LatticeVision: Image to Image Networks for Modeling Non-Stationary Spatial Data

Antony Sikorski[1] **Michael Ivanitskiy**[1] **Nathan Lenssen**[1,2]
Douglas Nychka[1] **Daniel McKenzie**[1]
Colorado School of Mines[1]
NSF National Center for Atmospheric Research[2]

Abstract

In many applications, we wish to fit a parametric statistical model to a small ensemble of spatially distributed random variables ('fields'). However, parameter inference using maximum likelihood estimation (MLE) is computationally prohibitive, especially for large, non-stationary fields. Thus, many recent works train neural networks to estimate parameters given spatial fields as input, sidestepping MLE completely. In this work we focus on a popular class of parametric, spatially autoregressive (SAR) models. We make a simple yet impactful observation; because the SAR parameters can be arranged on a regular grid, both inputs (spatial fields) and outputs (model parameters) can be viewed as images. Using this insight, we demonstrate that image-to-image (I2I) networks enable faster and more accurate parameter estimation for a class of non-stationary SAR models with unprecedented complexity.

1 INTRODUCTION

Modeling large, gridded spatial data has become a central challenge in many scientific and industrial applications. This typically involves fitting parametric spatial models to enable prediction, data fusion, and, when data are limited, rapid simulation of additional fields. The bottleneck in this framework is inferring the parameters of the statistical model using maximum likelihood estimation (MLE), which becomes computationally intractable as dataset size increases (Stein, 2008; Sun et al., 2012). Moreover, spatial data over large domains frequently exhibit *non-stationarity*, meaning key parameters vary over space, further complicating parameter estimation. Many recent works have replaced MLE with neural networks, mapping spatial fields directly to local parameter estimates (Liu et al., 2020; Banesh et al., 2021; Zammit-Mangion et al., 2024; Lenzi et al., 2023). Like MLE, these methods are limited to dividing large fields into smaller sections, each assumed to be stationary, and estimating parameters independently for each section. Although faster than MLE, the number of forward passes required scales linearly with the number of sections. Additionally, such local neural estimators struggle to capture long-range, global context.

In this work we introduce LatticeVision, a global estimation and emulation framework for large, non-stationary spatial data. We observe that for a popular group of statistical models known as spatial autoregressive (SAR) models, the parameters themselves are naturally arranged on a grid. *Thus, both the spatial fields of interest and their associated parameters can be viewed as images.* Consequently, we adapt image-to-image (I2I) networks—both fully convolutional (Ronneberger et al., 2015) and vision transformer based (Dosovitskiy, 2020; Chen et al., 2021)—to the parameter estimation task. Unlike local neural estimators, I2I networks estimate all model parameters at once, in a single forward pass. Our networks are chosen to evaluate whether incorporating the attention mechanism improves performance over purely convolutional approaches (Liu et al., 2022b), which have far fewer parameters. We find that a hybrid approach offers the best performance.

A key challenge in training these networks is ensuring they recognize complex non-stationarity patterns encountered in large, geoscientific data. Since we require (field, parameters) pairs for training, the existing corpora of application specific fields (Nguyen et al., 2023;

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

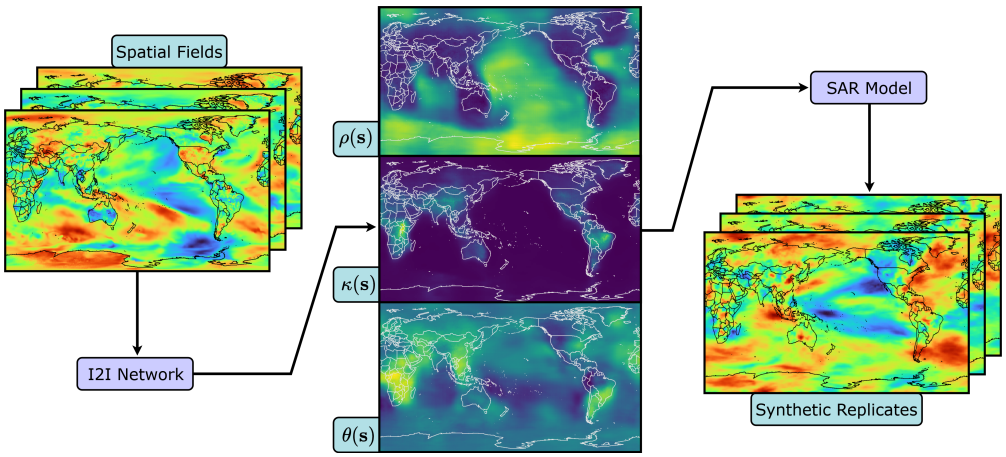

Figure 1: An illustration of the LatticeVision workflow applied to ESM outputs. Spatial fields are fed into an I2I network, which in turn produces estimates of the non-stationary parameter fields. These are encoded into a SAR model from which synthetic replicates are efficiently simulated.

Kaltenborn et al., 2023; Watson-Parris et al., 2022) are inapplicable. Thus we generate our own training data, encoding priors that represent the kind of non-stationary spatial processes expected for geophysical variables.

In experiments with simulated data, I2I networks outperform local neural estimators in both speed and accuracy. This advantage stems from the fact that I2I networks process the entire field at once, rather than by section (Sainsbury-Dale et al., 2024b) or by pixel (Wiens et al., 2020). We also show that I2I networks estimate weakly-identifiable parameters from a small number of replicate spatial fields more reliably than previous, local approaches.

As an illustrative example, we employ the LatticeVision framework on data from multiple Earth System Models (ESMs). ESM simulations ('runs') model the long-term evolution of Earth's climate, providing decision-makers with critical projections. The computational cost of performing more than a handful of ESM runs is extremely prohibitive, limiting ensemble sizes to 3-100 members (Kay et al., 2015; Rodgers et al., 2021); too few for many applications (Milinski et al., 2020; Deser et al., 2020; Schwarzwald and Lenssen, 2022; Eyring et al., 2024). Following parameter estimation, we generate realistic ensembles containing thousands of fields in a matter of seconds using the LatticeKrig package (Nychka et al., 2015); a stark contrast to the tens of millions of core-hours required by ESMs. We show that ensembles simulated with parameters estimated by I2I networks better capture spatial relationships—especially long-range anisotropic correlations—than those produced by local methods.

In summary, we make the following contributions:

- We propose a *global* framework for efficiently estimating non-stationary parameters with I2I networks, demonstrate that hybrid architectures outperform those that are purely convolutional or transformer-based for this task, and address the limitations of existing *local* approaches.
- We provide a strategy for generating training data that encodes scientifically meaningful priors that future estimators and emulators can use.
- We pair our novel, global estimators with a computationally efficient and flexible SAR model (Nychka et al., 2015, 2019), and validate this framework on ESM outputs.

All of the accompanying code is available at github.com/antonyxsik/LatticeVision.

2 BACKGROUND: GAUSSIAN PROCESSES AND SAR MODELS

The past thirty years have seen the development of statistical models for spatial data that are invaluable for spatial prediction and emulation (Heaton et al., 2019; Katzfuss, 2017; Fuentes, 2002; Guhaniyogi and Banerjee, 2018). Despite recent advances in generative deep learning for spatial data (Rühling Cachay et al., 2023; Price et al., 2023; Li et al., 2023), statistical models outperform other approaches when data is limited (Lütjens et al., 2025). Moreover, statistical models contain interpretable parameters, which are useful for downstream applications.

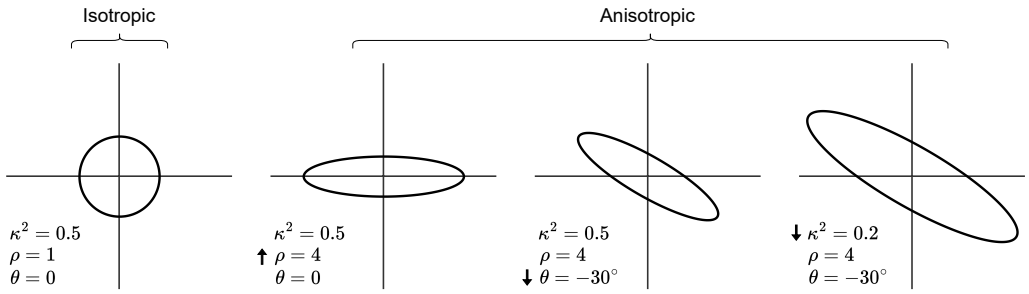

Figure 2: Illustration of the effects of κ^2, ρ, and θ. The ellipses represent contours of constant correlation, e.g. all locations with correlation 0.5 with the origin. κ^2 controls the radii of the ellipse, ρ controls the ratio of the semi-major and semi-minor radii (i.e., the 'aspect ratio' of the ellipse), and θ is the angle the semi-major ellipse makes with the positive x-axis.

We consider statistical models built around Gaussian processes (GPs). With locations $\mathbf{s} \in \mathbb{R}^2$, a mean-zero GP $f(\mathbf{s})$ is fully specified by its covariance function $k(\mathbf{s}, \mathbf{s}') = \mathbb{E}[f(\mathbf{s})f(\mathbf{s}')]$. The kernel k must be positive definite and this requirement is typically enforced by assuming *stationarity* and *isotropy* (Cressie, 2015; Higdon et al., 2022). That is, k is independent of the location in the domain (stationarity), and depends only on the separation distance (isotropy):

$$k(\mathbf{s}, \mathbf{s}') = \sigma^2 \mathcal{C}(\kappa_m \|\mathbf{s} - \mathbf{s}'\|). \tag{1}$$

Here $\mathcal{C}(0) = 1$, σ^2 is the variance of the GP, and $\kappa_m > 0$ governs the spatial correlation range. A popular choice for $\mathcal{C}$ is the Matérn family[1] with an additional smoothness parameter, $\nu > 0$. The Matérn class includes several common kernels as special cases (e.g., exponential and Gaussian). The primary obstacle to working with GP-based models is the computational cost ($\mathcal{O}(n^3)$ operations and $\mathcal{O}(n^2)$ memory for n spatial locations) of factorizing the covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$ where $\Sigma_{ij} = k(\mathbf{s}_i, \mathbf{s}_j)$ (Sun et al., 2012).

The SPDE Method GPs from the Matérn family have an equivalent representation in terms of a stochastic partial differential equation (SPDE) (Matérn, 2013; Whittle, 1954):

$$(\kappa^2 - \Delta)^{(\nu+1)/2} f(\mathbf{s}) = \mathcal{W}(\mathbf{s}), \tag{2}$$

where κ^2 controls the correlation range, and is similar but not identical to κ_m in the Matérn, Δ is the Laplacian operator, and $\mathcal{W}(\mathbf{s})$ is a white noise Gaussian process with zero mean and variance σ^2. The so-called SPDE method (Lindgren et al., 2011, 2022) connects Matérn GPs to Gaussian Markov Random Fields (GMRFs) by demonstrating that discretizing Equation (2) on a regular grid yields a GMRF which approximates the Matérn GP well. This approach can be extended[2] to allow non-stationarity and anisotropy by incorporating a spatially varying dispersion matrix $D(\mathbf{s}) \in \mathbb{R}^{2\times 2}$ and letting κ^2 vary in space:

$$(\kappa^2(\mathbf{s}) - \nabla \cdot D(\mathbf{s})\nabla) f(\mathbf{s}) = \mathcal{W}(\mathbf{s}). \tag{3}$$

Following Haskard et al. (2007), we construct $D(\mathbf{s})$ via its eigendecomposition: $D(\mathbf{s}) = R(\mathbf{s})^\top \Lambda(\mathbf{s}) R(\mathbf{s})$ where

$$R(\mathbf{s}) = \begin{bmatrix} \cos\theta(\mathbf{s}) & -\sin\theta(\mathbf{s}) \\ \sin\theta(\mathbf{s}) & \cos\theta(\mathbf{s}) \end{bmatrix}, \Lambda(\mathbf{s}) = \begin{bmatrix} \rho(\mathbf{s}) & 0 \\ 0 & \frac{1}{\rho(\mathbf{s})} \end{bmatrix} \tag{4}$$

The generalized Laplacian in (3) can be written as

$$\nabla \cdot D(\mathbf{s}) \nabla \equiv D_{1,1}(\mathbf{s}) \frac{\partial^2}{\partial x^2} + 2D_{2,1}(\mathbf{s}) \frac{\partial^2}{\partial x \partial y} + D_{2,2}(\mathbf{s}) \frac{\partial^2}{\partial y^2}. \tag{5}$$

Interpreting the Parameter Fields By specifying the "parameter fields" $\kappa^2(\mathbf{s}), \theta(\mathbf{s})$, and $\rho(\mathbf{s})$ we obtain a rich class of nonstationary GPs. Because we are defining this model in terms of (3), the problem of explicitly specifying an analytical form for the covariance that is positive definite is avoided. Moreover, these parameter fields are interpretable and can yield physical insights into the spatial dependence of the field. Specifically, κ^2 controls the overall range of correlation (larger κ^2 means more localized dependence), ρ controls the degree of anisotropy ($\rho = 1$ corresponds to isotropy), and θ controls the direction of anisotropy; see Figure 2.

Discretizing the SPDE To obtain a computable model from (3), one approximates this SPDE using either a finite element or finite difference method. Following Wiens et al. (2020)[3] we use the finite difference

[1] In reviewing commonly used covariance kernels, Stein (1999) concludes: 'Use the Matérn'.

[2] For simplicity, we shall henceforth focus on the $\nu = 1$ case, the so-called Whittle covariance (Whittle, 1954)

[3] We correct a minor error in their derivation. Our derivation can be found in Appendix A.3

method on a regular grid, yielding the stencil:

$$\begin{array}{c|c|c} \frac{D_{1,2}(\mathbf{s})}{2} & -D_{2,2}(\mathbf{s}) & \frac{-D_{1,2}(\mathbf{s})}{2} \\ \hline -D_{1,1}(\mathbf{s}) & \kappa^2(\mathbf{s}) + 2D_{1,1}(\mathbf{s}) + 2D_{2,2}(\mathbf{s}) & -D_{1,1}(\mathbf{s}) \\ \hline \frac{-D_{1,2}(\mathbf{s})}{2} & -D_{2,2}(\mathbf{s}) & \frac{D_{1,2}(\mathbf{s})}{2}. \end{array} \tag{6}$$

Let $\mathbf{y} \in \mathbb{R}^n$ denote a discretized solution to (3) on this grid, with $y_{i,j}$ denoting the value at grid location (i, j). Then $\mathbf{y}$ is the solution to $B\mathbf{y} = \mathbf{e}$ where $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, I)$ is a sample from the standard multivariate normal distribution and $B \in \mathbb{R}^{n\times n}$ is the spatial autoregressive (SAR) matrix associated to the stencil (6). Lindgren et al. (2011) show that $\mathbf{y}$ approximates a sample from the Matérn GP associated to (3). As B is sparse and structured (*i.e.*, it is a banded matrix) this linear system may be solved at a computational cost $\mathcal{O}(n^{3/2})$, thus sidestepping the bottleneck associated with working with the GP directly. This formulation results in a GMRF with a particular SAR structure, where, by linear statistics, the precision matrix is $Q = B^\top B$.

3 NON-STATIONARY DATA GENERATION

I2I Data In order for the I2I networks to be successful, the training data needs to encode priors appropriate for geoscientific applications. Previous work (Wiens et al., 2020) has shown that coastlines, long-range East-West correlations due to jet streams, and oceanic circulation yield parameter fields that are important yet challenging to detect. So, we construct a pipeline for generating synthetic fields exhibiting these key phenomena. First, we construct spatially varying parameter fields $\kappa^2(\mathbf{s}), \rho(\mathbf{s}), \theta(\mathbf{s})$ and then use them to produce an M-replicate ensemble of synthetic fields $Y = \{\mathbf{y}^{(m)}\}_{m=1}^M$, where each $\mathbf{y}^{(m)} \in \mathbb{R}^{H\times W}$ is generated using the same parameter fields. These parameter fields are concatenated along the channel dimension to form a three channel, ground truth "image" $\Phi \in \mathbb{R}^{3\times H\times W}$, where $\Phi(\mathbf{s}) = [\kappa^2(\mathbf{s}), \rho(\mathbf{s}), \theta(\mathbf{s})]^T$. The replicates of the synthetic fields are also concatenated along the channel dimension to form the input image to our networks $Y \in \mathbb{R}^{M\times H\times W}$. Thus, we have input-output pairs (Y, Φ) in our data.

Each time we construct a single parameter field, we first sample one of eight spatial patterns $p^{(i)}(\mathbf{s}; \mathbf{\Omega}^{(i)})$, $i = 1, 2, \ldots, 8$. The patterns are simple spatial functions that dictate how a parameter will vary across the domain, and are designed to be caricatures of real geophysical variability. We hypothesize that these patterns will serve as "building blocks", enabling our networks to generalize to the kinds of parameter changes present in geophysical settings. Each pattern is defined by its hyperparameters $\mathbf{\Omega}^{(i)}$, which are randomly sampled from a series of prior uniform distributions. For example, when a "Coastline" pattern is selected, values that dictate the position and variation of the "Coastline" are chosen as well. The relative frequency and qualitative descriptions of these patterns are illustrated in Figure 3, with detailed functional forms and hyperparameter priors provided in Appendix B. Once a pattern $p^{(i)}$ and its hyperparameters $\mathbf{\Omega}^{(i)}$ have

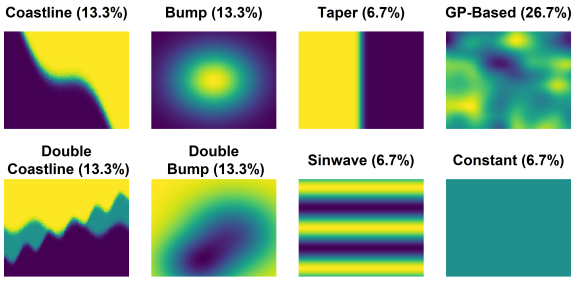

Figure 3: Spatial patterns and their frequencies.

been chosen, we sample values from the prior distribution of the specific parameter (κ^2, ρ, or θ) for which we are constructing a field. Each pattern (except "Constant") requires sampling two values which dictate the maximum and minimum value across the resulting parameter field. We set uniform priors on the anisotropy parameters $\rho \sim \mathcal{U}(1, 7)$ and $\theta \sim \mathcal{U}\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, and set a mixed prior of $\kappa^2 \sim 0.6 \log \mathcal{U}(10^{-4}, 2) + 0.4\mathcal{U}(10^{-4}, 2)$ for the correlation range. These choices ensure we capture a broad range of spatial relationships, ranging from isotropy ($\rho = 1$) to very elongated ellipses ($\rho \gg 1$) in any direction.

We follow this process each time to create a parameter field for $\kappa^2(\mathbf{s}), \rho(\mathbf{s})$, and $\theta(\mathbf{s})$, and then encode all parameters into the SAR matrix B, as described in Section 2. Drawing white noise $\mathbf{e} \sim \mathcal{N}(0, I)$ and solving $B\mathbf{y} = \mathbf{e}$ yields a random field $\mathbf{y}$ with the desired spatial covariance structure. Repeating this for M independent draws produces a small ensemble $Y = \{\mathbf{y}^{(m)}\}_{m=1}^M$ of different synthetic fields with identical covariance structures.

CNN Data For the local estimation setting, we assume local stationarity: within a small spatial window, all grid cells are governed by the same parameters. This makes data generation for the local CNNs comparatively straightforward. Rather than constructing parameter fields, we independently sample a single value for each of (κ^2, ρ, θ) from the prior distributions defined above. These parameters are then encoded into a SAR matrix B, resulting in stationary synthetic fields. We generate smaller fields than for the I2I networks, with each sample consisting of M replicates $\mathcal{Y} \in \mathbb{R}^{M\times h\times w}$, where $h \ll H$ and $w \ll W$. The associated ground truth is a parameter vector $\phi = [\kappa^2, \rho, \theta]^T \in \mathbb{R}^3$. Thus, our input-output pairs are $(\mathcal{Y}, \phi)$.

4 PARAMETER ESTIMATION NETWORKS

We adapt three I2I networks for parameter estimation: a fully convolutional U-Net, a modified ViT, and a hybrid network inspired by the TransUNet architecture. A range of CNNs representative of the local neural estimator literature serve as our baselines.

UNet We use a standard U-Net (Ronneberger et al., 2015) with a symmetric encoder-bottleneck-decoder structure and skip connections that propagate spatial information across resolutions. Our implementation replaces ReLU with GELU activations (Hendrycks and Gimpel, 2016), employs group normalization (Wu and He, 2018), and omits dropout (Srivastava et al., 2014). The number of channels in the bottleneck matches the transformer embedding dimension used in the transformer-based I2I networks.

ViT The original ViT (Dosovitskiy, 2020) was designed for image classification, so we remove classification-specific components such as the class token and final MLP head, and use a learnable linear layer to project the transformer output back to the original image resolution. The original 1D positional embeddings are replaced with a range of 2D alternatives; see Section 5.

STUN Our hybrid architecture, a spatial TransUNet (STUN), is based upon a network originally designed for image segmentation (Chen et al., 2021). It blends the two I2I networks discussed above, originally combining a pretrained convolutional encoder with a vision transformer and a shallow decoder. We retain the overall structure, but replace the encoder and decoder with the symmetric U-Net components used in our fully convolutional network.

CNN Our local neural estimators follow the standard design of convolutional layers and a max pooling layer followed by an MLP. We use three networks with varying receptive window sizes that are representative of the local estimation literature (Banesh et al., 2021; Lenzi et al., 2023; Gerber and Nychka, 2021; Sainsbury-Dale et al., 2024a; Wiens et al., 2020). The key difference is our networks have more parameters (1-2.5M), as compared to typical local estimators (50-700k).

4.1 Implementation and Training Details

All networks are trained with a batch size of $b = 64$ and varying numbers of replicate input fields $M \in \{1, 5, 15, 30\}$. Unless otherwise noted, all reported network sizes and metrics correspond to $M = 30$ replicates. For the I2I networks, each training example consists of M input fields of shape $[b, M, H, W] = [64, M, 192, 288]$, with corresponding output parameter fields $[64, 3, 192, 288]$. For the local CNN estimators, we vary the receptive field $h = w = \{9, 17, 25\}$, with inputs of shape $[64, M, h, w]$ and scalar outputs $[64, 3]$. We append the window size to the name of the CNN, thus $h = w = 25$ is denoted CNN25. To encourage permutation invariance, we randomly shuffle the replicates across the channel dimension each training step.

Certain data augmentation techniques do not preserve the statistical structure of the fields. For example, image rotation does not preserve the angle of anisotropy θ. Thus, we limit augmentation to spatial translation and random field negation. We generate datasets consisting of 8000 (I2I) and 80000 (CNN) samples with 30 replicates, and employ a 90/8/2 (train/validation/test) split, resulting in a test set of 160 samples for the I2I dataset, which both local and global methods are evaluated on. More details on implementation, storage, and software can be found in Appendix C.

5 SIMULATED DATA EXPERIMENTS

We generate a test set of 160 samples following the procedure in Section 3. Each sample is a small ensemble $Y \in \mathbb{R}^{M \times 192 \times 288}$ of M replicates and an associated parameter image $\Phi \in \mathbb{R}^{3 \times 192 \times 288}$. For the I2I networks (UNet, ViT, STUN) we simply forward–propagate Y to obtain $\hat{\Phi}_{\text{I2I}}$. For the local CNN baselines, we employ a pixel-by-pixel approach: we translate the CNN window across the field with a stride of 1, assigning the prediction to the central pixel of the window to build $\hat{\Phi}_{\text{CNN}}$. We use reflection padding—improving upon prior works which use zero padding—thus reducing edge-artifacts. Figure 4 contrasts an example Φ with estimates from the best performing global (STUN) and local (CNN25) networks. MLE is not attempted; local likelihood evaluation for one field would require hundreds of hours, exceeding the total runtime of our approach by orders of magnitude (Wiens et al., 2020).

Positional Embeddings The spatial nature of our problem leads us to study the effects of positional embeddings. We evaluate four embeddings for the transformer: none, 2-D sinusoidal (Parmar et al., 2018), learned 2-D (Dosovitskiy, 2020), and rotary (RoPE) (Su et al., 2024). RoPE yields the best performance across the majority of metrics for both ViT and STUN, and is adopted henceforth (see Appendix Table 2).

Table 1: Root mean square error (RMSE) for parameter estimation on a simulated test set using 1 and 30 replicates (reps). "Size" is the net parameter count (in millions). "Train" is the wall-clock time for training until early stopping, and "Eval" is the average time (out of 5) to process the 160 sample test dataset. Arrows indicate desirable direction, and bold values indicate best performance.

Net	30 Rep RMSE ↓			1 Rep RMSE ↓			Size (M)↓	Train (min)↓	Eval (sec)↓
	κ^2	ρ	θ	κ^2	ρ	θ			
CNN9	0.963	0.937	0.316	1.01	1.39	0.535	**1.3**	**8**	71.3
CNN17	0.765	0.994	0.293	0.766	1.27	0.443	1.9	19	163.0
CNN25	0.743	1.03	0.272	0.806	1.28	0.418	2.6	35	341.6
ViT	0.374	0.625	0.204	0.471	0.771	0.237	92	161	**0.30**
UNet	0.201	0.308	**0.087**	0.195	0.354	0.111	25	144	0.33
STUN	**0.189**	**0.302**	0.091	**0.189**	**0.351**	**0.097**	105	178	0.38

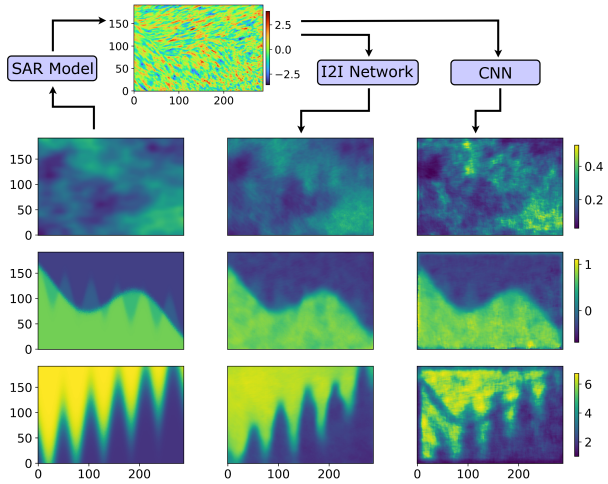

Figure 4: True parameters Φ (**left**) are encoded into the LatticeKrig SAR model to simulate a testing sample Y, of which one replicate $\mathbf{y}^{(0)}$ is displayed (**top**). Y is used as an input to STUN and a sliding window, local estimation strategy using CNN25, resulting in $\hat{\Phi}_{\text{STUN}}$ (**middle**), and $\hat{\Phi}_{\text{CNN25}}$ (**right**).

Number of Replicates We observe the effect on parameter estimation performance with varying numbers of replicates $M = \{1, 5, 15, 30\}$. Table 1 compares results for the extremes of this range. We find that I2I networks are more resilient to a low number of replicates, with STUN and UNet showing almost no difference in prediction RMSE. Full results are in Appendix Tables 3, 4.

Our experiments highlight three consistent trends. (i) Global I2I networks pose a significant improvement over local CNNs: STUN and UNet often display 4-5x lower RMSE and are almost unaffected by shrinking the ensemble size, whereas CNNs are noticeably sensitive. (ii) Adding attention helps only marginally in this setting, and attention on its own lags behind approaches that include multiscale convolutions. STUN edges out UNet but at 4x the parameter count. ViT lags behind the other I2I networks, perhaps as it requires more training data (Dosovitskiy, 2020). (iii) Global I2I networks take longer to train due to a higher parameter count, yet are *amortized* at inference time. In order to achieve the results of a single forward pass through an I2I network, the local estimators must perform $H \times W = 55,296$ forward passes. Consequently, I2I networks perform inference 100–1000 times faster than local neural estimators.

6 CLIMATE APPLICATION

We evaluate our framework by estimating parameters from climate model outputs, using these parameters to generate large, synthetic ensembles, and then comparing the quality of the I2I-based and local CNN-based emulators. We consider ensembles of surface temperature sensitivity fields from three climate models with differing numbers of replicates and resolutions: MPI-ESM (50 members, 192×96 resolution, Olonscheck et al. (2023)), CESM1 (30, 288×192, Kay et al. (2015)), and EC-Earth3 (72, 512×256, Eyring et al. (2016)). While diffusion networks have recently shown promise in adjacent emulation applications (Rühling Cachay et al., 2024; Bassetti et al., 2023), only having 30-72 fields per model renders the training of generative models impractical in this setting.

The fields represent the local changes in temperature given an increase of 1°C in global temperature. Due to the chaotic nature of the ocean-atmosphere system, we can treat each field within an ensemble as an independent replicate sampled from the model's "true" climate sensitivity. The data is preprocessed into standardized temperature sensitivity anomalies where each pixel has zero mean and unit variance. We then perform parameter estimation with all global I2I and local CNN estimators on 30 randomly selected fields from each ensemble. Regardless of the estimation method

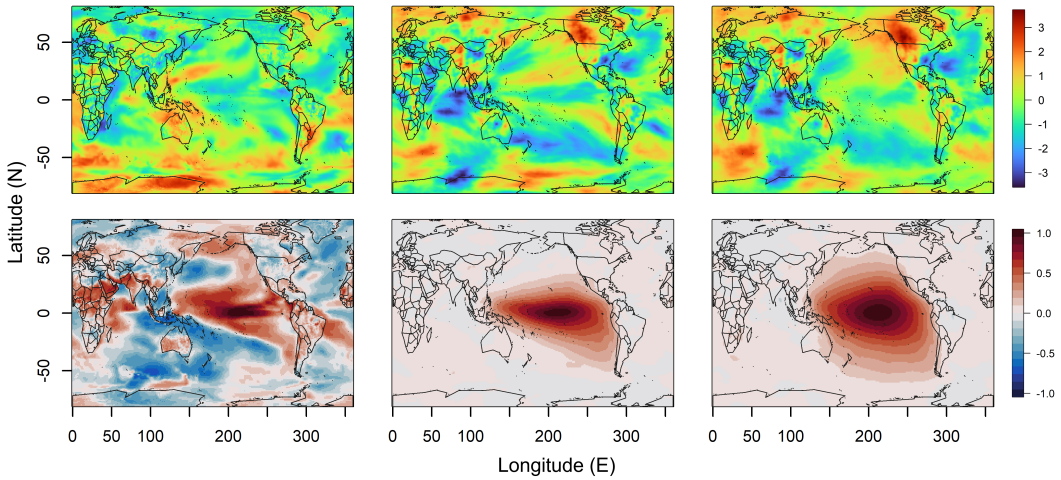

Figure 5: **Top:** Standardized temperature fields drawn from the CESM1 ensemble (**left**), the STUN-based emulator (**middle**), and the CNN25-based emulator (**right**). **Bottom:** Correlations with a chosen location in the Niño 3.4 region at (212°E, 1°N) for the same three ensembles. The STUN-based emulator better preserves spatial relationships, including the zonal correlation structure along the equator and the meridional oceanic correlation range.

used, the LatticeKrig SAR model simulates 1000 fields in less than one minute on a single laptop CPU, a stark contrast to the tens of millions of core hours required to generate the original ensembles.

For each climate model, we perform the following experiment: We use the I2I-based and CNN-based emulators to generate 1000 fields each, and evaluate how well these synthetic ensembles preserve the spatial relationships present in the climate model outputs. Absolute prediction error is not meaningful here as the "truth" is itself a Monte-Carlo sample of internal climate variability. Thus, we compare the second-order structure through a covariance analysis. Rather than computing the entire correlation matrices of the true and simulated fields, which can get as large as $131,072 \times 131,072$ (EC-Earth3), we compare a representative sample of 50 rows.

First, we randomly select 50 anchor locations. We then calculate each anchor's Pearson correlation with every other location for the original fields, and those from the I2I-based and CNN-based emulators. The average RMSE between the 50 true rows and their simulated counterparts is then calculated. **We find that I2I-based emulators consistently outperform CNN-based emulators in capturing spatial relationships**, exhibiting significantly lower RMSEs in eight of nine cases (paired t-tests; Appendix Table 5).

As a representative qualitative comparison, we visualize empirical correlations in CESM1 with a chosen point in the Niño 3.4 region—a region of the tropical Pacific most correlated with the El Niño–Southern Oscillation phenomenon (Barnston et al., 1997)—for the STUN-based and CNN25-based emulators (Figure 5). The STUN-based emulator better preserves the expected zonal east-west correlation structure along the equator with a more realistic correlation range. The CNN-based emulator systematically over-smooths and inflates the oceanic correlation range, particularly in the meridional north-south direction. The original ensemble correlation field is noisier as the CESM1 ensemble contains only 30 members; whereas our synthetic ensembles contain 1,000. These patterns remain consistent across climate models and network pairs. In sum, the I2I-based methods outperform the local CNN-based methods both quantitatively and qualitatively in representing the underlying correlation structure of climate sensitivity fields.

7 RELATED WORKS

Neural Parameter Estimation In situations where the likelihood function is intractable, but simulation from the model is feasible, neural networks have emerged as a powerful alternative to MLE (Liu et al., 2020; Zammit-Mangion et al., 2024). Neural parameter estimation combines ideas from simulation-based inference (Cranmer et al., 2020; Sisson et al., 2018) and learning to optimize (L2O) (Chen et al., 2022; Yin et al., 2022) by training networks to identify maxima of the Bayes risk (Sainsbury-Dale et al., 2024a). The training cost is then amortized by repeated use. Until now, these approaches have been limited to local estimation (Banesh et al., 2021; Lenzi et al., 2023; Sainsbury-Dale et al., 2024a; Rai et al.,

2024; Walchessen et al., 2024; Gerber and Nychka, 2021; Rai et al., 2025), typically using multi-layer perceptrons (MLPs) and CNNs. This line of research is the main inspiration for this work, which advances the field with simultaneous, non-stationary parameter inference and the use of I2I networks.

Climate Model Applications While climate model emulation is not the sole application of our method, it provides an illustrative example. Physics-based climate models require 10^7–10^8 core hours, necessitating data-driven emulation (Eyring et al., 2024). Despite recent advances in deterministic (Pathak et al., 2022; Lam et al., 2023; Nguyen et al., 2024; Bi et al., 2023; Nathaniel et al., 2024; Keisler, 2022; Chen et al., 2023) and ensemble-based probabilistic (Kochkov et al., 2024; Price et al., 2023; Li et al., 2023; Shi et al., 2024) weather forecasting, comparatively fewer methods focus explicitly on long-term climate projections (Lai et al., 2024; Kashinath et al., 2021). Our approach draws ideas from statistical methods, which explicitly link parameter estimation with emulation (Castruccio et al., 2014; Song et al., 2024; Nychka et al., 2018; Wiens et al., 2020; Chakraborty and Katzfuss, 2025), and machine learning (ML) methods, both deterministic (Nguyen et al., 2023; Watt-Meyer et al., 2023; Chapman et al., 2025) and probabilistic (Rühling Cachay et al., 2024, 2023) which emulate directly from initial forcings or prior timesteps. Specifically, we combine the straightforward uncertainty quantification and interpretable parameters from the statistical model with a deterministic ML approach for efficient parameter estimation. Purely ML-based climate emulators typically require an extensive corpus of training data (Watson-Parris et al., 2022; Yu et al., 2023) and can exhibit limited generalization capabilities beyond their training distributions (Kaltenborn et al., 2023). We sidestep this by generating synthetic training data that mimics non-stationarity in geophysical settings. Our framework does not serve as a replacement for ESMs, but rather a complementary method for augmenting ensemble sizes, and reducing the number of "ground truth" ESM runs that must be computed.

8 LIMITATIONS AND EXTENSIONS

The limitations of this work fall into two categories: those of the I2I networks, and those of the statistical model.

Estimation Networks Our I2I networks assume complete, regularly gridded data. Replicate count is also fixed at training time, with only a soft constraint to enforce permutation invariance. A hard constraint via aggregation (Zaheer et al., 2017) might resolve this, and warrants further exploration. Transformer-based networks would likely benefit from larger datasets or advanced training techniques such as student-teacher distillation (Touvron et al., 2021) and a larger dataset. Future work could extend the data generation pipeline to accommodate variable training dataset dimensions and explore scalable attention mechanisms for larger spatial domains (Cao et al., 2022). Our network choices establish a baseline, but there exist many additional architectures for future work to explore (Liu et al., 2022a; Rao et al., 2022; Kim et al., 2022; Wang et al., 2021; Bao et al., 2023).

Statistical Model We adopt a Gaussian process framework via SAR approximation, which enforces monotonic decay of the covariance and cannot capture teleconnections or nonlinear dynamics such as eddies. Our current formulation omits explicit modeling of an additional white noise process, and could benefit from increased smoothing in areas with long-range correlation structures. Extensions to this work could explore multi-resolution structures (Katzfuss, 2017; Sainsbury-Dale et al., 2021; Nychka et al., 2015), estimate a spatially varying noise term, and approximate nonlocal dependence patterns that arise in physical systems.

While our framework enables efficient parameter estimation and simulation of plausible ensembles, it ultimately inherits the assumptions and constraints of the underlying statistical model. Applications across a broad range of fields such as epidemiology, hydrology, or materials science are feasible, although they may require tailoring the data generation strategy and retraining.

9 CONCLUSION

We introduce LatticeVision, a global, image-to-image (I2I) framework for the estimation and emulation of non-stationary spatial processes. By representing both the spatial fields and parameters as images, we use I2I networks to estimate all parameters simultaneously, in a single forward pass. We also develop a novel pipeline for generating non-stationary training data. We show that I2I networks demonstrate improvements in accuracy, robustness with few replicates, estimation speed, and ability to capture long-range, anisotropic correlations, as compared to local approaches. We pair I2I parameter estimators with the LatticeKrig SAR model, enabling fast simulation of large ensembles for non-stationary spatial data.

10 ACKNOWLEDGEMENTS

We would like to thank Ryan Peterson, Ryker Fish, Brandon Knutson, Sweta Rai, Samy Wu Fung, and our anonymous reviewers for their insights and helpful suggestions. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2137099. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Nathan Lenssen is partially funded through NCAR which is sponsored by the National Science Foundation under Cooperative Agreement 1852977.

References

Banesh, D., Panda, N., Biswas, A., Van Roekel, L., Oyen, D., Urban, N., Grosskopf, M., Wolfe, J., and Lawrence, E. (2021). Fast Gaussian process estimation for large-scale in situ inference using convolutional neural networks. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 3731–3739. IEEE.

Bao, Y., Sivanandan, S., and Karaletsos, T. (2023). Channel vision transformers: An image is worth 1 x 16 x 16 words. *arXiv preprint arXiv:2309.16108*.

Barnston, A. G., Chelliah, M., and Goldenberg, S. B. (1997). Documentation of a highly ENSO-related SST region in the equatorial Pacific: Research note. *Atmosphere-ocean*, 35(3):367–383.

Bassetti, S., Hutchinson, B., Tebaldi, C., and Kravitz, B. (2023). Diffesm: Conditional emulation of earth system models with diffusion models. *arXiv preprint arXiv:2304.11699*.

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., and Tian, Q. (2023). Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538.

Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., and Wang, M. (2022). Swin-unet: Unet-like pure transformer for medical image segmentation. In *European conference on computer vision*, pages 205–218. Springer.

Castruccio, S., McInerney, D. J., Stein, M. L., Liu Crouch, F., Jacob, R. L., and Moyer, E. J. (2014). Statistical emulation of climate model projections based on precomputed GCM runs. *Journal of Climate*, 27(5):1829–1844.

Chakraborty, A. and Katzfuss, M. (2025). Learning non-Gaussian spatial distributions via Bayesian transport maps with parametric shrinkage. *Journal of Agricultural, Biological and Environmental Statistics*, pages 1–19.

Chapman, W. E., Schreck, J. S., Sha, Y., Gagne II, D. J., Kimpara, D., Zanna, L., Mayer, K. J., and Berner, J. (2025). Camulator: Fast emulation of the community atmosphere model. *arXiv preprint arXiv:2504.06007*.

Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A. L., and Zhou, Y. (2021). Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*.

Chen, K., Han, T., Gong, J., Bai, L., Ling, F., Luo, J.-J., Chen, X., Ma, L., Zhang, T., Su, R., et al. (2023). Fengwu: Pushing the skillful global medium-range weather forecast beyond 10 days lead. *arXiv preprint arXiv:2304.02948*.

Chen, T., Chen, X., Chen, W., Heaton, H., Liu, J., Wang, Z., and Yin, W. (2022). Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23(189):1–59.

Cranmer, K., Brehmer, J., and Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062.

Cressie, N. (2015). *Statistics for spatial data*. John Wiley & Sons.

Deser, C., Lehner, F., Rodgers, K. B., Ault, T., Delworth, T. L., DiNezio, P. N., Fiore, A., Frankignoul, C., Fyfe, J. C., Horton, D. E., et al. (2020). Insights from Earth system model initial-condition large ensembles and future prospects. *Nature Climate Change*, 10(4):277–286.

Dosovitskiy, A. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., and Taylor, K. E. (2016). Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development*, 9(5):1937–1958.

Eyring, V., Collins, W. D., Gentine, P., Barnes, E. A., Barreiro, M., Beucler, T., Bocquet, M., Bretherton, C. S., Christensen, H. M., Dagon, K., et al. (2024). Pushing the frontiers in climate modelling and analysis with machine learning. *Nature Climate Change*, 14(9):916–928.

Fuentes, M. (2002). Spectral methods for nonstationary spatial processes. *Biometrika*, 89(1):197–210.

Gerber, F. and Nychka, D. (2021). Fast covariance parameter estimation of spatial Gaussian process models using neural networks. *Stat*, 10(1):e382.

Guhaniyogi, R. and Banerjee, S. (2018). Meta-kriging: Scalable Bayesian modeling and inference for massive spatial datasets. *Technometrics*, 60(4):430–444.

Haskard, K. A., Cullis, B. R., and Verbyla, A. P. (2007). Anisotropic Matérn correlation and spatial prediction using REML. *Journal of agricultural, biological, and environmental statistics*, 12:147–160.

Heaton, M. J., Datta, A., Finley, A. O., Furrer, R., Guinness, J., Guhaniyogi, R., Gerber, F., Gramacy, R. B., Hammerling, D., Katzfuss, M., et al. (2019). A case study competition among methods for analyzing large spatial data. *Journal of agricultural, biological and environmental Statistics*, 24:398–425.

Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Higdon, D., Swall, J., and Kern, J. (2022). Nonstationary spatial modeling. *arXiv preprint arXiv:2212.08043*.

Kaltenborn, J., Lange, C., Ramesh, V., Brouillard, P., Gurwicz, Y., Nagda, C., Runge, J., Nowack, P., and Rolnick, D. (2023). Climateset: A large-scale climate model dataset for machine learning. *Advances in Neural Information Processing Systems*, 36:21757–21792.

Kashinath, K., Mustafa, M., Albert, A., Wu, J., Jiang, C., Esmaeilzadeh, S., Azizzadenesheli, K., Wang, R., Chattopadhyay, A., Singh, A., et al. (2021). Physics-informed machine learning: Case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, 379(2194):20200093.

Katzfuss, M. (2017). A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, 112(517):201–214.

Kay, J. E., Deser, C., Phillips, A., Mai, A., Hannay, C., Strand, G., Arblaster, J. M., Bates, S., Danabasoglu, G., Edwards, J., et al. (2015). The Community Earth System Model (CESM) large ensemble project: A community resource for studying climate change in the presence of internal climate variability. *Bulletin of the American Meteorological Society*, 96(8):1333–1349.

Keisler, R. (2022). Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575*.

Kim, S., Baek, J., Park, J., Kim, G., and Kim, S. (2022). Instaformer: Instance-aware image-to-image translation with transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18321–18331.

Kochkov, D., Yuval, J., Langmore, I., Norgaard, P., Smith, J., Mooers, G., Klöwer, M., Lottes, J., Rasp, S., Düben, P., et al. (2024). Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066.

Lai, C.-Y., Hassanzadeh, P., Sheshadri, A., Sonnewald, M., Ferrari, R., and Balaji, V. (2024). Machine learning for climate physics and simulations. *Annual Review of Condensed Matter Physics*, 16.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., et al. (2023). Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421.

Lenzi, A., Bessac, J., Rudi, J., and Stein, M. L. (2023). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, 185:107762.

Li, L., Carver, R., Lopez-Gomez, I., Sha, F., and Anderson, J. (2023). Seeds: Emulation of weather forecast ensembles with diffusion models. *arXiv preprint arXiv:2306.14066*.

Lindgren, F., Bolin, D., and Rue, H. (2022). The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running. *Spatial Statistics*, 50:100599.

Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 73(4):423–498.

Liu, S., Sun, X., Ramadge, P. J., and Adams, R. P. (2020). Task-agnostic amortized inference of Gaussian process hyperparameters. *Advances in Neural Information Processing Systems*, 33:21440–21452.

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al. (2022a). Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. (2022b). A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lütjens, B., Ferrari, R., Watson-Parris, D., and Selin, N. E. (2025). The impact of internal variability

on benchmarking deep learning climate emulators. *Journal of Advances in Modeling Earth Systems*, 17(8):e2024MS004619.

Matérn, B. (2013). *Spatial variation*, volume 36. Springer Science & Business Media.

Milinski, S., Maher, N., and Olonscheck, D. (2020). How large does a large ensemble need to be? *Earth System Dynamics*, 11(4):885–901.

Nathaniel, J., Qu, Y., Nguyen, T., Yu, S., Busecke, J., Grover, A., and Gentine, P. (2024). Chaosbench: A multi-channel, physics-based benchmark for subseasonal-to-seasonal climate prediction. *arXiv preprint arXiv:2402.00712*.

Nguyen, T., Brandstetter, J., Kapoor, A., Gupta, J. K., and Grover, A. (2023). ClimaX: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*.

Nguyen, T., Shah, R., Bansal, H., Arcomano, T., Maulik, R., Kotamarthi, R., Foster, I., Madireddy, S., and Grover, A. (2024). Scaling transformer neural networks for skillful and reliable medium-range weather forecasting. *Advances in Neural Information Processing Systems*, 37:68740–68771.

Nychka, D., Bandyopadhyay, S., Hammerling, D., Lindgren, F., and Sain, S. (2015). A multiresolution Gaussian process model for the analysis of large spatial datasets. *Journal of Computational and Graphical Statistics*, 24(2):579–599.

Nychka, D., Hammerling, D., Krock, M., and Wiens, A. (2018). Modeling and emulation of nonstationary Gaussian fields. *Spatial statistics*, 28:21–38.

Nychka, D., Hammerling, D., Sain, S., Lenssen, N., and Nychka, M. D. (2019). Package 'LatticeKrig'.

Olonscheck, D., Suarez-Gutierrez, L., Milinski, S., Beobide-Arsuaga, G., Baehr, J., Fröb, F., Ilyina, T., Kadow, C., Krieger, D., Li, H., et al. (2023). The new max planck institute grand ensemble with cmip6 forcing and high-frequency model output. *Journal of Advances in Modeling Earth Systems*, 15(10):e2023MS003790.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., and Tran, D. (2018). Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR.

Paszke, A. (2019). Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Pathak, J., Subramanian, S., Harrington, P., Raja, S., Chattopadhyay, A., Mardani, M., Kurth, T., Hall, D., Li, Z., Azizzadenesheli, K., et al. (2022). Fourcastnet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv preprint arXiv:2202.11214*.

Price, I., Sanchez-Gonzalez, A., Alet, F., Andersson, T. R., El-Kadi, A., Masters, D., Ewalds, T., Stott, J., Mohamed, S., Battaglia, P., et al. (2023). Gencast: Diffusion-based ensemble forecasting for medium-range weather. *arXiv preprint arXiv:2312.15796*.

Rai, S., Hoffman, A., Lahiri, S., Nychka, D. W., Sain, S. R., and Bandyopadhyay, S. (2024). Fast parameter estimation of generalized extreme value distribution using neural networks. *Environmetrics*, 35(3):e2845.

Rai, S., Nychka, D. W., and Bandyopadhyay, S. (2025). Modeling spatial extremes using non-Gaussian spatial autoregressive models via convolutional neural networks. *arXiv preprint arXiv:2505.03034*.

Rao, Y., Zhao, W., Tang, Y., Zhou, J., Lim, S. N., and Lu, J. (2022). Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *Advances in Neural Information Processing Systems*, 35:10353–10366.

Rodgers, K. B., Lee, S.-S., Rosenbloom, N., Timmermann, A., Danabasoglu, G., Deser, C., Edwards, J., Kim, J.-E., Simpson, I. R., Stein, K., et al. (2021). Ubiquity of human-induced changes in climate variability. *Earth System Dynamics*, 12(4):1393–1411.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer.

Rühling Cachay, S., Henn, B., Watt-Meyer, O., Bretherton, C. S., and Yu, R. (2024). Probablistic emulation of a global climate model with spherical DYffusion. *Advances in Neural Information Processing Systems*, 37:127610–127644.

Rühling Cachay, S., Zhao, B., Joren, H., and Yu, R. (2023). Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. *Advances in neural information processing systems*, 36:45259–45287.

Sainsbury-Dale, M., Zammit-Mangion, A., and Cressie, N. (2021). Modelling big, heterogeneous, non-Gaussian spatial and spatio-temporal data using FRK. *arXiv preprint arXiv:2110.02507*.

Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2024a). Likelihood-free parameter estimation with neural Bayes estimators. *The American Statistician*, 78(1):1–14.

Sainsbury-Dale, M., Zammit-Mangion, A., Richards, J., and Huser, R. (2024b). Neural Bayes estimators for irregular spatial data using graph neural networks. *Journal of Computational and Graphical Statistics*, (just-accepted):1–28.

Schwarzwald, K. and Lenssen, N. (2022). The importance of internal climate variability in climate impact projections. *Proceedings of the National Academy of Sciences*, 119(42):e2208095119.

Shi, J., Jin, B., Han, J., and Narasimhan, G. (2024). Codicast: Conditional diffusion model for weather prediction with uncertainty quantification. *arXiv preprint arXiv:2409.05975*.

Sikorski, A., McKenzie, D., and Nychka, D. (2024). Normalizing basis functions: Approximate stationary models for large spatial data. *Stat*, 13(4):e70015.

Sisson, S. A., Fan, Y., and Beaumont, M. (2018). *Handbook of approximate Bayesian computation*. CRC press.

Song, Y., Khalid, Z., and Genton, M. G. (2024). Efficient stochastic generators with spherical harmonic transformation for high-resolution global climate simulations from CESM2-LENS2. *Journal of the American Statistical Association*, 119(548):2493–2507.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Stein, M. L. (1999). *Interpolation of spatial data: Some theory for kriging*. Springer Science & Business Media.

Stein, M. L. (2008). A modeling approach for large spatial datasets. *Journal of the Korean Statistical Society*, 37:3–10.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. (2024). Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Sun, Y., Li, B., and Genton, M. (2012). Geostatistics for large datasets.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR.

Walchessen, J., Lenzi, A., and Kuusela, M. (2024). Neural likelihood surfaces for spatial processes with computationally intensive or intractable likelihoods. *Spatial Statistics*, 62:100848.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. (2021). Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578.

Watson-Parris, D., Rao, Y., Olivié, D., Seland, Ø., Nowack, P., Camps-Valls, G., Stier, P., Bouabid, S., Dewey, M., Fons, E., et al. (2022). ClimateBench v1.0: A benchmark for data-driven climate projections. *Journal of Advances in Modeling Earth Systems*, 14(10):e2021MS002954.

Watt-Meyer, O., Dresdner, G., McGibbon, J., Clark, S. K., Henn, B., Duncan, J., Brenowitz, N. D., Kashinath, K., Pritchard, M. S., Bonev, B., et al. (2023). ACE: A fast, skillful learned global atmospheric model for climate prediction. *arXiv preprint arXiv:2310.02074*.

Whittle, P. (1954). On stationary processes in the plane. *Biometrika*, pages 434–449.

Wiens, A., Nychka, D., and Kleiber, W. (2020). Modeling spatial data using local likelihood estimation and a Matérn to spatial autoregressive translation. *Environmetrics*, 31(6):e2652.

Wu, Y. and He, K. (2018). Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19.

Yin, W., McKenzie, D., and Fung, S. W. (2022). Learning to optimize: Where deep learning meets optimization and inverse problems. *SIAM News*.

Yu, S., Hannah, W., Peng, L., Lin, J., Bhouri, M. A., Gupta, R., Lütjens, B., Will, J. C., Behrens, G., Busecke, J., et al. (2023). ClimSim: A large multi-scale dataset for hybrid physics-ML climate emulation. *Advances in Neural Information Processing Systems*, 36:22070–22084.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. *Advances in neural information processing systems*, 30.

Zammit-Mangion, A., Sainsbury-Dale, M., and Huser, R. (2024). Neural methods for amortized inference. *Annual Review of Statistics and Its Application*, 12.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] We detail the statistical model in Section 2, and the estimation networks in Section 4.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] We explain these in Section 2.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes] We provide all source code in the supplementary material. Dependencies are explicitly specified in both the requirements.txt and pyproject.toml files.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Not Applicable]
   (b) Complete proofs of all theoretical results. [Not Applicable]
   (c) Clear explanations of any assumptions. [Not Applicable]
   Explanation for all: While this paper does provide mathematical background and derivations both in the main body and in the supplementary material, we do not prove any new, theoretical results.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] We provide this in the supplementary material. A google drive with sample data is linked, data generation scripts are provided, and instructions to reproduce all experiments are available in a README.md file.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes] The experimental setting presented in the core of the paper allows the reader to appreciate the results, while additional implementation details such as the hyperparameters and optimizer are detailed in the supplementary material (Appendix C).
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes] Confidence intervals and statistical significance are provided for the comparisons in Section 6 (Climate application). We do not provide error bars in the results for the simulated data in Section 5 and the accompanying appendix Tables 3, 4, as this would require too large of a computational effort. All networks are evaluated across minor changes in the ablation tables in the appendix, providing an estimate of the variability and consistency in those results.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] Details regarding software, hardware (CPU, GPU), and storage are all provided in the supplementary material (Appendix C).

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes] We use open source software packages (within their terms of usage) such as PyTorch (Python, license: Modified BSD) and LatticeKrig (R, license: GPL-3).
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes] We will release open source, well documented code via Github with all necessary details to reproduce this work. Currently, it is anonymized and included in the supplementary material.
   (d) Information about consent from data providers/curators. [Not applicable] We generate our own data.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]
   Explanation for all: We do not use crowdsourcing or conduct research with human subjects.

Appendix

A ANISOTROPIC SAR DERIVATION

A.1 The Matérn Covariance

Under the Matérn family, Equation (1) takes the form:

$$k(\mathbf{s}, \mathbf{s}') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\kappa_m \|\mathbf{s} - \mathbf{s}'\|)^\nu \mathcal{K}_\nu(\kappa_m \|\mathbf{s} - \mathbf{s}'\|), \tag{7}$$

where $\Gamma(\cdot)$ is the gamma function, and $\mathcal{K}_\nu(\cdot)$ is the modified Bessel function of the second kind.

A.2 Isotropic SAR

In two dimensions ($\mathbf{s} \in \mathbb{R}^2$), with $\nu = 1$, Equation (2) can be written as

$$(\kappa^2 - \Delta) f(\mathbf{s}) = \mathcal{W}(\mathbf{s}), \tag{8}$$

where $\Delta = \partial^2/\partial x^2 + \partial^2/\partial y^2$ is the Laplacian operator. Let the domain be covered by a square lattice with unit spacing. We denote $f_{i,j} \equiv f(\mathbf{s}_{i,j})$ at grid point $\mathbf{s}_{i,j} = (i, j)$, $i, j \in \{1, \ldots, N\}$.

Using second order central-difference approximations for the Laplacian yields

$$-\Delta f_{i,j} \;\approx\; \big[4f_{i,j} - f_{i+1,j} - f_{i-1,j} - f_{i,j+1} - f_{i,j-1}\big]. \tag{9}$$

Substituting (9) into (8) results in

$$(\kappa^2 + 4)\, f_{i,j} - \big(f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1}\big) \;=\; e_{i,j}, \tag{10}$$

where $e_{i,j}$ denotes the discrete version of the noise $\mathcal{W}$ at location (i, j). Equation (10) can be visualized in lattice notation using the following stencil:

0	-1	0
-1	$\kappa^2 + 4$	-1
0	-1	0

(11)

illustrating the isotropic relationship between $f_{i,j}$ and its neighboring locations.

A.3 Anisotropic Extension

To incorporate geometric anisotropy we replace the Laplacian in (8) by the *generalised* Laplacian $\nabla \cdot D\nabla$, where the positive–definite dispersion matrix $D \in \mathbb{R}^{2\times 2}$ is

$$D \;=\; R^\top \Lambda R, \qquad R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \rho & 0 \\ 0 & \frac{1}{\rho} \end{bmatrix}. \tag{12}$$

Here, R is a rotation matrix parametrized by θ and Λ is a scaling matrix parametrized by ρ. For a constant D the anisotropic SPDE becomes

$$\left(\kappa^2 - \nabla \cdot D\nabla\right) f(\mathbf{s}) = \mathcal{W}(\mathbf{s}). \tag{13}$$

Using unit grid spacing and second-order central differences for the second–derivative terms,

$$\begin{aligned}\nabla \cdot D\nabla f_{i,j} \;\approx\; & D_{1,1}(f_{i+1,j} - 2f_{i,j} + f_{i-1,j}) + D_{2,2}(f_{i,j+1} - 2f_{i,j} + f_{i,j-1}) \\ & + \frac{D_{1,2}}{2}\big[f_{i+1,j+1} - f_{i+1,j-1} - f_{i-1,j+1} + f_{i-1,j-1}\big]. \end{aligned} \tag{14}$$

Substituting the above expression into (13) results in

$$\begin{aligned}\Big(\kappa^2 + 2D_{1,1} + 2D_{2,2}\Big) f_{i,j} \;-\; & D_{1,1}(f_{i+1,j} + f_{i-1,j}) \;-\; D_{2,2}(f_{i,j+1} + f_{i,j-1}) \\ & -\; \frac{D_{1,2}}{2}\big[f_{i+1,j+1} - f_{i+1,j-1} - f_{i-1,j+1} + f_{i-1,j-1}\big] \;=\; e_{i,j}. \end{aligned} \tag{15}$$

Equation (15) corresponds to the 3×3 stencil

$$\begin{array}{c|c|c} \dfrac{D_{1,2}}{2} & -\,D_{2,2} & -\dfrac{D_{1,2}}{2} \\ \hline -\,D_{1,1} & \kappa^2 + 2D_{1,1} + 2D_{2,2} & -\,D_{1,1} \\ \hline -\dfrac{D_{1,2}}{2} & -\,D_{2,2} & \dfrac{D_{1,2}}{2}. \end{array} \tag{16}$$

One incorporates non-stationarity by allowing D and κ^2 to vary in space, as shown in Equation (3).

B NON-STATIONARY DATA GENERATION DETAILS

The synthetic data generation pipeline for the I2I networks constructs parameter fields for $\kappa^2(\mathbf{s})$, $\rho(\mathbf{s})$, and $\theta(\mathbf{s})$. The parameters have the following prior distributions:

$$\kappa^2 \sim 0.6 \log \mathcal{U}(10^{-4}, 2) + 0.4\mathcal{U}(10^{-4}, 2), \quad \rho \sim \mathcal{U}(1, 7), \quad \theta \sim \mathcal{U}(-\frac{\pi}{2}, \frac{\pi}{2}) \tag{17}$$

Each parameter field is independently created by selecting one of several spatial patterns, sampling associated hyperparameters from prior distributions, sampling parameter values for the maxima and minima of the field, and optionally stacking two patterns to increase complexity.

B.1 Spatial Patterns

We design eight simple yet expressive spatial patterns $p(\mathbf{s})$, intended as caricatures of realistic geophysical variability. Here we express p as a generic value for κ^2, ρ, or θ, our pattern's associated hyperparameters as $\mathbf{\Omega}$, and the x and y coordinates on the grid as $\mathbf{s}_x, \mathbf{s}_y$, respectively.

Constant: Uniform value across the domain.

- Functional Form: $p(\mathbf{s}) = p_{\text{constant}}$, where p_{constant} is a constant sampled from the prior.

Coastline: A sharp sigmoidal boundary, perturbed by sinusoidal "bumps".

- Functional Form:
$$p(\mathbf{s}) = p_{\text{low}} + \frac{p_{\text{high}} - p_{\text{low}}}{1 + \exp\left(-\gamma(\mathbf{s}_y - v(\mathbf{s}_x))\right)} \tag{18}$$
where $v(\mathbf{s}_x) = \alpha \mathbf{s}_x + \beta \sin(2\pi\omega \mathbf{s}_x) + \epsilon$, and $p_{\text{low}}, p_{\text{high}}$ are the lower and higher of the two independently sampled parameter values, dictating the maxima and minima of the pattern.

- Hyperparameters: $\mathbf{\Omega} = (\alpha, \beta, \omega, \gamma)$, sampled from $\alpha \sim \mathcal{U}(-2, 2), \beta \sim \mathcal{U}(0.1, 0.5), \omega \sim \mathcal{U}(0.4, 3)$, and $\gamma \sim \mathcal{U}(3, 50)$.

Taper: Smooth transition between two values, based on a Gaussian CDF.

- Functional Form:
$$p(\mathbf{s}) = p_{\text{low}}\Psi(\mathbf{s}_x + \mathbf{s}_y; 0, \sigma) + p_{\text{high}}(1 - \Psi(\mathbf{s}_x + \mathbf{s}_y; 0, \sigma)) \tag{19}$$
where $\Psi(\cdot)$ denotes the standard normal cumulative distribution function (CDF).
- Hyperparameters: $\Omega = \sigma$, with $\sigma \sim \mathcal{U}(0.05, 1)$ controlling the sharpness of the transition.

Bump: A single, smooth, Gaussian peak.

- Functional Form:
$$p(\mathbf{s}) = p_{\text{constant}} + a_1 \exp\left(-\frac{\mathbf{s}_x^2 + \mathbf{s}_y^2}{\lambda_1}\right) \tag{20}$$
- Hyperparameters: $\mathbf{\Omega} = (a_1, \lambda_1)$, with a_1 controlling the peak height and λ_1 controlling spread. In all cases $\lambda_1 \sim \mathcal{U}(0.2, 0.5)$, while a_1 varies. When making a $\kappa^2(\mathbf{s})$ field, $a_1 \sim \mathcal{U}(0.1, 0.5)$. For $\rho(\mathbf{s})$, $a_1 \sim \mathcal{U}(0.1, 1.5)$, and for $\theta(\mathbf{s})$, $a_1 \sim \mathcal{U}(0.1, \frac{\pi}{4})$.

Sinwave: Periodic variation along one spatial axis.

- Functional Form:
$$p(\mathbf{s}) = \begin{cases} p_{\text{constant}} + a\sin(\pi\omega\mathbf{s}_x), & \text{(horizontal)} \\ p_{\text{constant}} + a\cos(\pi\omega\mathbf{s}_y), & \text{(vertical)} \end{cases} \tag{21}$$
- Hyperparameters: $\mathbf{\Omega} = (a, \omega, \text{orientation})$, where $\omega \sim \mathcal{U}(1.5, 5)$, orientation is sampled uniformly between horizontal and vertical, and a is constrained to stay within the bounds of the parameter one is constructing a field for.

Double Bump: Superposition of two independent Gaussian peaks located at different positions.

- Functional Form:
$$\begin{aligned} p(\mathbf{s}) = p_{\text{constant}} &+ a_1 \exp\left(-\frac{(\mathbf{s}_x - x_1)^2 + (\mathbf{s}_y - y_1)^2}{\lambda_1}\right) \\ &+ a_2 \exp\left(-\frac{(\mathbf{s}_x - x_2)^2 + (\mathbf{s}_y - y_2)^2}{\lambda_2}\right) \end{aligned} \tag{22}$$
- Hyperparameters: $\mathbf{\Omega} = (a_1, a_2, \lambda_1, \lambda_2, x_1, y_1, x_2, y_2)$, with amplitudes, widths, and locations sampled as previously described in the above "Bump" configuration, and with centers $(x_1, y_1), (x_2, y_2)$ being randomly sampled locations within the spatial domain.

Double Coastline: Weighted superposition of two independent coastline patterns.

- Functional Form:
$$p(\mathbf{s}) = p_{\text{low}} + w_1\text{Coastline}_1(\mathbf{s}) + w_2\text{Coastline}_2(\mathbf{s}) \tag{23}$$
- Hyperparameters: $w_1 \sim \mathcal{U}(0.1, 0.9), w_2 \sim \mathcal{U}(0.1, 1 - w_1)$, and each coastline has its own, independent $(\alpha, \beta, \omega, \gamma)$ parameters as described in the above "Coastline" configuration.

GP-Based: A smooth, stationary random field generated by a low-rank Gaussian process, either min-max rescaled or perturbed around a constant.

- Functional Form: A realization from a Gaussian process, constructed via low-rank basis function approximation:

$$p(\mathbf{s}) = \begin{cases} p_{\min}, p_{\max} \text{ rescaled field}, & \text{(min-max scaling)} \\ p_{\text{constant}} \times (1 \pm g(\mathbf{s})), & \text{(perturbation scaling)} \end{cases} \tag{24}$$

  where $g(\mathbf{s})$ is a normalized low-rank GP realization, p_{constant} is a parameter value sampled from the prior distribution, and $p_{\min}, p_{\max}$ represent the minimum and maximum of a parameter's prior distribution.

- Hyperparameters: $\mathbf{\Omega} = (n_{\text{basis}}, \text{scaling choice})$, where $n_{\text{basis}} \sim \mathcal{U}\{6, 7, \ldots, 32\}$ controls the number of basis functions used to generate the realization of the GP. The scaling choice is selected randomly: either
  - Min-max scaling: $p(\mathbf{s})$ is rescaled to span the full prior range, or
  - Perturbation scaling: $p(\mathbf{s})$ is a small multiplicative perturbation around p_{constant}, with perturbation magnitude drawn from a prior distribution.

B.2 Pattern Stacking

To further increase the complexity of our fields, for each parameter field we randomly decide whether to linearly combine two independently generated patterns. Given fields $p_1(\mathbf{s})$ and $p_2(\mathbf{s})$, the final field is

$$p(\mathbf{s}) = wp_1(\mathbf{s}) + (1 - w)p_2(\mathbf{s}), \quad w \sim \mathcal{U}(0.1, 0.9) \tag{25}$$

Stacking introduces complex non-stationarity patterns beyond those generated by a single functional form, increasing the range of spatial patterns seen during training.

B.3 Resulting Synthetic Data

We repeat the process of selecting a pattern, sampling the pattern hyperparameters $\mathbf{\Omega}$, sampling the parameter values, and generating the resulting parameter field for $\kappa^2(\mathbf{s}), \rho(\mathbf{s})$, and $\theta(\mathbf{s})$. We then encode all parameter fields into the SAR matrix B, draw white noise $\mathbf{e} \sim \mathcal{N}(0, I)$ and solve $B\,\mathbf{y} = \mathbf{e}$. We repeat this for M independent draws of the white noise, resulting in a small ensemble $Y = \{\mathbf{y}^{(m)}\}_{m=1}^{M}$, with each field having the same covariance structure. We process these synthetic fields identically to the way we process the input ESM fields: we perform pixelwise standardization to ensure each pixel has a mean of 0 and a standard deviation of 1. The non-stationarity in our problem does not allow for more sophisticated normalization techniques (Sikorski et al., 2024), which either become computationally intractable, require a stationary structure, or are not applicable in this setting. An example spatial field and its associated parameter fields can be seen in Figure 6.

C FURTHER IMPLEMENTATION DETAILS

C.1 Data and Storage

All data generation is done in the `R` programming language on a laptop with an Intel(R) Core(TM) i9-14900HX processor at 2.20 GHz, and 32GB of RAM. Fields are generated using the `LatticeKrig` package, and data is compressed and stored using the `hdf5` file format so it may be easily accessed in both `Python` and `R`. Storage proves to be more of a bottleneck as compared to data generation time: the I2I dataset (8,000 samples with 30 replicates) required 8 hours and occupies 108 GB, whereas the CNN dataset (80,000 samples with 30 replicates) is only 11 GB, and is generated in half an hour.

C.2 Training

All networks are implemented in PyTorch (Paszke, 2019) and trained on a single NVIDIA RTX A6000 GPU. The networks are trained using the AdamW optimizer (Loshchilov and Hutter, 2017) for 200 epochs with a step-wise learning rate decay and early stopping after 10 epochs without validation improvement. We use mean squared error (MSE) loss, computed on normalized parameter values within the training loop. This avoids loss imbalance caused by parameter scale differences while requiring less pre or post-processing from the user.

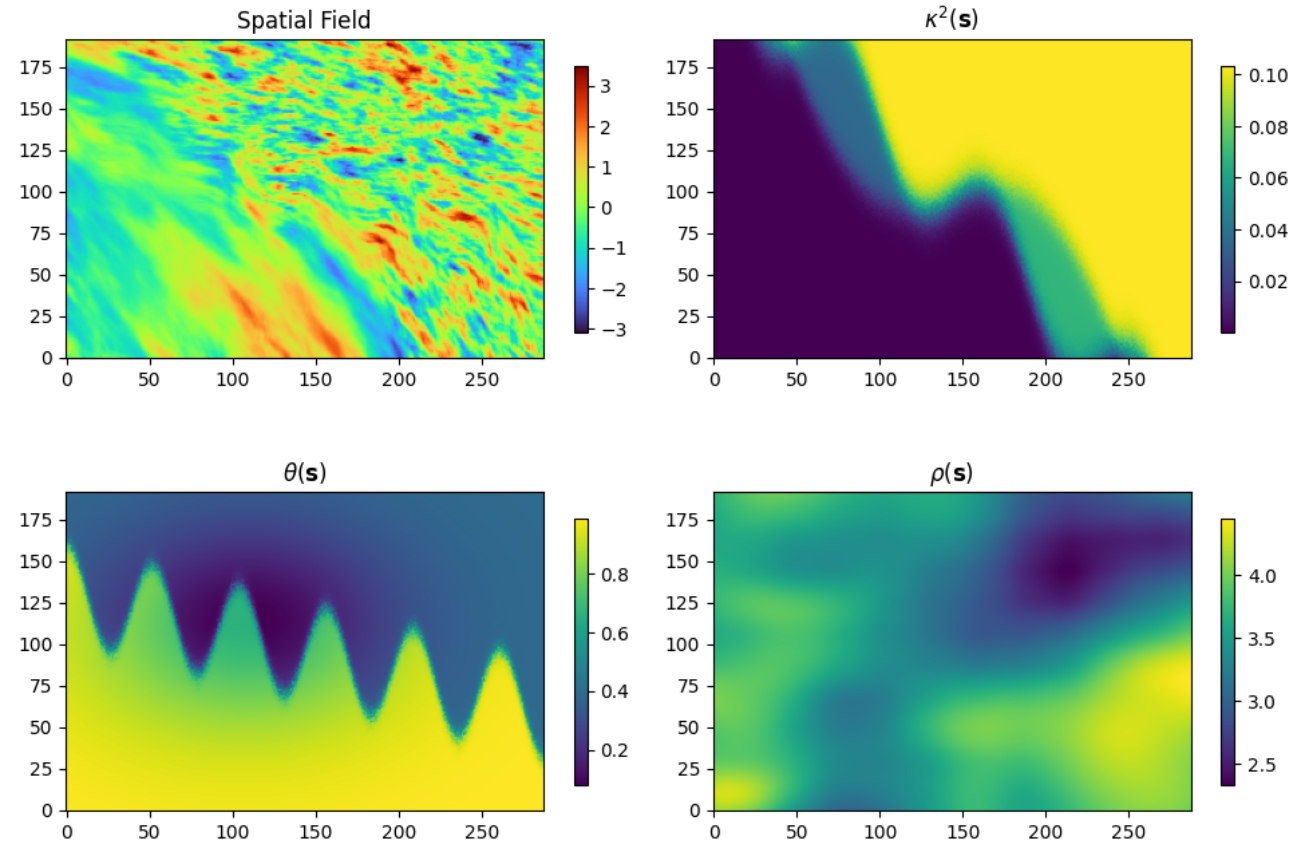

Figure 6: The first replicate in a training sample (**top-left**), and the accompanying parameter fields that were used to generate it (**remaining**). In this instance, a Double Coastline pattern was used for $\kappa^2(\mathbf{s})$, $\theta(\mathbf{s})$ is the result of stacked Coastline and Double Bump patterns, and $\rho(\mathbf{s})$ is created with a GP-Based pattern.

D FURTHER EXPERIMENTS WITH SIMULATED DATA

In this section, Table 2 contains the results of experimenting with different positional embeddings for STUN and ViT. Tables 3 and 4 contain the results of experimenting across a different number of replicates $M = \{1, 5, 15, 30\}$ for all networks. In all tables, root mean squared error (RMSE), mean absolute error (MAE), and normalized RMSE (NRMSE) are calculated, along with the structural similarity index measure (SSIM) and peak signal to noise ratio (PSNR) in decibels (db) image comparison metrics.

Table 2: Results on simulated test data across different types of positional embeddings for both STUN and ViT. Best results for each architecture are in bold.

Net	Embedding	Param	Metrics				
			RMSE ↓	MAE ↓	SSIM (↑ to 1)	PSNR ↑	NRMSE ↓
ViT	None	κ^2	0.398	0.279	0.252	10.9	0.040
		ρ	0.711	0.525	0.277	6.41	0.118
		θ	0.249	0.124	0.251	13.6	0.083
	Sinusoidal	κ^2	0.393	0.274	0.275	10.8	0.040
		ρ	0.913	0.723	0.294	3.61	0.152
		θ	0.280	0.143	0.253	12.4	0.094
	Learned	κ^2	**0.371**	**0.243**	0.291	**12.1**	**0.037**
		ρ	0.920	0.715	0.288	3.30	0.153
		θ	0.351	0.193	0.235	9.83	0.117
	Rotary	κ^2	0.374	0.250	**0.310**	11.9	0.038
		ρ	**0.625**	**0.465**	**0.337**	**7.01**	**0.104**
		θ	**0.204**	**0.103**	**0.328**	**15.1**	**0.068**
STUN	None	κ^2	0.198	0.118	**0.489**	18.4	0.020
		ρ	0.311	0.213	**0.548**	14.2	0.052
		θ	0.096	0.046	**0.604**	21.7	0.032
	Sinusoidal	κ^2	0.190	**0.113**	0.483	**18.7**	0.019
		ρ	0.302	**0.202**	0.542	**14.7**	0.050
		θ	0.126	0.050	0.592	21.5	0.042
	Learned	κ^2	0.204	0.122	0.462	18.0	0.021
		ρ	0.309	0.217	0.522	13.9	0.052
		θ	**0.090**	0.045	0.595	21.8	**0.030**
	Rotary	κ^2	**0.189**	0.117	0.469	18.4	**0.019**
		ρ	**0.302**	0.212	0.531	14.3	**0.050**
		θ	0.091	**0.044**	0.594	**21.9**	0.030

Table 3: Results on simulated test data across varying replicates for image-to-image networks.

Net	Reps	Param	Metrics				
			RMSE ↓	MAE ↓	SSIM (↑ to 1)	PSNR ↑	NRMSE ↓
UNet	1	κ^2	0.195	0.125	0.447	17.6	0.020
		ρ	0.354	0.250	0.534	13.1	0.059
		θ	0.111	0.052	0.571	20.8	0.037
	5	κ^2	0.199	0.125	0.438	17.7	0.020
		ρ	0.319	0.223	0.527	13.8	0.053
		θ	0.115	0.053	0.546	20.5	0.038
	15	κ^2	0.180	0.110	0.503	18.9	0.018
		ρ	0.294	0.207	0.592	14.6	0.049
		θ	0.102	0.044	0.628	22.3	0.034
	30	κ^2	0.201	0.124	0.447	17.9	0.020
		ρ	0.308	0.214	0.517	14.1	0.051
		θ	0.087	0.046	0.582	21.3	0.029
ViT	1	κ^2	0.471	0.303	0.296	10.5	0.048
		ρ	0.771	0.588	0.328	5.36	0.129
		θ	0.237	0.113	0.323	14.3	0.079
	5	κ^2	0.377	0.243	0.326	12.3	0.038
		ρ	0.633	0.470	0.358	7.06	0.106
		θ	0.211	0.098	0.359	15.6	0.071
	15	κ^2	0.354	0.229	0.324	12.6	0.036
		ρ	0.594	0.425	0.350	7.73	0.099
		θ	0.206	0.099	0.340	15.4	0.069
	30	κ^2	0.374	0.250	0.310	11.9	0.038
		ρ	0.625	0.465	0.337	7.01	0.104
		θ	0.204	0.103	0.328	15.1	0.068
STUN	1	κ^2	0.189	0.124	0.470	17.7	0.019
		ρ	0.351	0.243	0.535	13.3	0.058
		θ	0.097	0.047	0.593	21.3	0.033
	5	κ^2	0.180	0.109	0.502	18.8	0.018
		ρ	0.290	0.198	0.554	14.8	0.048
		θ	0.100	0.044	0.632	22.2	0.033
	15	κ^2	0.188	0.114	0.483	18.6	0.019
		ρ	0.303	0.209	0.551	14.4	0.051
		θ	0.102	0.047	0.589	21.6	0.034
	30	κ^2	0.189	0.117	0.469	18.4	0.019
		ρ	0.302	0.212	0.531	14.3	0.050
		θ	0.091	0.044	0.594	21.9	0.030

Table 4: Results on simulated test data across varying replicates for local CNNs.

Net	Reps	Param	Metrics				
			RMSE ↓	MAE ↓	SSIM (↑ to 1)	PSNR ↑	NRMSE ↓
CNN9	1	κ^2	1.01	0.637	0.069	4.64	0.101
		ρ	1.39	1.09	0.02	-0.980	0.231
		θ	0.535	0.295	0.072	5.27	0.179
	5	κ^2	0.983	0.547	0.129	6.97	0.099
		ρ	1.03	0.760	0.052	2.02	0.172
		θ	0.346	0.149	0.217	10.5	0.116
	15	κ^2	0.949	0.512	0.160	7.81	0.096
		ρ	0.962	0.689	0.070	2.90	0.160
		θ	0.330	0.135	0.300	11.3	0.110
	30	κ^2	0.963	0.508	0.172	8.09	0.097
		ρ	0.937	0.660	0.085	3.28	0.156
		θ	0.316	0.121	0.347	11.7	0.106
CNN17	1	κ^2	0.766	0.480	0.086	6.45	0.077
		ρ	1.27	0.960	0.026	-0.062	0.212
		θ	0.443	0.200	0.147	7.60	0.148
	5	κ^2	0.828	0.443	0.169	8.52	0.084
		ρ	1.03	0.714	0.081	2.75	0.171
		θ	0.317	0.126	0.289	11.1	0.106
	15	κ^2	0.730	0.377	0.222	9.94	0.074
		ρ	0.939	0.614	0.123	4.00	0.157
		θ	0.266	0.097	0.411	12.0	0.089
	30	κ^2	0.764	0.397	0.212	9.63	0.077
		ρ	0.994	0.669	0.103	3.37	0.166
		θ	0.293	0.116	0.349	11.5	0.098
CNN25	1	κ^2	0.806	0.488	0.111	6.81	0.081
		ρ	1.28	0.949	0.036	0.164	0.214
		θ	0.418	0.179	0.215	8.21	0.140
	5	κ^2	0.767	0.409	0.166	9.0	0.077
		ρ	1.09	0.736	0.088	2.77	0.181
		θ	0.340	0.124	0.316	10.2	0.114
	15	κ^2	0.732	0.372	0.249	10.2	0.074
		ρ	1.03	0.669	0.150	3.71	0.172
		θ	0.279	0.107	0.445	11.5	0.093
	30	κ^2	0.743	0.378	0.256	10.1	0.075
		ρ	1.03	0.676	0.144	3.55	0.172
		θ	0.272	0.106	0.434	11.8	0.091

E FURTHER CLIMATE APPLICATION RESULTS

We use data from three climate models: the Max Planck Institute Earth System Model at Low Resolution (MPI-ESM 1.2LR), the Community Earth System Model (CESM1), and the European Community Earth System Model (EC-Earth3). We pass the standardized ensembles of temperature sensitivity anomalies into each of our estimation networks: STUN, UNet, ViT, CNN25, CNN17, and CNN9. Using the predicted parameters, we generate 1,000 synthetic fields from each model. Generation is performed using the LatticeKrig R package (Nychka et al., 2019), which accommodates cylindrical geometry and ensures the correct treatment of spatial distances under the Mercator projection.

To evaluate how well each synthetic ensemble preserves the spatial correlation structure, we randomly select 50 anchor locations and compute the corresponding 50 rows from each simulated ensemble's empirical correlation matrix. These are then compared to the 50 rows from the original ensemble's correlation matrix using root mean squared error (RMSE). We systematically pair each I2I network with a local neural estimator of comparable rank (e.g., STUN vs. CNN25, UNet vs. CNN17), and compare their RMSEs across the same anchor locations. As the anchor locations are fixed for all pairings, we conduct paired t-tests between matched RMSEs to compare performance.

Our null hypothesis for each paired comparison is that the mean RMSE difference equals zero, $H_0 : \mu_d = 0$, where $d = \text{RMSE}_{\text{I2I}} - \text{RMSE}_{\text{CNN}}$. The one-sided alternative, $H_1 : \mu_d < 0$, states that the I2I emulator attains a lower RMSE than its CNN counterpart. All tests are conducted with a significance level $\alpha = 0.01$. Results are summarized in Table 5. For eight of the nine comparisons, the null hypothesis is rejected and the 99% confidence interval is entirely negative, demonstrating that I2I networks tend to produce significantly more accurate correlation structure estimates than their corresponding local CNNs.

Table 5: Paired t-test results comparing I2I-based emulators to corresponding CNN-based emulators on average RMSE across the same 50 anchor locations. CI denotes the 99% confidence interval of the paired differences.

Climate model	Network Pair	Paired t-test metrics				
		$\mu_{\text{I2I}} \downarrow$	$\mu_{\text{local}} \downarrow$	μ_d	99% CI	p_{value}
MPI-ESM (192 × 96)	STUN vs. CNN25	0.202	0.210	−0.008	$(-\infty, -0.004]$	2.2×10^{-6}
	UNet vs. CNN17	0.203	0.214	−0.011	$(-\infty, -0.005]$	1.0×10^{-5}
	ViT vs. CNN9	0.208	0.247	−0.039	$(-\infty, -0.028]$	7.1×10^{-12}
CESM1 (288 × 192)	STUN vs. CNN25	0.229	0.243	−0.013	$(-\infty, -0.007]$	1.1×10^{-6}
	UNet vs. CNN17	0.230	0.239	−0.008	$(-\infty, -0.004]$	3.4×10^{-5}
	ViT vs. CNN9	0.228	0.244	−0.016	$(-\infty, -0.008]$	5.9×10^{-6}
EC-Earth3 (512 × 256)	STUN vs. CNN25	0.271	0.285	−0.014	$(-\infty, -0.005]$	1.6×10^{-4}
	UNet vs. CNN17	0.275	0.288	−0.012	$(-\infty, -0.004]$	2.7×10^{-4}
	ViT vs. CNN9	0.283	0.304	−0.021	$(-\infty, 0.0003]$	1.1×10^{-2}