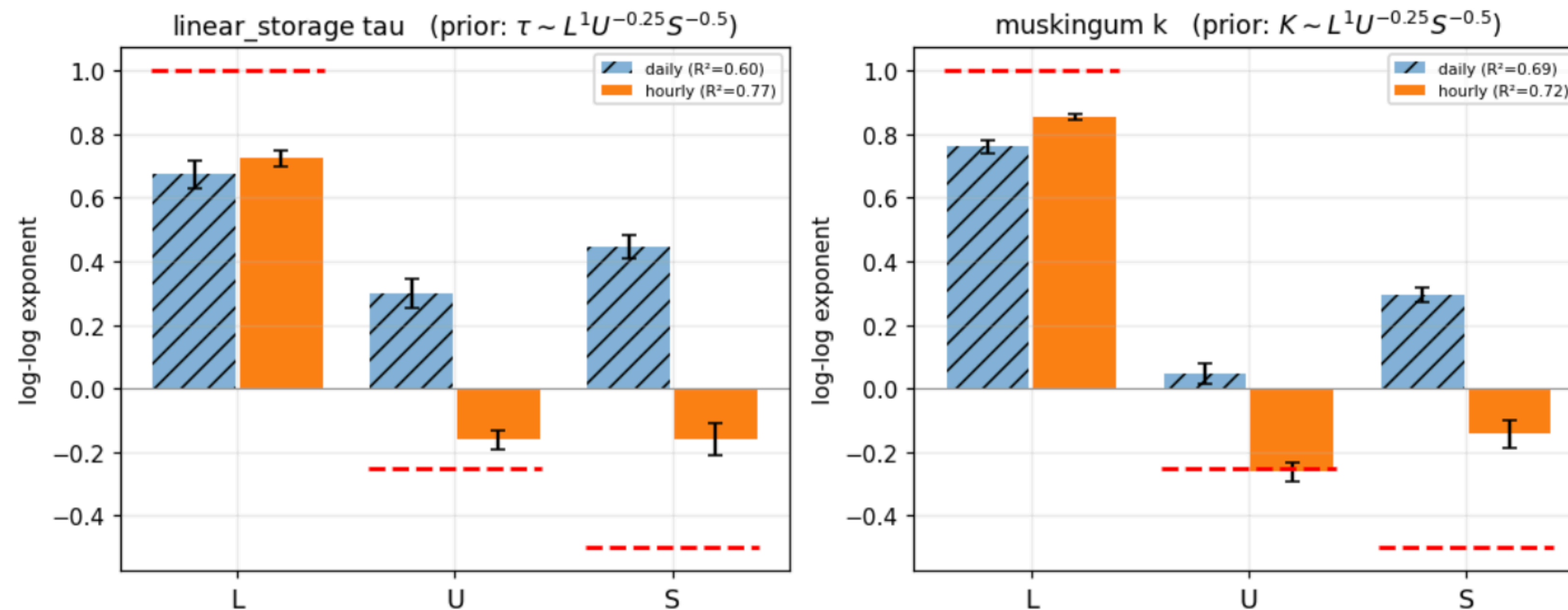


Consistent tendency to underestimate exponents

Daily resolution fails to recover the impact of slope and UPA

Results - Physical expectation

How do the learned power laws compare to prior physical expectations?



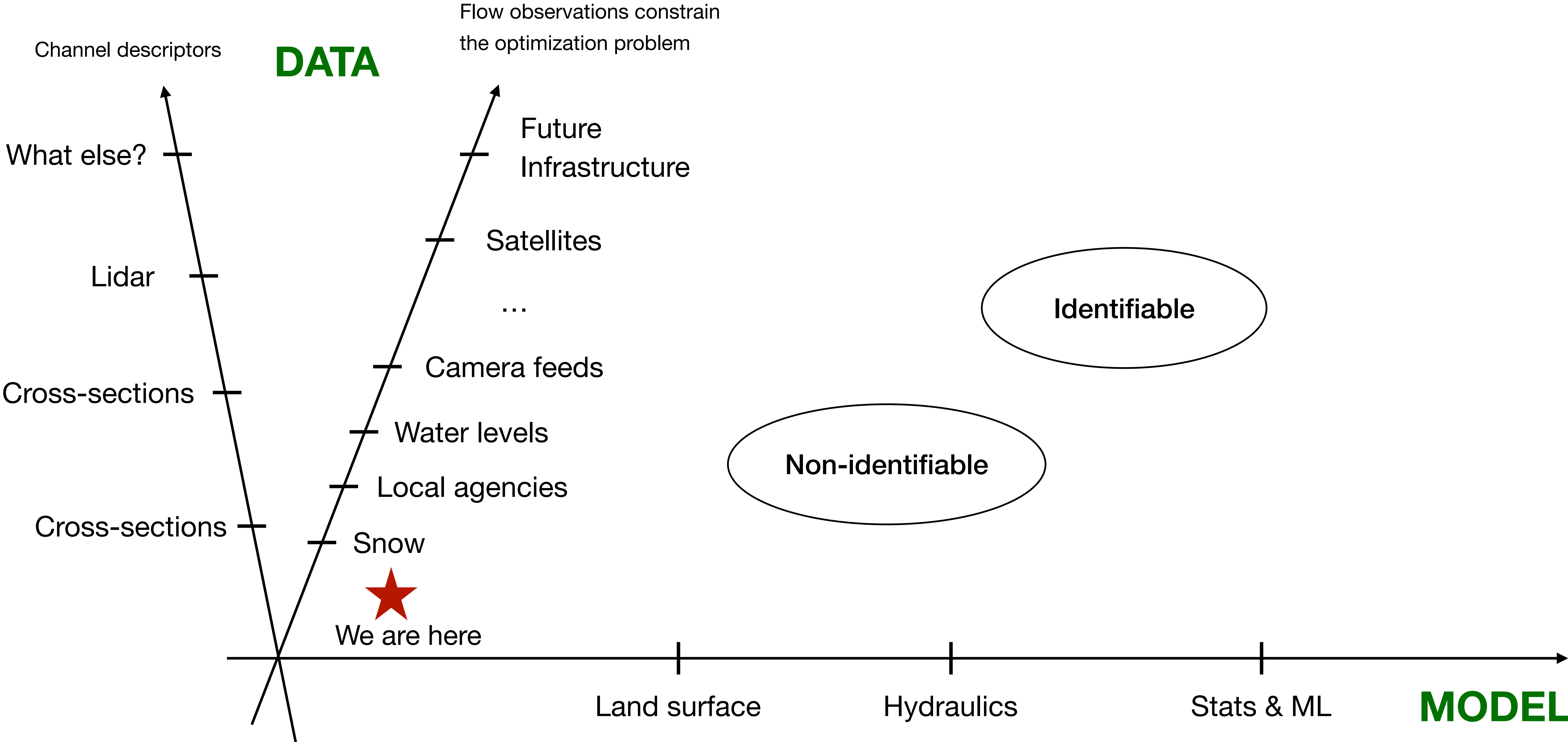
Consistent tendency to underestimate exponents

Daily resolution fails to recover the impact of slope and UPA

Conclusion

- At **daily** resolution, E2EL resolves **runoff vs 1st order routing component**
 - (correctly identifies the impact of distance on delays)
- At **hourly** resolution, E2EL also resolves a **2nd order routing component**
 - Identifies the impact of slope and UPA on wave celerity
- Those results are robust: Hold across all physical equations
- *Diffusivity parameters (Muskingum x and Hayami D) are **unidentifiable**:*
 - **This is an artifact of the routing schemes** due to auto-correlation of the advective and diffusive terms.
 - Using **orthogonal formulation** results in **identifiable diffusivity**

Opening



Outline

- 1. Differentiable routing (DiffRoute) enables end-to-end learning (**E2EL**)
 - “Differentiable river routing for end-to-end learning of hydrological processes”, AGU JGR Machine Learning and Computations, 2026, (published)
- 2. **E2EL** produces high accuracy predictions with physically consistent spatial distribution
 - “Leveraging Japan's streamflow records for end-to-end learning of hydrological simulations at national scale”, Water Resource Research, 2026 (preprint)
- 3. **E2EL** allows to identify physically meaningful dynamic parameters
 - “End-to-end learning from sparse and noisy observations largely resolves the equifinality of distributed river-discharge models over Japan”, ECMWF workshop, in preparation

Closing message

Thank you very much for your attention.

I look forward to fruitful discussions

We have much to learn from the hydraulics community.

Please feel free to contact me for discussion, early access to data, code or simulations

GitHub: @TristHas

@: *tristan.hascoet@riken.jp*