Neural Networks for Parameter Estimation in Intractable Models

Amanda Lenzi, Julie Bessac, Johann Rudi, Michael L. Stein

Reviewed By

Kaniz Fatema

STAT718-001-FALL-2025

Statistical Learning for Big Data

University of South Carolina

Why This Paper Was Chosen

- **Relevance:** It connects deep learning and Bayesian inference—two major themes in modern computational statistics.
- **Innovation:** Proposes a likelihood-free inference method using CNNs for estimating parameters of complex max-stable models.
- **Computational Importance:** Traditional pairwise-likelihood approaches are accurate but extremely slow for high-dimensional spatial data.
- **Contribution**
  - CNNs trained on simulated data can replace intractable likelihood calculations. The method achieves large computational speed-ups and comparable or better accuracy.

Problem Area & Importance

- **Problem Area:**
  - Statistical inference for complex environmental processes.
  - Focus on **max-stable models** for spatial extremes — essential for modeling rare events (**e.g., heatwaves, heavy rainfall**).
- **Challenge:**
  - Likelihood computation is *intractable* for large datasets.
  - Classical estimators (**MLE, composite likelihood**) are computationally expensive or inefficient.
- **Importance:**
  - Deep learning offers an automated, scalable alternative.
  - The approach can extend to many scientific areas where simulation is easy but likelihood evaluation is hard.

Relation to Course Topics

- **Related Course Themes:**
  - Neural networks and deep learning for statistical modeling.
- **Extension of Concepts:**
  - Converts traditional approximate **Bayesian computation(ABC)/likelihood-free** ideas into a data-driven framework using CNNs.
  - Learns the mapping between simulated data and parameters directly—no need for handcrafted summary statistics.
  - Reduces simulation and fitting cost by training the network once, then reusing it for inference.

Limitations of Prior Work & Paper's Contribution

- **Limitations of Prior Work:**
  - Classical methods (**e.g., pairwise likelihood**) are slow and lose efficiency for large spatial datasets.
  - ABC methods rely on manually chosen summary statistics, limiting accuracy and scalability.
  - Deep neural networks (DNNs) previously used mainly for Gaussian or simple models.
- **How This Paper Fills the Gap:**
  - Introduces a CNN-based framework for parameter inference in max-stable processes.
  - Demonstrates higher accuracy and faster computation than pairwise likelihood estimators.
  - Proposes a bootstrap-based uncertainty quantification method compatible with the neural estimator.
  - Provides real-world validation on 29 years of Midwest U.S. temperature data.

Key Takeaways From the Introduction

- Deep learning can be used for **statistical parameter estimation**—not just prediction.
- The method bypasses intractable likelihoods by learning from simulations.
- The framework offers:
  - Computational efficiency.
  - Accuracy comparable or superior to traditional estimators.
  - A scalable foundation for high-dimensional inference problems.
- This work bridges machine learning and environmental statistics, providing a template for future cross-disciplinary studies.

Methodology: Key Idea of the Framework

Main Components:

- Simulation-based data generation
- Neural network (CNN) training
- Likelihood-free parameter estimation
- Application to spatial max-stable models
- Comparison with pairwise likelihood method
- Computational efficiency and scalability
- Automatic feature extraction by CNN
- Uncertainty quantification via bootstrap

Parameter Estimation Framework

- Estimate parameters $\boldsymbol{\theta}$ of a statistical model from observed data $\mathbf{y}$.

Maximum Likelihood Estimation (MLE)

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta} \in \Theta} p(\mathbf{y}; \boldsymbol{\theta})$$

- **Challenge:** For many non-Gaussian or complex models, the likelihood function $p(\mathbf{y}; \boldsymbol{\theta})$ is **analytically intractable or computationally expensive**.
- Classical optimization strategies (e.g., numerical MLE) become impractical even at moderate dimensions.

Proposed Solution

- Replace direct likelihood optimization with a neural network function $\mathcal{F}_w(\mathbf{y})$ that maps data $\mathbf{y}$ directly to parameter estimates $\boldsymbol{\theta}$.

Learning Objective

$$\mathcal{F}_w : \mathbf{y} \mapsto \boldsymbol{\theta}, \qquad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} d\big(\boldsymbol{\theta}, \mathcal{F}_w(\mathbf{y})\big)$$

- $\mathcal{F}_w$ is a deep neural network with trainable weights and biases $w = (w_1, \ldots, w_{l_1}, b_1, \ldots, b_{l_2})$.

Loss Function and Optimization

- Instead of maximizing a likelihood, the model minimizes a **loss function** that penalizes prediction errors.

Mean Squared Error (MSE) Loss

$$\mathrm{MSE}(w) = \mathbb{E}\left\{ \|\boldsymbol{\theta} - \mathcal{F}_w(\mathbf{y})\|^2 \right\}$$

- For regression-type inference, MSE is a natural and widely used loss .
- Training uses iterative methods such as **stochastic gradient descent** (SGD) or **Adam** .

Optimization and Training Setup

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathrm{MSE}(\mathbf{w}),$$

$\nabla_{\mathbf{w}} L(\mathbf{w})$computed via automatic differentiation.
Training data:

$$\{(\boldsymbol{\theta}_j^{\text{train}}, \mathbf{y}_j^{\text{train}})\}_{j=1}^{J}, \quad \boldsymbol{\theta}_j^{\text{train}} = (\theta_{1,j}, \ldots, \theta_{K,j})^\top,$$

$$\mathrm{y}_j^{\text{train}} = (y_{1,j}, \ldots, y_{n,j})^\top.$$

Parameter sampling: $\boldsymbol{\theta}_j^{\text{train}} \sim \mathrm{Unif}(a_{\boldsymbol{\theta}}, b_{\boldsymbol{\theta}}), \quad \mathbf{y}_j^{\text{train}} \sim \mathrm{Model}(\boldsymbol{\theta}_j^{\text{train}})$.

Goal: $\mathcal{F}_{\mathbf{w}} : \mathbf{y} \mapsto \boldsymbol{\theta}$ trained to minimize MSE and predict $\hat{\boldsymbol{\theta}} = \mathcal{F}_{\mathbf{w}^*}(\mathbf{y})$.

Introduction to Convolutional Neural Networks

For an input image $\mathbf{y}$ and kernel (filter) K, the discrete convolution is:

$$K[s_1, s_2] * \mathbf{y}[s_1, s_2] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} K[i, j]\, \mathbf{y}[s_1 + i, s_2 + j].$$

- Each kernel K is a matrix of **trainable weights** (filters) that scan across the image.

Limit Definition of a Max-stable Process

$$Z(\mathbf{s}) \;=\; \lim_{m\to\infty} a_m(\mathbf{s})^{-1}\Big[\max_{1\le i\le m} Y_i(\mathbf{s}) \;-\; b_m(\mathbf{s})\Big], \qquad \mathbf{s}\in\mathcal{S}\subset\mathbb{R}^d,$$

where,

- $\{Y_i(\mathbf{s})\}_{i\ge 1}$: i.i.d. **stochastic processes** (block maxima come from e.g. yearly maxima).
- $a_m(\mathbf{s}) > 0$ and $b_m(\mathbf{s})$ are normalizing functions (scale and location) chosen so the limit is nondegenerate.
- The limit (if it exists) defines a **max-stable process** $Z(\mathbf{s})$; marginally each $Z(\mathbf{s})$ belongs to a GEV family.

Spectral Representation

$$Z(\mathbf{s}) \;=\; \max_{i\geq 1} \xi_i \, W_i(\mathbf{s}), \qquad \mathbf{s} \in \mathcal{S},$$

where,

- $\{\xi_i\}_{i\geq 1}$ are the points of a **Poisson process** on $(0,\infty)$ with intensity $d\Lambda(\xi) = \xi^{-2}\, d\xi$ (this choice yields unit Fréchet margins).
- $\{W_i(\mathbf{s})\}_{i\geq 1}$ are i.i.d. nonnegative **stochastic processes** with $\mathbb{E}[W_i(\mathbf{s})] = 1$ for every $\mathbf{s}$, independent of the Poisson points.
- This representation (**de Haan spectral representation**) constructs any max-stable process with **unit Fréchet margins** by random scaling of nonnegative shape functions $W_i(\cdot)$.
- Intuition: each term $\xi_i W_i(\mathbf{s})$ is a random **"storm" or extremal function**; the process takes the pointwise maximum over infinitely many such storms.

Unit Fréchet Marginal Distribution And Implication

$$\Pr\big(Z(\mathbf{s}) \le z\big) \;=\; \exp\big(-1/z\big), \qquad z > 0,$$

Remarks and implications:

- The choice $d\Lambda(\xi) = \xi^{-2}\, d\xi$ together with $\mathbb{E}[W(\mathbf{s})] = 1$ yields unit Fréchet margins: $F_Z(z) = \exp(-1/z)$.
- More general margins can be obtained by monotone transformations (GEV family with location, scale, shape).

Two Max-Stable Models Considered

- **1. Brown–Resnick Model**
  - $W_i(\mathbf{s}) = \exp\{\epsilon_i(\mathbf{s}) - \gamma(\mathbf{s})\}$,
  - $\epsilon_i(\mathbf{s})$ are Gaussian with $\gamma(\mathbf{h}) = \|\mathbf{h}\|^\nu / \lambda^\nu$,
  - $\lambda > 0$ (range), $\nu \in (0, 2]$ (smoothness).

- **2. Schlather Model**
  - $W_i(\mathbf{s}) = \sqrt{2\pi}\, \max\{0,\, \epsilon_i(\mathbf{s})\}$,
  - $\epsilon_i(\mathbf{s})$ Gaussian with correlation $\rho(\mathbf{h}) = \exp\{-(\|\mathbf{h}\|/\lambda)^\nu\}$.

Joint Cumulative Distribution Function (CDF)

Joint Distribution of $Z(\mathbf{s})$

$$p(Z(\mathbf{s}_1) \leq z_1, \ldots, Z(\mathbf{s}_D) \leq z_D) = \exp[-V(z_1, \ldots, z_D)],$$

- This defines the joint CDF of a max-stable process at D spatial sites $\mathbf{s}_1, \ldots, \mathbf{s}_D$.
- $V(z_1, \ldots, z_D)$ is the **exponent function**, defined as:

$$V(z_1, \ldots, z_D) = \mathbb{E}\left[\max_i \frac{W(\mathbf{s}_i)}{z_i}\right],$$

  where $W(\mathbf{s}_i)$ are nonnegative stochastic processes with $\mathbb{E}[W(\mathbf{s}_i)] = 1$.
- $V(\cdot)$ ensures the joint CDF is homogeneous and has correct marginal behavior.

Full Likelihood Function

Density Function of Max-Stable Process

$$f(z_1, \ldots, z_D) = \exp[-V(z_1, \ldots, z_D)] \sum_{\pi \in \mathcal{P}} \prod_{l=1}^{L} [-V_{\pi_l}(z_1, \ldots, z_D)],$$

- $\mathcal{P}$ is the set of all partitions $\pi = \{\pi_1, \ldots, \pi_L\}$ of $\{1, \ldots, D\}$.
- V_{π_l} is the partial derivative of V with respect to the variables indexed by block π_l.
- The number of partitions equals the **Bell number** of order D — grows super-exponentially.
- Hence, this full likelihood is **computationally infeasible** when $D > 12$ (Castrocuccio et al., 2016).

Pairwise Log-Likelihood (Simplified Estimation)

Weighted Pairwise Log-Likelihood

$$\ell(\boldsymbol{\phi}) = \sum_{(j_1,j_2)\in\mathcal{P}} \alpha_{j_1,j_2} \Big[\log\big\{ V_1(z_{j_1}, z_{j_2}) V_2(z_{j_1}, z_{j_2}) - V_{12}(z_{j_1}, z_{j_2}) \big\} - V_1(z_{j_1}, z_{j_2}) \Big]$$

- $\boldsymbol{\phi}$ is the vector of model parameters. $\alpha_{j_1,j_2} \geq 0$ are weighting coefficients for each pair of sites (j_1, j_2).
- Pairwise likelihood is used instead of the full likelihood — reduces computational burden. It remains **consistent** and **asymptotically Gaussian**, providing a practical compromise.

Proposed CNN Parameter Estimation Setup

$$\hat{\boldsymbol{\theta}}_i^{\text{CNN}} = \begin{pmatrix} \hat{\lambda}_i^{\text{CNN}} \\ \hat{\nu}_i^{\text{CNN}} \end{pmatrix}, \qquad \boldsymbol{\theta}_j^{\text{train}} = \begin{pmatrix} \lambda_j^{\text{train}} \\ \nu_j^{\text{train}} \end{pmatrix}, \quad j = 1, \ldots, 2000.$$

$$\lambda_j^{\text{train}} \sim \text{Unif}(a_\lambda^{\text{train}}, b_\lambda^{\text{train}}), \qquad \nu_j^{\text{train}} \sim \text{Unif}(a_\nu^{\text{train}}, b_\nu^{\text{train}}).$$

The choice of $a_{\boldsymbol{\theta}}^{\text{train}}$ and $b_{\boldsymbol{\theta}}^{\text{train}}$, for $\boldsymbol{\theta} = (\lambda, \nu)$, is informed by testing pa

$\boldsymbol{\theta}_i$ to ensure that the training set covers the region of interest.

$$\mathbf{y}^{\text{train}} \in \mathbb{R}^{2000 \times 25 \times 25}, \qquad \text{output: } \left\{ \log(\lambda^{\text{train}}),\ \log\left(\frac{\nu^{\text{train}}}{2 - \nu^{\text{train}}} \right) \right\}^\top.$$

The logarithm is used as a variance-stabilizing transformation to help with numerical issues during training. The denominator corresponding to ν is chosen such that this parameter is not greater than two.

$$\hat{\boldsymbol{\theta}}_i^{\text{CNN}} = \mathcal{F}_w(\mathbf{y}_i), \qquad i = 1, \ldots, 16.$$

Proposed Parameter Estimation Setup — Pairwise Likelihood Estimator

Alternative Estimator: Pairwise-likelihood estimator

$$\hat{\boldsymbol{\theta}}_i^{\mathrm{PL}} = (\hat{\lambda}_i^{\mathrm{PL}}, \hat{\nu}_i^{\mathrm{PL}})^\top.$$

- Uses the **pairwise log-likelihood function** implemented via the **R** function `fitmaxstab`.
- Optimization handled with `optim` (**method = L-BFGS-B**).
- To reduce computational cost:
  - Use only pairs of sites within 3 spatial units.
  - Assign equal weights $\alpha_{i_1,i_2} = 1$ to these pairs; others set to 0.
- Multiple random initializations used for robustness.
- Final parameter estimates chosen from the five runs with the highest pairwise likelihood.

Summary of CNN Model

Table 1
Summary of the CNN model. It is a sequential model taking input of shape [–, 25, 25, 128] and mapping it to two scalar values of shape [–, 2].

Layer Type	Output Shape	Filters	Kernel Size	Parameters
2D conv	[-, 25, 25, 128]	128	3×3	1280
2D conv	[-, 13, 13, 128]	128	3×3	147584
2D conv	[-, 7, 7, 16]	16	3×3	18448
dense	[-, 4]			1028
dense	[-, 8]			40
dense	[-, 16]			144
dense	[-, 2]			34
Total trainable weights:				168,558

Results for the Brown-Resnick Model

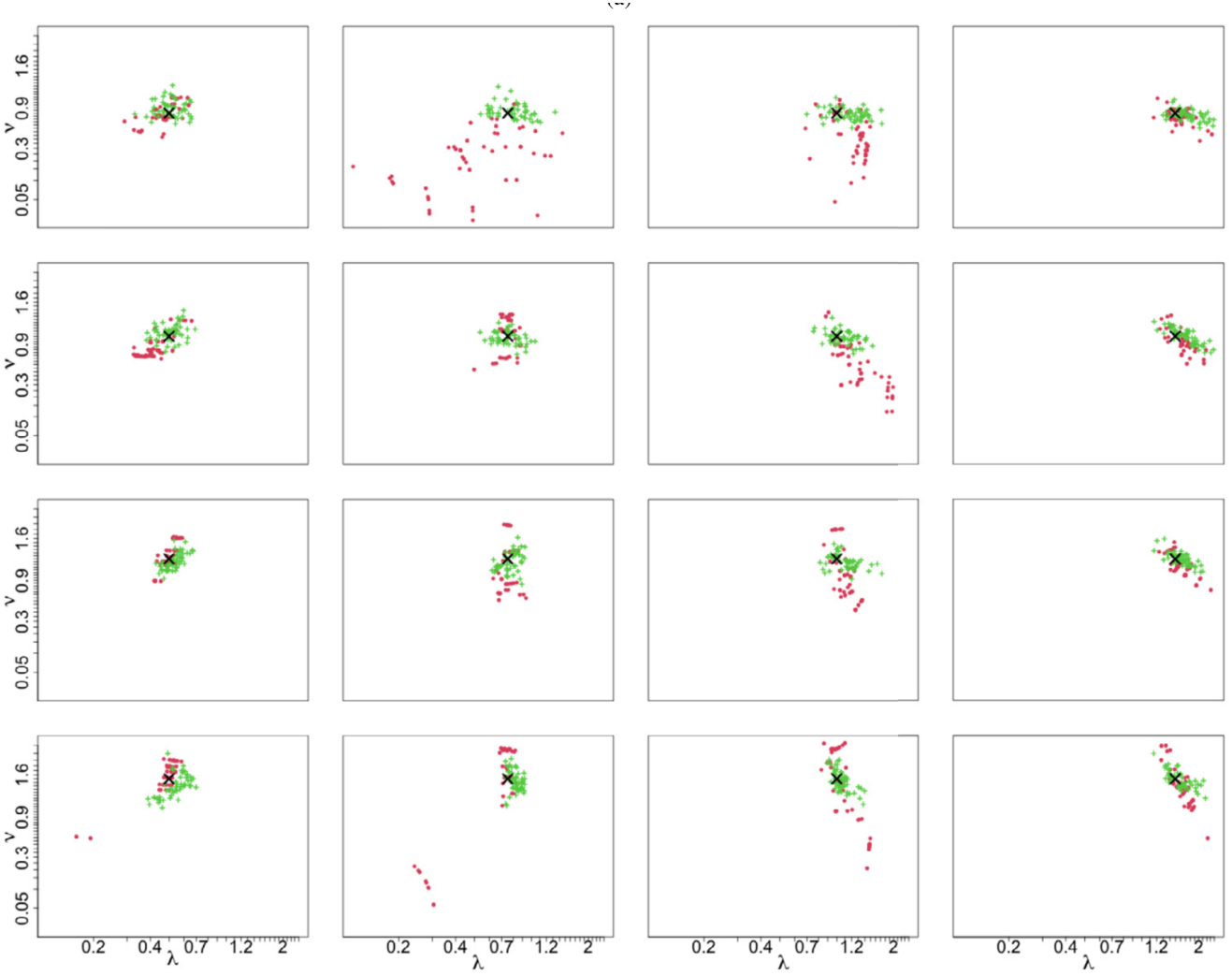

Fig. 2. Scatterplots of estimated parameters on the transformed scales (with numbers on the axes on the untransformed scales). Each plot shows 50 independent estimates from the Brown-Resnick (a) and Schlather's (b) models using the CNN (green) or PL (red) with the first initial values that maximizes the PL (red). Small to large ranges are shown from left to right, and rough to smooth are from top to bottom. The ×'s are the true values. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Results for Schlather's Model

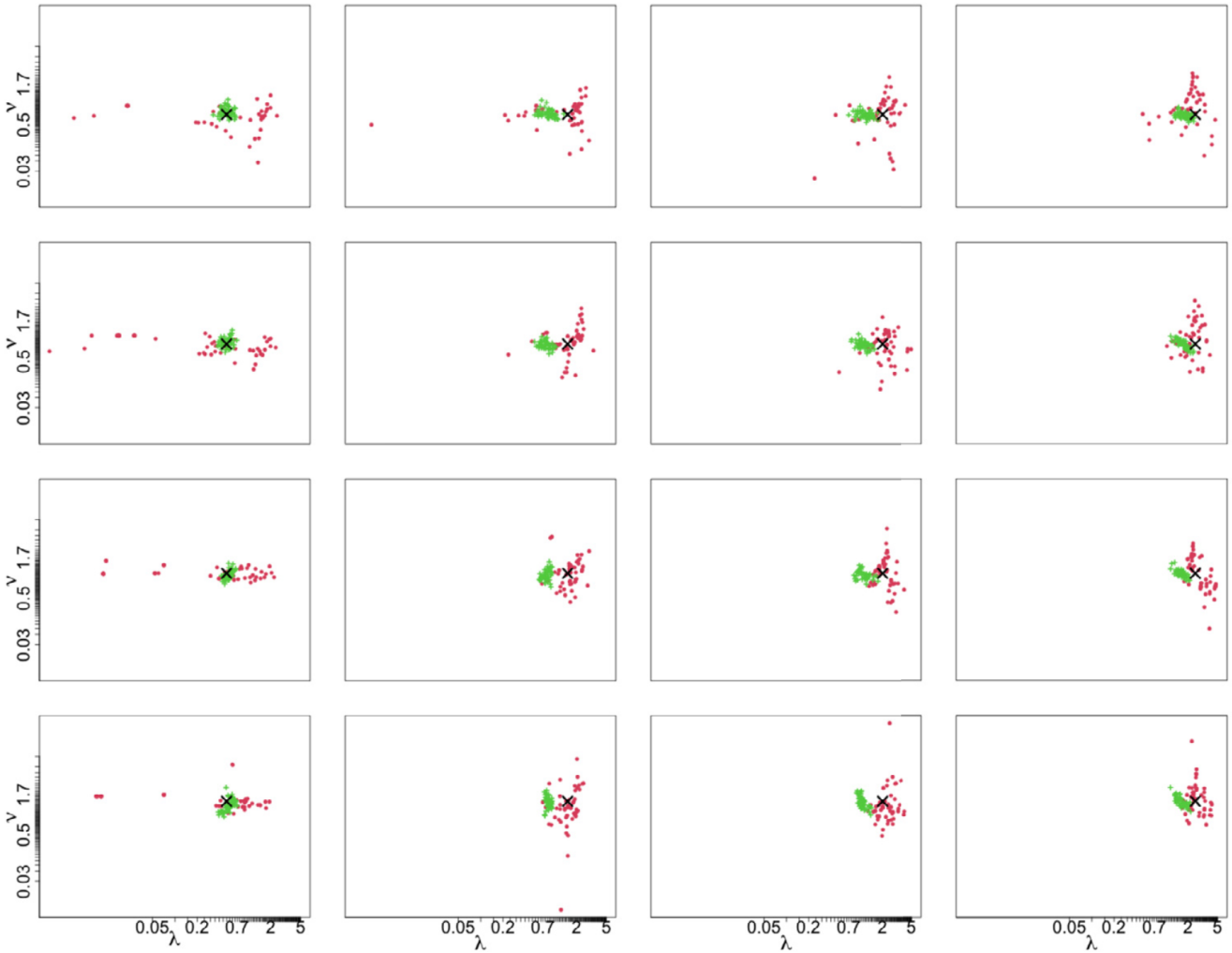

Fig. 2. Scatterplots of estimated parameters on the transformed scales (with numbers on the axes on the untransformed scales). Each plot shows 50 independent estimates from the Brown-Resnick (a) and Schlather's (b) models using the CNN (green) or PL (red) with the first initial values that maximizes the PL (red). Small to large ranges are shown from left to right, and rough to smooth are from top to bottom. The ×'s are the true values. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 2
RMSE, MAE, mean bias, and SD from the Brown-Resnick and Schlather's models using the CNN and the pairwise likelihood approaches. The two numbers in each column for the first three rows represent scores for estimating range and smoothness, respectively.

	Brown-Resnick		Schlather's	
	CNN	PL	CNN	PL
RMSE	**0.45;0.38**	0.56;0.62	**0.66;0.49**	0.83;0.62
MAE	**0.14;0.11**	0.24;0.30	**0.33;0.18**	0.55;0.31
Mean Bias	0.09;**-0.02**	**0.05**;-0.15	-0.23;**0.01**	**0.12**;-0.02
SD	**0.47;0.29**	0.60;0.46	**0.66; 0.31**	1.00;0.43

Application to Temperature Data

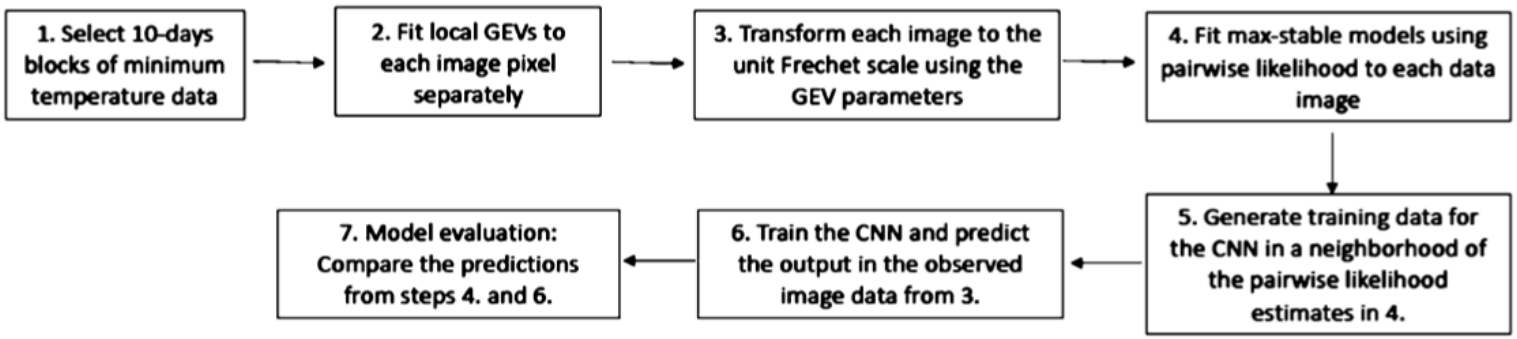

Fig. 4. Diagram of the proposed framework for estimating parameters from temperature data. Knowledge about the mapping between simulated data and parameters is compactly encoded within the weights of the CNN.

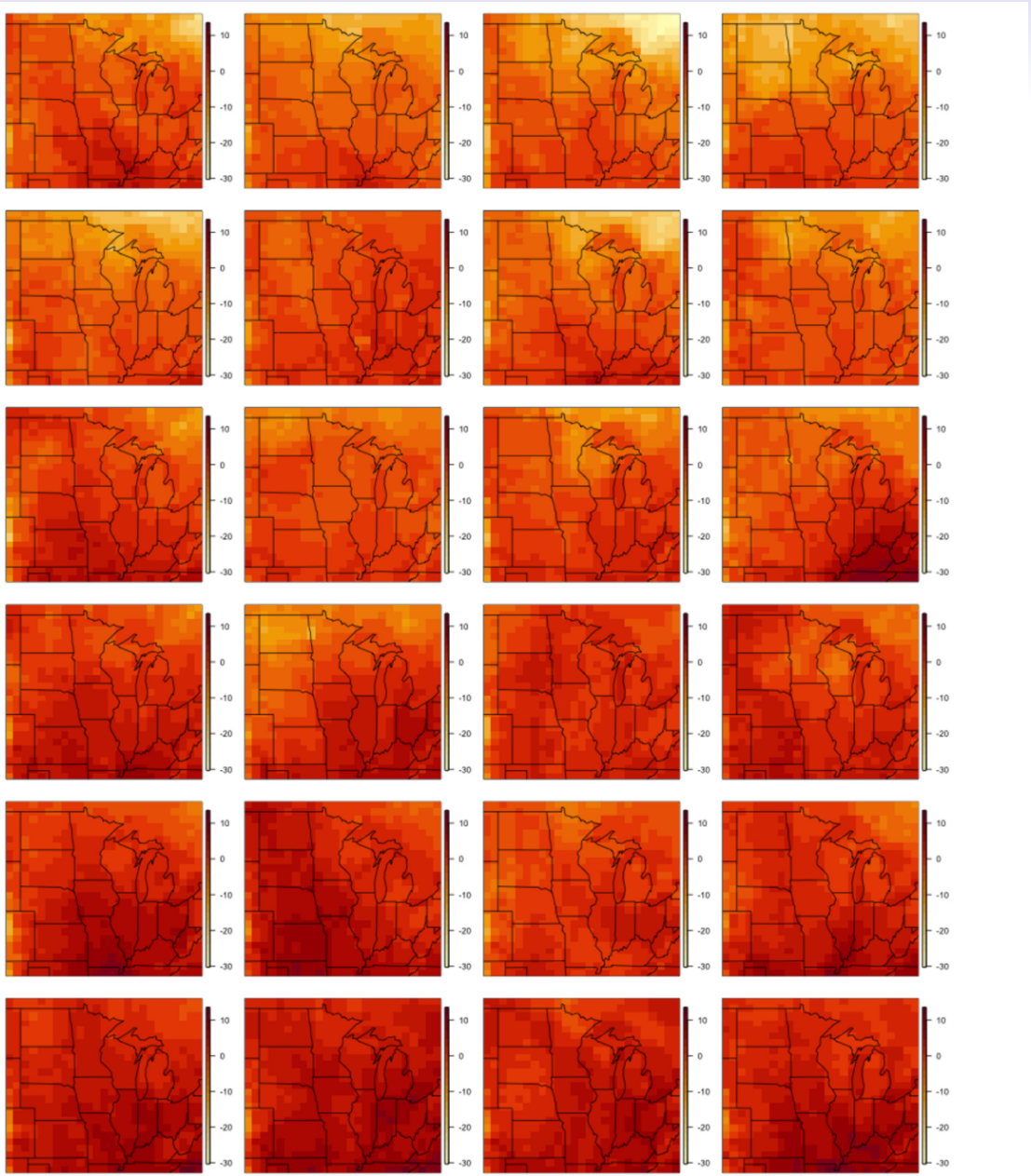

Fig. 3. Temporal evolution of temperature minima from six 10-day data periods over April-May (rows 1-6) during 1991, 2000, 2009, 2019 (columns 1-4).

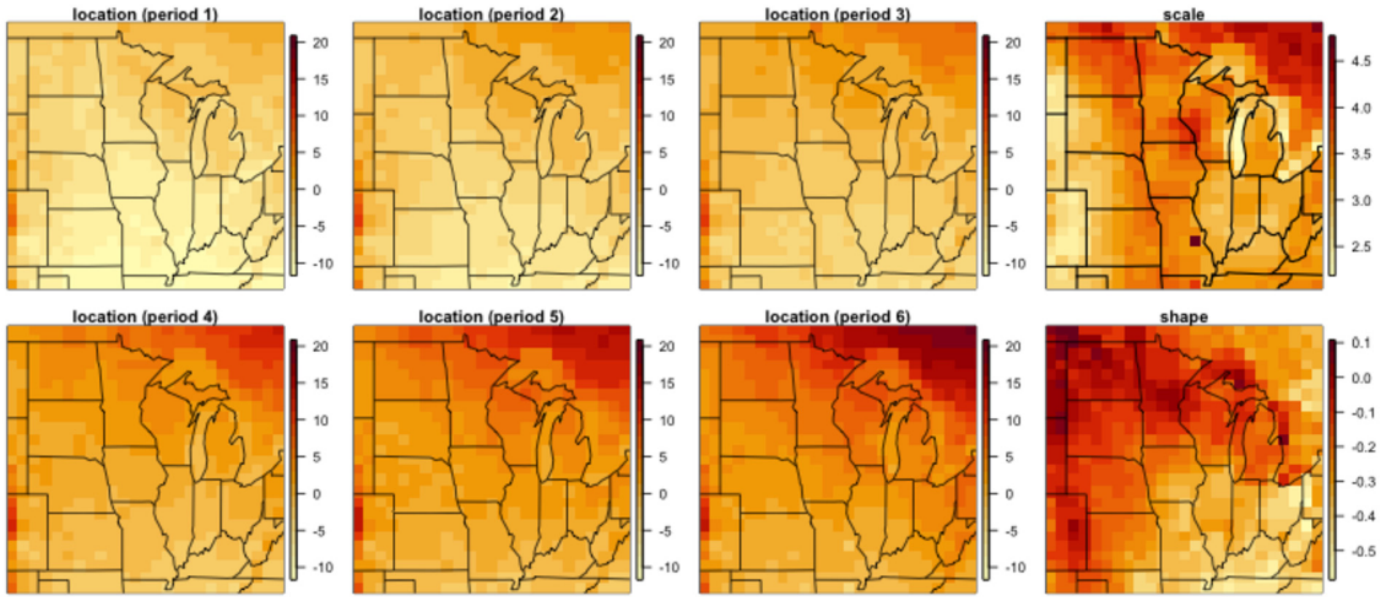

Fig. 5. (a) Estimated location for each of the 6 periods of data, scale, and shape parameters obtained from the individual fit of the GEV model.

Training Parameter Bounds and Sampling

$$\lambda_j^{\text{train}} \sim \text{Unif}\Big(\hat{\lambda}_{\min}^{\text{PL}} - 3\,\text{sd}(\hat{\lambda}^{\text{PL}}) \vee 0,\ \hat{\lambda}_{\max}^{\text{PL}} + 3\,\text{sd}(\hat{\lambda}^{\text{PL}})\Big)$$

$$\nu_j^{\text{train}} \sim \text{Unif}\big(\hat{\nu}_{\min}^{\text{PL}} - 3\,\text{sd}(\hat{\nu}^{\text{PL}}) \vee 0,\ \hat{\nu}_{\max}^{\text{PL}} + 3\,\text{sd}(\hat{\nu}^{\text{PL}}) \vee 2\big)$$

$$\text{j} = 1, \ldots, 2000.$$

$$\hat{\boldsymbol{\Theta}}_{\min}^{\text{PL}} = \min(\hat{\boldsymbol{\Theta}}_1^{\text{PL}}, \ldots, \hat{\boldsymbol{\Theta}}_{174}^{\text{PL}}), \quad \hat{\boldsymbol{\Theta}}_{\max}^{\text{PL}} = \max(\hat{\boldsymbol{\Theta}}_1^{\text{PL}}, \ldots, \hat{\boldsymbol{\Theta}}_{174}^{\text{PL}}).$$

Where,

- These samples generate simulated Brown–Resnick processes ($D = 25^2$) for CNN training.

Results And Model Comparison

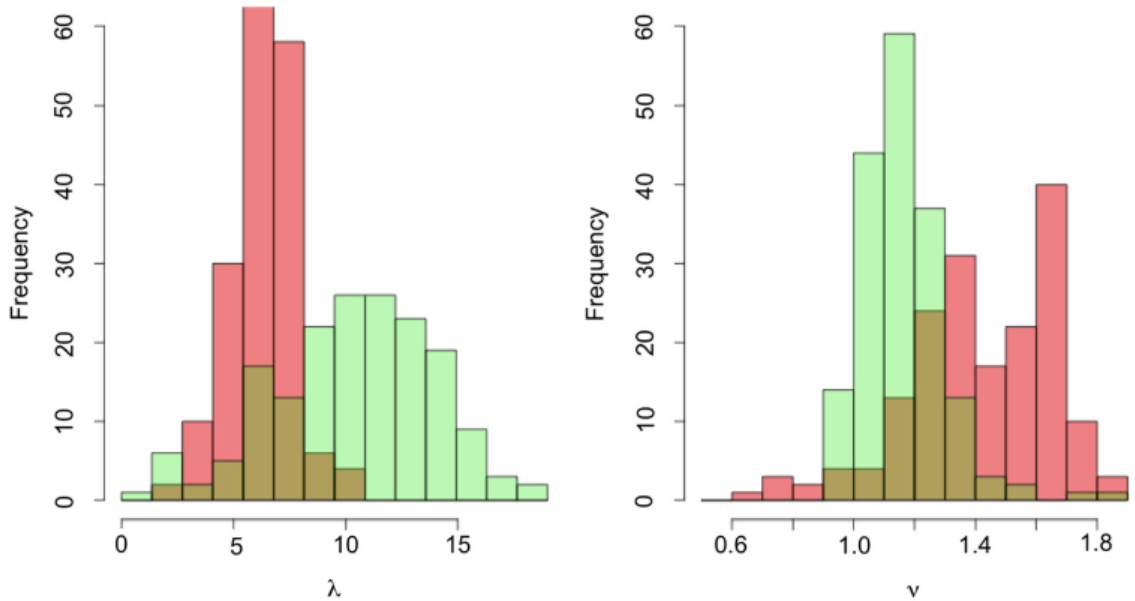

Fig. 6. Histograms of the estimates of range (left) and smoothness (right) from a Brown-Resnick model fitted to the Fréchet transformed data by using pairwise likelihood (red) and CNN (green).

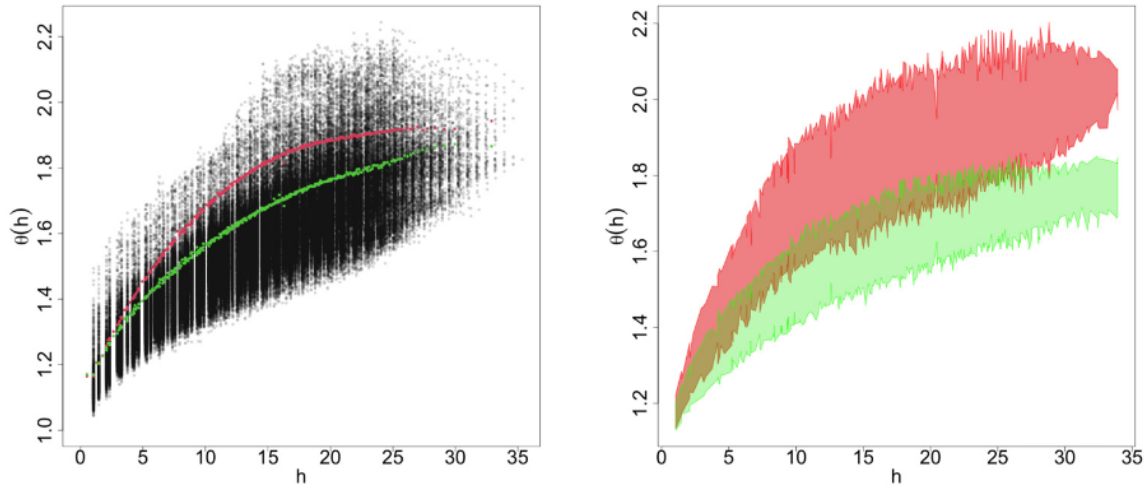

Fig. 7. Left: F-madogram estimates for the validation datasets (black points) and the estimated extremal coefficient functions from the pairwise likelihood (red) and CNN (green) using 100 bins. Right: Example of nonbinned F-madogram estimates using as parameter values the estimates obtained from one of the images in the testing set.

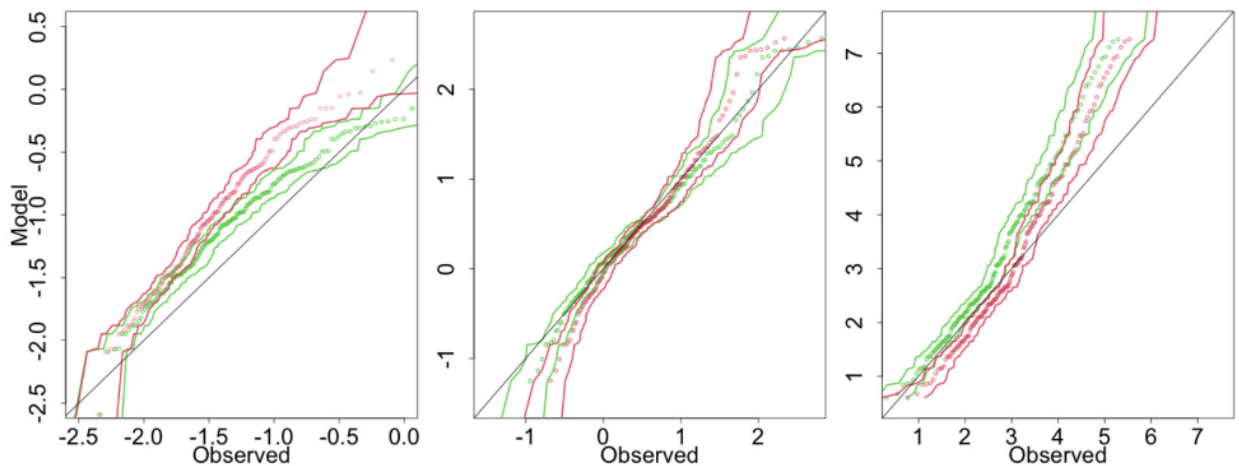

Fig. 8. Comparison of the observed versus predicted minima (left), mean (middle), and maxima (right) with 95% confidence intervals from the pairwise likelihood (red) and CNN (green) fits.

Uncertainty Assessment of The CNN Estimates

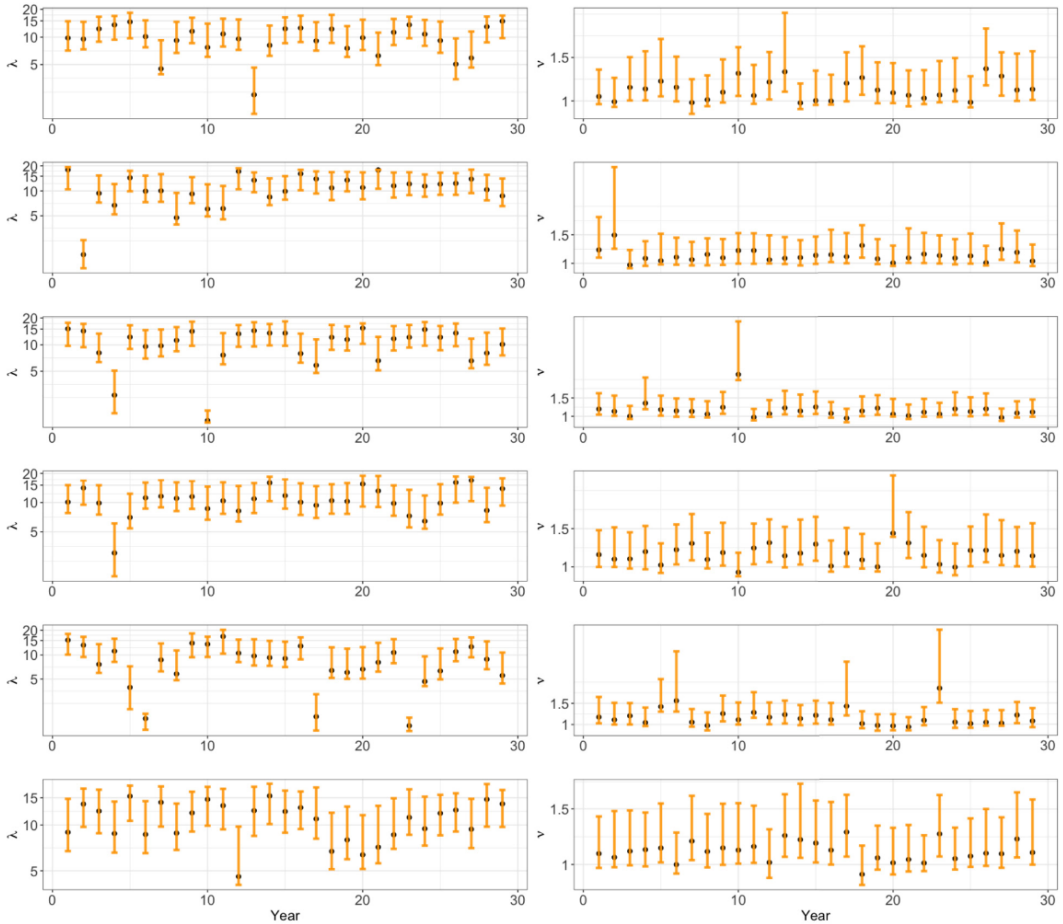

Fig. 9. 95% bootstrap confidence intervals of estimated range (left column) and smoothness (right column) on the transformed scales (with numbers on the axes on the untransformed scales) according to the CNN model. The confidence intervals are calculated for the 29 years of data and are displayed separately for the six 10-day periods over April-May (rows 1-6). Black dots indicate true values.

Uncertainty Assessment of The CNN Estimates

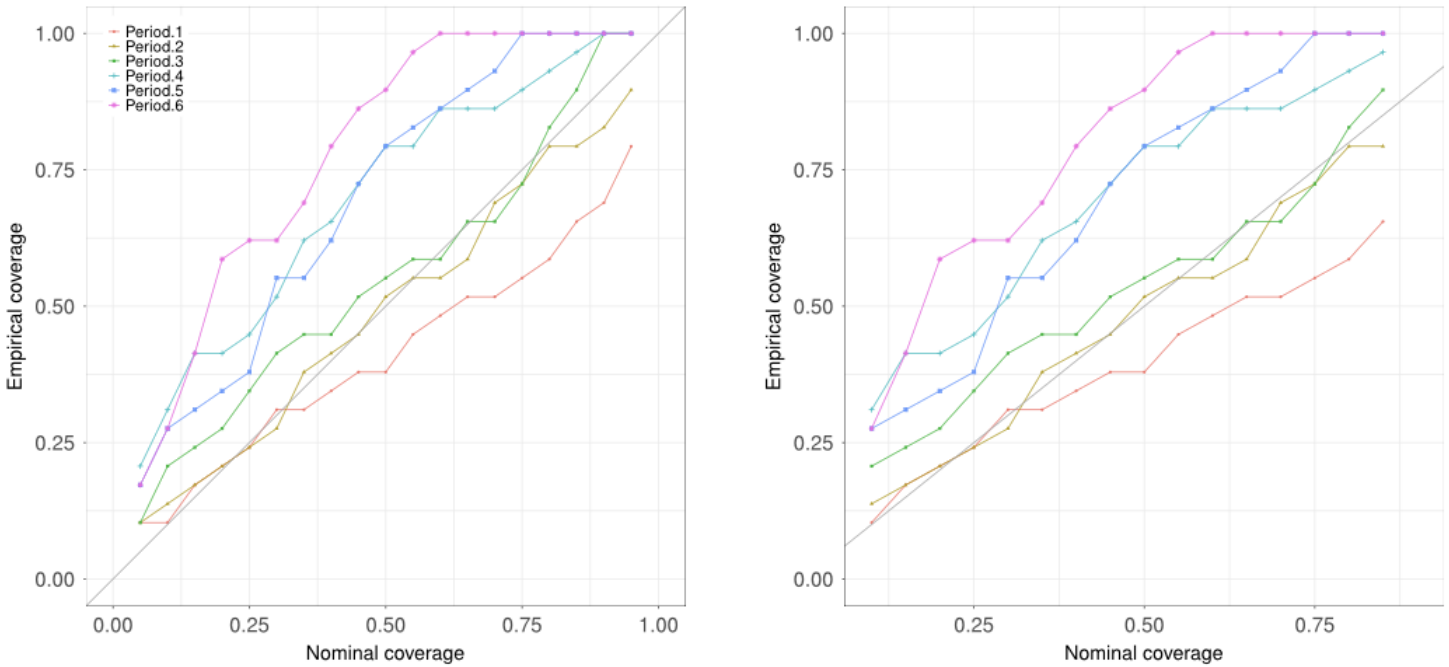

Fig. 10. Empirical versus nominal coverage for different quantiles of the predicted range (left) and smoothness (right) parameters. The lines represent data from each of the six periods over April-May.

Discussion

- Proposed a new **deep learning-based framework** for parameter estimation in statistical models where likelihoods are intractable.
- CNN acts as a **stand-in for classical inference**, learning the mapping between simulated data and parameters directly.
- Demonstrated on **max-stable models (Brown–Resnick, Schlather)** with:
  - Comparable or superior accuracy to pairwise likelihood estimation.
  - Reduced bias and variance in parameter estimates.
- Bootstrapping used for uncertainty quantification, showing calibrated confidence intervals.

Weaknesses and Possible Extensions

- Requires **simulated training data**, which may be computationally demanding for high-dimensional models.
- The framework currently assumes a limited number of parameters — scaling to many parameters remains challenging.
- Future work could explore:
  - Hybrid models (e.g., CNN + LSTM or CNN + MLP).
  - Applications to other statistical models: Poisson, Ising/Potts, stochastic volatility, epidemiology.