

FAST, NEURAL, AND SPARSE METHODS FOR BIG SPATIAL DATA

by

Antony Yuri Sikorski

© Copyright by Antony Yuri Sikorski, 2026

All Rights Reserved

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Statistics).

Golden, Colorado

Date _____

Signed: _____

Antony Yuri Sikorski

Signed: _____

Dr. Daniel McKenzie
Thesis Advisor

Signed: _____

Dr. Douglas Nychka
Thesis Advisor

Golden, Colorado

Date _____

Signed: _____

Dr. Stephen Pankavich
Professor and Department Head
Department of Applied Mathematics and Statistics

ABSTRACT

Advances in technologies such as climate and weather models, satellites, and sensor networks have resulted in terabytes of high-resolution, spatially correlated data being produced each day. Analyses of these data can help us understand the accelerating effects of climate change, the health of ecosystems, air quality in populated regions, and other phenomena critical to a sustainable future. However, the volume and complexity of these data have introduced computational challenges that render traditional methods inefficient or infeasible. This thesis develops fast statistical and machine learning methods that address three such challenges in the modeling and analysis of large spatial data.

The Gaussian Process (GP) is a foundational tool for the analysis of spatial data, but becomes computationally intractable on large datasets. A powerful class of methods represent the GP as a sum of basis functions with random coefficients. While efficient, these methods often introduce unwanted artifacts and edge effects, which can be mitigated through a computationally demanding normalization step. The first work in this thesis introduces two fast algorithms for this normalization step, enabling efficient prediction on fine grids and demonstrating significant speedups on large spatial datasets.

Large spatial data are often non-stationary, making the estimation of model parameters difficult, even for modern spatial methods. The second work introduces LatticeVision, a method that views large, gridded data and its accompanying parameters as images, enabling the use of image-to-image neural networks to estimate millions of parameters in seconds. Paired with a fast spatial autoregressive model, LatticeVision is applied to generate realistic ensembles of global climate model fields in seconds, in contrast to the millions of core-hours required by the models themselves. The third work extends LatticeVision to another central problem: the fusion of data sources on different spatial supports. By transferring information from chemical transport model outputs to ground-station measurements, we predict and quantify the uncertainty of atmospheric air pollution across Italy.

For each work in this thesis, we provide open-source software to facilitate the application of our methods by practitioners. Links to all works and software can also be found at antonyxsik.github.io.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF FIGURES	ix
LIST OF TABLES	xii
LIST OF SYMBOLS	xiii
LIST OF ABBREVIATIONS	xiv
ACKNOWLEDGMENTS	xvi
CHAPTER 1 OVERVIEW	2
1.1 Goals	2
1.2 General Overview	2
CHAPTER 2 BACKGROUND	4
2.1 Spatial Statistics	4
2.2 The Spatial Model	4
2.3 Gaussian Processes	5
2.3.1 Stationarity and Isotropy	6
2.3.2 Limitations of the Gaussian Process	6
2.4 The Stationary LatticeKrig Model	7
2.4.1 Multi-Resolution Model	8
CHAPTER 3 NORMALIZING BASIS FUNCTIONS: APPROXIMATE STATIONARY MODELS FOR LARGE SPATIAL DATA	10
3.1 Abstract	10
3.2 Introduction	10
3.2.1 Paper Outline	12
3.3 Basis Function Methods for Spatial Prediction	12
3.3.1 Prediction Artifacts in Basis Function Models	14
3.3.2 LatticeKrig Spatial Model	15

3.4	Methods	16
3.4.1	Normalization using the FFT	16
3.4.2	Approximate periodicity of the variance function	16
3.4.3	FFT-based upsampling	17
3.4.4	Normalization using Kronecker Structure	19
3.4.5	Multi-Resolution Implementation	20
3.5	Results	21
3.5.1	Timing	21
3.5.2	Error in the FFT Method	22
3.6	Application	23
3.7	Conclusion	25
3.8	Acknowledgments	27
3.9	Conflict of Interest Statement	27
3.10	Data Availability Statement	27
CHAPTER 4 NON-STATIONARITY, ANISOTROPY, AND NEURAL PARAMETER ESTIMATION		28
4.1	Non-Stationarity and Anisotropy	28
4.2	The Non-Stationary LatticeKrig Model	28
4.2.1	The Trouble with Non-Stationarity	30
4.3	Neural Parameter Estimation	30
4.4	The Transformer Architecture	32
4.4.1	The Math Behind a Simple Transformer	33
CHAPTER 5 LATTICEVISION: IMAGE TO IMAGE NETWORKS FOR MODELING NON-STATIONARY SPATIAL DATA		36
5.1	Abstract	36
5.2	Introduction	36
5.3	Background: Gaussian Processes and SAR Models	38
5.3.1	The SPDE Method	39

5.3.2	Interpreting the Parameter Fields	39
5.3.3	Discretizing the SPDE	40
5.4	Non-Stationary Data Generation	40
5.4.1	I2I Data	40
5.4.2	CNN Data	42
5.5	Parameter Estimation Networks	42
5.5.1	Implementation and Training Details	43
5.6	Simulated Data Experiments	43
5.7	Climate Application	45
5.8	Related Works	47
5.8.1	Neural Parameter Estimation	47
5.8.2	Climate Model Applications	47
5.9	Limitations and Extensions	47
5.10	Conclusion	48
5.11	Acknowledgements	48
CHAPTER 6 A NON-STATIONARY, AMORTIZED, TRANSFER LEARNING APPROACH FOR MODELING ITALIAN AIR QUALITY		50
6.1	Abstract	50
6.2	Introduction	50
6.3	Related Works	52
6.4	Data	54
6.4.1	Air quality data	54
6.4.2	Additional covariate data	55
6.5	Methods	55
6.5.1	The Latent and Observed Processes	56
6.5.2	Data Fusion and Change of Support Problem	57
6.5.3	The Non-stationary LatticeKrig Model	58
6.5.4	Neural Parameter Estimation	60

6.5.5	Training Procedure	60
6.5.6	Implementation and Evaluation	61
6.5.7	Extensions for Data Fusion and Change of Support Problem	61
6.6	Application to Numerical Model Outputs	62
6.6.1	Parameter Estimation	62
6.6.2	CTM Reconstruction Experiment	64
6.7	Application to Ground-based Measurements	66
6.7.1	Model Estimation	66
6.7.2	Cross-validation	67
6.7.3	Prediction to Fine Grids	68
6.8	Discussion	70
6.9	Acknowledgements	71
CHAPTER 7 MULTI-RESOLUTION NON-STATIONARITY WITH LATTICEKRIG		73
7.1	Non-Stationarity and Anisotropy across Multiple Resolutions	73
7.2	Demonstrating Non-Stationary, Multi-Resolution Layer Weighting	74
CHAPTER 8 CONCLUSIONS AND DISCUSSION		78
8.1	A Conclusion in the Words of a Language Model	78
8.2	Future Directions for this Research	79
8.3	A Personal Reflection	80
REFERENCES		82
APPENDIX A NORMALIZING BASIS FUNCTIONS		94
A.1	Further details on solving the screened Poisson equation	94
A.2	Software Integration	94
APPENDIX B LATTICEVISION		96
B.1	Anisotropic SAR Derivation	96
B.1.1	The Matérn Covariance	96
B.1.2	Isotropic SAR	96

B.1.3	Anisotropic Extension	96
B.2	Non-Stationary Data Generation Details	97
B.2.1	Spatial Patterns	97
B.2.2	Pattern Stacking	100
B.2.3	Resulting Synthetic Data	100
B.3	Further Implementation Details	100
B.3.1	Data and Storage	100
B.3.2	Training	100
B.4	Further Experiments with Simulated Data	101
B.5	Further Climate Application Results	104
APPENDIX C	MODELING ITALIAN AIR QUALITY	106
C.1	Change of support	106
C.2	Neural Estimator Details	108
C.3	Additional Cross Validation Results	109
APPENDIX D	COPYRIGHT PERMISSIONS	111

LIST OF FIGURES

Figure 3.1	A simple, one-dimensional scenario, where a small amount of noise is added to data sampled from a quadratic. <i>Left</i> : Both normalized and un-normalized basis function models are fitted to the data. <i>Right</i> : Finite difference approximation of the gradient for both fits reveals the artifacts more clearly.	11
Figure 3.2	<i>Top row</i> : $\phi_1(s), \dots, \phi_5(s)$ where ψ is a Wendland polynomial, dilated to yield large overlap (<i>left</i>) and small overlap (<i>right</i>). <i>Bottom row</i> : the corresponding $\Phi \in \mathbb{R}^{21 \times 5}$, demonstrating the sparsity pattern. While less overlap results in a sparser Φ (<i>Right column</i>), it also results in a less uniform variance (magenta dotted line in <i>Top row</i>).	15
Figure 3.3	Diagrammatic description of the proposed method. In this case $r = 5$ and $\tilde{n} = 13$, with basis function centers denoted by white points. The calculation is then upsampled to a grid with $n = 31$. A visual comparison of the true (exact) variance on the finer grid is provided. The log of the values is taken in the FFT domain to provide a better visual demonstration.	19
Figure 3.4	Timing results on a logarithmic scale for each method where r is varied: 25 (<i>top left</i>), 35 (<i>top right</i>), 50 (<i>bottom left</i>), 100 (<i>bottom right</i>), as is n : (500, 750, 1000, 1250, 1500, 2000). Dots represent the median time of 5 iterations of the simulation, and shading represents maximum and minimum values.	22
Figure 3.5	Mean and maximum error on a logarithmic scale for the FFT normalization method. Once again, both r : (25, 35, 50, 100), and n : (500, 750, 1000, 1250, 1500, 2000) are varied.	23
Figure 3.6	Full data set of $N = 1,329,409$ locations (<i>left</i>) along with training visualizations of both the MAR (<i>center</i>) and Blocks (<i>right</i>) sampling schemes.	24
Figure 3.7	Artifacts obtained by taking the difference of the unnormalized and exactly normalized prediction surfaces in the MAR case (<i>left</i>). The remnants of those artifacts after approximate normalization (<i>right</i>).	25
Figure 5.1	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.	37
Figure 5.2	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 axis makes with the positive x -axis.	40
Figure 5.3	Spatial patterns and their frequencies.	41
Figure 5.4	True parameters $\mathcal{P}$ (<i>left</i>) are encoded into the LatticeKrig SAR model to simulate a testing sample Y , of which one replicate $\mathbf{y}^{(0)}$ is displayed (<i>top</i>). Y is used as an input to STUN and a sliding window, local estimation strategy using CNN25, resulting in $\hat{\mathcal{P}}_{\text{STUN}}$ (<i>middle</i>), and $\hat{\mathcal{P}}_{\text{CNN25}}$ (<i>right</i>).	44

Figure 5.5	<i>Top:</i> Standardized temperature fields drawn from the CESM1 ensemble (<i>left</i>), the STUN-based emulator (<i>middle</i>), and the CNN25-based emulator (<i>right</i>). <i>Bottom:</i> 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.	46
Figure 6.1	From the left: NO ₂ concentrations from ground monitoring stations, NO ₂ concentrations from chemical transport models, temperature from reanalysis (ERA5 datasets), NO _x emission fluxes from CAMS-REG-ANT. Data for Italy, on 13th April 2023.	56
Figure 6.2	2023 annual averages of CTM-based parameter estimates. Top row from left to right: the average CTM NO ₂ field, elliptical dependence contours overlaid on an elevation (m) map of the full domain, and a zoomed-in view of enlarged ellipses over land only. Bottom row from left to right: annual averages of STUN estimates $\hat{\kappa}^2$, $\hat{\theta}$, and $\hat{\rho}$	63
Figure 6.3	CTM NO ₂ reconstruction experiment for April 13, 2023. From left to right: the actual CTM NO ₂ field, masked data used for model fitting (pixels at locations with stations), the full prediction from the non-stationary model, and the difference between the true and predicted field.	64
Figure 6.4	Left: time series of daily RMSE percent differences ((nonstationary / stationary - 1)*100). A solid black line illustrates equality in both cases, while the pink line is a GAM-based fit with a cyclic cubic spline basis (20 basis functions), chosen to capture seasonal structure in the temporal trend. Dots below the black line indicate days on which the nonstationary model outperforms the stationary one. Center: difference in pixel-wise RMSE (nonstationary - stationary); negative values indicate locations where the nonstationary model outperforms the stationary model. Right: the pixel-wise nonstationary RMSE divided by the local annual mean NO ₂ concentration to show relative errors.	65
Figure 6.5	Differences between the range parameters estimated by the stationary model and the adjusted non-stationary one, restricted to basis-function centers on land. Left: daily log-scale differences between the stationary range parameter and the spatial average of the nonstationary one. The dotted line indicates the mean difference of 0.8. Right: annual average of the daily log-scale difference at each basis function center.	67
Figure 6.6	Estimated spatial component on November 1, 2023. Left: stationary model. Right: adjusted nonstationary model	68
Figure 6.7	Fine-grid predictions for November 1, 2023. Left: posterior mean NO ₂ concentration ($\mu\text{g}/\text{m}^3$) on the $0.05^\circ \times 0.05^\circ$ prediction grid. Right: posterior standard error ($\mu\text{g}/\text{m}^3$) derived from a 1,000-member conditional simulation ensemble.	69
Figure 7.1	<i>Left:</i> Realization of the spatial process from the first layer. <i>Right:</i> Covariance with the point (0, 0).	75
Figure 7.2	<i>Left:</i> Realization from the two-layer model with constant, equal marginal variances $\alpha_1 = \alpha_2 = 0.5$. <i>Right:</i> Covariance with the point (0, 0).	76
Figure 7.3	<i>Left:</i> Marginal variance field α_1 for the first layer. <i>Right:</i> Marginal variance field α_2 for the second layer. By construction, $\alpha_1(\mathbf{u}) + \alpha_2(\mathbf{u}) = 1$ at every center.	76

Figure 7.4	<i>Left:</i> Realization of the spatial process from the combined, multi-resolution model with non-stationary marginal variance fields. <i>Right:</i> Covariance with the point $(0, 0)$, showing a secondary peak in the top-right corner where the first layer dominates.	77
Figure B.1	The first replicate in a training sample (<i>top-left</i>), and the accompanying parameter fields that were used to generate it (<i>remaining</i>). 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.	101
Figure C.1	Station-level cross-validation metrics for the nonstationary (adjusted) model, aggregated over all 349 days. From left to right: RMSE weighted by the local annual mean NO_2 concentration, 95% prediction interval coverage probability, and mean prediction interval width ($\mu\text{g}/\text{m}^3$).	110

LIST OF TABLES

Table 3.1	Mean absolute error (MAE), root-mean-squared prediction error (RMSPE), and total run time are recorded for each choice of normalization method for both the MAR and Blocks sampling schemes. Arrows ($\uparrow, \downarrow$) indicate the desirable direction for a metric.	24
Table 5.1	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.	45
Table 6.1	STUN parameter-estimation performance (all true vs. predicted pixels) on held-out test data. Here, NRMSE is computed relative to the range (max - min) of parameter values. Relative error is summarized by the median and is reported as a percentage. Arrows indicate the desirable direction.	61
Table 6.2	RMSE, 95% Prediction Interval Coverage Probability (PICP), Mean 95% Prediction Interval Width (MPIW). Arrows indicate the desirable direction.	68
Table 7.1	Parameter values for each layer of the multi-resolution model.	75
Table B.1	Results on simulated test data across different types of positional embeddings for both STUN and ViT. Best results for each architecture are in bold.	102
Table B.2	Results on simulated test data across varying replicates for image-to-image networks. . . .	103
Table B.3	Results on simulated test data across varying replicates for local CNNs.	104
Table B.4	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.	105

LIST OF SYMBOLS

Basis function	ϕ
Covariance matrix	Σ
Error/white noise process	ε
Gaussian process	$\mathcal{GP}$
Identity matrix	I
LatticeKrig angle parameter	θ
LatticeKrig correlation range parameter	κ^2
LatticeKrig elliptical ratio parameter	ρ
Linear model coefficient	β
Mean	μ
Normal distribution	$\mathcal{N}$
Observed spatial data	z
Order of magnitude	$\mathcal{O}$
Precision matrix	Q
Spatial autoregressive matrix	B
Spatial location	$\mathbf{s}$
Spatial process	y
Stochastic process	g
The set of Real Numbers	$\mathbb{R}$
Uniform distribution	$\mathcal{U}$

LIST OF ABBREVIATIONS

Artificial Intelligence	AI
Autoregressive	AR
Central Processing Unit	CPU
Change of Support Problem	COSP
Chemical Transport Model	CTM
Colorado School of Mines	CSM
Confidence Interval	CI
Convolutional Neural Network	CNN
Discrete Fourier Transform	DFT
Earth System Model	ESM
European Environment Agency	EEA
Fast Fourier Transform	FFT
Fixed Rank Kriging	FRK
Gaussian Markov Random Field	GMRF
Gaussian Process	GP
General Circulation Model	GCM
Generalized Additive Model	GAM
Generative Pre-trained Transformer	GPT
Graduate Research Fellowship Program	GRFP
Graph Neural Network	GNN
Graphics Processing Unit	GPU
Image-to-Image	I2I
LatticeKrig	LK
Learning to Optimize	L2O
Machine Learning	ML

Maximum Likelihood Estimation	MLE
Mean Absolute Error	MAE
Multi-Headed Self-Attention	MHSA
Multi-Layer Perceptron	MLP
National Center for Atmospheric Research	NCAR
National Science Foundation	NSF
Neural Network	NN
Nitrogen Dioxide	NO ₂
Normalized Root Mean Square Error	NRMSE
Peak Signal-to-Noise Ratio	PSNR
Recurrent Neural Network	RNN
Root Mean Square Error	RMSE
Rotary Positional Embeddings	RoPE
Spatial Autoregressive	SAR
Spatial TransUNet	STUN
Stochastic Partial Differential Equation	SPDE
Structural Similarity Index Measure	SSIM
Vision Transformer	ViT

ACKNOWLEDGMENTS

In my opinion, luck is the most influential variable in the model of anyone's achievement. The completion of my PhD is no exception to this rule. Throughout this degree, I have been incredibly lucky, with much of my luck showing itself in the form of the wonderful people who surround me and support me to this day. This document may be a synthesis of my technical work, but it is shaped by countless direct and indirect contributions from those wonderful people, to whom I owe enormous thanks. As I try my best to express this gratitude, I know the words I write here will be insufficient, I will likely fail to name everyone, and I will definitely ramble. Ah well, I hope it's a good start.

My parents have been "brainwashing" me since I was a young child, telling me all sorts of "heresy" like "you should do a PhD", "you'll meet amazing people", "you'll have a more interesting career", "it will be worth it", and so on. As usual, they were right on all counts. It was totally worth it. Thank you, Mom and Dad, for all of your love and support, and for the ridiculous number of things you sacrificed to provide me with all of these incredible opportunities. Thank you, Katherine, for being my oldest friend and my favorite (only) sister. Thank you to our dog, Cookie, the greatest puppy that ever lived. Thank you to my partner, Annalea, for your love and your otherworldly patience. Signing up to be separated by 1000 miles for half a decade isn't exactly an easy thing, but it was one of the best decisions I've made yet. On a totally unrelated note, Annalea's family only lives 50 miles away from me, and every time I stop by, Junnie graciously gifts me multiple bags of the world's best dumplings, wontons, and bao buns that she spent hours making. As a grad student, these have constituted half of my diet throughout this time. Thank you Ethan, Morgan, and especially Junnie.

I won the lottery when it comes to friends. Spanning Fort Collins, San Diego, and grad school, I have stumbled upon some pretty fantastic people. Thank you, Carse, Zachy, and Jakey. You guys have known me since I was less than 4 feet tall, and to this day I know you'd lie down in traffic for me. That's rare, and I don't take it for granted. Thank you Suj, Neel, Ferm, Alain, Andrey, Willy, and too many to name from Sig Ep and UCSD. Thank you to Ryan for letting me rant to you across the desk for 4 years. It's such a joy to share an office with a friend. Thank you Souvick, for being a great friend and seemingly knowing everything there is to know about probability. I don't know if I would have made it through the first year without you. Thank you Brandon, Boris, Misha, Ibrohim, and Charlie, you turned every house we lived in into a home that I was stoked to be in. Thank you to the rest of my friends and colleagues at Mines for ski days, GCB Fridays, and the most collaborative and supportive PhD program I've seen or heard of yet.

Thank you to all of the folks and professors in the AMS department for creating the environment that allows us PhD students to have this kind of collaborative, supportive experience. A special thank you to Lisa Maddux, for patiently guiding me through every step of the process. I'm pretty sure the department would collapse without you. My summers in the PhD program would not have been possible without the people who took a chance on hiring me and guiding me, and the friends I made each time. I must admit, during my first year, I only applied to internships in LA, so I could close the 1000 mile gap for a couple months. I was so lucky to end up working with Lisa, Harvey, Eshan, and the rest of the wonderful folks at JPL. Thank you, so much. Thank you to Sara for your meaningful guidance at LEAP, and to Obin for an awesome summer in NYC. Finally, thank you to Zhen and Breno for your valuable lessons, and to Huy and Brad for making SF so fun. Thank you to the members of my thesis committee, Samy, Soutir, and Adrienne, for your wise words and constant encouragement.

I would like to acknowledge my support from the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2137099. Public funding for science is not a luxury but a necessity, and work like this simply does not happen without it. I am grateful to have benefited from that investment, and hopeful that it will remain available to those who come after me.

I remember a few wise folks told me that the research will figure itself out, and the actual most important decision I will make in my PhD is choosing who advises me. They were completely right. Doug and Daniel, you have not only served as research advisors, but also mentors. The vast majority of the time, I was excited to go to our advisor meetings. Excited! That's not the word I've heard most people use to describe their emotions about their weekly advisor meeting. You supported me in graduating early, let me do internships every summer, and put an enormous amount of energy into guiding me in the right direction. I traveled to conferences in amazing places all over the world because you were kind enough to send me. You were willing to sit down and dig through code with me to find bugs, read paper drafts over and over, listen to my anxious rants when life got hard, and so much more. Thank you, thank you, thank you.

CHAPTER 1

OVERVIEW

This chapter contains the goals of this work and a general outline of the remainder of the thesis.

1.1 Goals

The primary aims of this work are to:

- Motivate the development of fast and accurate methods for modeling large spatial data.
- Establish a short background on key concepts in spatial statistics, data science, and machine learning.
- Present novel advances in research at the intersection of machine learning and spatial statistics.
- Discuss the limitations of the research presented in this thesis, and provide ideas for potential extensions or improvements to continue the advancement of the field.

1.2 General Overview

In Chapter 2, we provide essential background material to support the subsequent chapters. This includes a brief introduction to spatial data and spatial statistics, Gaussian Processes (GPs), and the stationary LatticeKrig spatial model [1]. This chapter provides the necessary background for Chapter 3.

In Chapter 3, we introduce a computational improvement to basis function methods for spatial statistics. Although these methods offer efficient GP representations for working with large datasets, they often suffer from artifacts and edge effects caused by a non-constant marginal variance. To address this, a costly normalization of the basis functions is required. This chapter presents two fast and accurate algorithms that dramatically reduce the cost of normalization [2]. We showcase the practical value of these algorithms via a prediction task on a large dataset, and implement them in the LatticeKrig model, a well known, basis function method with an accompanying, open-source R package [3].

In Chapter 4, we introduce additional concepts that motivate the remainder of the thesis. We begin with non-stationarity and anisotropy, discuss the challenges they pose for parameter estimation, and define the non-stationary LatticeKrig model. We then introduce neural parameter estimation, in which neural networks replace traditional likelihood-based estimators. Finally, we review key neural network architectures relevant to spatial problems.

In Chapter 5, we turn our attention to the application of deep learning to modeling non-stationary spatial data. We present a novel method for estimating spatially varying parameters using deep learning

architectures inspired by those in computer vision [4]. Specifically, we adapt image-to-image (I2I) neural networks, both transformer-based and fully convolutional, to predict parameters of spatial autoregressive (SAR) models directly from input fields. Our approach bypasses both the computational bottlenecks of traditional, maximum likelihood based approaches, and the locality restrictions of prior deep learning methods. Once the parameters have been estimated, they can be encoded into the LatticeKrig SAR model to generate large ensembles of additional, synthetic fields. This offers a practical solution for uncertainty quantification in high-cost domains such as climate modeling. We apply our end-to-end methodology to outputs from an Earth System Model, and generate thousands of additional fields in a matter of seconds.

In Chapter 6, we extend the methodology of Chapter 5 to a data fusion setting, contributing to the longstanding problem of combining data sources defined on different spatial supports. We develop a transfer-learning framework that uses LatticeVision to learn spatially varying parameters from gridded chemical transport model outputs and then uses these parameters as a prior for modeling sparse ground-station measurements [5]. A likelihood-based refinement step then adjusts the parameters to recover fine-scale variability smoothed out by the gridded data. We apply this framework to produce daily, fine-resolution maps of nitrogen dioxide concentrations across Italy for 2023, with uncertainty quantification.

In Chapter 7, we present software contributions that extend the methodology of Chapter 5 and Chapter 6. Specifically, we show that the LatticeKrig model is capable of representing fully independent, non-stationary, and anisotropic parameter fields across multiple resolutions, along with spatially varying weights that control the contribution of each level. This flexibility enables the representation of even richer spatial structures, including long-range teleconnections, opening the door for future applications.

In Chapter 8, we conclude this thesis with discussions, broader applications, and future directions.

CHAPTER 2

BACKGROUND

2.1 Spatial Statistics

Spatial statistics is a broad subset of statistical techniques used in dealing with data that has an element of spatial correlation, meaning there exist relationships in the data that can be attributed to the “location” of the observations. While this field naturally lends itself to applications in geography and climate science, the ubiquity of spatially-referenced data has led to diverse use cases, ranging from tracking the spread of viral infections like Covid-19 [6], determining dark matter distribution in galaxies [7], and identifying crime hotspots in major cities [8]. The field has roots in geostatistics, multivariate statistics, cartography, and even mining, where it was primarily used for creating accurate maps of mineral resources and predicting their occurrences. One of the cornerstone principles of spatial statistics is Tobler’s “First Law of Geography” [9], which essentially states that observations that are closer together are typically more related than those that are further apart. While a seemingly simple law, this principle captures the essence of spatial dependence, and moves beyond the classical assumption of independent, identically distributed observations.

2.2 The Spatial Model

Let $y(\mathbf{s})$ be a curve, a spatial surface, or a function defined at locations $\mathbf{s}$. We define a model for y such that

$$y(\mathbf{s}) = \mu(\mathbf{s}) + g(\mathbf{s}) \tag{2.1}$$

where $y(\mathbf{s})$ is the value of the spatial process at $\mathbf{s}$, $\mu(\cdot)$ is a mean function responsible for fixed effects, and $g(\cdot)$ is a stochastic process. In practice we do not have access to the latent surface $y(\mathbf{s})$ directly; instead we collect data in the form of possibly noisy observations at a finite set of locations. Let $z(\mathbf{s})$ be an observation at spatial location $\mathbf{s}$. The observational model can be written as

$$z(\mathbf{s}) = y(\mathbf{s}) + \varepsilon(\mathbf{s}) = \mu(\mathbf{s}) + g(\mathbf{s}) + \varepsilon(\mathbf{s}) \tag{2.2}$$

where $\varepsilon(\cdot)$ encapsulates independent, normally distributed noise with mean zero and a variance of τ^2 . Often, we assume the mean function $\mu(\cdot)$ depends linearly on covariates $x_1(\cdot), \dots, x_d(\cdot)$

$$\mu(\mathbf{s}) = \mathbf{x}(\mathbf{s})^\top \boldsymbol{\beta}, \quad \text{where } \mathbf{x}(\mathbf{s}) = [x_1(\mathbf{s}) \quad \cdots \quad x_d(\mathbf{s})], \tag{2.3}$$

and $\boldsymbol{\beta}$ represents the linear coefficients.

For illustration, consider the example of mapping annual average $PM_{2.5}$ concentration over a city. $PM_{2.5}$, or fine-particle air pollution, is defined as particulate matter with a diameter of 2.5 microns or less, which can be harmful to the human respiratory system in large quantities [10]. We assume an underlying process y , which denotes the true but unknown pollution level at location $\mathbf{s}$, and suppose we have n monitoring stations that report noisy readings

$$z(\mathbf{s}_i) = \mu(\mathbf{s}_i) + g(\mathbf{s}_i) + \varepsilon(\mathbf{s}_i), \quad i = 1, \dots, n \quad (2.4)$$

with an error term $\varepsilon(\mathbf{s}_i) \sim \mathcal{N}(0, \tau^2)$ which encapsulates sensor noise and microscale variation. In $\mu(\cdot)$, we might include covariates $\mathbf{x}$ such as humidity, population density, and elevation, which can help explain broad spatial trends and aid the prediction of $PM_{2.5}$. The spatial structure that can not be explained by the fixed effects is modeled by the stochastic process $g(\cdot)$, and the rest is captured by $\varepsilon(\cdot)$

There are many choices for the form of $g(\cdot)$, with the most common one being a Gaussian process, which is favored for its flexibility, well-understood probabilistic structure, and ability to yield closed-form spatial predictions with built-in uncertainty quantification. In the following section we define the Gaussian process and discuss its advantages and limitations in more detail.

2.3 Gaussian Processes

A spatial Gaussian process (GP) is a collection of random variables indexed by spatial locations $\mathbf{s}$ such that any finite subset of the values $(y(\mathbf{s}_1), y(\mathbf{s}_2), \dots, y(\mathbf{s}_n))$ has a multivariate Gaussian distribution. The process is fully characterized by two components. The first is a mean function $\mu(\mathbf{s})$, which creates the mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ for a finite set of observations. The second is the covariance function C , which describes how correlated the values of the process are between any two locations $\mathbf{s}$ and $\mathbf{s}'$ and builds the covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$, where $\Sigma_{a,b} = C(\mathbf{s}_a, \mathbf{s}_b)$. Throughout this thesis, we use the notation $\mathbb{R}^n$ for the space of real-valued vectors of length n , $\mathbb{R}^{m \times n}$ for the space of real $m \times n$ matrices, and so on for higher-order tensors. The Gaussian process and resulting multivariate normal distribution can be written as:

$$y(\cdot) \sim \mathcal{GP}(\mu, C), \quad \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma) \quad (2.5)$$

When modeling spatial data, we may specify $y(\cdot) \sim \mathcal{GP}(\mu, C)$ directly, or—equivalently—write $y(\mathbf{s}) = \mu(\mathbf{s}) + g(\mathbf{s})$ with $g(\cdot) \sim \mathcal{GP}(\mathbf{0}, C)$. The covariance function $C(\mathbf{s}, \mathbf{s}')$, also commonly referred to as a *kernel* in machine learning contexts, must be symmetric ($C(\mathbf{s}, \mathbf{s}') = C(\mathbf{s}', \mathbf{s})$) and positive semi-definite so that the resulting covariance matrix is valid for every finite set of locations.

There are many covariance function choices, such as the Exponential, Cauchy, Spherical, and Matérn, to name only a few. The Matérn class is most popular due to its balance of interpretability and expressiveness [11]. Under the Matérn family, the covariance function takes the form:

$$C(\mathbf{s}, \mathbf{s}') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\kappa_{\mathcal{M}} \|\mathbf{s} - \mathbf{s}'\|)^\nu \mathcal{K}_\nu(\kappa_{\mathcal{M}} \|\mathbf{s} - \mathbf{s}'\|), \quad (2.6)$$

where $\nu > 0$ is smoothness, $\kappa_{\mathcal{M}} > 0$ is the spatial correlation range, $\mathcal{K}_\nu$ is a modified Bessel function of the second kind, and σ^2 is the marginal variance. It is important to note that the value of the covariance function decays as the distance between two points ($\|\mathbf{s} - \mathbf{s}'\|$) increases.

2.3.1 Stationarity and Isotropy

The standard Matérn covariance function, along with many other analytical choices for the covariance kernel in spatial modeling, is by construction *isotropic* and *stationary*.

- *Stationarity* implies invariance under translation, meaning the covariance between two locations does not depend on their location within the domain. More precisely, when we refer to “stationarity” in this text, we are referring to weak (second-order) stationarity, where $C(\mathbf{s}, \mathbf{s}') = C(\mathbf{s} + \mathbf{h}, \mathbf{s}' + \mathbf{h})$ for every shift $\mathbf{h}$. Simply put, this means that the kernel and key statistical parameters (such as the correlation range) do not vary across the domain.
- *Isotropy* implies invariance under rotation, meaning the covariance between two locations does not depend on the direction of their separation. Formally, $C(\mathbf{s}, \mathbf{s}') = C(\|\mathbf{s} - \mathbf{s}'\|)$. In two dimensions, one can imagine isotropy as the property that all points sharing a fixed correlation with a given point form a perfect circle around it.

While these assumptions make analytic kernels such as the Matérn easy to implement and interpret, real-world processes often exhibit anisotropy (direction-dependent correlation) and non-stationarity (key spatial parameters vary across the domain). Relaxing these simplifying assumptions is therefore an active area of research and a key focus of the later chapters in this thesis, specifically Chapter 4 to Chapter 7.

2.3.2 Limitations of the Gaussian Process

Despite the many attractive properties of the GP as a tool for modeling spatial data, it becomes computationally intractable for even moderately sized datasets (*i.e.* $n \sim 10^4$). The primary barrier to working with the GP in such settings is the fact that Σ is dense and of size $n \times n$, with n being the number of locations. Thus, computing the likelihood and conditional expectation—both of which involve factorizing Σ —require $\mathcal{O}(n^3)$ operations and $\mathcal{O}(n^2)$ memory. This computational bottleneck has motivated an active field of developing fast GP representations such as covariance tapering [12], basis function methods [13], spatial autoregressive (SAR) models [14], and methods such as the LatticeKrig model, which relies on sparse linear algebra [1], and is further described in the following section. This thesis extends that

line of work, and integrates sparse statistical models with deep learning methods, resulting in models that can handle larger and more heterogeneous spatial datasets.

2.4 The Stationary LatticeKrig Model

The LatticeKrig (LK) model [1] represents a GP as a sum of basis functions with coefficients governed by a spatial autoregression (SAR). This construction results in the spatial dependence being controlled by a sparse precision matrix, providing scalability for large spatial data. The original LK model is a multi-resolution model, comprising several stacked layers of basis functions and coefficients at progressively finer scales. For ease of exposition, we first present the single-layer version of the model, and return to the multi-resolution extension at the end of this section. Additionally, the LK framework can be extended to accommodate nonstationarity and anisotropy. We restrict our attention to the stationary, isotropic case here, and develop the nonstationary, anisotropic extension in Chapter 4. Under the LK specification, the spatial component $g(\mathbf{s})$ is written as $\boldsymbol{\phi}(\mathbf{s})^\top \mathbf{c}$, and the observational model in Equation (2.2) becomes

$$z(\mathbf{s}) = \mathbf{x}(\mathbf{s})^\top \boldsymbol{\beta} + \boldsymbol{\phi}(\mathbf{s})^\top \mathbf{c} + \varepsilon(\mathbf{s}), \quad (2.7)$$

where $\boldsymbol{\phi}(\mathbf{s}) = (\phi_1(\mathbf{s}), \dots, \phi_m(\mathbf{s}))^\top$ is the vector of m basis functions evaluated at $\mathbf{s}$, and $\mathbf{c} = (c_1, \dots, c_m)^\top$ is the corresponding vector of random coefficients.

Each basis function ϕ_j is a radially symmetric, compactly supported Wendland function centered at one of the m lattice points $\{\mathbf{u}_j\}_{j=1}^m$ arranged on a regular grid over the spatial domain:

$$\phi_j(\mathbf{s}) = \psi\left(\frac{\|\mathbf{s} - \mathbf{u}_j\|}{\gamma}\right), \quad \psi(d) = \begin{cases} \frac{(1-d)^6(35d^2+18d+3)}{3} & 0 \leq d \leq 1, \\ 0 & \text{otherwise,} \end{cases} \quad (2.8)$$

with γ a scaling factor that sets the radius of support. Because ψ vanishes outside the unit interval, each ϕ_j contributes only within a small neighborhood of its center $\mathbf{u}_j$, so the regression matrix $\Phi \in \mathbb{R}^{n \times m}$ with entries $\Phi_{i,j} = \phi_j(\mathbf{s}_i)$ is sparse.

The coefficients are distributed as $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, Q^{-1})$, where $Q \in \mathbb{R}^{m \times m}$ is the sparse precision matrix, which is specified through a spatial autoregressive (SAR) representation. Given a sparse linear operator B and a random vector $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, I)$, we construct the distribution of $\mathbf{c}$ such that $B\mathbf{c} = \mathbf{e}$. Thus, B is a SAR operator that decorrelates $\mathbf{c}$ to white noise. In the stationary case, the rows of B are populated according to the five-point stencil

$$\begin{array}{ccc} & -1 & \\ -1 & 4 + \kappa^2 & -1 \\ & -1 & \end{array}, \quad (2.9)$$

where the central value gives the diagonal entry of B , the four off-center values give the entries corresponding to the four nearest neighbors on the lattice, and all other entries in the row are zero. The

parameter $\kappa^2 > 0$ controls the spatial correlation range for all observations. By linear statistics, $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, (B^\top B)^{-1})$, and so the precision matrix Q equals $B^\top B$ and is also sparse. This particular form of the stencil arises from the connection between Matérn GPs and stochastic partial differential equations (SPDEs). Whittle [15] showed that a Matérn GP $g(\mathbf{s})$ on $\mathbb{R}^2$ with smoothness $\nu = 1$ satisfies

$$(\kappa^2 - \Delta)g(\mathbf{s}) = \mathcal{W}(\mathbf{s}), \quad (2.10)$$

where Δ is the Laplacian and $\mathcal{W}$ is a white noise process. Lindgren et al. [14] demonstrated that discretizing (2.10) on a regular lattice via finite differences yields a Gaussian Markov random field (GMRF) that approximates the Matérn GP, with a sparse precision matrix whose structure is determined by the discretization stencil. The derivation can be found in the Appendix B.1.2.

The sparsity in both Q and Φ is what makes the LK model scalable. The simulation, likelihood evaluation, and prediction steps perform operations with sparse matrices, sidestepping dense matrix factorizations that limit the original GP. In practice, this allows LK to be applied to spatial datasets with $n \sim 10^5$ or larger, well beyond the reach of standard GP methods.

2.4.1 Multi-Resolution Model

Although a single layer model is sufficient for many applications, it is the introduction of multiple resolutions that allows the LK model to approximate standard families of covariance functions. In two dimensions, the model is defined as the sum of L independent layers, with each successive layer containing roughly four times more basis functions than the previous layer. The full spatial process is thus a sum

$$g(\mathbf{s}) = \sum_{\ell=1}^L \sum_{j=1}^{m_\ell} c_{\ell j} \phi_{\ell j}(\mathbf{s}) \quad (2.11)$$

where each layer $\ell = 1, \dots, L$ has its own correlation range parameter κ_ℓ^2 , its own coefficients $\mathbf{c}_\ell$ governed by an independent SAR, and a marginal variance α_ℓ that weights the contribution of the layer to the final process. The marginal variances are constrained to sum to one, $\sum_{\ell=1}^L \alpha_\ell = 1$, so that they partition the total variance across the layers. The resulting precision matrix has the sparse, block diagonal form

$$Q = \begin{bmatrix} (1/\alpha_1) B_1^\top B_1 & 0 & \cdots & 0 \\ 0 & (1/\alpha_2) B_2^\top B_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (1/\alpha_L) B_L^\top B_L \end{bmatrix}, \quad (2.12)$$

where B_ℓ corresponds to the SAR matrix of layer ℓ , constructed from the five-point stencil with correlation range parameter κ_ℓ^2 .

By appropriately choosing the marginal variances $\{\alpha_\ell\}$ and correlation ranges $\{\kappa_\ell^2\}$, this multi-resolution model has been shown to approximate stationary covariance functions from the Matérn

family successfully, while preserving the sparsity that makes the model fast [1]. The single-layer model in this notation corresponds to $L = 1$ and serves as the basis for the nonstationary extension developed in Chapter 4. Both nonstationarity and multiple resolutions are then combined to demonstrate the full flexibility of the LK model in Chapter 7.

CHAPTER 3

NORMALIZING BASIS FUNCTIONS: APPROXIMATE STATIONARY MODELS FOR LARGE SPATIAL DATA

Modified from a paper published in *Stat*¹.

Antony Sikorski^{2,3}, Daniel McKenzie⁴, Douglas Nychka⁵

3.1 Abstract

In geostatistics, traditional spatial models often rely on the Gaussian Process (GP) to fit stationary covariances to data. It is well known that this approach becomes computationally infeasible when dealing with large data volumes, necessitating the use of approximate methods. A powerful class of methods approximate the GP as a sum of basis functions with random coefficients. Although this technique offers computational efficiency, it does not inherently guarantee a stationary covariance. To mitigate this issue, the basis functions can be “normalized” to maintain a constant marginal variance, avoiding unwanted artifacts and edge effects. This allows for the fitting of nearly stationary models to large, potentially non-stationary datasets, providing a rigorous base to extend to more complex problems. Unfortunately, the process of normalizing these basis functions is computationally demanding. To address this, we introduce two fast and accurate algorithms to the normalization step, allowing for efficient prediction on fine grids. The practical value of these algorithms is showcased in the context of a spatial analysis on a large dataset, where significant computational speedups are achieved. While implementation and testing are done specifically within the LatticeKrig framework, these algorithms can be adapted to other basis function methods operating on regular grids.

3.2 Introduction

Generating prediction surfaces in geostatistics frequently involves fitting models with a stationary covariance to spatially referenced data on a regular grid and then computing model predictions at many unobserved grid points. Historically, the Gaussian Process (GP) has served as a primary model for such analysis, as GPs allow for straightforward prediction and uncertainty quantification [16, 17]. Well known computational bottlenecks arise from the need to evaluate the conditional expectation to obtain a spatial

¹Reprinted with permission of *Stat* (2025), 13 (4), e70015

²Graduate student at the Colorado School of Mines

³Author for correspondence

⁴Assistant Professor at Colorado School of Mines

⁵Professor at Colorado School of Mines

prediction, and the likelihood to estimate covariance parameters. These operations generically require $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ memory, where N represents the number of spatial locations [18, 19].

A flexible and computationally efficient approach for avoiding these bottlenecks is provided by a class of “basis function models”, which approximate a spatial process as a weighted sum of basis functions [20]. Some notable examples include Fixed Rank Kriging (FRK) [13, 21], INLA [22], mgcv [23], Multi-Resolutional Approximation (MRA) [24], Predictive Processes (PP) [25, 26], Spatial Partitioning [27], and LatticeKrig [1, 3]. Despite variations in basis function choice and modeling approaches, a common disadvantage is the absence of guaranteed stationarity in the covariance structure, which mandates that the spatial dependence between two points rely solely on their distance. Although data are frequently non-stationary, well fitting stationary models allow for easier and more efficient parameter estimation due to the simplicity of the covariance structure, along with the use of standard estimation methods. Moreover, non-stationary models are often built by modifying a stationary covariance kernel.

A drawback of basis function methods is the presence of undesirable artifacts in the predicted surface, as demonstrated in Figure 3.1. Here a simple model consisting of eight overlapping basis functions is fit to noisy observations from a quadratic function. A naive fit introduces oscillations in the predicted curve that are solely due to the placement and shape of the basis functions. The reduction or removal of these artifacts is critical to achieving stationarity, and fitting models that are representative of the physical data. One way of doing so is by “normalizing” the basis functions to ensure a constant marginal variance.

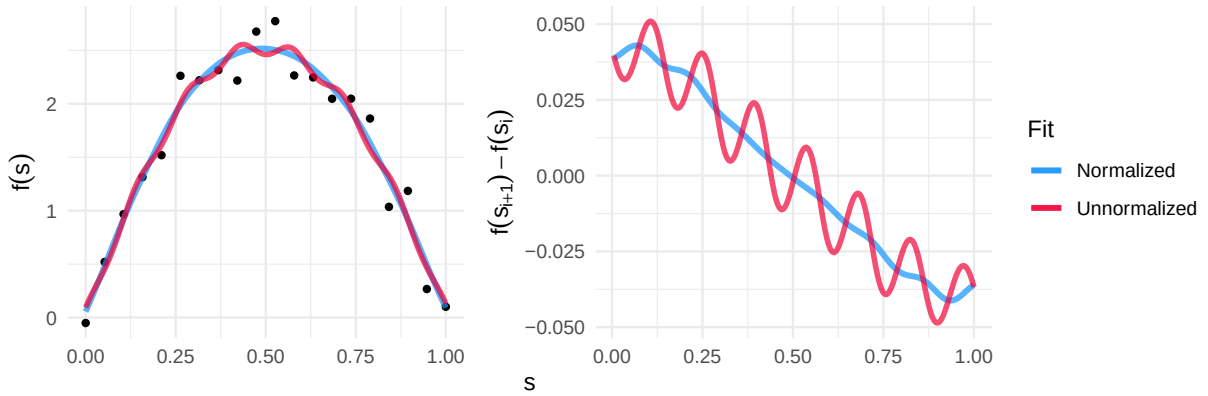


Figure 3.1 A simple, one-dimensional scenario, where a small amount of noise is added to data sampled from a quadratic. *Left:* Both normalized and un-normalized basis function models are fitted to the data. *Right:* Finite difference approximation of the gradient for both fits reveals the artifacts more clearly.

Although it is an improvement to the spatial model, this normalization step constitutes a subsequent computational bottleneck. In order to analyze datasets on the order of millions or more spatial locations ($N \sim 10^6$) efficiently, it has become necessary to explore, understand, and accelerate this operation.

In this paper, we significantly reduce the computational cost of ensuring the stationarity of basis function models on a regular grid. We provide two algorithms, and software for reproducibility. The first algorithm uses the fast Fourier transform (FFT) to approximate the variance of the basis functions on large grids rapidly. This replaces an $\mathcal{O}(N^3)$ exact computation with an $\mathcal{O}(N \log(N))$ approximate one, at almost no loss of accuracy. The second algorithm calculates the variance exactly by decomposing key matrices into a series of Kronecker products of smaller matrices. All supplementary code for the figures and experiments in this paper, along with a development version of the `LatticeKrig` R package can be found on github.com/antonyxsik/Normalization-Paper.

3.2.1 Paper Outline

The remainder of this paper is as follows. Section 3.3 provides a brief introduction to basis function methods, and as a concrete example, the `LatticeKrig` model. In Section 3.4, two fast normalization methods are described: one based on discretization using Kronecker matrices and the other based on image upsampling using the Fast Fourier Transform (FFT). Section 3.5 provides experimental timing and accuracy results. Section 3.6 demonstrates the practical value of the algorithms during the analysis of a large, simulated dataset, while conclusions and recommendations for future work are contained in Section 3.7.

3.3 Basis Function Methods for Spatial Prediction

A ubiquitous model for a spatial process z is

$$z(\mathbf{s}) = y(\mathbf{s}) + \varepsilon(\mathbf{s}) \quad (3.1)$$

where $z(\mathbf{s})$ is the observation at spatial location $\mathbf{s}$, $y(\cdot)$ is a Gaussian Process, and $\varepsilon(\cdot)$ encapsulates independent, normally distributed measurement error with mean zero and a variance of τ^2 . We typically assume that the mean of $y(\cdot)$ depends linearly on a number of observed covariates $x_1(\cdot), \dots, x_d(\cdot)$

$$y(\mathbf{s}) = \mathbf{x}(\mathbf{s})^\top \boldsymbol{\beta} + g(\mathbf{s}), \quad \text{where } \mathbf{x}(\mathbf{s}) = [x_1(\mathbf{s}) \ \cdots \ x_d(\mathbf{s})], \quad (3.2)$$

although more sophisticated models are possible [28–30]. Note that $g(\cdot)$ is a zero-mean Gaussian Process; we denote its covariance function as $C(\cdot, \cdot)$:

$$\text{cov}(g(\mathbf{s}), g(\mathbf{s}')) = C(\mathbf{s}, \mathbf{s}'). \quad (3.3)$$

A central problem in this framework is to predict $y(\mathbf{s})$ at large numbers of unobserved locations. We focus on the case where these locations lie on a square grid⁶ $\{\mathbf{s}_{i,j}\}_{i,j=1}^n$. Fitting a model for this purpose with standard frequentist methodology involves parameter estimation (*i.e.* computing $\boldsymbol{\beta} \in \mathbb{R}^d$ and any free

⁶Our methods apply equally to rectangular grids; we restrict to square grids in our exposition for notational clarity.

parameters in $C(\cdot, \cdot)$ by maximum likelihood. These computations are dominated by evaluating the inverse and determinant of matrices of size $N \times N$, where $N = n \times n$, with a generic cost of $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ memory. For small problems this framework has the advantage that any covariance kernel can be considered to model spatial dependence in y . However, these computations become prohibitive for even moderately sized datasets (*i.e.* $N \sim 10^4$).

Basis function models, also known as fixed-rank approaches, sidestep this computational barrier by either reducing the size of key matrices or introducing sparsity into the model. We focus on approaches in which a set of $R = r \times r$ basis functions $\{\phi_{k,l}(\cdot)\}_{k,l=1}^r$ is constructed via translations and dilations of a fixed, compactly supported, unimodal function ψ :

$$\phi_{k,l}(\mathbf{s}) = \psi \left(\left\| \frac{\mathbf{s} - \mathbf{u}_{k,l}}{\gamma} \right\| \right). \quad (3.4)$$

Here γ is a scaling factor that controls the overlap of the basis functions. Common choices for ψ include bisquare functions [13], Wendland polynomials [1], and wavelets [31]. We further assume that the *basis function centers* $\{\mathbf{u}_{k,l}\}_{k,l=1}^r$ form a square, regular grid. Basis function models assume an expansion for g of the form

$$g(\mathbf{s}) = \sum_{k,l=1}^r c_{k,l} \phi_{k,l}(\mathbf{s}). \quad (3.5)$$

Randomness and correlation in $g(\cdot)$ is now captured by the coefficients $c_{1,1}, \dots, c_{r,r}$. We assume $\mathbf{c}$ is drawn from a multivariate Gaussian, $\mathbf{c} \sim \mathcal{N}(0, \Sigma)$, with covariance matrix $\Sigma \in \mathbb{R}^{R \times R}$. We define $\Phi \in \mathbb{R}^{N \times R}$ as the *regression matrix*, with columns indexing the basis functions and rows indexing locations. The full “Kriging” prediction for this model is then given by

$$\hat{y}(\mathbf{s}) = \mathbf{x}(\mathbf{s})^\top \hat{\boldsymbol{\beta}} + \hat{g}(\mathbf{s}), \quad \text{where } \hat{g}(\mathbf{s}) = \Phi \hat{\mathbf{c}} \quad (3.6)$$

The coefficients $\hat{\boldsymbol{\beta}}$ are found using generalized least squares (GLS), and

$$\hat{\mathbf{c}} = (\Phi^\top \Phi + \tau^2 \Sigma^{-1})^{-1} \Phi^\top (z - X \hat{\boldsymbol{\beta}}) \quad (3.7)$$

denotes our estimate for the basis function coefficients, where X is the covariate matrix. These forms are derived from the usual form of the spatial predictions using the Sherman-Morrison-Woodbury identity. When Σ is unstructured, the computational costs in this model reduce from $\mathcal{O}(N^3)$ to $\mathcal{O}(NR^2)$ [13], with R chosen to be small relative to N . A greater number of basis functions intuitively provides a better approximation, yielding a tradeoff between computational complexity and accuracy. Reduced cost has allowed these models to be applied to numerous environmental problems with $N \sim 10^5$ or greater [13, 24, 32].

If ψ has compact support, so will $\{\phi_{k,l}\}$, thus γ can be chosen such that Φ and its *Gram matrix* $\Phi^T \Phi$ are sparse. This motivates a final elaboration of the basis function model. Rather than computing Σ^{-1} , one can directly prescribe $Q = \Sigma^{-1}$ as a sparse precision matrix. This strategy is used in LatticeKrig [1] and other popular spatial models stemming from the stochastic partial differential equation (SPDE) formulation [14]. Although it will later be shown that in certain settings, one must evaluate the inverse of Q , its structured form allows computation to be sped up using sparse linear algebra.

Finally, we note that in practice a *multi-layered* approach is favored, in which basis functions centered on multiple nested, increasingly fine grids are used. For simplicity we limit our initial exposition to the single grid case, but note that our techniques directly extend to such *multi-resolution* approaches.

3.3.1 Prediction Artifacts in Basis Function Models

Although basis function models can be computationally efficient, they can also produce undesirable artifacts in their prediction surfaces. Figure 3.2 illustrates how the choice of basis function overlap creates a trade-off between the sparsity of Φ and the uniformity of the process marginal variance. Although a sparser Φ is desirable as it reduces computational cost, the features in the marginal variance induce similar patterns in the resulting predictions.

A natural solution to this issue is to *normalize* the basis functions to guarantee a constant process variance. We define $\phi_s = \begin{bmatrix} \phi_{1,1}(s) & \cdots & \phi_{r,r}(s) \end{bmatrix}^T$ as the vector formed by evaluating all basis functions at s . Accordingly, the basis functions are redefined as

$$\phi_{k,l}^*(s) = \frac{\phi_{k,l}(s)}{\sqrt{\text{Var}(g(s))}}, \quad \text{where } \text{Var}(g(s)) = \phi_s^T \Sigma \phi_s \quad (3.8)$$

It has previously been shown that this transformation provides a good approximation of a stationary GP [1]. For methods that define Σ explicitly, this computation is straightforward. However, for methods that are phrased in terms of the precision matrix this normalization is problematic. Here the variance computation is

$$\text{Var}(g(s)) = \phi_s^T Q^{-1} \phi_s \quad (3.9)$$

which requires evaluating Q^{-1} . Note that the normalization as proposed requires evaluation not only at all observed data locations, but also at all locations for prediction. When generating large prediction surfaces or predicting on fine grids, this becomes a bottleneck for otherwise very fast methods.

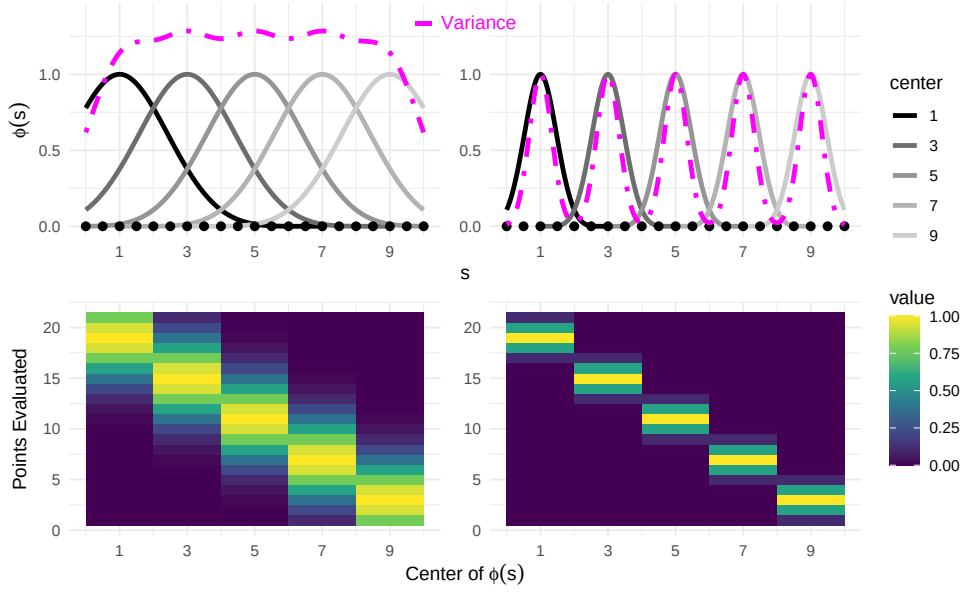


Figure 3.2 *Top row:* $\phi_1(s), \dots, \phi_5(s)$ where ψ is a Wendland polynomial, dilated to yield large overlap (*left*) and small overlap (*right*). *Bottom row:* the corresponding $\Phi \in \mathbb{R}^{21 \times 5}$, demonstrating the sparsity pattern. While less overlap results in a sparser Φ (*Right column*), it also results in a less uniform variance (magenta dotted line in *Top row*).

3.3.2 LatticeKrig Spatial Model

The LatticeKrig (LK) methodology again expands g in the same basis-function form as (3.5), namely

$$g(\mathbf{s}) = \sum_{k,l=1}^r c_{k,l} \phi_{k,l}(\mathbf{s}). \quad (3.10)$$

with a sparse precision matrix Q such that $\mathbf{c} \sim \mathcal{N}(0, Q^{-1})$. Here, the choice of ψ is the Wendland basis function. This choice comes with two major advantages: the functions have compact support $[0, 1]$, inducing sparsity into Φ , and are also positive definite. In radial form, this Wendland polynomial can be written as:

$$\psi(d) = \begin{cases} \frac{(1-d)^6(35d^2+18d+3)}{3} & \text{for } 0 \leq d \leq 1 \\ 0 & \text{otherwise.} \end{cases} \quad (3.11)$$

The sparse precision matrix Q is prescribed by modeling the spatial dependence among the coefficients $\mathbf{c}$ using a Gaussian Markov random field (GMRF), specifically a spatial autoregressive (SAR) model. The coefficients $\mathbf{c}$ are defined such that $B\mathbf{c} = \mathbf{e}$, where B is a sparse, SAR matrix which de-correlates the coefficients to white noise $\mathbf{e} \sim \mathcal{N}(0, 1)$. For two dimensions, the diagonal elements of B are $4 + \kappa^2$, while the four nearest neighboring grid points are -1 . The κ^2 parameter is adjustable, and defines the correlation

range of the SAR model. For interior grid centers this results in the corresponding row of B having only 5 non-zero values. Linear statistics then implies that $Q = BB^\top$ is the precision matrix of the coefficients $\mathbf{c}$.

The sparsity in both Φ and Q results in a model which allows for the total number of basis functions R to exceed the number of spatial locations N , with the limiting factor being the sparsity of the matrix $(\Phi^\top \Phi + \tau^2 \Sigma^{-1})$ rather than its $R \times R$ nominal size. This allows for fitting spatial data where the measurement error is small and g is close to interpolating the observations.

Although the LK model enjoys many benefits, the direct prescription of the precision matrix makes normalization problematic, as evaluating Q^{-1} is required for the variance calculation. Since Q is positive definite, the default method to calculate $\text{Var}(g(\mathbf{s}))$ in LK first computes the sparse Cholesky decomposition $Q = \mathcal{C}\mathcal{C}^\top$ and then solves $\mathcal{C}\mathbf{v} = \phi_{\mathbf{s}}$, whence

$$\text{Var}(g(\mathbf{s})) = \|\mathbf{v}\|_2^2 = \sum_{i,j=1}^r v_i^2. \quad (3.12)$$

In this work we propose a different approach, based on an alternative factorization of Q that is readily available, namely $Q = BB^\top$ where B is the SAR matrix described above. In fact, B can be decomposed into $B = L + \kappa^2 I$ where L is the five-point stencil discrete Laplacian [33]. Thus we propose to solve

$$(L + \kappa^2 I)\mathbf{v} = \phi_{\mathbf{s}} \quad (3.13)$$

instead, and then compute $\text{Var}(g(\mathbf{s}))$ as the norm of $\mathbf{v}$, as in (3.12). Exploiting the structure of (3.13) provides multiple avenues to speed up the computation of $\text{Var}(g(\mathbf{s}))$.

3.4 Methods

3.4.1 Normalization using the FFT

In deriving this method, we were inspired by the observation that the variance function $\mathbf{s} \mapsto \text{Var}(g(\mathbf{s}))$ appears nearly periodic. Before continuing, we offer a justification of why this is the case.

3.4.2 Approximate periodicity of the variance function

Because the $\phi_{k,l}$ are constructed as grid translations of a single parent function ψ ,

$$\begin{aligned} \phi_{\mathbf{s}} &= [\phi_{1,1}(\mathbf{s}) \quad \dots \quad \phi_{r,r}(\mathbf{s})] \\ &= [\psi(\|\mathbf{s} - \mathbf{s}_{1,1}\|/\gamma) \quad \dots \quad \psi(\|\mathbf{s} - \mathbf{s}_{r,r}\|/\gamma)] \\ &= [\psi_{\mathbf{s}}(\mathbf{s}_{1,1}) \quad \dots \quad \psi_{\mathbf{s}}(\mathbf{s}_{r,r})]. \end{aligned}$$

To simplify matters and avoid edge effects, we consider the case where the basis function centers form an infinite grid, specifically the integer lattice $\mathbb{Z}^2$. In this case, the variance is

$$\text{Var}(g(\mathbf{s})) = \sum_{\mathbf{m} \in \mathbb{Z}^2} v_{\mathbf{m}}^2, \quad (3.14)$$

where we ignore—for now—issues of summability. The linear system of interest is now a difference equation that is a discretization of a well known partial differential equation (PDE), the *screened Poisson equation*,

$$-\Delta v_{\mathbf{s}} + \kappa^2 v_{\mathbf{s}} = \psi_{\mathbf{s}}(\mathbf{z}). \quad (3.15)$$

where the solution $v_{\mathbf{s}} : \mathbb{R}^2 \rightarrow \mathbb{R}$ depends on the source term $\psi_{\mathbf{s}}(\mathbf{z})$. The PDE (3.15) may be solved using the Fourier transform, as discussed further in Appendix A.1, yielding the solution

$$v_{\mathbf{s}}(\mathbf{z}) = \iint_{\mathbb{R}^2} \frac{e^{-2\pi i(\mathbf{s}-\mathbf{z}) \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}(\boldsymbol{\omega}) d\boldsymbol{\omega}, \quad (3.16)$$

where $\tilde{\psi}(\cdot)$ denotes the Fourier transform of ψ . Returning to (3.14):

$$\text{Var}(g(\mathbf{s})) = \sum_{\mathbf{m} \in \mathbb{Z}^2} |\mathbf{v}_{\mathbf{s}}(\mathbf{m})|^2 = \sum_{\mathbf{m} \in \mathbb{Z}^2} \left| \iint_{\mathbb{R}^2} \frac{e^{-2\pi i(\mathbf{s}-\mathbf{m}) \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}(\boldsymbol{\omega}) d\boldsymbol{\omega} \right|^2 \quad (3.17)$$

It is now apparent that $\text{Var}(g(\mathbf{s}))$ is $\mathbb{Z}^2$ -periodic, as for any $\mathbf{n} \in \mathbb{Z}^2$

$$\text{Var}(g(\mathbf{s} + \mathbf{n})) = \sum_{\mathbf{m} \in \mathbb{Z}^2} |\mathbf{v}_{\mathbf{s}+\mathbf{n}}(\mathbf{m})|^2 = \sum_{\mathbf{m} \in \mathbb{Z}^2} \left| \iint_{\mathbb{R}^2} \frac{e^{-2\pi i(\mathbf{s}+\mathbf{n}-\mathbf{m}) \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}(\boldsymbol{\omega}) d\boldsymbol{\omega} \right|^2 \quad (3.18)$$

$$= \sum_{\mathbf{m}' \in \mathbb{Z}^2} \left| \iint_{\mathbb{R}^2} \frac{e^{-2\pi i(\mathbf{s}-\mathbf{m}') \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}(\boldsymbol{\omega}) d\boldsymbol{\omega} \right|^2 \quad \text{where } \mathbf{m}' := \mathbf{m} - \mathbf{n} \quad (3.19)$$

$$= \text{Var}(g(\mathbf{s})) \quad (3.20)$$

and indeed we approximately observe this periodicity in practice. We conjecture that the lack of exact periodicity in real applications is primarily due to the edge effects arising from using a finite lattice, which can be reduced by adding an additional buffer region with minimal computational overhead.

3.4.3 FFT-based upsampling

Based on the argument in Section 3.4.2, we expect $\text{Var}(g(\mathbf{s}))$ to be nearly periodic, thus we propose to apply a commonly used image upsampling technique—*Fourier interpolation* [34]—to approximately evaluate $\text{Var}(g(\mathbf{s}))$ on the square⁷ grid $\{\mathbf{s}_{i,j}\}_{i,j=1}^n$. Fourier interpolation begins by *subsampling*, that is evaluating $\text{Var}(g(\mathbf{s}))$ on a coarser grid which contains the grid of basis function centers:

$$\{\mathbf{u}_{k,l}\}_{k,l=1}^r \subset \{\mathbf{s}_{i_p,j_q}\}_{p,q=1}^{\tilde{n}} \subset \{\mathbf{s}_{i,j}\}_{i,j=1}^n \quad (3.21)$$

and $\tilde{n} < n$. In the ideal case where $\text{Var}(g(\mathbf{s}))$ is periodic, as discussed in Section 3.4.2, we observe that it is naturally *band-limited*, with the spacing between the $\mathbf{u}_{k,l}$ determining the highest frequency. Consequently, by the Shannon-Nyquist sampling theorem [35, Appendix C.1], as long as $\tilde{n} \geq 2r + 1$, the set of evaluations $\{\text{Var}(g(\mathbf{s}_{i_p,j_q}))\}_{p,q=1}^{\tilde{n}}$ determine $\text{Var}(g(\mathbf{s}))$ exactly. More precisely, letting $\{\tilde{f}_{p,q}\}_{p,q=1}^{\tilde{n}}$ denote the discrete Fourier transform (DFT) of $\{\text{Var}(g(\mathbf{s}_{i_p,j_q}))\}_{p,q=1}^{\tilde{n}}$:

⁷This also applies to rectangular grids, but we restrict to square grids in our exposition for notational clarity

Algorithm 1 Fourier Interpolation

- 1: **Inputs:** Parent function ϕ and fine evaluation grid $\{\mathbf{s}_{i,j}\}_{i,j=1}^n$
 - 2: Choose basis function centers $\{\mathbf{u}_{k,l}\}_{k,l=1}^r$ and coarse evaluation grid $\{\mathbf{s}_{i_p,j_q}\}_{p,q=1}^{\tilde{n}}$ such that $\tilde{n} \geq 2r + 1$ and $n = M(\tilde{n} - 1) + 1$.
 - 3: Compute $\{\text{Var}(g(\mathbf{s}_{i_p,j_q}))\}_{p,q=1}^{\tilde{n}}$ exactly by solving (3.13) for all $\mathbf{s} = \mathbf{s}_{i_p,j_q}$.
 - 4: Compute $\{\tilde{f}_{p,q}\} = \mathcal{F}(\{\text{Var}(g(\mathbf{s}_{i_p,j_q}))\})$
 - 5: Compute $\{f_{i,j}\}_{i,j=1}^{n^*}$ by zero-padding, as per (3.24), where $n^* = M\tilde{n}$.
 - 6: Compute $\{\widehat{\text{Var}}_{i,j}\}_{i,j=1}^{n^*} = \mathcal{F}^{-1}(\{\tilde{f}_{i,j}\}_{i,j=1}^{n^*})$.
 - 7: Compute $\{\widehat{\text{Var}}_{i,j}\}_{i,j=1}^n$ by removing the last $M - 1$ rows and columns of $\{\widehat{\text{Var}}_{i,j}\}_{i,j=1}^{n^*}$.
 - 8: **Return:** $\{\widehat{\text{Var}}_{i,j}\}_{i,j=1}^n \approx \{\text{Var}(g(\mathbf{s}_{i,j}))\}_{i,j=1}^n$.
-

$$\tilde{f}_{p,q} = \sum_{p=1}^{\tilde{n}} \sum_{q=1}^{\tilde{n}} \text{Var}(g(\mathbf{s}_{i_p,j_q})) e^{-2\pi i p/\tilde{n}} e^{-2\pi i q/\tilde{n}}, \quad (3.22)$$

for any $\mathbf{s} \in \mathbb{R}^2$ we may evaluate $\text{Var}(g(\mathbf{s}))$ as

$$\text{Var}(g(\mathbf{s})) = \sum_{p=1}^{\tilde{n}} \sum_{q=1}^{\tilde{n}} \tilde{f}_{p,q} e^{2\pi i s_1 p/\tilde{n}} e^{2\pi i s_2 q/\tilde{n}} \quad \text{where } \mathbf{s} = (s_1, s_2). \quad (3.23)$$

Thus, we may use (3.23) to evaluate $\text{Var}(g(\mathbf{s}_{i,j}))$ *without solving the associated linear system*. Instead of evaluating (3.23) directly, we use a standard construction whereby the $\{\tilde{f}_{p,q}\}_{p,q=1}^{\tilde{n}}$ is zero-padded to obtain $\{f_{i,j}\}_{i,j=1}^n$ as:

$$f_{i,j} = \begin{cases} \tilde{f}_{p,q} & \text{if } 1 \leq p \leq \frac{\tilde{n}}{2}, 1 \leq q \leq \frac{\tilde{n}}{2} \text{ and } i = p, j = q \\ \tilde{f}_{p,q} & \text{if } 1 \leq p \leq \frac{\tilde{n}}{2}, \frac{\tilde{n}}{2} < q \leq \tilde{n} \text{ and } i = p, j = q + n - \tilde{n} \\ \tilde{f}_{p,q} & \text{if } \frac{\tilde{n}}{2} < p \leq \tilde{n}, 1 \leq q \leq \frac{\tilde{n}}{2} \text{ and } i = p + n - \tilde{n}, j = q \\ \tilde{f}_{p,q} & \text{if } \frac{\tilde{n}}{2} < p \leq \tilde{n}, \frac{\tilde{n}}{2} < q \leq \tilde{n} \text{ and } i = p + n - \tilde{n}, j = q + n - \tilde{n} \\ 0 & \text{otherwise} \end{cases} \quad (3.24)$$

whence the inverse DFT is applied to $\{f_{i,j}\}_{i,j=1}^n$, yielding $\{\text{Var}(g(\mathbf{s}_{i,j}))\}_{i,j=1}^n$. We use the ubiquitous Cooley-Tukey Fast Fourier Transform (FFT) [36] to compute the DFT and inverse DFT, at a cost of $\mathcal{O}(\tilde{n} \log \tilde{n})$ and $\mathcal{O}(n \log n)$ operations respectively. Thus, Fourier interpolation reduces the number of linear system solves required to compute $\{\text{Var}(g(\mathbf{s}_{i,j}))\}_{i,j=1}^n$ from n^2 to $\tilde{n}^2$ with negligible computational overhead.

In the realistic setting where $\text{Var}(g(\mathbf{s}))$ is only approximately periodic, Fourier interpolation yields an approximation to $\{\text{Var}(g(\mathbf{s}_{i,j}))\}_{i,j=1}^n$. Nonetheless, as verified by our experiments in Section 3.5.2, this approximation is in most cases quite good. While increasing $\tilde{n}$ beyond the theoretical minimum of $\tilde{n} = 2r + 1$ improves accuracy, it also raises the computational expense. In practice, we find sampling rates of $\tilde{n} = 4r - 3$ or $\tilde{n} = 4r$ yield a good balance between these two competing factors. To avoid errors resulting from content bleeding from the left edge to the right (and the top edge to the bottom) we choose $\tilde{n}$ such that $n = M(\tilde{n} - 1) + 1$, for positive integer scale factor M , and ensure that the corners of the coarse

grid coincide with those of the fine grid. In scenarios such as timing the method, where content bleeding is not a concern, simple modification allows for any choice of $\tilde{n} < n$ to be made. The resulting algorithm is represented pictorially in Figure 3.3 and in pseudocode as Algorithm 1, where we use $\mathcal{F}$ to denote the operation of applying the FFT.

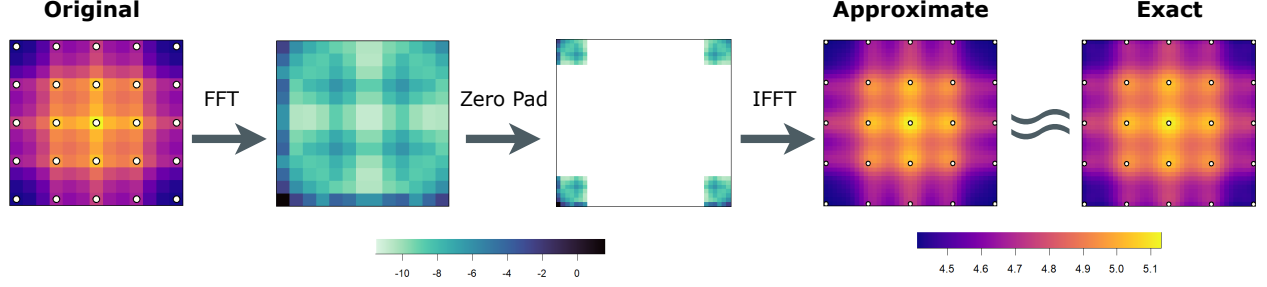


Figure 3.3 Diagrammatic description of the proposed method. In this case $r = 5$ and $\tilde{n} = 13$, with basis function centers denoted by white points. The calculation is then upsampled to a grid with $n = 31$. A visual comparison of the true (exact) variance on the finer grid is provided. The log of the values is taken in the FFT domain to provide a better visual demonstration.

3.4.4 Normalization using Kronecker Structure

An alternate approach to that proposed in Section 3.4.3, where the number of linear system solves is reduced, one may also consider speeding up each linear system solve. Elliptic PDEs such as the screened Poisson equation are well-studied, and many fast schemes exist for solving their discretizations (3.13). One approach utilizes the Kronecker product to represent operations along each dimension separately [37]. Here, the computation for the variance is exact, but requires constant κ^2 , meaning all diagonal elements of B are the same.

With reference to (3.13), we note that $\mathbf{v} \in \mathbb{R}^R$ can naturally be reshaped into a matrix $V \in \mathbb{R}^{r \times r}$. Using Kronecker product identities and the “vec” vectorization operation

$$B\mathbf{v} = \text{vec}(AV + VA^\top) = \text{vec}(AVI_r + I_rVA^\top) = (A \otimes I_r + I_r \otimes A) \text{vec}(V) = (A \otimes I_r + I_r \otimes A)\mathbf{v} \quad (3.25)$$

where $A \in \mathbb{R}^{r \times r}$ is the tridiagonal matrix

$$A = \begin{bmatrix} 2 + \kappa^2/2 & -1 & 0 & \dots & 0 \\ -1 & 2 + \kappa^2/2 & -1 & \dots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & \dots & -1 & 2 + \kappa^2/2 & -1 \\ 0 & \dots & 0 & -1 & 2 + \kappa^2/2 \end{bmatrix} \quad (3.26)$$

Pre-multiplication by A represents column operations, while post-multiplying by A operates on rows, formulating a discretization across two dimensions. In other words,

$$B = L + \kappa^2 I = A \otimes I_r + I_r \otimes A \quad (3.27)$$

Equation (3.27) may be further simplified using the eigen-decomposition $A = UDU^\top$:

$$B = (UDU^\top \otimes I_r + I_r \otimes UDU^\top) = \mathcal{U}\mathcal{D}\mathcal{U}^\top \quad (3.28)$$

with $\mathcal{U} = U \otimes U$ and $\mathcal{D} = (D \otimes I_r + I_r \otimes D)$. It also follows that $\mathcal{U}$ is an orthonormal matrix and $\mathcal{D}$ is diagonal, so we have the eigen decomposition for B . We use this decomposition to get a compact expression of the process variance:

$$\text{Var}(g(\mathbf{s})) = \phi_s^\top Q^{-1} \phi_s = \phi_s^\top B^{-1} B^{-\top} \phi_s = \|B^{-\top} \phi_s\|^2 = \|\mathcal{U}\mathcal{D}^{-1}\mathcal{U}^\top \phi_s\|^2 = \|\mathcal{D}^{-1}\mathcal{U}^\top \phi_s\|^2 \quad (3.29)$$

where the final inequality holds as $\mathcal{U}$ is orthogonal. As $\mathcal{U}^\top = U^\top \otimes U^\top$, we observe that

$$\mathcal{U}^\top \phi_s = \text{vec}(U^\top \Phi_s + \Phi_s U^\top) \quad (3.30)$$

where $\Phi_s \in \mathbb{R}^{r \times r}$ is the reshaping of $\phi_s \in \mathbb{R}^R$ into matrix form. Consequently, $\mathcal{U}^\top \phi_s$ may be computed using only two $r \times r$ matrix-matrix multiplies at a cost of $2(2r^3 - r^2)$. As $\mathcal{D}$ is diagonal, multiplication by $\mathcal{D}^{-1}$ requires r^2 operations. Thus finding the variance at all N spatial locations costs

$$N(4r^3 - 2r^2 + r^2) = \mathcal{O}(Nr^3) = \mathcal{O}(NR^{3/2}), \quad (3.31)$$

This method is particularly powerful in settings where the FFT method is limited, such as when the number of basis functions approaches or exceeds the number of points. Timing for this method is provided in Section 3.5.1. More sophisticated numerical methods for solving elliptic PDEs, such as the multigrid method [38] may yield even greater savings; we leave this for future work.

3.4.5 Multi-Resolution Implementation

To guarantee stationarity in a multi-resolution model, each layer of basis functions must be normalized. In order to integrate both the FFT and Kronecker speedups into the `LatticeKrig` R package, two new options were created. The user can choose to use either the Kronecker algorithm alone, or a combined FFT and Kronecker approach. When normalizing a multi-resolution model, the combined approach uses the FFT method for coarser levels, where the number of spatial locations significantly exceeds the number of points, while the Kronecker algorithm is used for the remaining levels. When appropriate, the Kronecker algorithm is also utilized in the inner workings of the FFT method, where it is used to perform the small, exact calculation. This use of both speedups is necessary to avoid hindering the full capabilities of `LatticeKrig`, as the framework was designed for settings where one can choose $R > N$. Although the bulk of computational overhead lies in the normalization of levels with finer resolutions, the combined approach

still demonstrates a significant speedup in Section 3.6.

3.5 Results

We present timing results to demonstrate the computational gains from both algorithms, and assess the consequent approximation errors of the FFT normalization. Our experiments were conducted on a laptop with an 11th Gen Intel(R) Core(TM) i9-11900H processor at 2.50GHz, 16 GB RAM, and 8 cores. For simplicity, all experiments are performed with regular, square grids for both basis functions $\{\mathbf{u}_{k,l}\}_{k,l=1}^r$, and spatial locations $\{\mathbf{s}_{i,j}\}_{i,j=1}^n$, with a spatial domain $[-1, 1] \times [-1, 1]$. Here the coarse grid $\{\mathbf{s}_{i_p,j_q}\}_{p,q=1}^{\tilde{n}}$ will be chosen with $\tilde{n} = 4r$, providing a balance between computational efficiency and approximation accuracy.

3.5.1 Timing

Our first aim is to measure the wall clock time required to perform the basis function computation, specifically using the `LKrig.basis` function. We evaluate four distinct scenarios:

1. computation without any normalization,
2. computation with exact normalization using sparse linear algebra (default),
3. computation leveraging Kronecker products for exact normalization, and
4. computation using the approximate FFT normalization method.

These experiments contextualize the speedup from both algorithms by recording the time to compute the basis functions; a step that often requires the most computational overhead in a realistic spatial analysis. For each scenario, r is varied between 25 and 100, and the side length of the fine grid n ranges from 500 to 2,000. Furthermore, we have the option to adjust the κ^2 parameter and the number of buffer points. For these experiments, we set $\kappa^2 = 0.05$ and employ 10 buffer points in our analysis to mitigate edge effects associated with a finite grid. These choices result in the largest dataset having size $N = 4,000,000$ and a maximum of $R = 10,000$ total basis functions, which are consistent with testing spatial models where traditional, dense linear algebra is not feasible. These results also provide an opportunity to understand how timing will scale for much larger problems.

As depicted in Figure 3.4, the FFT normalization algorithm is consistently the fastest, often requiring only marginally more time than the un-normalized calculation. The FFT method achieves maximum efficiency when $r = 100$, yielding a speedup factor of $56\times$ faster than the default method. As expected, the timing slope of the FFT method flattens as the number of basis functions increases. This is because the algorithm is primarily limited by the initial, exact calculation on the coarse grid, which is determined by the number of basis functions. Conversely, the Kronecker method emerges as an alternative when accuracy

cannot be compromised, offering a speedup of at most $8\times$ over the default method. To roughly illustrate scale between the three normalization methods, we take the case with $r = 50$ and $n = 2000$. The default method requires approximately 20 minutes, the Kronecker algorithm takes about 5 minutes, and the FFT algorithm takes a mere 25 seconds.

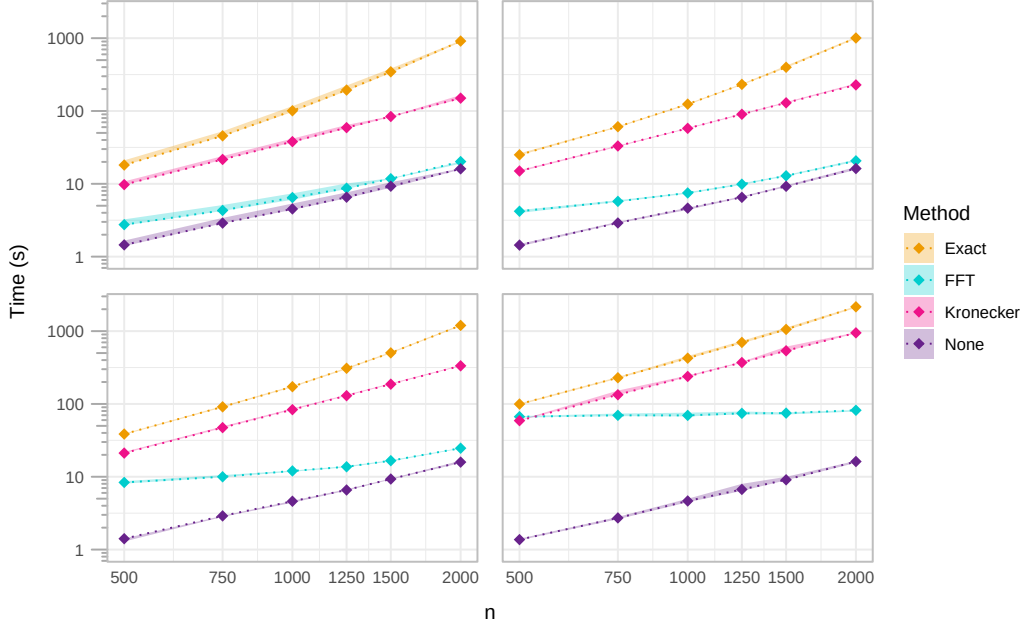


Figure 3.4 Timing results on a logarithmic scale for each method where r is varied: 25 (*top left*), 35 (*top right*), 50 (*bottom left*), 100 (*bottom right*), as is n : (500, 750, 1000, 1250, 1500, 2000). Dots represent the median time of 5 iterations of the simulation, and shading represents maximum and minimum values.

3.5.2 Error in the FFT Method

It is important to determine the approximation error from the FFT-based method. Given the lack of physical significance attached to the variances, we use relative error metrics for comparison. Figure 3.5 shows both the mean percentage error and maximum absolute percentage errors, computed by subtracting the FFT approximation $\{\widehat{\text{Var}}(g(\mathbf{s}_{i,j}))\}_{i,j}$ from the results of the exact calculation at each $\mathbf{s}_{i,j}$. Across all tested configurations of basis function counts and grid side lengths, we find that the error introduced by the FFT method remains consistently low. Specifically, the mean percent error hovers around 0.01%, while the maximum values range between 1% and 2%. We demonstrate that the impact of this error is essentially negligible in Section 5, when prediction accuracy and presence of artifacts are evaluated.

Additionally, we find these results are robust to different values of κ^2 and variations in the number of buffer points. Error is minimized at the default value of $\kappa^2 = 0.05$, with the maximum error increasing to roughly 4.5% at a large value of $\kappa^2 = 1$. A choice of 10 buffer points—slightly higher than the default value

of 5—minimizes the error, although all tested values between (and including) 5 and 40 do not surpass a maximum error of 3%. When the number of buffer points decreases to 0, error rises as high as 35%, as the assumption of periodicity is strongly violated in this scenario. Edge effects due to finite grids are a well known issue, rendering the high error from a lack of buffer points irrelevant, as this scenario would not be seen in practice.

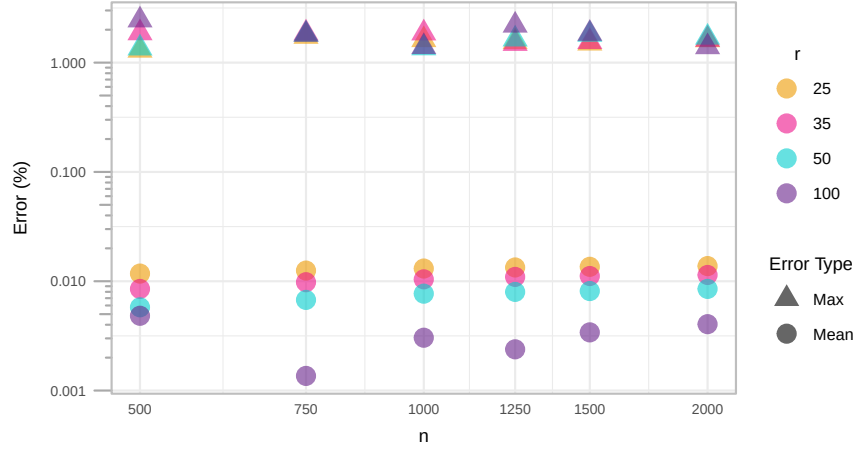


Figure 3.5 Mean and maximum error on a logarithmic scale for the FFT normalization method. Once again, both r : (25, 35, 50, 100), and n : (500, 750, 1000, 1250, 1500, 2000) are varied.

3.6 Application

We perform a comparison of prediction results and timing using the different normalization methods on a large, simulated dataset. Circulant embedding [39] is used to simulate a Gaussian field on a 1153×1153 grid with a spatial domain of $[0, 90] \times [0, 90]$, a Matern covariance parameterized by a range of $\kappa_{\mathcal{M}} = 6$ and a smoothness $\nu = 1$, and added white noise $\varepsilon \sim \mathcal{N}(0, 0.2)$. Observation values $z(\mathbf{s})$ are scaled using an affine transformation that results in a range of roughly -14.05 to 15.45 . We employ two sampling schemes, shown in Figure 3.6. The first is Missing at Random (MAR), in which 80% of the values are randomly removed, leaving the remaining 20% as training data. The second is Missing Blocks (Blocks), in which we remove three square blocks of size 100×100 —in total removing 30,000 observations from the image—and test the models’ long range predictive abilities. We record the run time of the full spatial analysis, and generate prediction surfaces to assess the accuracy of out-of-sample predictions relative to the test data for each of the four normalization choices:

1. None: No normalization at all.
2. Both: The combined algorithm, using the FFT at coarser layers and the Kronecker for the rest.

3. Kronecker: The exact Kronecker algorithm.
4. Exact: The default, exact computation.

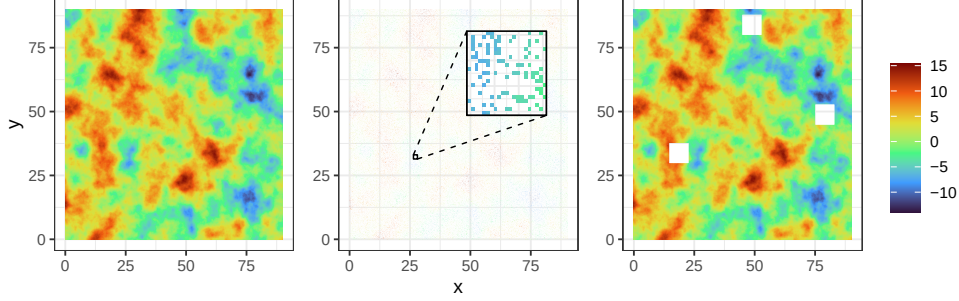


Figure 3.6 Full data set of $N = 1,329,409$ locations (*left*) along with training visualizations of both the MAR (*center*) and Blocks (*right*) sampling schemes.

We fit a LatticeKrig model with 4 levels such that r is 25, 49, 97, and 193, respectively ($R = 65,844$ total basis functions). When the “both” speedup is used in this experiment, FFT normalization occurs on the first 3 levels, and the last level uses the Kronecker speedup. We choose a value of $\kappa^2 = 0.015$ and a buffer of 10 basis functions, following the findings in Section 3.5.2. The times for both fitting the model and generating a prediction surface on the whole grid are combined to form total run time.

Table 3.1 Mean absolute error (MAE), root-mean-squared prediction error (RMSPE), and total run time are recorded for each choice of normalization method for both the MAR and Blocks sampling schemes. Arrows ($\uparrow, \downarrow$) indicate the desirable direction for a metric.

Method	Blocks			MAR		
	MAE $\downarrow$	RMSPE $\downarrow$	Time (min)	MAE $\downarrow$	RMSPE $\downarrow$	Time (min)
None	1.1672	1.6432	28.68	0.1983	0.2486	16.57
Both	1.0527	1.4735	78.82	0.2051	0.2570	45.08
Kronecker	1.0508	1.4712	111.49	0.2051	0.2570	57.90
Exact	1.0508	1.4712	173.75	0.2051	0.2570	95.51

As seen in Table 3.1, the “Kronecker” and “Both” options offer significant speedups in the context of the full spatial analysis. The combined option is roughly 25% faster than the strictly Kronecker-based counterpart, and roughly $2\times$ faster than the classic normalization method. We expect time differences between the three normalization methods to continue to widen as data-set size increases due to their differing computational complexities. The Blocks case takes more time as the number of points one needs to fit the model to initially is much greater than the MAR case. As for accuracy, all choices of normalization provide similarly excellent fit. The impact from the artifacts in the un-normalized case is

first noticeable in the differing values of MAE and RMSPE, while the results from the approximate, combined approach are near identical to those of the exact methods. In the Blocks case, the un-normalized method is less accurate, while in the MAR case it performs slightly better; an unexpected result. Despite this observation, we do not deem these differences in accuracy significant, as the true detriment of artifacts comes from their inaccurate representation of physical data. This is clearly demonstrated in Figure 3.7, which zooms in to a small subset of the data (lat: [40, 50], lon: [65, 75]) to display the artifacts more clearly.

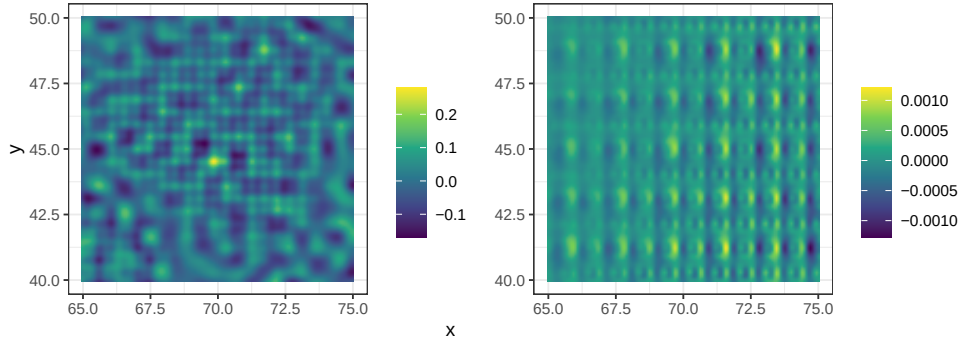


Figure 3.7 Artifacts obtained by taking the difference of the unnormalized and exactly normalized prediction surfaces in the MAR case (*left*). The remnants of those artifacts after approximate normalization (*right*).

The periodic pattern from not normalizing is striking, and is reminiscent of the basis functions and the shape of their variance. The average amplitude of the unnormalized artifacts is on the order of 10^{-2} . The combined approach removes roughly 99% of the artifacts, reducing their average amplitude to an order of 10^{-4} . If scaled to match the range of the no-normalization image (left), the image on the right side of Figure 3.7 displays as a solid color due to the greatly diminished amplitude. Note that both the default and Kronecker normalization methods would remove the artifacts entirely.

3.7 Conclusion

A useful model for analyzing large spatial data is the representation of a Gaussian Process as a sum of basis functions with a multivariate normal distribution prescribed for the coefficients. An important adjustment to this model is a normalization to avoid artifacts in the predicted surface that arise from the arrangement of the locally supported basis functions. Despite the value from eliminating these artificial features, this normalization step constitutes a computational bottleneck. In this work we have introduced two fast algorithms for this step, reducing the cost of estimation and prediction for large spatial data sets. These methods also constrain the spatial model to have a constant marginal variance, another attractive feature.

The Kronecker method takes advantage of the SAR structure of the model’s precision matrix, providing a way to speed up each individual linear system solve required for normalization. This is an exact calculation and enjoys a speedup of roughly $5\times$ over the default method for computing normalization weights. However, it is limited to the case of a constant κ^2 and a two dimensional, rectangular geometry. In addition, even if these conditions are met, there exist faster ways of solving discretizations of elliptic PDEs, which could be utilized for this purpose. A future research direction is to leverage numerical and graphical algorithms for PDEs to improve the computation for the normalization, or to derive approximate but analytical expressions for the variance function.

An alternative to the Kronecker approach, the FFT method requires fewer linear systems solves, and instead upsamples the variance to much finer grids. Although this method is approximate, it is very accurate, and can be faster than the default by almost two orders of magnitude. Additionally, it can be used for any non-stationary covariance model for the coefficients, and could be expanded to work for higher dimensions. The FFT method is currently confined to models or layers where the number of basis functions is significantly less than that of the number of points, and requires the coarse, exact calculation to be done on a nested subgrid of the data. There exists a broad range of other image upsampling techniques that could be implemented, using our general methodology as a guide. In particular, the self-similarity of the variance function could be exploited to avoid many exact evaluations.

To streamline the examples in this work, we have assumed that the data locations are registered to a grid. This restriction can be overcome by computing the normalization on a fine grid, on the order of the data location spacing, and then using a fast local interpolation for the off-grid locations. Because the FFT method can interpolate to very fine grids efficiently, handling irregularly spaced locations will not impact the timing.

All numerical experiments for this paper were done specifically within the LatticeKrig framework. This was done for the convenience of working within a tested R package and also to facilitate distribution of the algorithms. However, provided the radial basis functions are organized on a regular grid and the coefficients follow a sparse precision matrix, these algorithms could be useful for other basis function models. Future work could study other basis functions that are more amenable to normalization, such as tensor products, and also geometries where rectangular grids are not available, such as spheres. In general, we believe that basis function models are an important tool for large spatial data analysis, and fine tuning their properties to match standard spatial models is an important area for further research.

3.8 Acknowledgments

The authors acknowledge the work of Nathan Lenssen in implementing the Kronecker normalization method into the `LatticeKrig` package. We would also like to thank Brennan Sprinkle for helpful discussion. Finally, we are grateful to the reviewers and editors for their comments and suggestions that improved the quality of this manuscript.

3.9 Conflict of Interest Statement

The authors declare no potential conflict of interests.

3.10 Data Availability Statement

The data that support the findings of this study are openly available in Normalization-Paper at github.com/antonyxsik/Normalization-Paper. The `LatticeKrig` package is freely available on the CRAN repository for R cran.r-project.org/web/packages/LatticeKrig/.

CHAPTER 4

NON-STATIONARITY, ANISOTROPY, AND NEURAL PARAMETER ESTIMATION

4.1 Non-Stationarity and Anisotropy

While isotropy and stationarity are convenient, simplifying assumptions, they are rarely defensible for real spatial data, particularly over large domains or at fine resolutions. Relaxing these assumptions yields the more general notions of *non-stationarity* and *anisotropy*, which can be described as follows:

- *Non-Stationarity* relaxes invariance under translation, allowing the covariance structure to depend on absolute location within the domain. That is, in general $C(\mathbf{s}, \mathbf{s}') \neq C(\mathbf{s} + \mathbf{h}, \mathbf{s}' + \mathbf{h})$ for a shift $\mathbf{h}$. In practice this means that key statistical parameters (such as the correlation range) are permitted to vary across the domain.
- *Anisotropy* relaxes invariance under rotation, allowing the covariance between two locations to depend on the direction of their separation in addition to its magnitude. In two dimensions, one can imagine anisotropy as the property that the set of all points sharing a fixed correlation with a given point forms a more complex shape, such as an ellipse, rather than a perfect circle.

As an illustrative example, there may be a dramatic shift in the range of correlation when measuring key variables (such as temperature and precipitation [40]) on land as compared to in the ocean, implying non-stationarity. Along the coastline separating the ocean from the land, a variable may be highly correlated with distant locations in the ocean, but may exhibit little correlation with distant locations on land, implying anisotropy. Flexibly modeling such behavior is essential for accurate interpolation of observations, realistic emulation of gridded data products for large-ensembles, and other applications. The following sections in this chapter describe how non-stationarity and anisotropy are incorporated into the LK model, the inferential challenges introduced by this added flexibility, and one line of research aimed at addressing them.

4.2 The Non-Stationary LatticeKrig Model

We extend the LK model to accommodate non-stationarity and anisotropy by generalizing the SAR construction on the coefficients first shown in Section 2.4. We first introduce anisotropy by incorporating additional parameters into the model, and then introduce non-stationarity by allowing them to vary across the spatial domain.

In the isotropic case, the contour of constant correlation around a location can be visualized as a circle whose radius is governed by κ^2 . Anisotropy generalizes this by replacing the circle with an ellipse, which requires two additional parameters: ρ , which sets the aspect ratio of the ellipse, and θ , which sets its orientation. The combined effect of κ^2 , ρ , and θ on the correlation contours is illustrated in Figure 5.2.

These parameters are added into the SAR matrix B , which controls the dependence between the coefficients. Concretely, we modify the SPDE in Equation (2.10) by replacing the Laplacian Δ with the generalized Laplacian $\nabla \cdot D \nabla$, where $D \in \mathbb{R}^{2 \times 2}$ is a positive definite dispersion matrix constructed via its eigendecomposition:

$$D = R^\top \Lambda R, \quad R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \rho & 0 \\ 0 & \frac{1}{\rho} \end{bmatrix}. \quad (4.1)$$

Here R is a rotation matrix parametrized by θ and Λ is a diagonal scaling matrix parametrized by ρ . With this generalization, the anisotropic SPDE becomes

$$(\kappa^2 - \nabla \cdot D \nabla) g(\mathbf{s}) = \mathcal{W}(\mathbf{s}). \quad (4.2)$$

Discretizing (4.2) on a uniform grid via finite differences yields the nine-point stencil

$$\begin{array}{c|c|c} \frac{D_{1,2}}{2} & -D_{2,2} & -\frac{D_{1,2}}{2} \\ \hline -D_{1,1} & \kappa^2 + 2D_{1,1} + 2D_{2,2} & -D_{1,1} \\ \hline -\frac{D_{1,2}}{2} & -D_{2,2} & \frac{D_{1,2}}{2} \end{array} \quad (4.3)$$

which determines the rows of the SAR matrix B . The stationary stencil can be recovered by setting $\rho = 1$ and $\theta = 0$, in which case $D = I$, the 2×2 identity matrix.

The extension to non-stationarity is made by allowing the parameters κ^2 , ρ , and θ to vary across the domain. Non-stationary capabilities of the LK model were first demonstrated by Wiens et al. [41] in a formulation that omitted the basis-function layer, leaving a SAR where the parameters varied at every spatial location. This is the same construction used by the LatticeVision framework introduced in Chapter 5. The work in Chapter 6 was the first to incorporate this idea into the full basis-function model, where the parameters vary across the lattice of basis-function centers $\{\mathbf{u}_j\}_{j=1}^m$ instead. To remain cohesive with the basis function model in Section 2.4, we describe the case in which the parameters vary across the lattice here. With $\kappa^2(\mathbf{u})$ and $D(\mathbf{u})$ varying across the lattice, the rows of the SAR matrix B are determined by the spatially varying stencil

$$\begin{array}{c|c|c} \frac{D_{1,2}(\mathbf{u})}{2} & -D_{2,2}(\mathbf{u}) & -\frac{D_{1,2}(\mathbf{u})}{2} \\ \hline -D_{1,1}(\mathbf{u}) & \kappa^2(\mathbf{u}) + 2D_{1,1}(\mathbf{u}) + 2D_{2,2}(\mathbf{u}) & -D_{1,1}(\mathbf{u}) \\ \hline -\frac{D_{1,2}(\mathbf{u})}{2} & -D_{2,2}(\mathbf{u}) & \frac{D_{1,2}(\mathbf{u})}{2} \end{array}. \quad (4.4)$$

Further details can be found in Appendix B.1.3. This anisotropic, non-stationary formulation has several notable advantages. First, the model is defined by the dependence structure in the B (and consequently Q)

matrix, avoiding the difficult specification of an analytical covariance function that accommodates such extreme flexibility. Because $Q = B^\top B$, the precision matrix is positive semi-definite by construction, and is positive definite whenever B is non-singular. It follows that the implied covariance Q^{-1} is then guaranteed to be a valid, positive definite covariance matrix. Second, this B (and Q) matrix remains very sparse, retaining many of the computational advantages from the isotropic, stationary model. Finally, the model is now controlled by three interpretable parameter fields κ^2 , ρ , and θ . The interpretability is easily demonstrated by returning to the ellipse analogy: each lattice center $\mathbf{u}_j$ now carries its own correlation ellipse, with size, aspect ratio, and orientation set locally by $\kappa^2(\mathbf{u}_j)$, $\rho(\mathbf{u}_j)$, and $\theta(\mathbf{u}_j)$. The interplay between these local ellipses yields rich, flexible spatial dependence structures that allow better modeling of big and heterogeneous spatial data.

4.2.1 The Trouble with Non-Stationarity

The added flexibility of non-stationary spatial models comes at the cost of substantially more difficult parameter estimation. In the case of the LK model, estimating the parameter *fields* κ^2 , ρ , and θ is substantially more difficult than estimating their stationary counterparts.

In the stationary case, the model is governed by a small, fixed number of scalar parameters, and standard maximum likelihood or Bayesian inference is tractable. When dealing with data such as climate model outputs, or high resolution images, a natural choice is to allow these parameters to take a distinct value at every basis-function center, $\mathbf{u}_j$. For a lattice with m centers, this produces $3m$ unknowns to be estimated jointly, where $3m$ can easily be on the order of thousands or tens of thousands of parameters. Although the cost of a single likelihood evaluation is determined by the number of spatial locations in both the stationary and non-stationary settings, it is the number of evaluations required that scales with the number of parameters in the model. As a result, standard maximum likelihood optimization and traditional Bayesian sampling become infeasible, both because the optimization landscape is high-dimensional and non-convex, and because each step is itself expensive.

One significant line of work addresses this problem by replacing repeated likelihood evaluation with neural networks that are trained to map directly from observed spatial fields to estimates of the underlying parameters. We give a brief introduction to this line of work in the following section.

4.3 Neural Parameter Estimation

Neural parameter estimation is a recent line of work in which a neural network is trained to map directly from observed data to estimates of the parameters of an underlying statistical model. It is also referred to as neural amortized inference [42, 43], and the trained networks themselves are commonly called

Neural Bayes Estimators [44]. The field draws ideas from two more established areas: learning to optimize (L2O) [45, 46], a line of work which uses neural networks to improve optimization procedures, and simulation-based inference (SBI) [47, 48].

Let Θ denote the *parameter space* of the underlying statistical model and $\mathcal{S}$ the *sample space* of possible data. Data $Y \in \mathcal{S}$ are generated according to a density $f(Y | \mathcal{P})$ for parameters $\mathcal{P} \in \Theta$. A point estimator is any mapping from the sample space to the parameter space,

$$\hat{\mathcal{P}} : \mathcal{S} \longrightarrow \Theta, \quad (4.5)$$

where $\hat{\mathcal{P}}(Y)$ estimates the parameters that originally generated Y . The goal of neural parameter estimation is to learn this map with a neural network. Not every such map is a good estimator, and in order to define optimal ones, we adopt a decision-theoretic viewpoint [44]. Given a non-negative loss $\mathcal{L}(\mathcal{P}, \hat{\mathcal{P}}(Y))$, the risk (R) of an estimator is its loss averaged over all data realizations for a fixed parameter, and the Bayes risk ($\mathcal{R}$) averages this risk over some prior measure Ω on the parameters:

$$R(\mathcal{P}, \hat{\mathcal{P}}(\cdot)) = \int_{\mathcal{S}} \mathcal{L}(\mathcal{P}, \hat{\mathcal{P}}(Y)) f(Y | \mathcal{P}) dY, \quad \mathcal{R}_{\Omega}(\hat{\mathcal{P}}(\cdot)) = \int_{\Theta} R(\mathcal{P}, \hat{\mathcal{P}}(\cdot)) d\Omega(\mathcal{P}). \quad (4.6)$$

An estimator that minimizes the Bayes risk is called a *Bayes estimator*, and the choice of loss determines which posterior summary it recovers. A common choice is the squared-error loss, which yields the posterior mean. Such estimators are rarely available in closed form, so we approximate one by restricting (4.5) to a neural network $\hat{\mathcal{P}}(\cdot; \mathbf{W})$ with weights $\mathbf{W}$, and seeking $\mathbf{W}^* = \arg \min_{\mathbf{W}} \mathcal{R}_{\Omega}(\hat{\mathcal{P}}(\cdot; \mathbf{W}))$. Because neural networks are universal function approximators, a well trained neural network will approximate the Bayes estimator with respect to $\mathcal{L}$ and Ω , hence the name *Neural Bayes Estimators*.

For many spatial models, simulating realizations of spatial data from known parameter values is fast, while estimating those parameters from realizations is expensive and difficult. Simulation-based inference exploits this asymmetry: although the Bayes risk in (4.6) often cannot be evaluated directly, it can be approximated by Monte Carlo using only cheap simulations from the statistical model. We draw $\mathcal{P}^{(k)} \sim \Omega$ and simulate $Y^{(k)} \sim f(\cdot | \mathcal{P}^{(k)})$ for $k = 1, \dots, n_{\text{train}}$, and replace the Bayes risk with the Monte Carlo estimate

$$\mathcal{R}_{\Omega}(\hat{\mathcal{P}}(\cdot; \mathbf{W})) \approx \frac{1}{n_{\text{train}}} \sum_{k=1}^{n_{\text{train}}} \mathcal{L}(\mathcal{P}^{(k)}, \hat{\mathcal{P}}(Y^{(k)}; \mathbf{W})). \quad (4.7)$$

Here, (4.7) requires no evaluation of the likelihood, only the ability to simulate. The Bayes risk is minimized over $\mathbf{W}$ by standard backpropagation.

The neural parameter estimation strategy is often referred to as an “amortized” approach. This emphasizes a rather obvious observation: while training the network is expensive, once trained, inference is fast, since it consists of forward passes through the network, and the same network can be applied to many

datasets. In settings with big spatial data, or problems where parameters need to be estimated repeatedly, this trade-off is favorable. We note, however, that the “amortized” component of the approach is often unnecessary. In fact, for sufficiently large problems, the full pipeline of generating training data, training a network, and applying it once can already be cheaper than a single maximum likelihood fit.

The choice of network architecture is dictated by the structure of the data. Vanilla, multi-layer perceptrons (MLPs) lack any built-in spatial inductive bias and are generally less effective for spatial fields. When data lies on a regular grid, the field can be treated as an image, and convolutional neural networks (CNNs) have been shown to make effective estimators [49–53]. Since it is well known that having replicates of the spatial data allows for more stable parameter estimation, replicates can be stacked as image channels in the convolutional setting. For irregular spatial data, graph neural networks (GNNs) provide an analogous inductive bias by treating the locations as nodes of a graph [54]. In the irregular setting, replicates can also be used, but can no longer be interpreted as channels of an input image.

When applied to large, non-stationary datasets, these networks were previously deployed in a *local* fashion: the spatial domain was partitioned into smaller subsections (e.g., square tiles for gridded data), and the network was applied independently to each block to produce local parameter estimates. This framework had two key drawbacks. First, it implicitly assumed local stationarity within each subsection. Second, the estimator lacked global context, missing relationships between subsections of the data that were sufficiently far apart.

The contributions in Chapters 5 and 6 move away from the local framework, using global estimators that are capable of estimating entire fields of non-stationary parameters in a single pass. A key component to the success of this work is the transformer architecture, in which the attention mechanism has been shown to learn spatial inductive biases directly from data, accommodate inputs with varying numbers of locations, and capture global dependence without partitioning large domains. Because the transformer architecture was relatively recently introduced compared to other networks, we provide a description in the following section.

4.4 The Transformer Architecture

The central advantage of the transformer—and the reason it is the main ingredient behind state-of-the-art artificial intelligence (AI) systems such as ChatGPT, Claude, and Gemini—is the *attention* mechanism [55]. More specifically, this refers to scaled dot-product attention. This mechanism is the “secret sauce” of the transformer architecture, allowing it to dynamically determine *what to pay attention to*, rather than relying on fixed inductive biases. Although this leads to a more complex architecture with significantly more parameters, it provides a substantial advantage over prior approaches like convolutional

and recurrent neural networks, which hard-code assumptions about spatial or sequential structure. For instance, convolutional neural networks impose a spatial prior by design, focusing attention on a certain number of neighboring pixels, where that number depends on the size of the convolutional kernel. While effective in many settings, this can be inadequate for tasks involving long-range dependencies where correlations can span large distances in unintuitive ways. In contrast, transformers can learn to attend to these long-range, nonlocal interactions.

4.4.1 The Math Behind a Simple Transformer

The transformer takes in an input matrix $X \in \mathbb{R}^{d_{\text{in}} \times C}$ where C represents the *context length*, and d_{in} is our input dimension. The matrix X is made by collating together a *sequence* of column vectors $\mathbf{x}^{(c)} \in \mathbb{R}^{d_{\text{in}}}$ where $c = 1, \dots, C$ which we call *tokens*. These tokens can be words (turned into vectors via some embedding) in the case of a language model, or patches of an image (unrolled into a vector) in the case of a vision transformer. In the vision case, we can take the simple example of an image with 20×20 pixels, split it into 4 patches of size 10×10 , and vectorize each patch into a vector $\mathbf{x} \in \mathbb{R}^{100}$. Thus, in this case, we have a value $d_{\text{in}} = 100$. By definition, the transformer can take in inputs of variable length, but a maximum context length C_{max} must be chosen, and is typically dictated by available compute. In our simple image example, we could make the choice to limit to the maximum size of 120×120 pixels, thus our $C_{\text{max}} = 144$ patches, and the maximum size of our input matrix is $X \in \mathbb{R}^{100 \times 144}$.

The architecture of a transformer can be dissected as follows: The input matrix X is multiplied by a learned embedding matrix to conform to some chosen model dimensions, is passed through a series of *transformer blocks*, and then is de-embedded so that the output can then be converted into the task-appropriate form (words, images, class labels, etc). We can denote the output of the model as $\hat{Y} \in \mathbb{R}^{d_{\text{out}} \times C}$, and we train the model using some loss function which compares the output to the truth $Y \in \mathbb{R}^{d_{\text{out}} \times C}$. Each transformer block consists of multi-headed attention (parallel operations containing the *attention mechanism*), followed by a dense multi-layer perceptron (MLP). In practice, one also sprinkles in normalizations (LayerNorm), dropout, and many other engineering tweaks. None of those extra engineering components change the core equations we present next, so we omit them for notational and expositional clarity:

Input: The model receives a matrix (sequence) $X \in \mathbb{R}^{d_{\text{in}} \times C}$.

Embedding: The input sequence $X \in \mathbb{R}^{d_{\text{in}} \times C}$ is first transformed to the model (or *hidden*) dimension of d_m via a single learned linear map

$$H^{(0)} = W_e X + P_e, \quad W_e \in \mathbb{R}^{d_m \times d_{\text{in}}}, \quad P_e \in \mathbb{R}^{d_m \times C}. \quad (4.8)$$

where W_e is a weight matrix and P_e is a matrix whose columns are positional encoding vectors, that indicate the position of the token in the sequence. There are many different methods for choosing these positional encodings, such as just setting them to be fixed sinusoidal vectors, or allowing them to be learned through the network.

Transformer Blocks: The resulting $H^{(0)} \in \mathbb{R}^{d_m \times C}$ serves as the input to the first transformer block. In practice we have L transformer blocks, where $\ell = 1, \dots, L$, with the matrix $H^{(\ell-1)} \in \mathbb{R}^{d_m \times C}$ serving as the input to the ℓ -th block. The intermediate matrices $H^{(\ell)}$ in between transformer blocks are often called the *residual stream*, with $\mathbb{R}^{d_m \times C}$ being the dimension of this residual stream. For the incoming matrix $H^{(0)} \in \mathbb{R}^{d_m \times C}$, we apply the two sub layers below.

1. *Multi-headed Self-Attention (MHSA).* We have multiple attention *heads*, through which the $H^{(0)}$ is passed through in parallel. For each head $i = 1, \dots, h$, we learn three weight matrices with a head dimension $\mathbb{R}^{d_h \times d_m}$, where $d_h = d_m/h$. The three weight matrices are:

$$W_Q^i, W_K^i, W_V^i \in \mathbb{R}^{d_h \times d_m}. \quad (4.9)$$

Given the incoming matrix $H^{(0)} \in \mathbb{R}^{d_m \times C}$ we form

$$Q_i = W_Q^i H^{(0)}, \quad K_i = W_K^i H^{(0)}, \quad V_i = W_V^i H^{(0)}, \quad Q_i, K_i, V_i \in \mathbb{R}^{d_h \times C}. \quad (4.10)$$

The scaled dot-product self-attention for head i is

$$A_i = V_i S_i, \quad S_i = \text{softmax}\left(\frac{K_i^T Q_i}{\sqrt{d_h}}\right)^T \in \mathbb{R}^{C \times C}, \quad A_i \in \mathbb{R}^{d_h \times C} \quad (4.11)$$

The matrix S_i represents how much each token attends to every other token, based on their learned query (Q) and key (K) representations. The softmax function ensures these attention weights are positive and sum to one across each column, effectively forming a probability distribution over the tokens being attended to. The output $A_i = V_i S_i$ is then the corresponding mixture of value vectors. Each head will learn a different matrix S_i , which can be interpreted as each head learning a different concept that the model is paying attention to.

We concatenate the outputs from all h heads along the row dimension to form a matrix

$\mathcal{A} = [A_1; A_2; \dots; A_h] \in \mathbb{R}^{d_m \times C}$ where $[\cdot; \cdot]$ denotes concatenation along the row (feature) dimension.

We then multiply by a learned weight matrix $W_a \in \mathbb{R}^{d_m \times d_m}$:

$$H_{\text{attn}}^{(0)} = W_a \mathcal{A} \in \mathbb{R}^{d_m \times C} \quad (4.12)$$

This is done to mix the information from all the attention heads, that were parallel and separately computed. The final step is the application of a residual connection, in which we add the original

input $H^{(0)}$ to the output of the MHSA:

$$H_{\text{res}}^{(0)} = H^{(0)} + H_{\text{attn}}^{(0)}. \quad (4.13)$$

2. Dense MLP The output of the attention section is a matrix $H_{\text{res}}^{(0)} \in \mathbb{R}^{d_m \times C}$, which now serves as the input to a position-wise, feed-forward MLP which independently applies a non-linear transformation to each token. We set a feed forward dimension of d_{ff} , which is commonly chosen to be $d_{\text{ff}} = 4d_m$ as a standard rule-of-thumb. The MLP then consists of two learned linear projections with a nonlinearity in between:

$$W_1 \in \mathbb{R}^{d_{\text{ff}} \times d_m}, \quad W_2 \in \mathbb{R}^{d_m \times d_{\text{ff}}},$$

and we write

$$H_{\text{mlp}}^{(0)} = W_2 \sigma(W_1 H_{\text{res}}^{(0)}) \in \mathbb{R}^{d_m \times C}, \quad (4.14)$$

where $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is a non-linear activation function that is applied element-wise, which is typically chosen to be GELU [56] or ReLU.

We again apply a residual connection:

$$H^{(1)} = H_{\text{res}}^{(0)} + H_{\text{mlp}}^{(0)} \in \mathbb{R}^{d_m \times C}. \quad (4.15)$$

Thus, the output of our first transformer block is $H^{(1)}$. We repeat this process for L transformer blocks (MHSA + MLP), resulting in a final matrix $H^{(L)} \in \mathbb{R}^{d_m \times C}$.

De-Embedding: The final matrix $H^{(L)} \in \mathbb{R}^{d_m \times C}$ is passed through a learned output projection to produce the final model output:

$$\hat{Y} = W_{\text{out}} H^{(L)}, \quad W_{\text{out}} \in \mathbb{R}^{d_{\text{out}} \times d_m}. \quad (4.16)$$

The dimension d_{out} is task-dependent. For instance, in image-to-image tasks, we may set $d_{\text{out}} = d_{\text{in}}$ to reconstruct spatial features with the same structure as the input. In contrast, for classification tasks, the sequence is typically pooled (e.g., via average pooling or selection of a “class” token), and the output projection maps to a vector of class logits in $\mathbb{R}^K$, where K is the number of categories.

The model output $\hat{Y}$ is compared to a ground truth target Y using a task-specific loss function, and all learnable parameters are updated via backpropagation using an optimizer such as AdamW [57].

CHAPTER 5

LATTICEVISION: IMAGE TO IMAGE NETWORKS FOR MODELING NON-STATIONARY SPATIAL DATA

Modified from a paper published in *PMLR in the Proceedings of AISTATS* ¹.

Antony Sikorski^{2,3}, Michael Ivanitsky ², Nathan Lenssen ^{4,5}, Douglas Nychka⁶, Daniel McKenzie⁷

5.1 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.

5.2 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 [18, 19]. 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 [42, 43, 49, 50]. Like MLE, these methods are limited to dividing large fields into smaller sections, each assumed to be stationary, and estimating parameters independently

¹Reprinted with permission of PMLR, currently available at <https://openreview.net/forum?id=km8KA33TUu>. The final link to the published work in PMLR will be available at <https://github.com/antonyxsik/LatticeVision>

²Graduate student at the Colorado School of Mines

³Author for correspondence

⁴Teaching Assistant Professor at Colorado School of Mines

⁵Scientist at NSF National Center for Atmospheric Research

⁶Professor at Colorado School of Mines

⁷Assistant Professor at Colorado School of Mines

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.

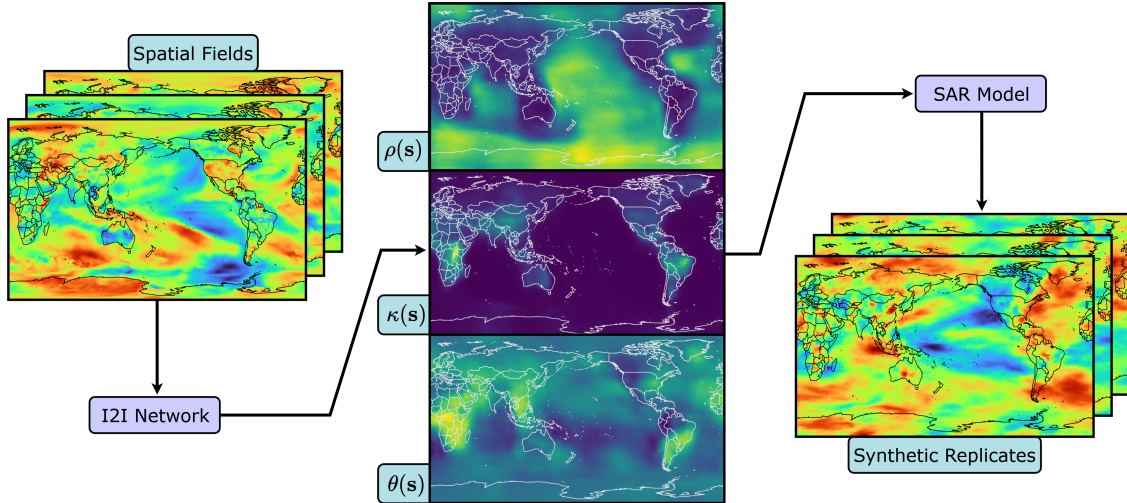


Figure 5.1 An illustration of the LatticeVision workflow applied to ESM outputs. Spatial fields are fed into an I2I network, which in turn produces estimates of the non-stationary parameter fields. These are encoded into a SAR model from which synthetic replicates are efficiently simulated.

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 [58] and vision transformer based [59, 60]—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 [61], 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 [62–64] 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 [54] or by pixel [41]. 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) (see Figure 5.1). 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 [65, 66]; too few for many applications [67–70]. Following parameter estimation, we generate realistic ensembles containing thousands of fields in a matter of seconds using the LatticeKrig package [1]; 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 [1, 3], and validate this framework on ESM outputs.

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

5.3 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 [71–74]. Despite recent advances in generative deep learning for spatial data [75–77], statistical models outperform other approaches when data is limited [78]. Moreover, statistical models contain interpretable parameters, which are useful for downstream applications.

We consider statistical models built around Gaussian processes (GPs). With locations $\mathbf{s} \in \mathbb{R}^2$, a mean-zero GP $g(\mathbf{s})$ is fully specified by its covariance function $C(\mathbf{s}, \mathbf{s}') = \mathbb{E}[g(\mathbf{s})g(\mathbf{s}')]$. The kernel C must be positive definite and this requirement is typically enforced by assuming *stationarity* and *isotropy* [16, 79]. That is, C is independent of the location in the domain (stationarity), and depends only on the separation distance (isotropy):

$$C(\mathbf{s}, \mathbf{s}') = \sigma^2 \mathcal{C}(\kappa_{\mathcal{M}} \|\mathbf{s} - \mathbf{s}'\|). \quad (5.1)$$

Here $\mathcal{C}(0) = 1$, σ^2 is the variance of the GP, and $\kappa_{\mathcal{M}} > 0$ governs the spatial correlation range. A popular choice for $\mathcal{C}$ is the Matérn family⁸ 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 $\Sigma_{ij} = C(\mathbf{s}_i, \mathbf{s}_j)$ [18].

5.3.1 The SPDE Method

GPs from the Matérn family have an equivalent representation in terms of a stochastic partial differential equation (SPDE) [15, 80]:

$$(\kappa^2 - \Delta)^{(\nu+1)/2} g(\mathbf{s}) = \mathcal{W}(\mathbf{s}), \quad (5.2)$$

where κ^2 controls the correlation range, and is similar but not identical to $\kappa_{\mathcal{M}}$ 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 [14, 81] connects Matérn GPs to Gaussian Markov Random Fields (GMRFs) by demonstrating that discretizing Equation (5.2) on a regular grid yields a GMRF which approximates the Matérn GP well. This approach can be extended⁹ 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:

$$(\kappa^2(\mathbf{s}) - \nabla \cdot D(\mathbf{s}) \nabla) g(\mathbf{s}) = \mathcal{W}(\mathbf{s}). \quad (5.3)$$

Following [82], we construct $D(\mathbf{s})$ via its eigendecomposition: $D(\mathbf{s}) = R(\mathbf{s})^\top \Lambda(\mathbf{s}) R(\mathbf{s})$ where

$$R(\mathbf{s}) = \begin{bmatrix} \cos \theta(\mathbf{s}) & -\sin \theta(\mathbf{s}) \\ \sin \theta(\mathbf{s}) & \cos \theta(\mathbf{s}) \end{bmatrix}, \quad \Lambda(\mathbf{s}) = \begin{bmatrix} \sqrt{\rho(\mathbf{s})} & 0 \\ 0 & \frac{1}{\sqrt{\rho(\mathbf{s})}} \end{bmatrix} \quad (5.4)$$

The generalized Laplacian in (5.3) can be written as

$$\nabla \cdot D(\mathbf{s}) \nabla \equiv D_{1,1}(\mathbf{s}) \frac{\partial^2}{\partial x^2} + 2D_{2,1}(\mathbf{s}) \frac{\partial^2}{\partial x \partial y} + D_{2,2}(\mathbf{s}) \frac{\partial^2}{\partial y^2}. \quad (5.5)$$

5.3.2 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 (5.3), 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 5.2.

⁸In reviewing commonly used covariance kernels, [11] concludes: ‘Use the Matérn’.

⁹For simplicity, we shall henceforth focus on the $\nu = 1$ case, the so-called Whittle covariance [15]

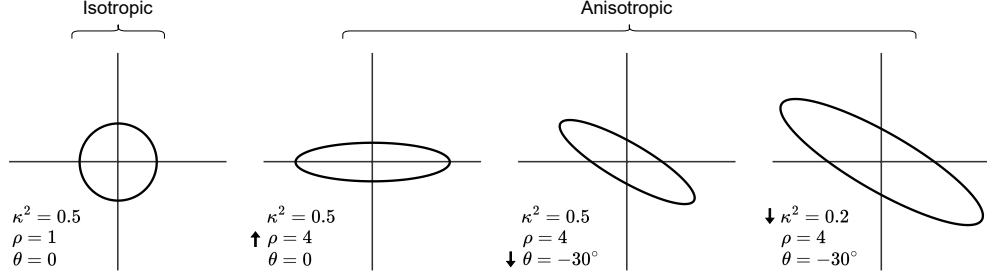


Figure 5.2 Illustration of the effects of κ^2 , ρ , and θ . The ellipses represent contours of constant correlation, e.g. all locations with correlation 0.5 with the origin. κ^2 controls the radii of the ellipse, ρ controls the ratio of the semi-major and semi-minor radii (i.e., the ‘aspect ratio’ of the ellipse), and θ is the angle the semi-major axis makes with the positive x -axis.

5.3.3 Discretizing the SPDE

To obtain a computable model from (5.3), one approximates this SPDE using either a finite element or finite difference method. Following [41]¹⁰ we use the finite difference method on a regular grid, yielding the stencil:

$$\begin{array}{c|c|c} \frac{D_{1,2}(\mathbf{s})}{2} & -D_{2,2}(\mathbf{s}) & \frac{-D_{1,2}(\mathbf{s})}{2} \\ \hline -D_{1,1}(\mathbf{s}) & \kappa^2(\mathbf{s}) + 2D_{1,1}(\mathbf{s}) + 2D_{2,2}(\mathbf{s}) & -D_{1,1}(\mathbf{s}) \\ \hline \frac{-D_{1,2}(\mathbf{s})}{2} & -D_{2,2}(\mathbf{s}) & \frac{D_{1,2}(\mathbf{s})}{2} \end{array} \quad (5.6)$$

Let $\mathbf{y} \in \mathbb{R}^n$ denote a discretized solution to (5.3) 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 (5.6). [14] show that $\mathbf{y}$ approximates a sample from the Matérn GP associated to (5.3). As B is sparse and structured, 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 $Q = B^\top B$.

5.4 Non-Stationary Data Generation

5.4.1 I2I Data

In order for the I2I networks to be successful, the training data needs to encode priors appropriate for geoscientific applications. Previous work [41] 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

¹⁰We correct a minor error in their derivation. Our derivation can be found in Appendix B.1.3

same parameter fields. These parameter fields are concatenated along the channel dimension to form a three channel, ground truth “image” $\mathcal{P} \in \mathbb{R}^{3 \times H \times W}$, where $\mathcal{P}(\mathbf{s}) = [\kappa^2(\mathbf{s}), \rho(\mathbf{s}), \theta(\mathbf{s})]^T$. 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 $(Y, \mathcal{P})$ in our data.

Each time we construct a single parameter field, we first sample one of eight spatial patterns $p^{(i)}(\mathbf{s}; \boldsymbol{\Omega}^{(i)})$, $i = 1, 2, \dots, 8$. 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 $\boldsymbol{\Omega}^{(i)}$, 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 5.3, with detailed functional forms and hyperparameter priors provided in Appendix B.2.

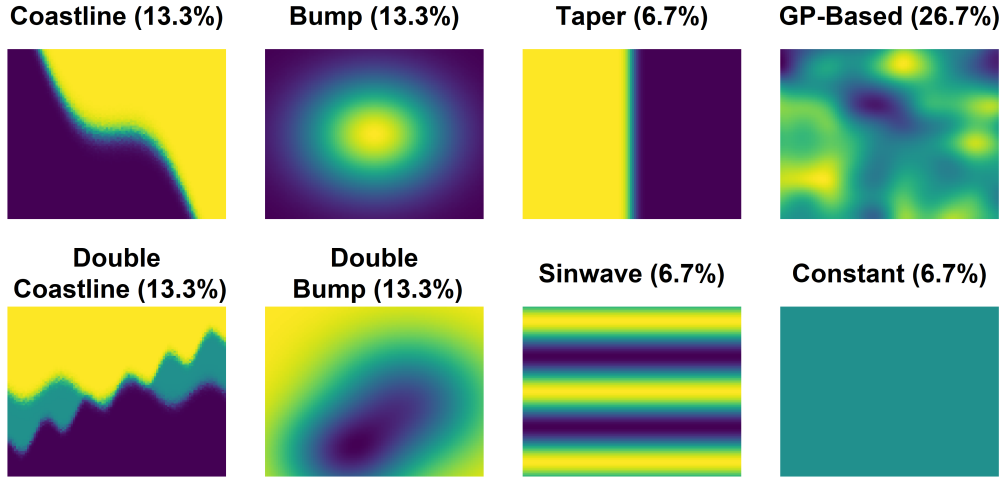


Figure 5.3 Spatial patterns and their frequencies.

Once a pattern $p^{(i)}$ and its hyperparameters $\boldsymbol{\Omega}^{(i)}$ have 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 $\theta \sim \mathcal{U}(-\frac{\pi}{2}, \frac{\pi}{2})$, 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 5.3. 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.

5.4.2 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 $\xi = [\kappa^2, \rho, \theta]^T \in \mathbb{R}^3$. Thus, our input-output pairs are $(\mathcal{Y}, \xi)$.

5.5 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 [58] with a symmetric encoder-bottleneck-decoder structure and skip connections that propagate spatial information across resolutions. Our implementation replaces ReLU with GELU activations [56], employs group normalization [83], and omits dropout [84]. The number of channels in the bottleneck matches the transformer embedding dimension used in the transformer-based I2I networks.

ViT: The original ViT [59] 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 5.6.

STUN: Our hybrid architecture, a spatial TransUNet (STUN), is based upon a network originally designed for image segmentation [60]. 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 [41, 44, 49, 50, 52]. The key difference is our networks have more parameters (1-2.5M), as compared to typical local estimators (50-700k).

5.5.1 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 B.3.

5.6 Simulated Data Experiments

We generate a test set of 160 samples following the procedure in Section 5.4. Each sample is a small ensemble $Y \in \mathbb{R}^{M \times 192 \times 288}$ of M replicates and an associated parameter image $\mathcal{P} \in \mathbb{R}^{3 \times 192 \times 288}$. For the I2I networks (UNet, ViT, STUN) we simply forward-propagate Y to obtain $\hat{\mathcal{P}}_{\text{I2I}}$. 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 $\hat{\mathcal{P}}_{\text{CNN}}$. We use reflection padding—improving upon prior works which use zero padding—thus reducing edge-artifacts. Figure 5.4 contrasts an example $\mathcal{P}$ 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 [41].

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 [85], learned 2-D [59], and rotary (RoPE) [86]. RoPE yields the best performance across the majority of metrics for both ViT

and STUN, and is adopted henceforth (see Appendix Table B.1).

Number of Replicates: We observe the effect on parameter estimation performance with varying numbers of replicates $M = \{1, 5, 15, 30\}$. Table 5.1 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 Table B.2 and Table B.3.

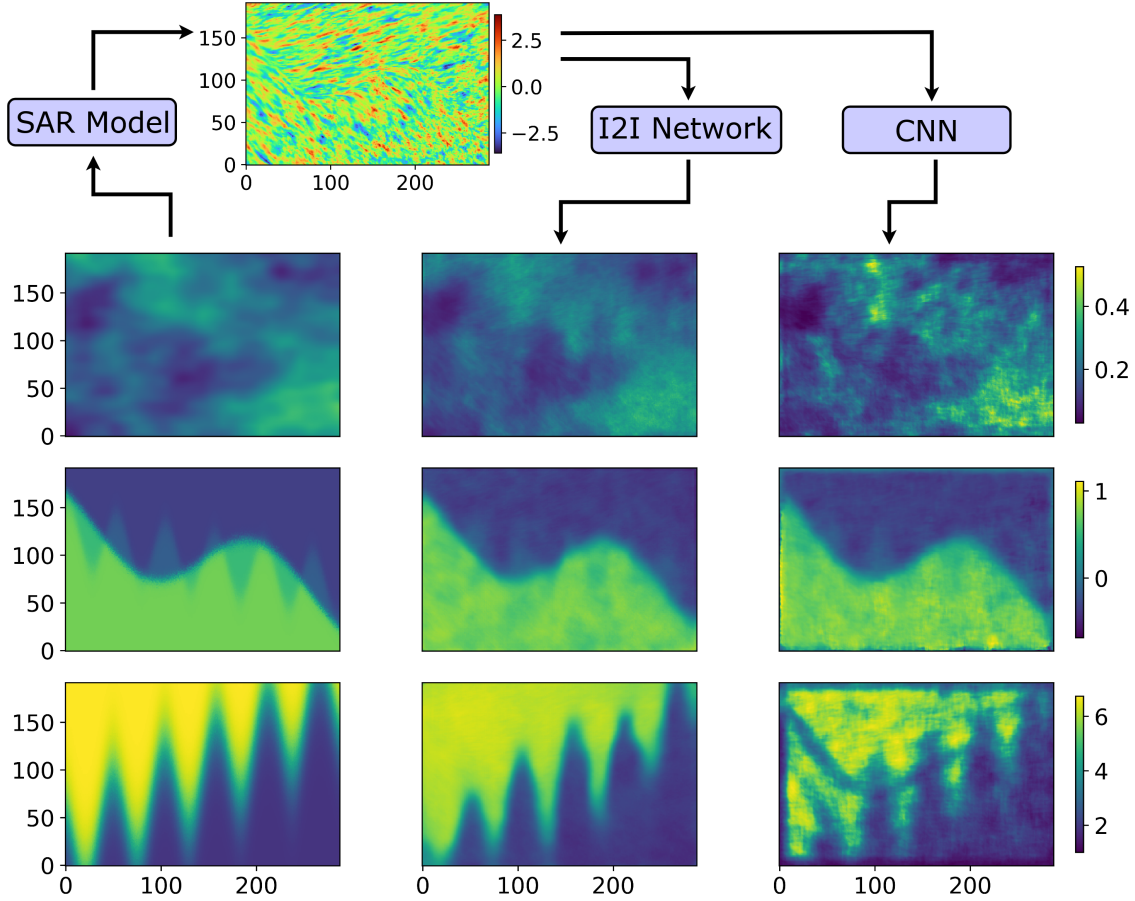


Figure 5.4 True parameters $\mathcal{P}$ (left) are encoded into the LatticeKrig SAR model to simulate a testing sample Y , of which one replicate $\mathbf{y}^{(0)}$ is displayed (top). Y is used as an input to STUN and a sliding window, local estimation strategy using CNN25, resulting in $\hat{\mathcal{P}}_{\text{STUN}}$ (middle), and $\hat{\mathcal{P}}_{\text{CNN25}}$ (right).

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 [59]. (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.

Table 5.1 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

5.7 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, [87]), CESM1 (30, 288×192 , [65]), and EC-Earth3 (72, 512×256 , [88]). While diffusion networks have recently shown promise in adjacent emulation applications [89, 90], 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 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.

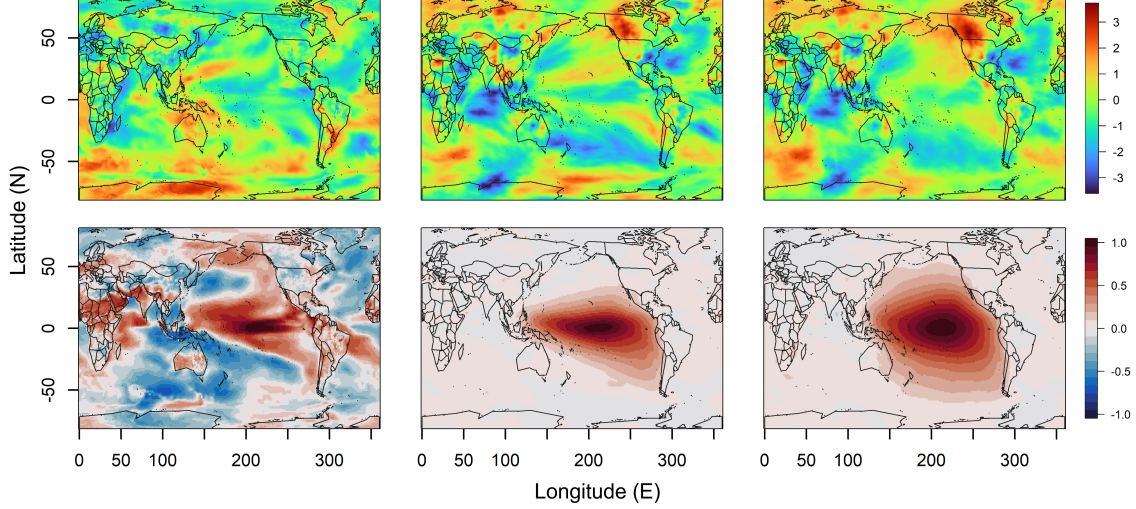


Figure 5.5 *Top*: Standardized temperature fields drawn from the CESM1 ensemble (*left*), the STUN-based emulator (*middle*), and the CNN25-based emulator (*right*). *Bottom*: Correlations with a chosen location in the Niño 3.4 region at (212°E, 1°N) for the same three ensembles. The STUN-based emulator better preserves spatial relationships, including the zonal correlation structure along the equator and the meridional oceanic correlation range.

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 B.4).

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 [91]—for the STUN-based and CNN25-based emulators (Figure 5.5). 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.

5.8 Related Works

5.8.1 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 [42, 43]. Neural parameter estimation combines ideas from simulation-based inference [47, 48] and learning to optimize (L2O) [45, 46] by training networks to identify maxima of the Bayes risk [44]. The training cost is then amortized by repeated use. Until now, these approaches have been limited to local estimation [44, 49–53, 92], 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.

5.8.2 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 [70]. Despite recent advances in deterministic [93–99] and ensemble-based probabilistic [76, 77, 100, 101] weather forecasting, comparatively fewer methods focus explicitly on long-term climate projections [102, 103]. Our approach draws ideas from statistical methods, which explicitly link parameter estimation with emulation [41, 104–107], and machine learning (ML) methods, both deterministic [62, 108, 109] and probabilistic [75, 89] 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 [64, 110] and can exhibit limited generalization capabilities beyond their training distributions [63]. 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.

5.9 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. The STUN and UNet architectures assume that each side of the grid is divisible by 2^L , with L being the number of convolutional layers in the encoder of the network. In our implementation, this means each side of the grid must be divisible by 16. Replicate count is also fixed at training time, with only a soft constraint to enforce

permutation invariance. A hard constraint via aggregation [111] 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 [112] 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 [113]. Our network choices establish a baseline, but there exist many additional architectures for future work to explore [114–118].

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 [1, 21, 72], 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.

5.10 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.

5.11 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.

CHAPTER 6
A NON-STATIONARY, AMORTIZED, TRANSFER LEARNING APPROACH FOR MODELING
ITALIAN AIR QUALITY

Modified from a paper submitted to *Spatial Statistics* ¹.

Alessandro Fusta Moro^{2,3,4}, Antony Sikorski^{4,5}, Daniel McKenzie⁶, Alessandro Fassò⁷, Douglas Nychka⁸

6.1 Abstract

Air quality monitoring in Italy relies on sparse, irregular, ground-based stations that provide high-quality but incomplete measurements of pollution. Chemical transport models (CTMs) offer full spatial and temporal coverage but smooth over local variability. We develop a spatial transfer-learning framework that integrates these two data sources to produce daily, fine-grid predictions of nitrogen dioxide (NO₂) concentrations across Italy for 2023, with uncertainty quantification. The resulting maps provide a resource for decision making in downstream applications such as epidemiology and environmental policy.

Our approach builds on the geostatistical LatticeKrig framework, which uses compactly supported basis functions and coefficients governed by a sparse precision matrix. We learn a nonstationary, anisotropic correlation structure from the gridded CTM outputs using an image-to-image neural architecture that estimates millions of spatially varying parameters in a matter of seconds. The basis-function representation enables this covariance structure to be transferred to the point-level station data and projected onto a finer prediction grid, a key extension for handling the change of support between data sources. A likelihood-based refinement step then adjusts the correlation range to recover fine-scale variability smoothed out by the gridded data. The proposed methodology results in a flexible, non-stationary, and anisotropic representation of the spatial process, better accommodating the complex geography of Italy. Performance is assessed through experiments on both gridded CTM outputs and point-level station measurements, demonstrating improvements over the stationary formulation.

6.2 Introduction

Air quality in Italy has been a persistent scientific and political concern due to the high concentrations of atmospheric pollutants regularly detected by Italian and European pollution monitoring systems.

¹Currently on Arxiv [5]

²Graduate Student at University of Bergamo

³Author for correspondence

⁴Co-first author, equal contribution

⁵Graduate student at the Colorado School of Mines

⁶Assistant Professor at Colorado School of Mines

⁷Professor at University of Bergamo

⁸Professor at Colorado School of Mines

Northern Italy is particularly affected: high anthropogenic presence results in emissions from traffic, industrial activity, and livestock farming, which are subsequently trapped by the surrounding mountain ranges [119, 120]. This has prompted the European Union to open repeated infringement proceedings against Italy for not meeting air quality standards. Beyond regulatory considerations, a deeper understanding of air pollutant concentrations is critical to furthering studies in public health [121, 122], COVID-19 [123, 124], and urban planning [125]. Among regulated pollutants, nitrogen dioxide (NO_2) is considered to be one of the most hazardous. Short-term exposure to high concentrations is associated with lung inflammation and airway irritation, while long-term exposure has been linked to chronic respiratory illnesses [126]. The serious consequences of the pollutant motivate the improvement of methods for characterizing the spatiotemporal distribution of its concentration.

Networks of ground-based monitoring stations provide high-quality, point-level measurements of air pollutant concentrations and are widely regarded as reliable observational data. However, these networks are inherently biased and irregular in space, with station locations often concentrated in areas with high population density. Coverage is often limited in rural regions and complex terrain, and the set of active stations varies over time due to maintenance or operational constraints. Numerical chemical transport models (CTMs), such as the Community Multiscale Air Quality (CMAQ) model developed by the U.S. Environmental Protection Agency [127], and reanalysis products that combine ensembles of CTMs with observational data through data assimilation, such as the European air quality reanalysis produced by the Copernicus Atmosphere Monitoring Service [128] (CAMS), can provide a large-scale view of air pollution dynamics. These models simulate atmospheric transport and chemical processes using emissions inventories and meteorological inputs, producing spatially and temporally complete pollutant concentrations on regular grids. Although CTM outputs offer uniform spatial coverage and capture large-scale spatiotemporal dynamics, their relatively coarse resolution can lead to bias or over-smoothing at local scales.

In this work, we develop a scalable, transfer-learning framework that allows us to integrate the gridded CTM outputs with irregular, point-level monitoring data. The CTM outputs are used to inform a prior covariance that is subsequently adjusted when fitting the spatial model to point-level data. Our ultimate objective is to produce accurate, daily estimates of NO_2 concentrations on a fine grid over all of Italy, for the year 2023. Achieving this goal introduces three methodological challenges. First, the spatial domain and temporal frequency imply very large datasets that lead to computational bottlenecks for classical Gaussian process methods. Second, Italy’s geography includes coastlines, mountain ranges, and heterogeneous emission regions, motivating a flexible, non-stationary, and anisotropic model. Third, we must explicitly accommodate the change of support between gridded CTM output and point-level stations in a way that preserves interpretability and allows prediction on a target fine grid.

We choose to model the NO_2 concentrations using the LatticeKrig framework [1, 3], a basis function method that provides an efficient Gaussian process approximation. The framework uses sparse linear algebra in key matrices to enable working with large spatial data, and the use of basis functions allows for interpolation and prediction across different spatial supports. Additionally, the LatticeKrig model allows for highly flexible, nonstationary, and anisotropic covariance function through the direct specification of a sparse precision matrix. Although this flexibility is attractive, it introduces a substantial inference challenge: the need to estimate a large number of spatially varying covariance parameters. As maximum likelihood estimation (MLE) is computationally prohibitive at the scale considered here, we adopt the LatticeVision framework [129], which uses an amortized, neural, image-to-image (I2I) estimation approach.

The change of support problem is explicitly addressed by two key extensions. First, we extend amortized parameter estimation [43] to the change-of-support setting by incorporating the basis-function representation of the original LatticeKrig formulation into the LatticeVision parameter estimation pipeline. This extension is essential for change-of-support problems, as the basis functions allow for the transfer of information across different spatial resolutions and supports. In our application, we transfer information between CTM fields and point-level station data, as well as predict onto fine grids. Second, we introduce a likelihood-based refinement that adjusts the covariance structure when transitioning between data sources. In our setting, the CTM outputs capture large-scale dependence but tend to under-represent local variability. We therefore treat CTM-derived anisotropy parameters as informative at regional scales and adjust the local correlation range using station data, combining amortized neural estimation for large, regular grids with likelihood-based refinement for smaller, irregular spatial data.

The remainder of the paper is organized as follows. Section 6.3 reviews related literature on change-of-support modeling, spatial data fusion, scalable Gaussian processes, and neural parameter estimation. Section 6.4 describes the station, CTM, and auxiliary covariate datasets. Section 6.5 introduces our statistical model and the nonstationary LatticeKrig formulation, along with the neural parameter estimation approach. Section 6.6 presents CTM-based parameter estimation and associated experiments. Section 6.7 applies the proposed method to the station data, including cross-validation and fine-grid prediction. Section 6.8 concludes with a discussion and directions for future work. All of the supplementary code and data used for the experiments in this paper are publicly available at github.com/afustamo/ItalyAQ-amortized-transfer-learning.

6.3 Related Works

The need to integrate spatial data sources that do not share the same spatial support is often referred to as a change-of-support problem (COSP; [130]). The transition from point-level data to gridded areal

data, and vice versa, was initially addressed through the use of block kriging [16, 131]. Subsequent data fusion approaches interpreted measurements collected over different spatial supports as distinct observations of the same latent process. In the context of air pollution, [132] expanded upon this approach with a Bayesian framework, estimating the true process from station measurements and CTM outputs. More recently, a closely related approach was adopted by [133], who investigated NO_2 concentrations in Italy during the COVID-19 pandemic. However, Bayesian techniques often involve substantial computational costs, mainly due to complex posterior distribution computation. A second line of methodology aimed at integrating information collected over different spatial supports relies on regression-based approaches. Here the response variable is represented by the most reliable measurements (i.e., ground monitoring stations), and covariates consist of additional information sources such as climate variables and CTM outputs. These covariates are often characterized by higher uncertainty but broader spatial coverage. These methods are commonly referred to as calibration approaches [e.g., 134–137]. To account for local discrepancies between ground-based measurements and gridded data, spatially varying regression coefficients have been introduced [134–136].

More recently, other approaches have emerged that incorporate machine learning techniques to predict ground-level concentrations from CTM outputs or satellite observations. These include tree-based methods [138–140] and neural networks [141]. Although satellite observations have been shown to improve performance in some settings, their overpass times for Italy typically occur during periods of low traffic and correspond to daily minima of NO_2 concentrations, making them less representative of daily average pollution patterns. Our approach is most closely related to the transfer-learning framework of Gong et al. [142], in which gridded CTM outputs inform a prior covariance that is subsequently adjusted when fitting the model to point-level data.

Beyond change-of-support considerations, the spatial domain and temporal frequency in modern environmental problems often comprise large datasets that lead to computational bottlenecks for classical Gaussian process methods [18, 19]. A substantial literature addresses scalability through approximations and sparse representations, including Vecchia-type factorizations [143, 144], covariance tapering [12, 145], spectral representations [146], low-rank methods such as fixed-rank kriging [147], SPDE/GMRF formulations [14, 148], and lattice-based models [1]. Heaton et al. [71] assessed many of these techniques in a competition, where the application of the models to the same case study enabled a fair comparison among them.

Big environmental data often exhibit nonstationarity due to heterogeneous terrain and complex meteorological drivers. Modeling this structure requires estimating many spatially varying parameters, rendering classical maximum likelihood estimation computationally infeasible at scale. Neural parameter

estimation has recently emerged as a powerful alternative to MLE, where the cost of training the neural network is amortized by its repeated use [42, 43]. Initial approaches for large domains relied on local neural estimators, which used simple dense or convolutional networks to infer parameters at each location or subsection of the domain independently, effectively assuming local stationarity [e.g., 44, 49, 50, 52, 53, 92]. Recent work has leveraged the observation that in regularly gridded settings, both ensembles of spatial fields and their corresponding spatially varying parameter “fields” can be represented as images with multiple channels [129, 149]. The LatticeVision framework uses this representation to move beyond local estimation and instead use modern I2I (image-to-image) architectures such as vision transformers to estimate thousands of spatially varying parameters across the entire domain in a single forward pass. Our work builds most directly on this line of research, extending it to explicitly handle change-of-support settings.

6.4 Data

6.4.1 Air quality data

We use two complementary sources of NO_2 concentration data: observations from air pollution monitoring stations and outputs from chemical transport models. The station data are obtained from the AQCLIM dataset [150], an open-access resource which contains daily measurements of air pollutant concentrations across Italy in the years 2013–2023. The AQCLIM dataset is derived from observations provided by the European Environmental Agency (EEA) air quality monitoring network and processed into a cleaned, harmonized format suitable for statistical analysis. In this study, we focus on daily mean NO_2 concentrations for the year 2023. An example daily realization (April 13, 2023) is shown in Figure 6.1. It is important to note that the monitoring network is not fixed over time, as the set of operating stations may vary slightly across days.

The second source of NO_2 concentrations is derived from an ensemble of data assimilation systems that rely on Chemical Transport Models (CTMs). CTMs combine emission data with external factors such as wind, temperature, and humidity to simulate the behaviour of air pollutants in the atmosphere. Observations from satellites and monitoring networks are also assimilated into these systems as additional information. We refer to this source as “CTM data” throughout the remainder of the paper. The CTM data are sourced from the Copernicus Atmosphere Monitoring Service [128] (CAMS), providing hourly average concentrations of various pollutants on a complete, uniform $0.1^\circ \times 0.1^\circ$ grid across the European continent. For this study, we compute daily mean NO_2 concentrations for the year 2023 over the spatial domain covering Italy, spanning longitudes $6.1^\circ - 18.8^\circ\text{E}$ and latitudes $35.2^\circ - 47.9^\circ\text{N}$. An example concentration field for April 13, 2023 is shown in Figure 6.1. In addition to NO_2 concentrations, we also

use a variety of covariates which are described in the following section.

6.4.2 Additional covariate data

The chosen meteorological variables represent external drivers of atmospheric transport and dispersion, and are commonly used in air-quality modeling studies [e.g., 120, 151]. We obtain these covariates from the ERA5 family of reanalysis datasets [152], which are generated by the European Centre for Medium-Range Weather Forecasts (ECMWF). We use ERA5-Land [153] as our primary data source, which provides hourly land-surface variables at a spatial resolution of $0.1^\circ \times 0.1^\circ$ and is produced by downscaling the original ERA5 fields. In regions not covered by ERA5-Land, such as coastal areas and small islands, we fill the gaps with ERA5, which provides hourly atmospheric and land-surface variables at a $0.25^\circ \times 0.25^\circ$ resolution. From these data products, we find the daily averages for temperature, relative humidity, surface solar radiation, wind speed, and boundary layer height. An example temperature field for April 13, 2023 is shown in Figure 6.1.

Air pollutant concentrations are largely driven by emissions from ground-based sources. To capture such information, we use the CAMS-REG-ANT dataset [154], a gridded anthropogenic emission inventory for Europe based on national emission reports. National totals are spatially distributed onto a $0.05^\circ \times 0.1^\circ$ grid using auxiliary information such as facility locations and land-surface typology. We combine the annual gridded emissions with detailed temporal profiles from the CAMS-REG-TEMPO dataset [155], which provides sector-specific temporal profiles at monthly, weekly, daily, and hourly scales. Combining these two datasets enables the extension of annual totals to daily estimates. We select daily values of NO_x emissions ($\text{NO} + \text{NO}_2$) as a key covariate, as primary emissions from ground-based sources are dominated by NO , which is converted to NO_2 in the atmosphere due to the presence of volatile organic compounds. An example emission field for April 13, 2023 is shown in Figure 6.1.

Atmospheric NO_2 concentration is closely related to elevation. Low elevation, enclosed regions such as the Po Valley are prone to pollutant accumulation, while high elevations tend to exhibit lower concentrations. We obtain an elevation covariate from the Copernicus Digital Elevation Model (DEM), originally available at 10m^2 , and aggregate it by averaging to the same spatial resolution of the final prediction grid (i.e., $0.05^\circ \times 0.05^\circ$). Together, these seven covariates characterize known, large-scale drivers of NO_2 concentrations, and will be incorporated into the linear mean component of our spatial modeling scheme.

6.5 Methods

In this section, we develop a statistical framework for data fusion in which the objective is to model an underlying spatio-temporal latent process. We observe this process through two distinct data sources:

sparse, irregular point-level measurements from monitoring stations, and spatially complete numerical model outputs defined on a coarse regular grid. Both are treated as noisy observations of the same latent field, but differ in spatial support and measurement error structure. Our approach follows a transfer-learning strategy: first, we learn the large-scale, nonstationary, and anisotropic spatial correlation structure from the gridded data using an image-to-image (I2I), neural estimator. We then transfer this structure to the geostatistical model when fitting to the point-level observations, allowing for an additional refinement to account for the change of support problem.

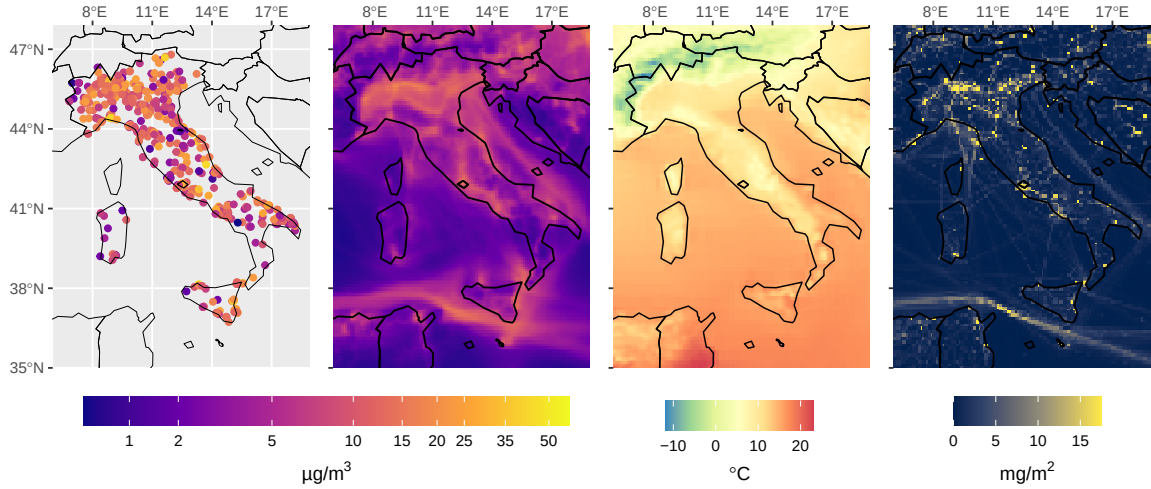


Figure 6.1 From the left: NO_2 concentrations from ground monitoring stations, NO_2 concentrations from chemical transport models, temperature from reanalysis (ERA5 datasets), NO_x emission fluxes from CAMS-REG-ANT. Data for Italy, on 13th April 2023.

6.5.1 The Latent and Observed Processes

Consider a real-valued spatio-temporal latent process defined over a continuous spatial domain and discrete time (i.e., $\{y_t(\mathbf{s}) : \mathbf{s} \in \mathcal{S} \subset \mathbb{R}^2, t \in \mathcal{T} \subset \mathbb{N}^+\}$). For a spatial location $\mathbf{s} \in \mathcal{S}$ and day $t \in \mathcal{T}$, $y_t(\mathbf{s})$ follows an autoregressive model where

$$y_t(\mathbf{s}) = \mu_t(\mathbf{s}) + \alpha_t y_{t-1}(\mathbf{s}) + g_t(\mathbf{s}), \quad (6.1)$$

where $\mu_t(\mathbf{s})$ represents the exogenous fixed effects, $\alpha_t y_{t-1}(\mathbf{s})$ captures the temporal dynamics, and $g_t(\mathbf{s})$ is a mean zero spatial process. The exogenous fixed effects are modeled as $\mu_t(\mathbf{s}) = \mathbf{x}_t(\mathbf{s})^\top \boldsymbol{\beta}_t$, with covariates $\mathbf{x}_t(\mathbf{s})$ and regression coefficients $\boldsymbol{\beta}_t$. The residual spatial field $g_t(\mathbf{s})$ is modeled as a Gaussian process $g_t(\cdot) \sim \mathcal{GP}(0, C_t)$ where the covariance function, $C_t(\mathbf{s}, \mathbf{s}')$, is allowed to be nonstationary and anisotropic. We observe two distinct processes, $z_t^{(1)}(\mathbf{s})$ and $z_{i,t}^{(2)}$, that provide noisy measurements of the latent process, each observed on a different spatial support and subject to its own measurement error.

The process $z_t^{(1)}(\mathbf{s})$ is a sparse and irregular point process that is observed at n_t spatial locations that can vary with t . We let

$$z_t^{(1)}(\mathbf{s}) \equiv y_t(\mathbf{s}) + \varepsilon_t^{(1)}(\mathbf{s}) \quad (6.2)$$

where $\varepsilon_t^{(1)}(\mathbf{s}) \sim \mathcal{N}(0, \tau_t^2)$ is an independent measurement error term. In complement, $z_{i,t}^{(2)}$ is regularly gridded and has no missing values. The process $z_{i,t}^{(2)}$ is spatially defined on a discretised domain $\mathcal{S}^d$ made of non-overlapping areal units A_i , and $\mathcal{S} = \bigcup_{i=1}^N A_i$, (i.e., $\mathcal{S}^d \equiv \{A_i \subset \mathcal{S} : i = 1, \dots, N\}$). The process $z_{i,t}^{(2)}$ represents areal data and is defined as the areal averages

$$z_{i,t}^{(2)} \equiv \frac{1}{|A_i|} \int_{A_i} y_t(\mathbf{s}) \, d\mathbf{s} + \varepsilon_{i,t}^{(2)} \quad (6.3)$$

where $\varepsilon_{i,t}^{(2)} \sim \mathcal{N}(0, \xi_t^2)$ is an error term.

6.5.2 Data Fusion and Change of Support Problem

Data fusion and change of support problems (COSP) arise from the relationship between point-level data $z_t^{(1)}(\mathbf{s})$ and areal data $z_{i,t}^{(2)}$. Given the full spatial and temporal coverage of the areal data, we aim to transfer information from $z_{i,t}^{(2)}$ to the point-level model, thus improving our understanding of the hidden process. Specifically, we use areal data to learn the spatial correlation structure and transfer it to the point-level model as a prior estimate of the covariance function. The connection between $z_t^{(1)}(\mathbf{s})$ and $z_{i,t}^{(2)}$ is detailed in Appendix C.1 and a brief description is provided here. The model for the areal data can be written as

$$z_{i,t}^{(2)} = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \eta_{i,t}^{(2)}, \quad (6.4)$$

with

$$\eta_{i,t}^{(2)} = \bar{g}_{i,t} + \varepsilon_{i,t}^{(2)} + \alpha_t \varepsilon_{i,t-1}^{(2)}, \quad (6.5)$$

where $\bar{\mathbf{x}}_{i,t} = |A_i|^{-1} \int_{A_i} \mathbf{x}_t(\mathbf{s}) \, d\mathbf{s}$ and $\bar{g}_{i,t} = |A_i|^{-1} \int_{A_i} g_t(\mathbf{s}) \, d\mathbf{s}$. Therefore, $\eta_{i,t}^{(2)}$ captures the spatial component and the error term, in particular the collection $\boldsymbol{\eta}_t^{(2)} = (\eta_{1,t}^{(2)}, \dots, \eta_{N,t}^{(2)})^\top \sim \mathcal{N}(0, \Sigma_{\boldsymbol{\eta}_t}^{(2)})$ with $\Sigma_{\boldsymbol{\eta}_t}^{(2)} = \Sigma_{\bar{g}_t} + I(\xi_t^2 + \alpha_t^2 \xi_{t-1}^2)$. Elements of the matrix $\Sigma_{\bar{g}_t}$ correspond to the covariance between all pairs $\bar{g}_{i,t}$ and $\bar{g}_{j,t}$. From $\boldsymbol{\eta}_t^{(2)}$ parameters of $\Sigma_{\bar{g}_t}$ are found using a neural estimator, discussed in Section 6.5.4.

To establish the connection between areal and point-level data, we decompose $y_t(\mathbf{s})$ into the areal average $\bar{y}_{i,t}$ and a residual term $\psi_t(\mathbf{s})$ that captures the fine-scale variability that is smoothed out by areal averaging. Specifically, for any $\mathbf{s} \in A_i$,

$$y_t(\mathbf{s}) = \bar{y}_{i,t} + \psi_t(\mathbf{s}), \quad (6.6)$$

where $\psi_t(\mathbf{s}) \sim GP(0, C_{\psi_t}(\mathbf{s}, \mathbf{s}'))$ and C_{ψ_t} is a covariance function such that $C_{\psi_t} = f(C_{g_t})$. The point-level model can be rewritten as

$$z_t^{(1)}(\mathbf{s}) = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \eta_t^{(1)}(\mathbf{s}), \quad (6.7)$$

with

$$\eta_t^{(1)}(\mathbf{s}) = \bar{g}_{i,t} + \varepsilon_t^{(1)}(\mathbf{s}) - \alpha_t \varepsilon_{i,t-1}^{(2)} + \psi_t(\mathbf{s}), \quad (6.8)$$

where $\eta_t^{(1)}(\mathbf{s}) \sim GP(0, C_{\eta_t}^{(1)})$. The form of $C_{\eta_t}^{(1)}$ is described in Appendix C.1. In summary, both the point-level covariance function $C_{\eta_t}^{(1)}(\cdot, \cdot)$ and the areal covariance matrix $\Sigma_{\eta_t}^{(2)}$ encode the same large-scale spatial structure determined by $\Sigma_{\bar{g}_t}$, justifying a transfer learning approach between the two data sources. Moreover, for a sufficiently small $|A_i|$, the resulting covariances are approximately equivalent. In the case of larger $|A_i|$, $\Sigma_{\eta_t}^{(2)}$ does not include the short-range structure from C_{ψ_t} . As it is unclear whether $|A_i|$ is sufficiently small in our application, we account for this effect by estimating an additional parameter in the point-level model, as detailed in Section 6.5.7.

6.5.3 The Non-stationary LatticeKrig Model

The LatticeKrig model specifies a GP through an efficient basis-function representation with coefficients governed by a spatial autoregression (SAR). These ingredients result in spatial dependence being controlled by a sparse precision matrix and provide scalable computations for large spatial data. By the LatticeKrig specification, the spatial component $g_t(\mathbf{s})$ becomes $\Phi(\mathbf{s})^\top \mathbf{c}_t$ and the model for the true latent process for a single day becomes:

$$y_t(\mathbf{s}) = \mathbf{x}_t(\mathbf{s})^\top \boldsymbol{\beta}_t + \alpha_t y_{t-1}(\mathbf{s}) + \Phi(\mathbf{s})^\top \mathbf{c}_t, \quad (6.9)$$

where $\Phi(\mathbf{s})^\top \equiv (\phi_1(\mathbf{s}), \dots, \phi_m(\mathbf{s}))$ are the set of basis functions evaluated at location $\mathbf{s}$ and

$\mathbf{c}_t \equiv (c_{1,t}, \dots, c_{m,t})^\top$ are the coefficients. For ease of exposition, we consider a single day, omitting t from our notation.

The basis functions $\phi_j(\mathbf{s})$ are compactly supported, radially symmetric Wendland functions centered at m lattice locations $\{\mathbf{u}_j\}_{j=1}^m \in \mathcal{S}$, and the coefficients satisfy $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, Q^{-1})$, where Q is the sparse precision matrix. The precision matrix Q is specified through a spatial autoregressive (SAR) representation. Given a sparse linear operator B and a random vector $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, I)$, we construct the distribution of $\mathbf{c}$ such that $B\mathbf{c} = \mathbf{e}$. Thus, the operator B is a SAR matrix that decorrelates the coefficients to white noise. In the stationary case, the rows of the matrix B are filled according to the five-point stencil

$$\begin{array}{c|c|c} & -1 & \\ \hline -1 & 4 + \kappa^2 & -1 \\ \hline & -1 & \end{array}. \quad (6.10)$$

The entries of the stencil determine the nonzero coefficients in each row of B : the center value corresponds to the diagonal entry, the four off-center values correspond to the immediate neighbors on the lattice, and all remaining entries in the row are zero. Here, κ^2 controls the correlation range for the spatial domain. By linear statistics, $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, (B^\top B)^{-1})$, thus the corresponding precision matrix $Q = B^\top B$ is also sparse.

For the LatticeKrig model, nonstationarity and anisotropy are achieved by expanding the SAR stencil to second order neighbors and allowing the stencil weights to vary across the domain. Specifically, we allow the diagonal and off-diagonal entries of B to depend on interpretable and spatially varying parameters $\theta(\mathbf{u})$, $\rho(\mathbf{u})$ and $\kappa^2(\mathbf{u})$. Unlike prior work by Sikorski et al. [129] and Wiens et al. [41], these parameters vary across the lattice of basis functions centers $\{\mathbf{u}_j\}_{j=1}^m$, rather than the spatial locations $\{\mathbf{s}_i\}_{i=1}^N$. At each basis-function center $\mathbf{u}_j$, the parameter $\kappa^2(\mathbf{u}_j)$ controls the correlation range (i.e., a larger $\kappa^2(\mathbf{u}_j)$ means smaller correlation range), the parameter $\rho(\mathbf{u}_j)$ controls the degree of anisotropy (e.g., $\rho(\mathbf{u}_j) = 1$ corresponds to isotropy at $\mathbf{u}_j$), and $\theta(\mathbf{u}_j)$ controls the direction of anisotropy. The anisotropy parameters are used to build the rotation matrix $\Psi(\mathbf{u})$ and the anisotropic scaling matrix $\Lambda(\mathbf{u})$,

$$\Psi(\mathbf{u}) = \begin{bmatrix} \cos \theta(\mathbf{u}) & -\sin \theta(\mathbf{u}) \\ \sin \theta(\mathbf{u}) & \cos \theta(\mathbf{u}) \end{bmatrix}, \quad \Lambda(\mathbf{u}) = \begin{bmatrix} \sqrt{\rho(\mathbf{u})} & 0 \\ 0 & \frac{1}{\sqrt{\rho(\mathbf{u})}} \end{bmatrix}, \quad (6.11)$$

which are then used to create the 2×2 dispersion matrix: $D(\mathbf{u}) = \Psi(\mathbf{u})^\top \Lambda(\mathbf{u}) \Psi(\mathbf{u})$. The entries of B in the nonstationary case are filled according to the new spatially varying nine-point stencil

$$\begin{array}{c|c|c} \frac{D_{1,2}(\mathbf{u})}{2} & -D_{2,2}(\mathbf{u}) & \frac{-D_{1,2}(\mathbf{u})}{2} \\ \hline -D_{1,1}(\mathbf{u}) & \kappa^2(\mathbf{u}) + 2D_{1,1}(\mathbf{u}) + 2D_{2,2}(\mathbf{u}) & -D_{1,1}(\mathbf{u}) \\ \hline \frac{-D_{1,2}(\mathbf{u})}{2} & -D_{2,2}(\mathbf{u}) & \frac{D_{1,2}(\mathbf{u})}{2} \end{array}. \quad (6.12)$$

This formulation retains computational advantages, specifies a rich class of non-stationary and anisotropic GPs, and avoids the difficult specification of an analytical covariance function. Another key advantage is that the spatially varying parameters are interpretable and can yield physical insights into the spatial dependence of the field. At each basis-function center, $\mathbf{u}_j$, we can represent the combination of the parameters as an elliptical dependence contour, where the orientation is governed by $\theta(\mathbf{u}_j)$, the aspect ratio is controlled by $\rho(\mathbf{u}_j)$, and the overall size is controlled by $\kappa^2(\mathbf{u}_j)$. It is important to note that this specification is with respect to the SAR, and when the implied covariance is evaluated, the interplay between these elliptical profiles can result in very complex and flexible spatial dependence structures.

However, the estimation of the resulting, spatially varying parameter fields (κ^2, ρ, θ) is a challenging inference problem, especially for large datasets. Fitting the model directly via maximum likelihood or Bayesian methods is not feasible. To address this problem, we use the amortized, neural parameter estimation strategy from the LatticeVision framework [129]. Here, a neural network learns a global mapping from ensembles of spatial fields to the associated parameter fields. This approach enables fast and

accurate one-shot estimation of the parameters in seconds and is described in the following sections.

6.5.4 Neural Parameter Estimation

We use the neural, image-to-image (I2I) estimation strategy from the LatticeVision framework. Specifically, we use the STUN estimator, which is a hybrid architecture that combines the U-Net and Vision Transformer architectures to leverage both the convolution and attention mechanisms. The key feature underlying this approach is that both the input (spatial fields) and the output (parameter fields) are naturally arranged on a regular grid and can therefore be treated as images [129]. This method has been shown to outperform local neural estimators with simpler architectures in both computational efficiency and accuracy. Additionally, these I2I architectures are amortized. Training them takes significant computational resources, but this cost is recouped upon deployment, as the trained net maps an ensemble of spatial fields directly to spatially varying parameter fields in a single forward pass.

6.5.5 Training Procedure

We follow the synthetic data generation and training procedure of [129] with a few key modifications. In brief, we sample from scientifically motivated spatial pattern priors to generate three parameter fields

$$P = [\kappa^2, \rho, \theta]^\top,$$

on the basis function lattice $\{\mathbf{u}\}_{j=1}^m$. We stack the parameter fields along the channel dimension, treating each field as an image, yielding a target tensor $P \in \mathbb{R}^{3 \times h_{\mathbf{u}} \times w_{\mathbf{u}}}$, where $h_{\mathbf{u}} * w_{\mathbf{u}} = m$. These parameter fields define a nonstationary SAR matrix B , as described in Section 6.5.3. To create training inputs, we draw independent white-noise vectors $\mathbf{e}^{(k)} \sim \mathcal{N}(\mathbf{0}, I)$ and solve the linear systems

$$B \mathbf{c}^{(k)} = \mathbf{e}^{(k)}, \quad k = 1, \dots, r,$$

yielding coefficient vectors $\mathbf{c}^{(k)} \sim \mathcal{N}(\mathbf{0}, Q^{-1})$ with precision matrix $Q = B^\top B$. Each coefficient vector is then mapped to a spatial field via the LatticeKrig basis expansion

$$g^{(k)}(\mathbf{s}) = (\Phi \mathbf{c}^{(k)})(\mathbf{s}),$$

producing r independent realizations that share the same underlying nonstationary and anisotropic covariance structure. Following [129], we standardize these realizations: for each pixel, we subtract its mean and divide by its standard deviation across the r replicates, resulting in each pixel having zero mean and unit variance. Stacking the r simulated fields yields the input tensor

$G = \begin{bmatrix} g^{(1)} & \dots & g^{(r)} \end{bmatrix} \in \mathbb{R}^{r \times h_A \times w_A}$, where $h_A * w_A = N$. Given an input G , our STUN estimator is trained to minimize a pixelwise regression loss between the predicted ($\hat{P}$) and true (P) parameter images.

6.5.6 Implementation and Evaluation

For simplicity, we align the basis function lattice and the data grid, so both the input spatial fields and output parameter fields have size $h_A * w_A = N = h_{\mathbf{u}} * w_{\mathbf{u}} = m = 128 * 128$. When creating the target parameter images P , the parameter values are sampled from the following range: $\rho \in [1, 7]$, $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2})$, and $\log(\kappa^2) \in [-9.2, 2.3]$ (with $\kappa^2 \in [10^{-4}, 10]$ represented on a log scale during training and evaluation for numerical stability). These parameter values were chosen to capture a broad range of spatial relationships, ranging from isotropy ($\rho = 1$) to very elongated dependence ellipses ($\rho \gg 1$) in any direction. The upper bound of κ^2 is set to 10 to include shorter correlation ranges in the data. We choose an ensemble size of $r = 30$ replicates within each input G , and following the procedure in [129], we generate an entirely synthetic dataset with 20,000 training pairs (G, P) . We use the 90/8/2 (train/validation/test) split, resulting in a test set of 400 synthetic data samples, which we use to verify the accuracy of our estimator. All remaining implementation details are provided in Appendix C.2.

Table 6.1 summarizes validation performance of the neural estimator across all pixels with several standard image regression metrics. We find that STUN recovers all parameters with high accuracy, supporting its use as an efficient surrogate for exact, likelihood-based inference.

Table 6.1 STUN parameter-estimation performance (all true vs. predicted pixels) on held-out test data. Here, NRMSE is computed relative to the range (max - min) of parameter values. Relative error is summarized by the median and is reported as a percentage. Arrows indicate the desirable direction.

Parameter	RMSE ↓	NRMSE ↓	MAE ↓	R^2 ↑	Median Rel. Err. ↓
$\log(\kappa^2)$	0.2169	0.01884	0.1317	0.9898	6.87%
θ	0.0896	0.02987	0.0509	0.9832	6.89%
ρ	0.3912	0.06521	0.2779	0.9227	5.17%

6.5.7 Extensions for Data Fusion and Change of Support Problem

To address the change-of-support problem (COSP) arising from the integration of areal and point-referenced data, this work introduces two key extensions to the amortized neural estimation framework in [129]. First, we make use of the compactly supported basis functions in the original LatticeKrig formulation, with the SAR being applied to the coefficients. The basis functions enable the transfer from gridded to point-level data, along with interpolation to finer grids. Second, we allow for the refinement of the parameter κ^2 , which controls correlation range, when transferring information from the regularly gridded data to the point-level observations, as discussed below.

The initial estimates of the parameter fields for a particular day $\hat{P} = [\hat{\kappa}^2, \hat{\rho}, \hat{\theta}]^\top$ are first obtained from the regularly gridded data using STUN. To adjust for the change of support, we assume that the

correlation range should decrease because the coarse, gridded data is likely to over-smooth the fine-scale structure that is present in the point-level data (see Section 6.5.2 and Appendix C.1).

Therefore, we supplement the initial estimate of κ^2 with an additional, stationary term κ_{point}^2 , which is estimated from the point-level data. The adjusted estimate of κ^2 is then

$$\hat{\kappa}_{adj}^2 = \hat{\kappa}^2 + \mathbf{w}\hat{\kappa}_{point}^2 \quad (6.13)$$

where the weights $\mathbf{w} \in \mathbb{R}^m$ can be chosen to allow heterogeneous adjustments. In this paper, point-level data are only present on land. Accordingly, we set $w(\mathbf{u}) = 1$ for basis-function centers located on land and $w(\mathbf{u}) = 0$ for centers located over the ocean, where no point-level observations are available. Since the point level data is available at a small number of observations n_t , the additional κ_{point}^2 parameter is estimated via MLE. Regarding anisotropy, we assume the initial estimates are sufficiently informative, thus we do not adjust $\hat{\rho}$ and $\hat{\theta}$.

6.6 Application to Numerical Model Outputs

6.6.1 Parameter Estimation

The CTM outputs are formatted to a regular $0.1^\circ \times 0.1^\circ$ grid over longitudes 6.1° – 18.8° E and latitudes 35.2° – 47.9° N. Here, $\{A_i\}_{i=1}^N$ denotes the collection of non-overlapping grid cells and $N = 128 * 128$. For each day in 2023, $t \in \{1, \dots, 365\}$, the CTM field is identified as the areal process $\{z_{i,t}^{(2)}\}_{i=1}^m$, where each $z_{i,t}^{(2)}$ corresponds to the grid-cell value over A_i . All covariates are also aligned to this grid. When a covariate is available at a different resolution, we upscale to the CTM grid resolution by averaging (e.g., elevation and emissions) or downscale via nearest-neighbor resampling (e.g., ERA5 where ERA5-Land is unavailable). To isolate the residual spatial process $\bar{g}_{i,t}$ for parameter estimation, we fit and then subtract the conditional mean model for each day. Specifically, the mean of our process is the ARX(1) regression, described in Eq. 6.4, composed of $\bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)}$, where $\bar{\mathbf{x}}_{i,t}$ includes an intercept, latitude, longitude, and the seven covariates described in Section 6.4.2, and $z_{i,t-1}^{(2)}$ is the lagged NO_2 concentration from the numerical model. The choice to include lagged NO_2 values as a covariate is motivated by a well-known, strong AR(1) behavior in air pollutant concentrations [156].

We assume the residual spatial process evolves slowly and smoothly over time. This is supported by the structure of our mean model: the AR(1) term removes the bulk of temporal autocorrelation, while covariates such as wind speed and boundary layer height account for much of the daily variation in anisotropic parameters, leaving residuals that are largely driven by slower seasonal patterns. Neighboring residual fields can therefore be treated as pseudo-replicates from a common distribution, allowing for more accurate parameter estimation and stable covariances. We therefore adopt a moving window strategy, including the 15 days before and the 14 days after the day selected, constructing ensembles of $r = 30$

residual fields

$$G_t \in \mathbb{R}^{30 \times 128 \times 128},$$

which are centered at day t . To avoid temporal boundary effects at the beginning and the end of 2023, we extend our time coverage to the last 15 days of 2022 and the first 14 days of 2024. To align with the synthetic data procedure, within each ensemble G_t , the 30 residual fields have been standardized pixel-wise. The zero-mean and unit variance fields are then used as the multi-channel input image to STUN. Following Section 6.5.6, we align the grid cells and the lattice of basis function centers, so the parameter fields are estimated on a 128×128 lattice of basis-function centers. For each day t , STUN outputs

$$\hat{P}_t = [\hat{\kappa}_t^2, \hat{\rho}_t, \hat{\theta}_t]^\top \in \mathbb{R}^{3 \times 128 \times 128}.$$

Overall, this amounts to $365 * 128 * 128 * 3 = 17,940,480$ parameter values that must be estimated. The amortized nature of our estimator means inference is extremely fast: on a single GPU, STUN executes 365 forward passes to estimate all 17,940,480 parameter values in just 4 seconds (averaged over 30 trials). Hardware details can be found in Appendix C.2.

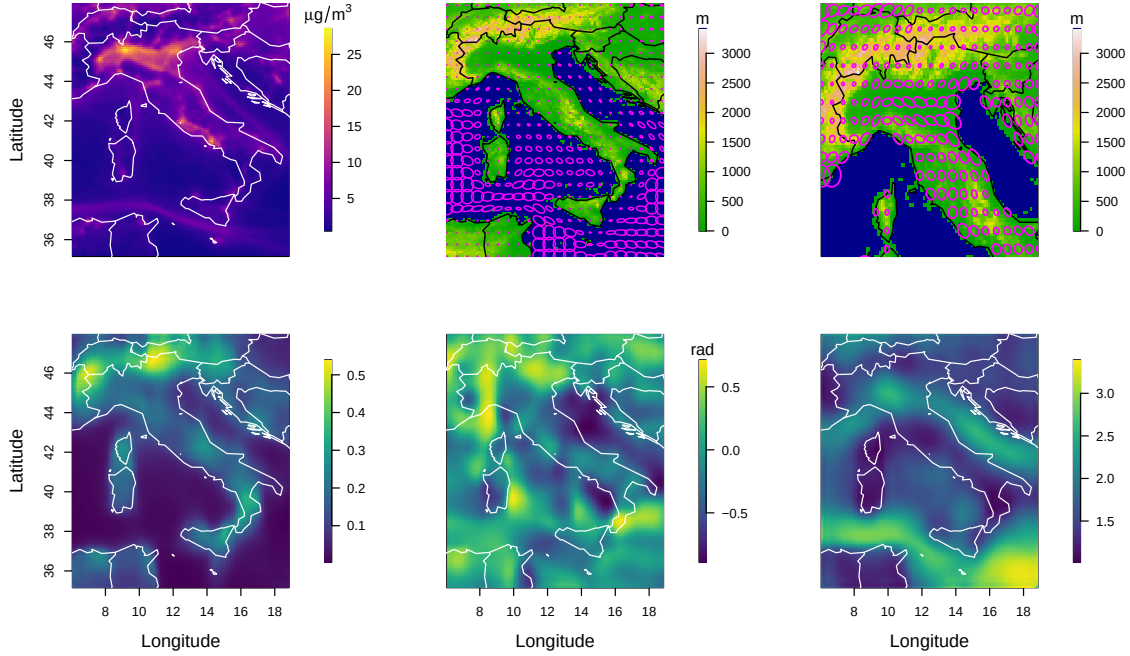


Figure 6.2 2023 annual averages of CTM-based parameter estimates. Top row from left to right: the average CTM NO_2 field, elliptical dependence contours overlaid on an elevation (m) map of the full domain, and a zoomed-in view of enlarged ellipses over land only. Bottom row from left to right: annual averages of STUN estimates $\hat{\kappa}^2$, $\hat{\theta}$, and $\hat{\rho}$.

Qualitatively, the resulting parameter estimates exhibit physically plausible spatial structure. Figure 6.2 summarizes spatial patterns through 2023 annual averages of the NO_2 field, the estimated parameters, and

elliptical contours representing the average local dependence structure. The inferred range fields $\{\hat{\kappa}_t^2\}$ vary smoothly at large scales while preserving sharp transitions near coastlines, with generally shorter ranges over land and longer ranges over surrounding water. The anisotropy fields $\{\hat{\rho}_t\}$ and $\{\hat{\theta}_t\}$ highlight directional dependence aligned with major geographic features: we observe pronounced anisotropy and consistent orientation patterns in the Po Valley, along the Adriatic Sea, and following maritime routes. Systematic changes in dependence structure are also evident near complex terrain such as mountain ranges.

6.6.2 CTM Reconstruction Experiment

Having obtained parameter fields for each day of 2023 from STUN, we assess whether these nonstationary parameters improve predictive performance. To do so, we design a controlled reconstruction experiment using the CTM data. This experiment mirrors our later application to monitoring station data, but with the advantage that, unlike monitoring station data, the full CTM NO_2 fields are known. For each day, we mask out all grid cells except those coinciding with locations where monitoring stations exist. On the 128×128 CTM grid (16,384 pixels), this leaves 483 pixels (approximately 3%) as observed locations, with the remaining pixels treated as missing and used for evaluation.

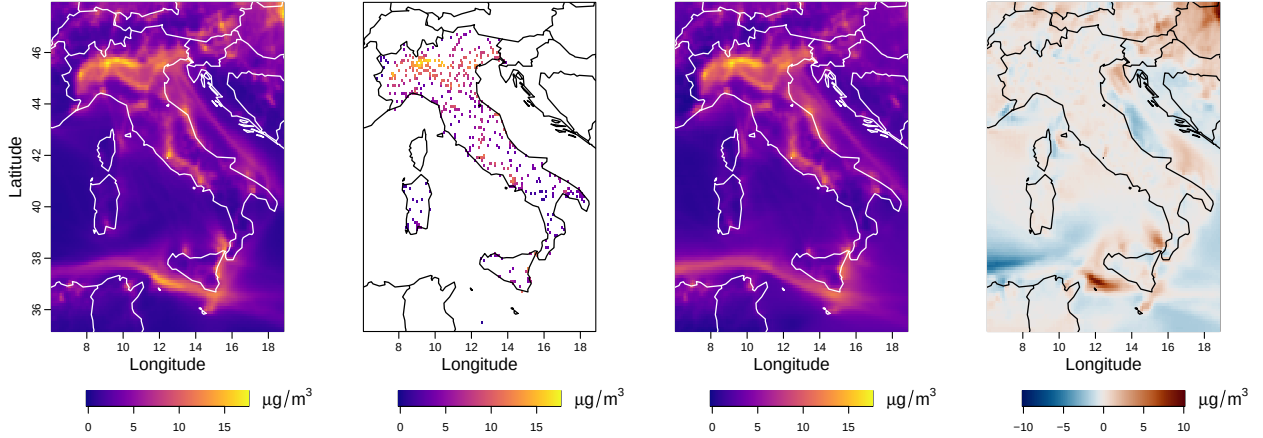


Figure 6.3 CTM NO_2 reconstruction experiment for April 13, 2023. From left to right: the actual CTM NO_2 field, masked data used for model fitting (pixels at locations with stations), the full prediction from the non-stationary model, and the difference between the true and predicted field.

We then predict the full CTM NO_2 field from sparse observations under different covariance assumptions. Figure 6.3 illustrates the true field and the masked data for a representative day. We compare two models. First, we fit a stationary, isotropic LatticeKrig model to each day. The mean function of this stationary model is fit using the ARX(1) structure described in Section 6.6.1. The model

uses the same single-resolution basis construction with $m = 128 * 128$ basis functions as the nonstationary model, and the covariance structure is governed by a single global range parameter κ^2 , estimated by maximum likelihood. This stationary Gaussian process model serves as a baseline, and is representative of standard approaches in spatial statistics. Second, we fit the nonstationary, anisotropic LatticeKrig model, again using the same mean structure and CTM NO_2 data, with covariance structure governed by the STUN-estimated spatially varying parameter fields $\hat{\kappa}^2$, $\hat{\rho}$, and $\hat{\theta}$ from Section 6.6.1.

Across the year, the nonstationary model achieves a lower reconstruction RMSE for the held-out pixels on 267 of 365 days, corresponding to an overall RMSE improvement of 3.2%. The relative improvement exhibits a seasonal structure, as the nonstationary model tends to outperform the stationary model more strongly during colder months, where daily RMSE improvements can be as large as 22.8% (see Figure 6.4).

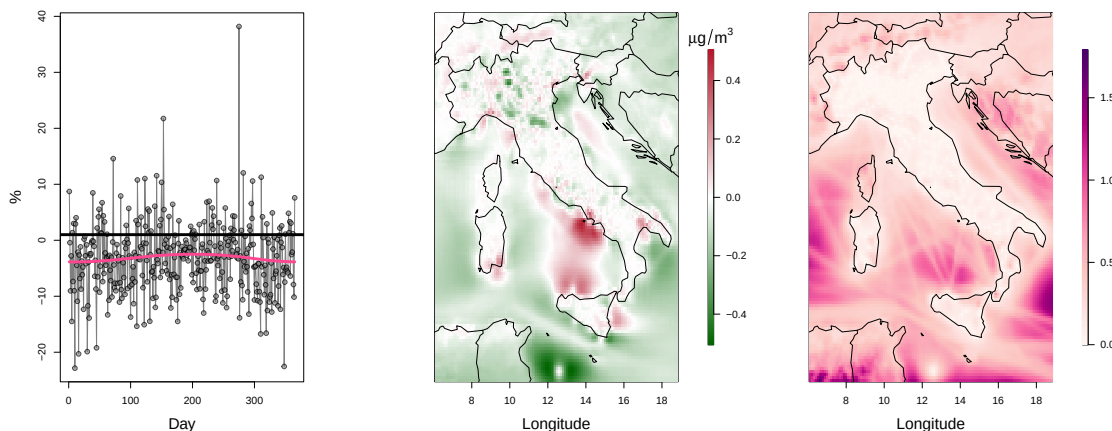


Figure 6.4 Left: time series of daily RMSE percent differences $((\text{nonstationary} / \text{stationary} - 1) * 100)$. A solid black line illustrates equality in both cases, while the pink line is a GAM-based fit with a cyclic cubic spline basis (20 basis functions), chosen to capture seasonal structure in the temporal trend. Dots below the black line indicate days on which the nonstationary model outperforms the stationary one. Center: difference in pixel-wise RMSE (nonstationary $-$ stationary); negative values indicate locations where the nonstationary model outperforms the stationary model. Right: the pixel-wise nonstationary RMSE divided by the local annual mean NO_2 concentration to show relative errors.

In absolute terms, the nonstationary model achieves an RMSE of $1.093 \mu\text{g}/\text{m}^3$ compared to $1.137 \mu\text{g}/\text{m}^3$ for the stationary model. Spatially, the largest relative reconstruction errors tend to occur in mountainous regions outside Italy and along major shipping routes, where emissions are highly variable and Italian monitoring stations are not present. In contrast, errors across Italy remain very small. The difference map in the center panel of Figure 6.4 shows that the nonstationary model achieves lower RMSE than the stationary model in key regions, including the Po Valley and areas with sparse or nonexistent station coverage. Overall, the results of this experiment provide an argument for the use of STUN-estimated parameters for the subsequent application to monitoring station data.

6.7 Application to Ground-based Measurements

For 2023, a total of 744 monitoring stations in Italy reported at least one daily NO_2 measurement. To ensure sufficient spatial coverage for reliable inference, we restrict our analysis to days with at least 250 active stations, yielding 349 valid days. The omitted 16 days exhibited a sharp decline in network coverage, with several days providing only a few dozen observations. Additionally, we utilize only “background” stations and remove anomalous measurement values ($> 80 \mu\text{g}/\text{m}^3$).

6.7.1 Model Estimation

Similarly to Section 6.6.2, we fit three different models of increasing complexity for each of the 349 days. The first two models are those used in the reconstruction experiment in Section 6.6.2. The first model is the stationary, isotropic LatticeKrig model with a single global range parameter estimated by maximum likelihood. The second is the nonstationary, anisotropic LatticeKrig model that uses the STUN-derived parameter fields estimated from the CTM data in Section 6.6.1, but without further correlation range adjustment. The third and most flexible model improves the CTM-informed covariance with an additional range refinement term using the method described in Section 6.5.7. Specifically, when fitting the model, we define the final κ^2 field as

$$\hat{\kappa}_{\text{adj}}^2 = \hat{\kappa}_{CTM}^2 + \mathbf{w}\hat{\kappa}_{EEA}^2 \quad (6.14)$$

where $\hat{\kappa}_{CTM}^2$ is the parameter field previously estimated from CTM data using STUN, and κ_{EEA}^2 is a single parameter that reduces the correlation range and is estimated from the station data by maximum likelihood. The weights $\mathbf{w} \in \mathbb{R}^m$ are specified such that $w(\mathbf{u}) = 1$ for basis-function centers located on land and $w(\mathbf{u}) = 0$ otherwise.

The temporal correlation tends to be strong across all models, with estimated values of $\hat{\alpha}_t$ consistently exceeding 0.8. Regarding spatial correlation, we analyze the changes in the range parameter between the stationary model and the adjusted non-stationary one. The comparison is made across both space and time, and is restricted to land cells where the adjustment occurs. From a temporal perspective, we compare the daily average of $\{\hat{\kappa}_{\text{adj}}^2(\mathbf{u}) : w(\mathbf{u}) = 1\}$ against the daily value of $\hat{\kappa}_{\text{stat}}^2$. Figure 6.5 shows the time series of their log-scale differences ($\log(\hat{\kappa}_{\text{stat}}^2) - \text{mean}_{\mathbf{u}}(\log(\hat{\kappa}_{\text{adj}}^2(\mathbf{u})))$), demonstrating that the nonstationary model exhibits a larger average correlation range on most days. This is likely due to the lack of flexibility permitted in the stationary model, causing it to artificially shrink the correlation range to avoid imposing dependence. From a spatial perspective, we compute the log-scale difference (i.e., $\log(\hat{\kappa}_{\text{stat}}^2) - \log(\hat{\kappa}_{\text{adj}}^2(\mathbf{u}))$) at all basis-function centers on land, averaged over all 349 valid days, and visualize the result as a map in Figure 6.5. The largest differences are concentrated in the Po Valley and

along the coastline, where the nonstationary model captures the expected longer-range, anisotropic correlations that are characteristic of these areas. In contrast, both models produce similar correlation ranges in mountainous regions, where complex terrain shrinks the range.

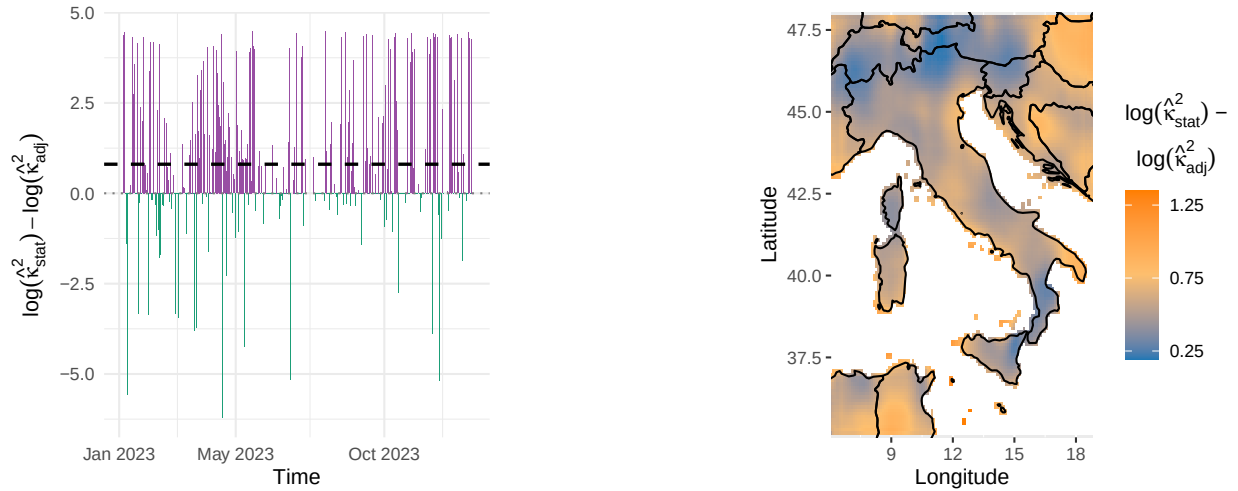


Figure 6.5 Differences between the range parameters estimated by the stationary model and the adjusted non-stationary one, restricted to basis-function centers on land. Left: daily log-scale differences between the stationary range parameter and the spatial average of the nonstationary one. The dotted line indicates the mean difference of 0.8. Right: annual average of the daily log-scale difference at each basis function center.

To demonstrate the effect of different correlation parameters between the two models, Figure 6.6 shows the estimated spatial components from both models for a representative day. The nonstationary model tends to produce a markedly smoother spatial component, which more coherently follows the underlying geographical patterns across the Italian peninsula. This is especially evident in the Po Valley, which appears as a more homogeneous region, in contrast to the localized “patchy” structures observed in the stationary model. Moreover, a larger correlation range in the sea induces a greater effect along the coast.

6.7.2 Cross-validation

We also assess how well the different models reproduce observed NO_2 concentrations at monitoring locations via 10-fold cross-validation, performed separately for each of the 349 eligible days. For a given day, the available stations are randomly partitioned into 10 approximately equal folds. Each model is then iteratively fit to 90% of the stations, and concentrations are predicted at the remaining 10%. We do not employ spatial stratification for the assignment of folds, as the set of active stations changes daily, and coverage can be highly uneven, which would result in imbalanced folds and unstable error estimates. Instead, we use random partitioning, ensuring each fold contains a comparable number of observations. Because the number of available stations varies by day, we report aggregated performance metrics for all

days below in Table 6.2, with additional diagnostic figures provided in Appendix C.3.

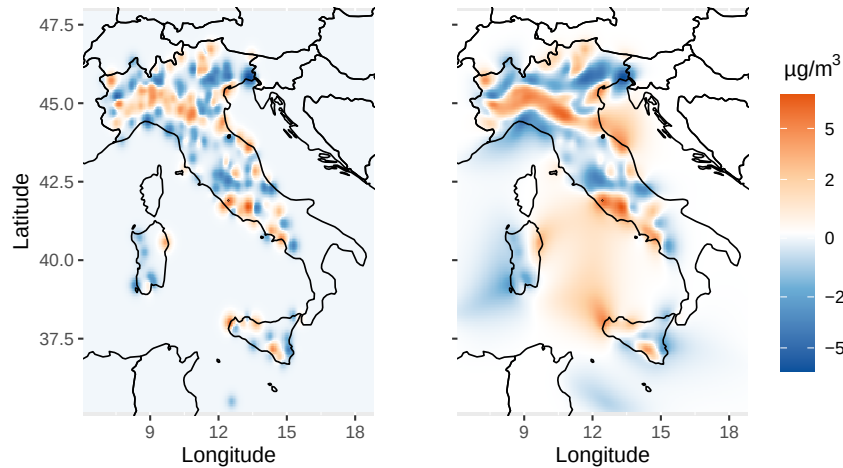


Figure 6.6 Estimated spatial component on November 1, 2023. Left: stationary model. Right: adjusted nonstationary model

As expected, all three spatial models perform near equivalently in aggregate, as station locations are frequently clustered together, reducing the ability to distinguish differences in model performance. This behavior is also consistent with existing literature on the effect of nonstationarity on predictive performance [157]. Moreover, the stationary LatticeKrig model is already highly flexible with a large number of basis functions ($128 * 128$), making the margin for improvement inherently small. Nevertheless, the correlation range refinement step systematically improves upon the original non-stationary model, and the adjusted non-stationary model achieves the lowest RMSE.

Table 6.2 RMSE, 95% Prediction Interval Coverage Probability (PICP), Mean 95% Prediction Interval Width (MPIW). Arrows indicate the desirable direction.

Model	RMSE ↓	PICP (95%)	MPIW ↓
Stationary	6.544	94.030%	23.845
Nonstationary	6.550	94.037%	24.038
Nonstationary (adjusted)	6.530	94.022%	23.826

6.7.3 Prediction to Fine Grids

We use the adjusted nonstationary model to generate spatially complete, daily maps of NO_2 concentrations across all of Italy. For each of the 349 retained days, the fitted model is used to predict the latent daily mean field on a finer $0.05^\circ \times 0.05^\circ$ grid (255×255 pixels) over the same spatial domain (longitudes 6.1° – 18.8°E , latitudes 35.2° – 47.9°N). All covariates are aligned to the prediction grid via

nearest-neighbor resampling, except for elevation, which is aggregated by averaging from its native higher resolution. This fine-grid prediction is made possible by embedding the STUN-estimated parameters within the LatticeKrig basis-function framework, rather than treating them as a strict GMRF on the observation grid. Because the basis functions $\Phi(\mathbf{s})$ can be evaluated at any spatial location $\mathbf{s}$, predictions are obtained on an arbitrarily fine target grid simply by evaluating $g(\mathbf{s})$ at the new locations, with no re-estimation of the covariance parameters. This decoupling of prediction resolution from estimation resolution is a direct consequence of extending LatticeVision to incorporate basis functions.

To accompany the predictions with measures of uncertainty, we perform conditional simulation from the fitted Gaussian process. We generate 1,000 conditional draws, producing an ensemble of simulated fields that exactly respect both the fitted covariance structure and the observed station data. Pixelwise summaries of uncertainty, such as the posterior standard error, are then computed directly from this ensemble. The resulting predictions and standard errors for a representative day are shown in Figure 6.7. As expected, uncertainty is lowest in areas with dense station coverage and increases in topographically complex regions such as the Alps, rural areas with few nearby observations, and the sea.

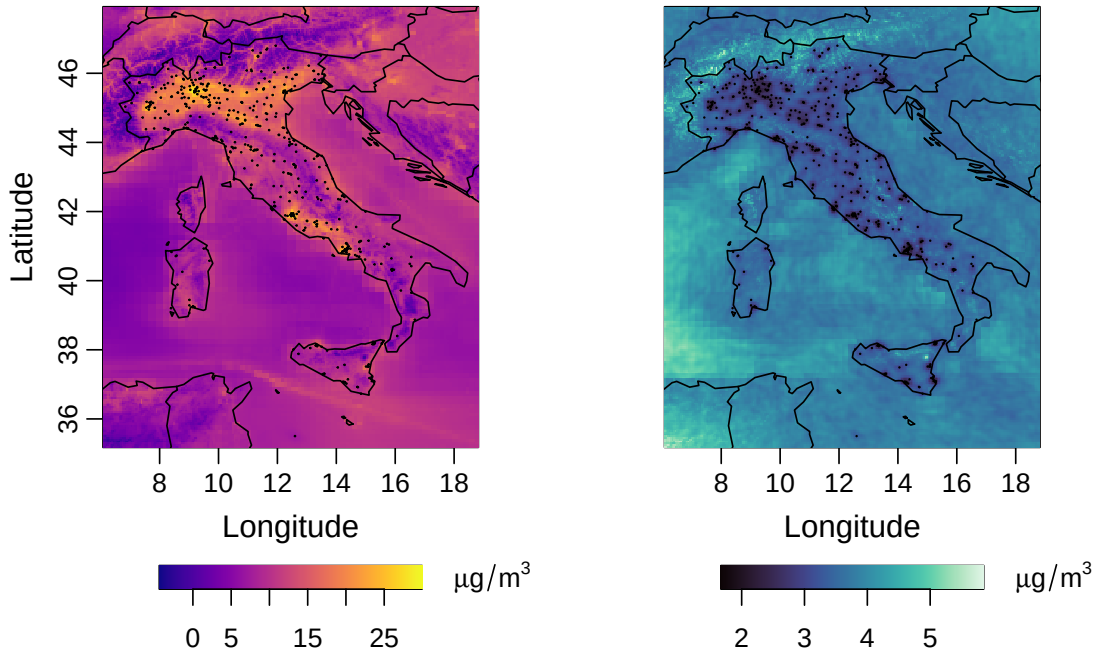


Figure 6.7 Fine-grid predictions for November 1, 2023. Left: posterior mean NO_2 concentration ($\mu\text{g}/\text{m}^3$) on the $0.05^\circ \times 0.05^\circ$ prediction grid. Right: posterior standard error ($\mu\text{g}/\text{m}^3$) derived from a 1,000-member conditional simulation ensemble.

It is important to note that the computational cost of prediction is modest. Generating the full fine-grid posterior mean for a single day requires 25 seconds on average, meaning the complete set of 349 daily maps can be produced in under three hours. Conditional simulation incurs a one-time required setup with the same computational cost as prediction (25 seconds), and each subsequent draw takes less than 2 seconds, allowing for very large ensembles to be generated. The resulting daily concentration maps and their associated uncertainty fields constitute a spatially complete, fine-resolution record of surface NO_2 over Italy, suitable for downstream applications in epidemiology, urban planning, and environmental policy assessment.

6.8 Discussion

In this work, we introduce a spatial transfer-learning approach for combining sparse, irregular, point-level observational data with gridded, spatially complete numerical model outputs. The underlying spatial dependence structure is first estimated from the regularly gridded data and then transferred to the observation-level model. We apply this methodology to daily NO_2 concentrations over Italy, using monitoring-station observations and chemical transport model (CTM) outputs to produce spatially complete concentration maps on a fine prediction grid, while retaining an interpretable covariance model and providing rigorous uncertainty quantification.

Our approach builds on the LatticeKrig formulation, which represents a Gaussian process through compactly supported basis functions with coefficients governed by a sparse precision matrix, enabling scalable inference over large spatial domains. We address the change-of-support problem using two strategies. First, we expand amortized neural parameter estimation by explicitly incorporating the basis-function representation into a LatticeVision-style estimation pipeline. This allows the covariance structure that is learned from the regularly gridded CTM outputs to be transferred to point-level observations and subsequently projected onto a finer prediction grid. Second, we introduce a likelihood-based refinement step for the correlation range parameter when transitioning between areal and point-level data sources. In our application, we expect that the CTM outputs capture large-scale dependence and anisotropy but tend to smooth over fine-scale variability. Treating the CTM-derived anisotropy parameters as informative, we refine the local correlation range using station data, conditioning on the nonstationary, anisotropic correlation structure learned from the gridded data.

We apply this framework to both areal and point-level data. The CTM reconstruction experiment provides the most reliable way to evaluate predictions off of station locations. In this case, the nonstationary model achieves the best performance, with the strongest gains occurring during colder months. When applied to the point-level measurements, the adjusted non-stationary correlation range is

larger than the stationary one, especially in the Po Valley and along coastlines. The different correlation structure is well demonstrated by the estimated spatial fields: the adjusted non-stationary process tends to show smoother behavior, which more coherently follows the underlying geographical and spatial patterns across the Italian peninsula. Moreover, the cross-validation scheme demonstrates the value of the correlation range refinement. Although cross-validation can be useful to assess predictive skill, in this application we feel it has limited value. This is due to the clustering of urban stations and the changing number of daily reporting stations.

Beyond point prediction, a key outcome of the proposed framework is the ability to generate spatially complete daily maps with uncertainty quantification. Using conditional simulation under the fitted nonstationary LatticeKrig model, we produce ensembles of plausible NO_2 fields on a $0.05^\circ \times 0.05^\circ$ grid, enabling pixel-wise uncertainty summaries and uncertainty propagation for derived spatial quantities. The resulting uncertainty maps exhibit intuitive spatial patterns, with low uncertainty in regions of dense monitoring coverage and increased uncertainty in topographically complex or observation-sparse areas. Both predictions and conditional ensembles for uncertainty quantification incur minimal computational expense.

Future research could explore several extensions to the framework proposed in this paper. First, while temporal dependence is incorporated in the mean through an ARX(1) specification, a more sophisticated handling of the temporal trend could reduce uncertainty and improve prediction on days with sparse station coverage. Second, although we focus on a single-resolution LatticeKrig model for clarity, this framework naturally extends to a multi-resolution construction. Including several resolutions results in covariance functions that more closely approximate standard families such as the Matérn. Third, the likelihood-based refinement step currently adjusts only the correlation range parameter; extending this refinement to additional parameters may improve predictive performance on observational data. Finally, more flexible weighting schemes in the refinement step based on station density, distance to observations, or terrain characteristics could be explored to improve localized adjustments.

6.9 Acknowledgements

We would like to thank Soutir Bandyopadhyay, Souvick Bera, and Adrienne Marshall for their insights and helpful discussions. 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. This project is co-funded under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.3 - Call for tender No. 341 of

15/03/2022 of Italian Ministry of University and Research funded by the European Union –
NextGenerationEU. Award Number: PE_00000018, Concession Decree No. 1558 of 11/10/2022 adopted by
the Italian Ministry of University and Research, CUP F83C22001720001, GROWING RESILIENT
INCLUSIVE AND SUSTAINABLE - GRINS.

CHAPTER 7

MULTI-RESOLUTION NON-STATIONARITY WITH LATTICEKRIG

The previous chapters of this thesis have introduced the capabilities of the LK model in successive stages of complexity. In Section 2.4, the stationary, isotropic single-layer model is introduced, with the multi-resolution component detailed in Section 2.4.1. The model is then extended to anisotropy and non-stationarity in Chapter 4, but only a single layer is used. In this chapter, we demonstrate the full flexibility of the LK model by combining these extensions. We introduce the multi-resolution, anisotropic, non-stationary LK model, opening the door to future research in which significantly more complex phenomena could be modeled.

7.1 Non-Stationarity and Anisotropy across Multiple Resolutions

In Chapter 4, the single-layer LK model was generalized to accommodate anisotropy and non-stationarity by introducing three spatially varying parameter fields κ^2 , ρ , and θ defined at the lattice of basis-function centers $\{\mathbf{u}_j\}_{j=1}^m$. These fields define the spatially varying nine-point stencil in Equation (4.4). Separately, in Section 2.4.1, the stationary multi-resolution model was expressed as the sum of L independent layers,

$$g(\mathbf{s}) = \sum_{\ell=1}^L \sum_{j=1}^{m_\ell} c_{\ell,j} \phi_{\ell,j}(\mathbf{s}), \quad (7.1)$$

where the marginal variance α_ℓ weights the contribution of each layer ℓ . In the combined model, each layer $\ell = 1, \dots, L$ has its own lattice of basis-function centers $\{\mathbf{u}_j^{(\ell)}\}_{j=1}^{m_\ell}$, on which the spatially varying parameters $\kappa_\ell^2(\mathbf{u}^{(\ell)})$, $\rho_\ell(\mathbf{u}^{(\ell)})$, and $\theta_\ell(\mathbf{u}^{(\ell)})$ are defined. These parameter fields construct the corresponding layer's SAR matrix B_ℓ .

To provide even more flexibility, the layer's scalar α_ℓ is promoted to a field α_ℓ , subject to the constraint that $\sum_{\ell=1}^L \alpha_\ell(\mathbf{u}_j^{(\ell)}) = 1$ at each center. This allows the contribution of each layer itself to vary across the spatial domain. When working with real data, this allows for even more flexibility, as in some regions of the hypothetical domain, the finer resolutions of the model may provide a better estimate of the process, while in others it is desirable for the larger scales to dominate. Although this is partially allowed by a non-stationary, single-layer model, the combined model expands the capability even further. The resulting precision matrix that determines the spatial dependence for all coefficients in the model then has a block diagonal structure such that

$$Q = \begin{bmatrix} A_1^{-1} B_1^\top B_1 A_1^{-1} & 0 & \cdots & 0 \\ 0 & A_2^{-1} B_2^\top B_2 A_2^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_L^{-1} B_L^\top B_L A_L^{-1} \end{bmatrix} \in \mathbb{R}^{M \times M}. \quad (7.2)$$

Here, for notational simplicity, each layer's marginal variance field has been written as a diagonal matrix $A_\ell = \text{diag}\left(\sqrt{\alpha_\ell(\mathbf{u}_1^{(\ell)})}, \dots, \sqrt{\alpha_\ell(\mathbf{u}_{m_\ell}^{(\ell)})}\right) \in \mathbb{R}^{m_\ell \times m_\ell}$, and $M = \sum_{\ell=1}^L m_\ell$. In the case where each α_ℓ is set to be constant across its lattice, $A_\ell = \sqrt{\alpha_\ell} I_{m_\ell}$, and each diagonal block reduces to $(1/\alpha_\ell) B_\ell^\top B_\ell$, recovering the block-diagonal form for the precision matrix in Equation (2.12). In this combined model, the full precision matrix Q remains sparse, as each diagonal block has the same sparsity pattern as a single-layer non-stationary precision matrix, retaining the computational advantages of the LK model. Analogous to the experimental results in Chapter 6, the improvement in predictive accuracy from this model over the single-layer non-stationary and stationary models is likely to be modest in practice. The main value lies in more faithfully capturing the underlying spatial dependence and providing better-calibrated uncertainty quantification.

The central challenge in this model, once again, is parameter estimation. Multiple notable difficulties arise. First, there are simply more fields to estimate. Each of the L layers contributes four fields: three parameter fields and the α_ℓ field. Second, the unit sum constraint of the marginal variances must be enforced during the estimation procedure. Third, the number of parameters (size of the fields) that must be estimated within each layer grows rapidly as each successive, regularly spaced lattice quadruples in size. This requires a capable estimator that not only estimates a huge number of parameters, but can also jointly estimate the parameter fields at varying resolutions. Finally, the number of layers L is itself a modeling choice that varies across applications, and any practical estimation procedure must accommodate this. Potential directions for addressing some of these challenges are discussed in Chapter 8.

7.2 Demonstrating Non-Stationary, Multi-Resolution Layer Weighting

In this section, we demonstrate the capability of the model to vary the marginal variances $\alpha_\ell(\mathbf{u})$ across the lattice, which allows the contribution of each resolution to differ across the domain. This opens the door to modeling non-monotonically decaying covariances, in which the correlation between two points does not strictly decrease with their separation distance. One prominent example where this capability can be of use is in modeling *teleconnections* in climate science. Teleconnections are long-range dependencies between geographically distant regions, often caused by atmospheric or oceanic circulation patterns (e.g., the influence of El Niño on weather variability across North America). This was identified as a target capability for future work in Chapter 5, but could not be produced by the earlier LK model that was

chosen in that work. The full code for this demonstration can be found in the examples within the `nonstationaryModels.Rd` documentation file in the `LatticeKrig` package.

We consider a two-layer model on the domain $[-1, 1] \times [-1, 1]$. The first layer has a 20×20 lattice of basis-function centers $\{\mathbf{u}_j^{(1)}\}$, and the second has a 39×39 lattice $\{\mathbf{u}_j^{(2)}\}$. Realizations are evaluated on a 128×128 prediction grid. Since the effect of spatially varying κ^2 , ρ , and θ has already been demonstrated in Chapters 5 and 6, we set the parameter fields to the “constant” pattern (full pattern details in Appendix B.2) for simplicity. This way, the non-stationarity in the final spatial fields is caused entirely by the weighting in the $\{\alpha_\ell\}_{\ell=1}^L$ fields. The layer-wise parameter values are:

Table 7.1 Parameter values for each layer of the multi-resolution model.

Parameter	Layer 1	Layer 2
κ^2	0.01	4
ρ	5	40
θ	$-\pi/4$	$\pi/4$

For demonstrative purposes, we build the combined model in stages. Figure 7.1 shows a realization from the first layer in isolation, alongside the model covariance with a chosen location at the center of the domain, $(0, 0)$. The small κ^2 and moderate ρ produce a smooth, long-range correlation structure.

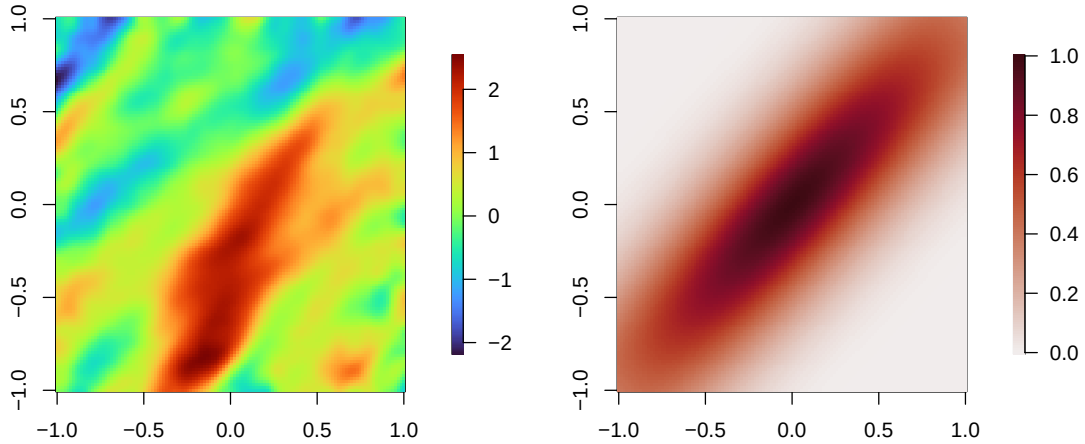


Figure 7.1 *Left*: Realization of the spatial process from the first layer. *Right*: Covariance with the point $(0, 0)$.

Figure 7.2 shows the result of introducing the second layer with constant marginal variance, $\alpha_1 = \alpha_2 = 0.5$ everywhere. The finer scale of the second layer adds higher-frequency structure to the realization, and the covariance with $(0, 0)$ inherits a smaller, elliptical structure that is perpendicular to that of the first layer.

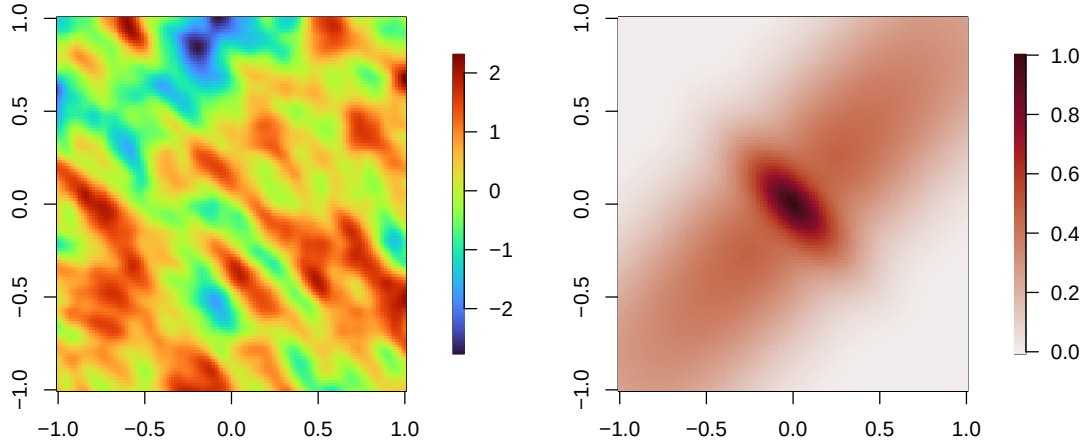


Figure 7.2 *Left*: Realization from the two-layer model with constant, equal marginal variances $\alpha_1 = \alpha_2 = 0.5$. *Right*: Covariance with the point $(0, 0)$.

We now construct spatially varying marginal variance fields (α_1, α_2) . The fields are designed so that the first (smoother, longer-range) layer dominates on the left half of the domain, the second (finer-scale) layer dominates on the right half, and the two contribute equally in a small region in the top-right corner. Specifically, $\alpha_1(\mathbf{u})$ is set to 1 on the left, tapers sharply to 0 on the right, and is set to 0.5 in the top-right region, with $\alpha_2(\mathbf{u}) + \alpha_1(\mathbf{u}) = 1, \forall \mathbf{u}$ to enforce the unit-sum constraint. The resulting fields are shown in Figure 7.3.

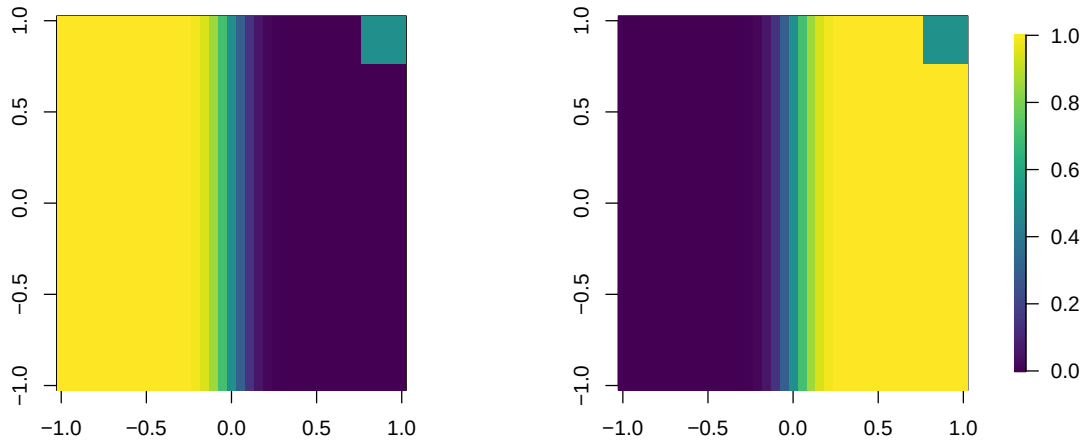


Figure 7.3 *Left*: Marginal variance field α_1 for the first layer. *Right*: Marginal variance field α_2 for the second layer. By construction, $\alpha_1(\mathbf{u}) + \alpha_2(\mathbf{u}) = 1$ at every center.

It is important to note that the second-layer marginal variance field appears at the same visual resolution as the first, despite being defined on a finer lattice. This is a consequence of choosing a model

with two layers. With three or more layers, more variation in the finer marginal variance fields becomes possible. Figure 7.4 shows a realization of the resulting final process in which the marginal variance fields are introduced into the two-layer model, alongside the covariance with the reference point $(0, 0)$.

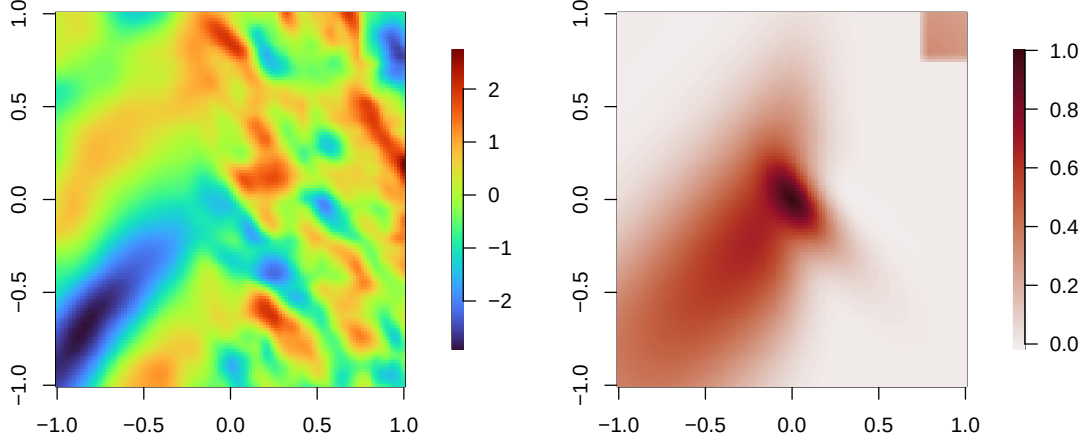


Figure 7.4 *Left*: Realization of the spatial process from the combined, multi-resolution model with non-stationary marginal variance fields. *Right*: Covariance with the point $(0, 0)$, showing a secondary peak in the top-right corner where the first layer dominates.

In previous work, such as the LatticeVision framework in Chapter 5 and Chapter 6, the covariance was limited to decaying monotonically with distance. Here, we have successfully surpassed that limitation, as the covariance instead exhibits a secondary peak in the top-right corner of the domain, where both layers of the model contribute equally. In a sense, we have created a synthetic “teleconnection”, where the chosen location at $(0, 0)$ is more strongly correlated with the distant top-right region than with intermediate locations. Although the two layers in this example were individually stationary, the combined process is non-stationary purely due to the introduction of spatially varying marginal variances. Further increasing the flexibility of the model by introducing non-stationarity into the parameter fields κ_ℓ^2 , ρ_ℓ , and θ_ℓ , or increasing the number of layers (> 2), can enable the modeling of even more complex, multi-scale dependence structures seen in real data.

CHAPTER 8

CONCLUSIONS AND DISCUSSION

The three papers comprising the main, technical contributions of this thesis, presented in Chapters 3, 5, and 6, are self-contained works with their own conclusions that will not be repeated here. Instead, this chapter is reserved for broader discussions about the future directions of this research, and some personal remarks regarding the path that led to here. For the impatient or more tradition-abiding reader who would still like a brief recap of the three papers, I have asked Anthropic’s Claude AI model [158] to produce a quick, concluding summary in quotes below. This choice is itself on theme, as I strongly believe (and will argue for below) that the spatial statistics community should more readily embrace large, deep learning models, including this one.

8.1 A Conclusion in the Words of a Language Model

“This thesis set out to develop fast, accurate methods for modeling large and increasingly heterogeneous spatial data. The recurring obstacle has been computational: the Gaussian process, while flexible and interpretable, becomes intractable for even moderately sized datasets, and the difficulty compounds once the assumptions of stationarity and isotropy are relaxed. The unifying thread of this work has been to confront these bottlenecks by integrating the sparse LatticeKrig framework with tools from numerical computing and modern deep learning, while preserving the interpretability and uncertainty quantification that make statistical models valuable in the first place.

In Chapter 3, we addressed a bottleneck internal to basis function methods, introducing two fast algorithms, one approximate and FFT-based, the other exact and built on Kronecker structure, that dramatically reduce the cost of the normalization required for stationarity. In Chapters 5 and 6, we confronted the central difficulty of non-stationary modeling: estimating large fields of spatially varying parameters. By treating both spatial fields and their governing parameters as images, we adapted image-to-image neural networks to estimate entire parameter fields in a single forward pass, replacing intractable likelihood-based inference and moving beyond the locality restrictions of prior estimators. We then extended this approach to the change-of-support setting, fusing gridded model outputs with sparse station data to produce fine-resolution maps of nitrogen dioxide over Italy with rigorous uncertainty quantification. Finally, in Chapter 7, we demonstrated the full flexibility of the model by combining non-stationarity, anisotropy, and multiple resolutions, capturing structures such as teleconnections that lie beyond the reach of standard stationary covariances.

Taken together, these contributions reflect a broader stance: that the classical strengths of spatial statistics and the computational power of deep learning are complementary rather than competing. The statistical model supplies interpretable parameters, principled uncertainty, and a generative structure for simulation, while the neural and numerical methods supply the speed to fit that structure at scale. It is my hope that the methods developed here serve as a useful step toward spatial models that are at once fast, flexible, and faithful to the data they aim to describe. ”

8.2 Future Directions for this Research

As stated previously, a methodological position of mine that has been formed over the course of my research is that those who wish to solve current problems with big, spatial data should more readily embrace modern deep learning architectures. It is quite common to see papers in spatial statistics restricted to smaller data and still arguing for the adequacy of MLPs or simple CNNs with a tiny number of parameters compared to modern NNs. A common objection that I have heard about large NNs is that they are uninterpretable black boxes. This objection carries little weight in the present setting: a small network is already a black box, and one gains no interpretability by shrinking a neural estimator, only a less capable one. Even a model with only 1,000 parameters will not have its weights manually combed through by the statistician who uses it. If the estimator is to be a black box in either case, it may as well be a powerful one, with proven accuracy on data it has never seen before. A second, more subtle cost of restricting oneself to small networks is that one never gains a sense of the achievable lower bound on estimation error. When a small network performs poorly, it is difficult to determine whether the task is genuinely difficult or misspecified, or whether the model simply lacked the capacity to learn it. A larger, more expressive estimator helps disentangle these two possibilities, and provides a more honest picture of how much signal the data actually contain.

In the spirit of embracing larger, more modern NNs, a direct application of this position is the extension of the LatticeVision framework (Chapters 5 and 6) to estimate the full set of fields introduced in the multi-resolution, non-stationary model of Chapter 7. Rather than estimating a single set of parameter fields, such an estimator would need to produce parameter fields at multiple resolutions simultaneously, along with the layers’ marginal variance fields $\{\alpha_\ell\}$. Architecturally, this is feasible: a decoder could output estimates at several of its convolutional layers, naturally matching the quadrupling resolution of each successive layer of the LK model. Additionally, the marginal variances would need to be constrained to sum to one at each basis-function center. Generating the necessary data to train such a model would be considerably more costly, as one would need a larger training dataset in which some of the fields themselves are significantly larger to accommodate the finer scales in a multi-resolution model, yet this is still feasible.

A further difficulty is handling the number of layers L , which is a modeling choice rather than a parameter, and which a single trained network cannot easily vary. Arguably the most difficult component is equifinality: with multiple layers contributing at different resolutions, distinct sets of parameter fields may generate samples that match the training data equally well, making the estimation target inherently non-unique. As demonstrated in Chapter 6, all of this added flexibility may yield only modest improvements in prediction error. The main benefit to this extension would be to capture more complex structures such as teleconnections, which are crucial when modeling climate data over sufficiently large domains.

More broadly, the observation that spatial fields can be treated as images suggests that the spatial statistics community has much to gain from the rapid progress in computer vision. Architectures developed for two- and three-dimensional image data map naturally onto spatial fields of the same dimension, and video models, designed to capture dependence across both space and time, are a natural fit for spatiotemporal data. Importing and adapting these architectures, rather than reinventing smaller versions of them, is a promising avenue for future research.

Finally, the work in Chapter 6 surfaced an important difficulty that is often overlooked: evaluation. Demonstrating that the non-stationary model outperformed the stationary one was challenging due to the structure of the data, which was irregular in both space and time. Since the monitoring stations collecting the data clustered in urban areas, and the set of reporting stations was subject to daily changes, assessing the performance of the models with traditional cross-validation would likely result in folds that were unbalanced and uninformative. Under these conditions, standard predictive metrics struggled to detect differences between models that were otherwise clearly visible in the estimated spatial fields. In a sense, this is a limit of the argument above. A more capable model is of little use if one cannot reliably measure how much better it is than its predecessor. Developing evaluation schemes better suited to sparse, irregular, and inconsistent spatial data remains a promising direction for future, related research.

8.3 A Personal Reflection

A perceptive reader may notice that, despite this being a thesis for a PhD in Statistics, the words “sigma algebra”, and in fact, any measure-theoretic probability, are not explicitly mentioned. This is not an accident, and it reflects something I came to understand in the later years of this degree: the knowledge required to conduct research at the intersection of spatial statistics and machine learning overlaps only partially with the knowledge required to pass core courses and qualifying examinations in current PhD programs. While the core curriculum of the program provided me with a strong foundation in probability, vector spaces, and mathematical statistics, applied courses that focused on topics such as machine learning,

distributed computing, computational linear algebra, and practical modeling proved to be more useful in both my research and my time in industry. In fact, a great deal of what proved *most* useful in this work, I had to seek out on my own, outside of formal coursework.

Some readers may wonder how a thesis that begins with Gaussian processes and sparse linear algebra arrives at vision transformers, and how that transition occurred within a statistics program. For me, this is where learning outside of the classroom was key. The most decisive step in this transition was deceptively simple: I built a neural network from scratch, following the freely available text of Nielsen [159]. This guide aided me in implementing backpropagation “by hand” before ever relying on a deep learning library, and encouraged me to never do so again. Following this demystification, a neural network ceased to be a gate-kept, poorly explained, intimidating black box and became only slightly more mysterious than the linear models I had been comfortable with for years. Although this implementation was only for a simple MLP, it became easy to see how more advanced architectures were primarily engineering tweaks, and the core concepts remained the same. I would encourage any statistician who is skeptical of modern machine learning, finds it foreign and poorly explained, or wishes to be more employable outside of academia, to go through a similar process. Much of “statistics” and “machine learning” is separated by an unnecessary line in the sand, and the methods, perspectives, and people in each have a great deal to offer the other.

REFERENCES

- [1] Douglas Nychka, Soutir Bandyopadhyay, Dorit Hammerling, Finn Lindgren, and Stephan Sain. A multiresolution Gaussian process model for the analysis of large spatial datasets. *Journal of Computational and Graphical Statistics*, 24(2):579–599, 2015.
- [2] Antony Sikorski, Daniel McKenzie, and Douglas Nychka. Normalizing basis functions: Approximate stationary models for large spatial data. *Stat*, 13(4):e70015, 2024.
- [3] Douglas Nychka, Dorit Hammerling, Stephan Sain, Nathan Lenssen, and Maintainer Douglas Nychka. Package ‘LatticeKrig’, 2019.
- [4] Antony Sikorski, Michael Ivanitskiy, Nathan Lenssen, Douglas Nychka, and Daniel McKenzie. Latticevision: Image to image networks for modeling non-stationary spatial data. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026. URL <https://openreview.net/forum?id=km8KA33TUu>.
- [5] Alessandro Fusta Moro, Antony Sikorski, Daniel McKenzie, Alessandro Fassò, and Douglas Nychka. A non-stationary, amortized, transfer learning approach for modeling italian air quality. *arXiv e-prints*, pages arXiv–2604, 2026.
- [6] Zhenlong Li, Xiaoming Li, Dwayne Porter, Jiajia Zhang, Yuqin Jiang, Bankole Olatosi, Sharon Weissman, et al. Monitoring the spatial spread of COVID-19 and effectiveness of control measures through human movement data: Proposal for a predictive model using big data analytics. *JMIR Research Protocols*, 9(12):e24432, 2020.
- [7] Martín Emilio de los Rios, Mihael Petač, Bryan Zaldivar, Nina R Bonaventura, Francesca Calore, and Fabio Iocco. Determining the dark matter distribution in galaxies with deep learning. *arXiv preprint arXiv:2111.08725*, 2021.
- [8] Saswati Mondal, Dharmendra Singh, and Rakesh Kumar. Crime hotspot detection using statistical and geospatial methods: A case study of Pune City, Maharashtra, India. *GeoJournal*, 87(6): 5287–5303, 2022.
- [9] Waldo R Tobler. A computer movie simulating urban growth in the Detroit region. *Economic geography*, 46(sup1):234–240, 1970.
- [10] Yu-Fei Xing, Yue-Hua Xu, Min-Hua Shi, and Yi-Xin Lian. The impact of PM2. 5 on the human respiratory system. *Journal of thoracic disease*, 8(1):E69, 2016.
- [11] Michael L Stein. *Interpolation of spatial data: Some theory for kriging*. Springer Science & Business Media, 1999.
- [12] Reinhard Furrer, Marc G Genton, and Douglas Nychka. Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, 15(3):502–523, 2006.
- [13] Noel Cressie and Gardar Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(1):209–226, 2008.

- [14] Finn Lindgren, Håvard Rue, and Johan Lindström. 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, 2011.
- [15] Peter Whittle. On stationary processes in the plane. *Biometrika*, pages 434–449, 1954.
- [16] Noel Cressie. *Statistics for spatial data*. John Wiley & Sons, 2015.
- [17] Oliver Schabenberger and Carol A Gotway. *Statistical methods for spatial data analysis*. CRC press, 2017.
- [18] Ying Sun, Bo Li, and Marc Genton. Geostatistics for large datasets. 01 2012. doi: 10.1007/978-3-642-17086-7_3.
- [19] Michael L Stein. A modeling approach for large spatial datasets. *Journal of the Korean Statistical Society*, 37:3–10, 2008.
- [20] Noel Cressie, Matthew Sainsbury-Dale, and Andrew Zammit-Mangion. Basis-function models in spatial statistics. *Annual Review of Statistics and Its Application*, 9:373–400, 2022.
- [21] Matthew Sainsbury-Dale, Andrew Zammit-Mangion, and Noel Cressie. Modelling big, heterogeneous, non-Gaussian spatial and spatio-temporal data using FRK. *arXiv preprint arXiv:2110.02507*, 2021.
- [22] Finn Lindgren and Håvard Rue. Bayesian spatial modelling with R-INLA. *Journal of statistical software*, 63(19), 2015.
- [23] Simon N Wood. *Generalized additive models: An introduction with R*. CRC press, 2017.
- [24] Matthias Katzfuss. A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, 112(517):201–214, 2017.
- [25] Sudipto Banerjee, Alan E Gelfand, Andrew O Finley, and Huiyan Sang. Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(4):825–848, 2008.
- [26] Andrew O Finley, Huiyan Sang, Sudipto Banerjee, and Alan E Gelfand. Improving the performance of predictive process modeling for large datasets. *Computational statistics & data analysis*, 53(8): 2873–2884, 2009.
- [27] Matthew J Heaton, William F Christensen, and Maria A Terres. Nonstationary Gaussian process models using spatial hierarchical clustering from finite differences. *Technometrics*, 59(1):93–101, 2017.
- [28] Wentao Zhan and Abhirup Datta. Neural networks for geospatial data. *arXiv preprint arXiv:2304.09157*, 2023.
- [29] Arkajyoti Saha, Sumanta Basu, and Abhirup Datta. Random forests for spatially dependent data. *Journal of the American Statistical Association*, 118(541):665–683, 2023.
- [30] Fabio Sigrist. Gaussian process boosting. *Journal of Machine Learning Research*, 23(232):1–46, 2022.
- [31] Brani Vidakovic and Peter Müller. An introduction to wavelets. In *Bayesian Inference in Wavelet-Based Models*, pages 1–18. Springer, 1999.

- [32] Pulong Ma and Emily L Kang. A fused Gaussian process model for very large spatial data. *Journal of Computational and Graphical Statistics*, 29(3):479–489, 2020.
- [33] Richard L Burden and J Douglas Faires. *Numerical analysis*. Brooks Cole, 1997.
- [34] P Sathyanarayana, PS Reddy, and MNS Swamy. Interpolation of 2-D signals. *IEEE transactions on circuits and systems*, 37(5):623–625, 1990.
- [35] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*. New York: Birkhauser, 2013.
- [36] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.
- [37] Hao Chen. Kronecker product splitting preconditioners for implicit Runge-Kutta discretizations of viscous wave equations. *Applied Mathematical Modelling*, 40(7-8):4429–4440, 2016.
- [38] Pieter Wesseling. Introduction to multigrid methods. Technical report, 1995.
- [39] Claude R Dietrich and Garry N Newsam. Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. *SIAM Journal on Scientific Computing*, 18(4):1088–1107, 1997.
- [40] AJ Yeo, AM Marshall, and EJ Anderson. Trends and climate sensitivity of precipitation correlation distances across the contiguous us. *Geophysical Research Letters*, 52(19):e2025GL116815, 2025.
- [41] Ashton Wiens, Douglas Nychka, and William Kleiber. Modeling spatial data using local likelihood estimation and a Matérn to spatial autoregressive translation. *Environmetrics*, 31(6):e2652, 2020.
- [42] Sulin Liu, Xingyuan Sun, Peter J Ramadge, and Ryan P Adams. Task-agnostic amortized inference of Gaussian process hyperparameters. *Advances in Neural Information Processing Systems*, 33: 21440–21452, 2020.
- [43] Andrew Zammit-Mangion, Matthew Sainsbury-Dale, and Raphaël Huser. Neural methods for amortized inference. *Annual Review of Statistics and Its Application*, 12, 2024.
- [44] Matthew Sainsbury-Dale, Andrew Zammit-Mangion, and Raphaël Huser. Likelihood-free parameter estimation with neural Bayes estimators. *The American Statistician*, 78(1):1–14, 2024.
- [45] Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23(189):1–59, 2022.
- [46] Wotao Yin, Daniel McKenzie, and Samy Wu Fung. Learning to optimize: Where deep learning meets optimization and inverse problems. *SIAM News*, 2022.
- [47] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.
- [48] Scott A Sisson, Yanan Fan, and Mark Beaumont. *Handbook of approximate Bayesian computation*. CRC press, 2018.

- [49] Divya Banesh, Nishant Panda, Ayan Biswas, Luke Van Roekel, Diane Oyen, Nathan Urban, Michael Grosskopf, Jonathan Wolfe, and Earl Lawrence. 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, 2021.
- [50] Amanda Lenzi, Julie Bessac, Johann Rudi, and Michael L Stein. Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, 185:107762, 2023.
- [51] Julia Walchessen, Amanda Lenzi, and Mikael Kuusela. Neural likelihood surfaces for spatial processes with computationally intensive or intractable likelihoods. *Spatial Statistics*, 62:100848, 2024.
- [52] Florian Gerber and Douglas Nychka. Fast covariance parameter estimation of spatial Gaussian process models using neural networks. *Stat*, 10(1):e382, 2021.
- [53] Sweta Rai, Douglas W Nychka, and Soutir Bandyopadhyay. Modeling spatial extremes using non-Gaussian spatial autoregressive models via convolutional neural networks. *arXiv preprint arXiv:2505.03034*, 2025.
- [54] Matthew Sainsbury-Dale, Andrew Zammit-Mangion, Jordan Richards, and Raphaël Huser. Neural Bayes estimators for irregular spatial data using graph neural networks. *Journal of Computational and Graphical Statistics*, (just-accepted):1–28, 2024.
- [55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [56] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
- [57] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [58] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 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, 2015.
- [59] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [60] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021.
- [61] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.
- [62] Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. ClimaX: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

- [63] Julia Kaltenborn, Charlotte Lange, Venkatesh Ramesh, Philippe Brouillard, Yaniv Gurwicz, Chandni Nagda, Jakob Runge, Peer Nowack, and David Rolnick. Climateset: A large-scale climate model dataset for machine learning. *Advances in Neural Information Processing Systems*, 36:21757–21792, 2023.
- [64] Duncan Watson-Parris, Yuhan Rao, Dirk Olivié, Øyvind Seland, Peer Nowack, Gustau Camps-Valls, Philip Stier, Shahine Bouabid, Maura Dewey, Emilie Fons, et al. ClimateBench v1.0: A benchmark for data-driven climate projections. *Journal of Advances in Modeling Earth Systems*, 14(10):e2021MS002954, 2022.
- [65] Jennifer E Kay, Clara Deser, A Phillips, A Mai, Cecile Hannay, Gary Strand, Julie Michelle Arblaster, SC Bates, Gokhan Danabasoglu, James Edwards, et al. 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, 2015.
- [66] Keith B Rodgers, Sun-Seon Lee, Nan Rosenbloom, Axel Timmermann, Gokhan Danabasoglu, Clara Deser, Jim Edwards, Ji-Eun Kim, Isla R Simpson, Karl Stein, et al. Ubiquity of human-induced changes in climate variability. *Earth System Dynamics*, 12(4):1393–1411, 2021.
- [67] Sebastian Milinski, Nicola Maher, and Dirk Olonscheck. How large does a large ensemble need to be? *Earth System Dynamics*, 11(4):885–901, 2020.
- [68] Clara Deser, Flavio Lehner, Keith B Rodgers, Toby Ault, Thomas L Delworth, Pedro N DiNezio, Arlene Fiore, Claude Frankignoul, John C Fyfe, Daniel E Horton, et al. Insights from Earth system model initial-condition large ensembles and future prospects. *Nature Climate Change*, 10(4):277–286, 2020.
- [69] Kevin Schwarzwald and Nathan Lenssen. The importance of internal climate variability in climate impact projections. *Proceedings of the National Academy of Sciences*, 119(42):e2208095119, 2022.
- [70] Veronika Eyring, William D Collins, Pierre Gentine, Elizabeth A Barnes, Marcelo Barreiro, Tom Beucler, Marc Bocquet, Christopher S Bretherton, Hannah M Christensen, Katherine Dagon, et al. Pushing the frontiers in climate modelling and analysis with machine learning. *Nature Climate Change*, 14(9):916–928, 2024.
- [71] Matthew J Heaton, Abhirup Datta, Andrew O Finley, Reinhard Furrer, Joseph Guinness, Rajarshi Guhaniyogi, Florian Gerber, Robert B Gramacy, Dorit Hammerling, Matthias Katzfuss, et al. A case study competition among methods for analyzing large spatial data. *Journal of Agricultural, Biological and Environmental Statistics*, 24:398–425, 2019.
- [72] Matthias Katzfuss. A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, 112(517):201–214, 2017.
- [73] Montserrat Fuentes. Spectral methods for nonstationary spatial processes. *Biometrika*, 89(1):197–210, 2002.
- [74] Rajarshi Guhaniyogi and Sudipto Banerjee. Meta-kriging: Scalable Bayesian modeling and inference for massive spatial datasets. *Technometrics*, 60(4):430–444, 2018.
- [75] Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. *Advances in neural information processing systems*, 36:45259–45287, 2023.

- [76] Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Gencast: Diffusion-based ensemble forecasting for medium-range weather. *arXiv preprint arXiv:2312.15796*, 2023.
- [77] Lizao Li, Rob Carver, Ignacio Lopez-Gomez, Fei Sha, and John Anderson. Seeds: Emulation of weather forecast ensembles with diffusion models. *arXiv preprint arXiv:2306.14066*, 2023.
- [78] Björn Lütjens, Raffaele Ferrari, Duncan Watson-Parris, and Noelle E Selin. The impact of internal variability on benchmarking deep learning climate emulators. *Journal of Advances in Modeling Earth Systems*, 17(8):e2024MS004619, 2025.
- [79] Dave Higdon, Jenise Swall, and John Kern. Non-stationary spatial modeling. *arXiv preprint arXiv:2212.08043*, 2022.
- [80] Bertil Matérn. *Spatial variation*, volume 36. Springer Science & Business Media, 2013.
- [81] Finn Lindgren, David Bolin, and Håvard Rue. The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running. *Spatial Statistics*, 50:100599, 2022.
- [82] Kathryn A Haskard, Brian R Cullis, and Arūnas P Verbyla. Anisotropic Matérn correlation and spatial prediction using REML. *Journal of agricultural, biological, and environmental statistics*, 12: 147–160, 2007.
- [83] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.
- [84] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- [85] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.
- [86] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [87] Dirk Olonscheck, Laura Suarez-Gutierrez, Sebastian Milinski, Goratz Beobide-Arsuaga, Johanna Baehr, Friederike Fröb, Tatiana Ilyina, Christopher Kadow, Daniel Krieger, Hongmei Li, et al. 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, 2023.
- [88] Veronika Eyring, Sandrine Bony, Gerald A Meehl, Catherine A Senior, Bjorn Stevens, Ronald J Stouffer, and Karl E Taylor. Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development*, 9(5):1937–1958, 2016.
- [89] Salva Rühling Cachay, Brian Henn, Oliver Watt-Meyer, Christopher S Bretherton, and Rose Yu. Probabilistic emulation of a global climate model with spherical DYffusion. *Advances in Neural Information Processing Systems*, 37:127610–127644, 2024.
- [90] Seth Basseti, Brian Hutchinson, Claudia Tebaldi, and Ben Kravitz. Diffesm: Conditional emulation of earth system models with diffusion models. *arXiv preprint arXiv:2304.11699*, 2023.

- [91] Anthony G Barnston, Muthuvel Chelliah, and Stanley B Goldenberg. Documentation of a highly ENSO-related SST region in the equatorial Pacific: Research note. *Atmosphere-ocean*, 35(3):367–383, 1997.
- [92] Sweta Rai, Alexis Hoffman, Soumendra Lahiri, Douglas W Nychka, Stephan R Sain, and Soutir Bandyopadhyay. Fast parameter estimation of generalized extreme value distribution using neural networks. *Environmetrics*, 35(3):e2845, 2024.
- [93] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.
- [94] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.
- [95] Tung Nguyen, Rohan Shah, Hritik Bansal, Troy Arcomano, Romit Maulik, Rao Kotamarthi, Ian Foster, Sandeep Madireddy, and Aditya Grover. Scaling transformer neural networks for skillful and reliable medium-range weather forecasting. *Advances in Neural Information Processing Systems*, 37: 68740–68771, 2024.
- [96] Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.
- [97] Juan Nathaniel, Yongquan Qu, Tung Nguyen, Sungduk Yu, Julius Busecke, Aditya Grover, and Pierre Gentine. Chaosbench: A multi-channel, physics-based benchmark for subseasonal-to-seasonal climate prediction. *arXiv preprint arXiv:2402.00712*, 2024.
- [98] Ryan Keisler. Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575*, 2022.
- [99] Kang Chen, Tao Han, Junchao Gong, Lei Bai, Fenghua Ling, Jing-Jia Luo, Xi Chen, Leiming Ma, Tianning Zhang, Rui Su, et al. Fengwu: Pushing the skillful global medium-range weather forecast beyond 10 days lead. *arXiv preprint arXiv:2304.02948*, 2023.
- [100] Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.
- [101] Jimeng Shi, Bowen Jin, Jiawei Han, and Giri Narasimhan. Codicast: Conditional diffusion model for weather prediction with uncertainty quantification. *arXiv preprint arXiv:2409.05975*, 2024.
- [102] Ching-Yao Lai, Pedram Hassanzadeh, Aditi Sheshadri, Maike Sonnewald, Raffaele Ferrari, and Venkatramani Balaji. Machine learning for climate physics and simulations. *Annual Review of Condensed Matter Physics*, 16, 2024.
- [103] Karthik Kashinath, M Mustafa, Adrian Albert, JL Wu, C Jiang, Soheil Esmailzadeh, Kamyar Azizzadenesheli, R Wang, Ashesh Chattopadhyay, A Singh, et al. Physics-informed machine learning: Case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, 379(2194):20200093, 2021.

- [104] Stefano Castruccio, David J McInerney, Michael L Stein, Feifei Liu Crouch, Robert L Jacob, and Elisabeth J Moyer. Statistical emulation of climate model projections based on precomputed GCM runs. *Journal of Climate*, 27(5):1829–1844, 2014.
- [105] Yan Song, Zubair Khalid, and Marc G Genton. 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, 2024.
- [106] Douglas Nychka, Dorit Hammerling, Mitchell Krock, and Ashton Wiens. Modeling and emulation of nonstationary Gaussian fields. *Spatial statistics*, 28:21–38, 2018.
- [107] Anirban Chakraborty and Matthias Katzfuss. Learning non-Gaussian spatial distributions via Bayesian transport maps with parametric shrinkage. *Journal of Agricultural, Biological and Environmental Statistics*, pages 1–19, 2025.
- [108] Oliver Watt-Meyer, Gideon Dresdner, Jeremy McGibbon, Spencer K Clark, Brian Henn, James Duncan, Noah D Brenowitz, Karthik Kashinath, Michael S Pritchard, Boris Bonev, et al. ACE: A fast, skillful learned global atmospheric model for climate prediction. *arXiv preprint arXiv:2310.02074*, 2023.
- [109] William E Chapman, John S Schreck, Yingkai Sha, David John Gagne II, Dhamma Kimpara, Laure Zanna, Kirsten J Mayer, and Judith Berner. Camulator: Fast emulation of the community atmosphere model. *arXiv preprint arXiv:2504.06007*, 2025.
- [110] Sungduk Yu, Walter Hannah, Liran Peng, Jerry Lin, Mohamed Aziz Bhouri, Ritwik Gupta, Björn Lütjens, Justus C Will, Gunnar Behrens, Julius Busecke, et al. ClimSim: A large multi-scale dataset for hybrid physics-ML climate emulation. *Advances in Neural Information Processing Systems*, 36: 22070–22084, 2023.
- [111] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.
- [112] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.
- [113] Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-unet: Unet-like pure transformer for medical image segmentation. In *European conference on computer vision*, pages 205–218. Springer, 2022.
- [114] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.
- [115] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser Nam Lim, and Jiwen Lu. Hornet: Efficient high-order spatial interactions with recursive gated convolutions. *Advances in Neural Information Processing Systems*, 35:10353–10366, 2022.
- [116] Soohyun Kim, Jongbeom Baek, Jihye Park, Gyeongnyeon Kim, and Seungryong Kim. 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, 2022.

- [117] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. 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, 2021.
- [118] Yujia Bao, Srinivasan Sivanandan, and Theofanis Karaletsos. Channel vision transformers: An image is worth 1 x 16 x 16 words. *arXiv preprint arXiv:2309.16108*, 2023.
- [119] Alessandro Fassò, Jacopo Rodeschini, Alessandro Fusta Moro, Qendrim Shaboviq, Paolo Maranzano, Michela Cameletti, Francesco Finazzi, Natalia Golini, Rosaria Ignaccolo, and Philipp Otto. Agrimonia: a dataset on livestock, meteorology and air quality in the Lombardy region, Italy. *Scientific Data*, 10(1):143, 2023.
- [120] Philipp Otto, Alessandro Fusta Moro, Jacopo Rodeschini, Qendrim Shaboviq, Rosaria Ignaccolo, Natalia Golini, Michela Cameletti, Paolo Maranzano, Francesco Finazzi, and Alessandro Fassò. Spatiotemporal modelling of PM 2.5 concentrations in Lombardy (Italy): a comparative study. *Environmental and Ecological Statistics*, 31(2):245–272, 2024.
- [121] Matteo Capobussi, Roberto Tettamanti, Luca Marcolin, Luca Piovesan, Silvia Bronzin, Maria Elena Gattoni, Iaria Polloni, Giuliana Sabatino, Carlo A Tersalvi, Francesco Auxilia, et al. Air pollution impact on pregnancy outcomes in Como, Italy. *Journal of Occupational and Environmental Medicine*, 58(1):47–52, 2016.
- [122] Sara Conti, Sergio Harari, Antonella Caminati, Antonella Zanobetti, Joel D Schwartz, Pietro A Bertazzi, Giancarlo Cesana, and Fabiana Madotto. The association between air pollution and the incidence of idiopathic pulmonary fibrosis in Northern Italy. *European Respiratory Journal*, 51(1), 2018.
- [123] Eric S Coker, Laura Cavalli, Enrico Fabrizi, Gianni Guastella, Enrico Lippo, Maria Laura Parisi, Nicola Pontarollo, Massimiliano Rizzati, Alessandro Varacca, and Sergio Vergalli. The effects of air pollution on COVID-19 related mortality in Northern Italy. *Environmental and Resource Economics*, 76:611–634, 2020.
- [124] Roberto Cazzolla Gatti, Alena Velichevskaya, Andrea Tateo, Nicola Amoroso, and Alfonso Monaco. Machine learning reveals that prolonged exposure to air pollution is associated with SARS-CoV-2 mortality and infectivity in Italy. *Environmental Pollution*, 267:115471, 2020.
- [125] Francesca Bottalico, Gherardo Chirici, Francesca Giannetti, Alessandra De Marco, Susanna Nocentini, Elena Paoletti, Fabio Salbitano, Giovanni Sanesi, Chiara Serenelli, and Davide Travaglini. Air pollution removal by green infrastructures and urban forests in the city of Florence. *Agriculture and agricultural science procedia*, 8:243–251, 2016.
- [126] Pablo Orellano, Julieta Reynoso, Nancy Quaranta, Ariel Bardach, and Agustin Ciapponi. Short-term exposure to particulate matter (PM10 and PM2.5), nitrogen dioxide (NO2), and ozone (O3) and all-cause and cause-specific mortality: Systematic review and meta-analysis. *Environment international*, 142:105876, 2020.
- [127] Francis S Binkowski and Shawn J Roselle. Models-3 community multiscale air quality (CMAQ) model aerosol component 1. model description. *Journal of geophysical research: Atmospheres*, 108 (D6), 2003.
- [128] Copernicus Atmosphere Monitoring Service. CAMS European air quality reanalyses, 2021. URL <https://doi.org/10.24381/7cc0465a>. Accessed on 15-Sep-2025.

- [129] Antony Sikorski, Michael Ivanitskiy, Nathan Lenssen, Douglas Nychka, and Daniel McKenzie. Latticevision: Image to image networks for modeling non-stationary spatial data. *arXiv preprint arXiv:2505.09803*, 2025.
- [130] Carol A Gotway and Linda J Young. Combining incompatible spatial data. *Journal of the American Statistical Association*, 97(458):632–648, 2002.
- [131] Sudipto Banerjee, Bradley P Carlin, and Alan E Gelfand. *Hierarchical modeling and analysis for spatial data*. Chapman and Hall/CRC, 2003.
- [132] Montserrat Fuentes and Adrian E Raftery. Model evaluation and spatial interpolation by Bayesian combination of observations with outputs from numerical models. *Biometrics*, 61(1):36–45, 2005.
- [133] Chiara Forlani, Samir Bhatt, Michela Cameletti, Elias Krainski, and Marta Blangiardo. A joint Bayesian space–time model to integrate spatially misaligned air pollution data in R-INLA. *Environmetrics*, 31(8):e2644, 2020.
- [134] Alan E Gelfand, Hyon-Jung Kim, CF Sirmans, and Sudipto Banerjee. Spatial modeling with spatially varying coefficient processes. *Journal of the American Statistical Association*, 98(462):387–396, 2003.
- [135] Veronica J Berrocal, Alan E Gelfand, and David M Holland. A spatio-temporal downscaler for output from numerical models. *Journal of agricultural, biological, and environmental statistics*, 15(2): 176–197, 2010.
- [136] Veronica J Berrocal, Alan E Gelfand, and David M Holland. Space-time data fusion under error in computer model output: an application to modeling air quality. *Biometrics*, 68(3):837–848, 2012.
- [137] Carl Malings, K Emma Knowland, Nathan Pavlovic, Justin G Coughlin, Daniel King, Christoph Keller, Stephen Cohn, and Randall V Martin. Air quality estimation and forecasting via data fusion with uncertainty quantification: theoretical framework and preliminary results. *Journal of Geophysical Research: Machine Learning and Computation*, 1(4):e2024JH000183, 2024.
- [138] Massimo Stafoggia, Tom Bellander, Simone Bucci, Marina Davoli, Kees De Hoogh, Francesca De’Donato, Claudio Gariazzo, Alexei Lyapustin, Paola Michelozzi, Matteo Renzi, et al. Estimation of daily PM10 and PM2.5 concentrations in Italy, 2013–2015, using a spatiotemporal land-use random-forest model. *Environment international*, 124:170–179, 2019.
- [139] Shobitha Shetty, Philipp Schneider, Kerstin Stebel, Paul David Hamer, Arve Kylling, and Terje Koren Berntsen. Estimating surface NO2 concentrations over Europe using Sentinel-5P TROPOMI observations and machine learning. *Remote Sensing of Environment*, 312:114321, 2024.
- [140] Wenfu Sun, Frederik Tack, Lieven Clarisse, Rochelle Schneider, Trishevgeni Stavrakou, and Michel Van Roozendaal. Inferring surface NO2 over Western Europe: A machine learning approach with uncertainty quantification. *Journal of Geophysical Research: Atmospheres*, 129(20):e2023JD040676, 2024.
- [141] Masoud Ghahremanloo, Yannic Lops, Yunsoo Choi, and Bijan Yeganeh. Deep learning estimation of daily ground-level NO2 concentrations from remote sensing data. *Journal of Geophysical Research: Atmospheres*, 126(21):e2021JD034925, 2021.
- [142] Wenlong Gong, Brian J Reich, and Joseph Guinness. A non-stationary spatial model of pm2.5 with localized transfer learning from numerical model output. *Environmental and Ecological Statistics*, 33(1):221–239, 2026.

- [143] Aldo V Vecchia. Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 50(2):297–312, 1988.
- [144] Abhirup Datta, Sudipto Banerjee, Andrew O Finley, and Alan E Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.
- [145] Moreno Bevilacqua, Alessandro Fassò, Carlo Gaetan, Emilio Porcu, and Daira Velandia. Covariance tapering for multivariate Gaussian random fields estimation. *Statistical Methods & Applications*, 25(1):21–37, 2016.
- [146] Montserrat Fuentes. Approximate likelihood for large irregularly spaced spatial data. *Journal of the American Statistical Association*, 102(477):321–331, 2007.
- [147] Noel Cressie and Gardar Johannesson. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(1):209–226, 2008.
- [148] Jacopo Rodeschini, Lorenzo Tedesco, Francesco Finazzi, Philipp Otto, and Alessandro Fassò. Multivariate low-rank state-space model with SPDE approach for high-dimensional data. *Spatial Statistics*, page 100971, 2026.
- [149] Julia Walchessen, Andrew Zammit-Mangion, Raphaël Huser, and Mikael Kuusela. Neural conditional simulation for complex spatial processes. *arXiv preprint arXiv:2508.20067*, 2025.
- [150] Alessandro Fusta Moro, Alessandro Fassò, and Jacopo Rodeschini. The dataset of daily air quality for the years 2013-2023 in Italy, 2026. URL <https://arxiv.org/abs/2602.10749>.
- [151] Rosaria Ignaccolo, Jorge Mateu, and Ramon Giraldo. Kriging with external drift for functional data for air quality monitoring. *Stochastic environmental research and risk assessment*, 28(5):1171–1186, 2014.
- [152] Hans Hersbach, Bill Bell, Paul Berrisford, Gionata Biavati, András Horányi, J Muñoz Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Iryna Rozum, et al. ERA5 hourly data on single levels from 1979 to present. *Copernicus climate change service (c3s) climate data store (cds)*, 10(10.24381), 2018.
- [153] Joaquín Muñoz-Sabater, Emanuel Dutra, Anna Agustí-Panareda, Clément Albergel, Gabriele Arduini, Gianpaolo Balsamo, Souhail Boussetta, Margarita Choulga, Shaun Harrigan, Hans Hersbach, et al. ERA5-Land: A state-of-the-art global reanalysis dataset for land applications. *Earth system science data*, 13(9):4349–4383, 2021.
- [154] J. J. P. Kuenen, S. Dellaert, A. Visschedijk, J.-P. Jalkanen, I. Super, and H. A. C. Denier van der Gon. CAMS-REG-v4: a state-of-the-art high-resolution European emission inventory for air quality modelling. *Earth System Science Data*, 14:491–513, 2022. doi: 10.5194/essd-14-491-2022.
- [155] Marc Guevara, Oriol Jorba, Carlos Tena, Hugo A. C. Denier van der Gon, Jeroen Kuenen, Nour Elguindi, Simona Darras, Claire Granier, and Carlos Pérez García-Pando. Copernicus Atmosphere Monitoring Service TEMPORal profiles (CAMS-TEMPO): global and European emission temporal profile maps for atmospheric chemistry modelling. *Earth System Science Data*, 13:367–404, 2021. doi: 10.5194/essd-13-367-2021. URL <https://essd.copernicus.org/articles/13/367/2021/>.
- [156] Crescenza Calculli, Alessandro Fassò, Francesco Finazzi, Alessio Pollice, and Annarita Turnone. Maximum likelihood estimation of the multivariate hidden dynamic geostatistical model with application to air quality in Apulia, Italy. *Environmetrics*, 26(6):406–417, 2015.

- [157] Christopher J Paciorek and Mark J Schervish. Spatial modelling using a new class of nonstationary covariance functions. *Environmetrics: The official journal of the International Environmetrics Society*, 17(5):483–506, 2006.
- [158] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Aspell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.
- [159] Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA, 2015.
- [160] Antony Sikorski, Daniel McKenzie, and Douglas Nychka. Normalizing basis functions: Approximate stationary models for large spatial data. *Stat*, 13(4):e70015, 2024.
- [161] A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

APPENDIX A

NORMALIZING BASIS FUNCTIONS

A.1 Further details on solving the screened Poisson equation

Recall that the equation we wish to solve is

$$-\Delta v_{\mathbf{s}} + \kappa^2 v_{\mathbf{s}} = \psi_{\mathbf{s}}(\mathbf{z}). \quad (\text{A.1})$$

Assuming $v_{\mathbf{s}}$ (respectively $\psi_{\mathbf{s}}$) is absolutely integrable, it may be expressed in terms of its Fourier transforms, $\tilde{v}_{\mathbf{s}}$ (resp. $\tilde{\psi}_{\mathbf{s}}$):

$$v_{\mathbf{s}}(\mathbf{z}) = \iint_{\mathbb{R}^2} e^{2\pi i \mathbf{z} \cdot \boldsymbol{\omega}} \tilde{v}_{\mathbf{s}}(\boldsymbol{\omega}) d\boldsymbol{\omega} \quad \text{and} \quad \psi_{\mathbf{s}}(\mathbf{z}) = \iint_{\mathbb{R}^2} e^{2\pi i \mathbf{z} \cdot \boldsymbol{\omega}} \tilde{\psi}_{\mathbf{s}}(\boldsymbol{\omega}) d\boldsymbol{\omega}. \quad (\text{A.2})$$

We substitute these expressions into (A.1), transforming the differential equation to an algebraic one in the frequency domain:

$$\|\boldsymbol{\omega}\|^2 \tilde{v}_{\mathbf{s}} + \kappa^2 \tilde{v}_{\mathbf{s}} = \tilde{\psi}_{\mathbf{s}}. \quad (\text{A.3})$$

$$\Rightarrow \tilde{v}_{\mathbf{s}} = \frac{1}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}_{\mathbf{s}} \quad (\text{A.4})$$

$$\Rightarrow v_{\mathbf{s}}(\mathbf{z}) = \iint_{\mathbb{R}^2} \frac{e^{-2\pi i \mathbf{z} \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}_{\mathbf{s}}(\boldsymbol{\omega}) d\boldsymbol{\omega} \quad (\text{A.5})$$

by the “shift” property of Fourier transforms:

$$\tilde{\psi}_{\mathbf{s}}(\boldsymbol{\omega}) := \iint_{\mathbb{R}^2} \psi_{\mathbf{s}}(\mathbf{z}) e^{2\pi i \mathbf{z} \cdot \boldsymbol{\omega}} d\mathbf{z} = \iint_{\mathbb{R}^2} \psi(\|\mathbf{z} - \mathbf{s}\|) e^{2\pi i \mathbf{z} \cdot \boldsymbol{\omega}} d\mathbf{z} = e^{2\pi i \mathbf{s} \cdot \boldsymbol{\omega}} \tilde{\psi}(\boldsymbol{\omega}) \quad (\text{A.6})$$

where $\tilde{\psi}(\cdot)$ is the Fourier transform of $\psi(\cdot)$. Returning to (A.5), we may rewrite the solution as

$$v_{\mathbf{s}}(\mathbf{z}) = \iint_{\mathbb{R}^2} \frac{e^{-2\pi i (\mathbf{s} - \mathbf{z}) \cdot \boldsymbol{\omega}}}{\kappa^2 + \|\boldsymbol{\omega}\|^2} \tilde{\psi}(\boldsymbol{\omega}) d\boldsymbol{\omega} \quad (\text{A.7})$$

A.2 Software Integration

In order to set up a model in `LatticeKrig`, one typically creates an `LKrig` object, in which they define critical parameters such as κ^2 , number of buffer points, overlap, and normalization choice.

One can choose between all three normalization options by simply modifying: `LKrigSetup(..., normalize = TRUE, normalizeMethod = "option", ...)`. The options for the normalization method are “exact”, “exactKronecker”, or “both”. The “exact” option selects the default, slow normalization method, and the “exactKronecker” option selects the Kronecker method. These will be used for each layer within a multi-resolution model. The “both” option uses the combined approach, as described in Section 3.4.5.

Setting `normalize = FALSE` makes all options irrelevant, and does not normalize the basis functions. The

implementation in `LatticeKrig` provides defaults for this strategy, along with examples and help files to illustrate software details. The development version can be found on github.com/antonyxsik/Normalization-Paper. The `LatticeKrig` package is freely available on the CRAN repository for R cran.r-project.org/web/packages/LatticeKrig/.

APPENDIX B

LATTICEVISION

B.1 Anisotropic SAR Derivation

B.1.1 The Matérn Covariance

Under the Matérn family, Equation (5.1) takes the form:

$$C(\mathbf{s}, \mathbf{s}') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\kappa_{\mathcal{M}} \|\mathbf{s} - \mathbf{s}'\|)^\nu \mathcal{K}_\nu(\kappa_{\mathcal{M}} \|\mathbf{s} - \mathbf{s}'\|), \quad (\text{B.1})$$

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

B.1.2 Isotropic SAR

In two dimensions ($\mathbf{s} \in \mathbb{R}^2$), with $\nu = 1$, Equation (5.2) can be written as

$$(\kappa^2 - \Delta)g(\mathbf{s}) = \mathcal{W}(\mathbf{s}), \quad (\text{B.2})$$

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 $g_{i,j} \equiv g(\mathbf{s}_{i,j})$ at grid point $\mathbf{s}_{i,j} = (i, j)$, $i, j \in \{1, \dots, N\}$. Using second order central-difference approximations for the Laplacian yields

$$-\Delta g_{i,j} \approx [4g_{i,j} - g_{i+1,j} - g_{i-1,j} - g_{i,j+1} - g_{i,j-1}]. \quad (\text{B.3})$$

Substituting (B.3) into (B.2) results in

$$(\kappa^2 + 4)g_{i,j} - (g_{i+1,j} + g_{i-1,j} + g_{i,j+1} + g_{i,j-1}) = e_{i,j}, \quad (\text{B.4})$$

where $e_{i,j}$ denotes the discrete version of the noise $\mathcal{W}$ at location (i, j) . Equation (B.4) can be visualized in lattice notation using the following stencil:

$$\begin{array}{c|c|c} 0 & -1 & 0 \\ \hline -1 & \kappa^2 + 4 & -1 \\ \hline 0 & -1 & 0 \end{array} \quad (\text{B.5})$$

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

B.1.3 Anisotropic Extension

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

$$D = R^\top \Lambda R, \quad R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \sqrt{\rho} & 0 \\ 0 & \frac{1}{\sqrt{\rho}} \end{bmatrix}. \quad (\text{B.6})$$

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

$$(\kappa^2 - \nabla \cdot D \nabla) g(\mathbf{s}) = \mathcal{W}(\mathbf{s}). \quad (\text{B.7})$$

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

$$\begin{aligned} \nabla \cdot D \nabla g_{i,j} \approx & D_{1,1}(g_{i+1,j} - 2g_{i,j} + g_{i-1,j}) + D_{2,2}(g_{i,j+1} - 2g_{i,j} + g_{i,j-1}) \\ & + \frac{D_{1,2}}{2}[g_{i+1,j+1} - g_{i+1,j-1} - g_{i-1,j+1} + g_{i-1,j-1}]. \end{aligned} \quad (\text{B.8})$$

Substituting the above expression into (B.7) results in

$$\begin{aligned} (\kappa^2 + 2D_{1,1} + 2D_{2,2})g_{i,j} - & D_{1,1}(g_{i+1,j} + g_{i-1,j}) - D_{2,2}(g_{i,j+1} + g_{i,j-1}) \\ - \frac{D_{1,2}}{2}[g_{i+1,j+1} - g_{i+1,j-1} - g_{i-1,j+1} + g_{i-1,j-1}] = & e_{i,j}. \end{aligned} \quad (\text{B.9})$$

Equation (B.9) corresponds to the 3×3 stencil

$$\begin{array}{c|c|c} \frac{D_{1,2}}{2} & -D_{2,2} & -\frac{D_{1,2}}{2} \\ \hline -D_{1,1} & \kappa^2 + 2D_{1,1} + 2D_{2,2} & -D_{1,1} \\ \hline -\frac{D_{1,2}}{2} & -D_{2,2} & \frac{D_{1,2}}{2} \end{array} \quad (\text{B.10})$$

One incorporates non-stationarity by allowing D and κ^2 to vary in space, as shown in Equation (5.3).

B.2 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:

$$\kappa^2 \sim 0.6 \log \mathcal{U}(10^{-4}, 2) + 0.4 \mathcal{U}(10^{-4}, 2), \quad \rho \sim \mathcal{U}(1, 7), \quad \theta \sim \mathcal{U}(-\frac{\pi}{2}, \frac{\pi}{2}) \quad (\text{B.11})$$

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.2.1 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 $\boldsymbol{\Omega}$, 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:

$$p(\mathbf{s}) = p_{\text{low}} + \frac{p_{\text{high}} - p_{\text{low}}}{1 + \exp(-\gamma(\mathbf{s}_y - v(\mathbf{s}_x)))} \quad (\text{B.12})$$

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: $\Omega = (\alpha, \beta, \omega, \gamma)$, 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:

$$p(\mathbf{s}) = p_{\text{low}} \Psi(\mathbf{s}_x + \mathbf{s}_y; 0, \sigma) + p_{\text{high}} (1 - \Psi(\mathbf{s}_x + \mathbf{s}_y; 0, \sigma)) \quad (\text{B.13})$$

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:

$$p(\mathbf{s}) = p_{\text{constant}} + a_1 \exp\left(-\frac{\mathbf{s}_x^2 + \mathbf{s}_y^2}{\lambda_1}\right) \quad (\text{B.14})$$

- Hyperparameters: $\Omega = (a_1, \lambda_1)$, 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:

$$p(\mathbf{s}) = \begin{cases} p_{\text{constant}} + a \sin(\pi\omega \mathbf{s}_x), & (\text{horizontal}) \\ p_{\text{constant}} + a \cos(\pi\omega \mathbf{s}_y), & (\text{vertical}) \end{cases} \quad (\text{B.15})$$

- Hyperparameters: $\Omega = (a, \omega, \text{orientation})$, 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:

$$p(\mathbf{s}) = p_{\text{constant}} + a_1 \exp\left(-\frac{(\mathbf{s}_x - x_1)^2 + (\mathbf{s}_y - y_1)^2}{\lambda_1}\right) + a_2 \exp\left(-\frac{(\mathbf{s}_x - x_2)^2 + (\mathbf{s}_y - y_2)^2}{\lambda_2}\right) \quad (\text{B.16})$$

- Hyperparameters: $\boldsymbol{\Omega} = (a_1, a_2, \lambda_1, \lambda_2, x_1, y_1, x_2, y_2)$, 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:

$$p(\mathbf{s}) = p_{\text{low}} + w_1 \text{Coastline}_1(\mathbf{s}) + w_2 \text{Coastline}_2(\mathbf{s}) \quad (\text{B.17})$$

- 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:

$$p(\mathbf{s}) = \begin{cases} p_{\min, p_{\max}} \text{ rescaled field,} & (\text{min-max scaling}) \\ p_{\text{constant}} \times (1 \pm g(\mathbf{s})), & (\text{perturbation scaling}) \end{cases} \quad (\text{B.18})$$

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: $\boldsymbol{\Omega} = (n_{\text{basis}}, \text{scaling choice})$, where $n_{\text{basis}} \sim \mathcal{U}\{6, 7, \dots, 32\}$ 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.2.2 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

$$p(\mathbf{s}) = wp_1(\mathbf{s}) + (1 - w)p_2(\mathbf{s}), \quad w \sim \mathcal{U}(0.1, 0.9) \quad (\text{B.19})$$

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

B.2.3 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 [160], 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 B.1.

B.3 Further Implementation Details

B.3.1 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.

B.3.2 Training

All networks are implemented in PyTorch [161] and trained on a single NVIDIA RTX A6000 GPU. The networks are trained using the AdamW optimizer [57] 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.

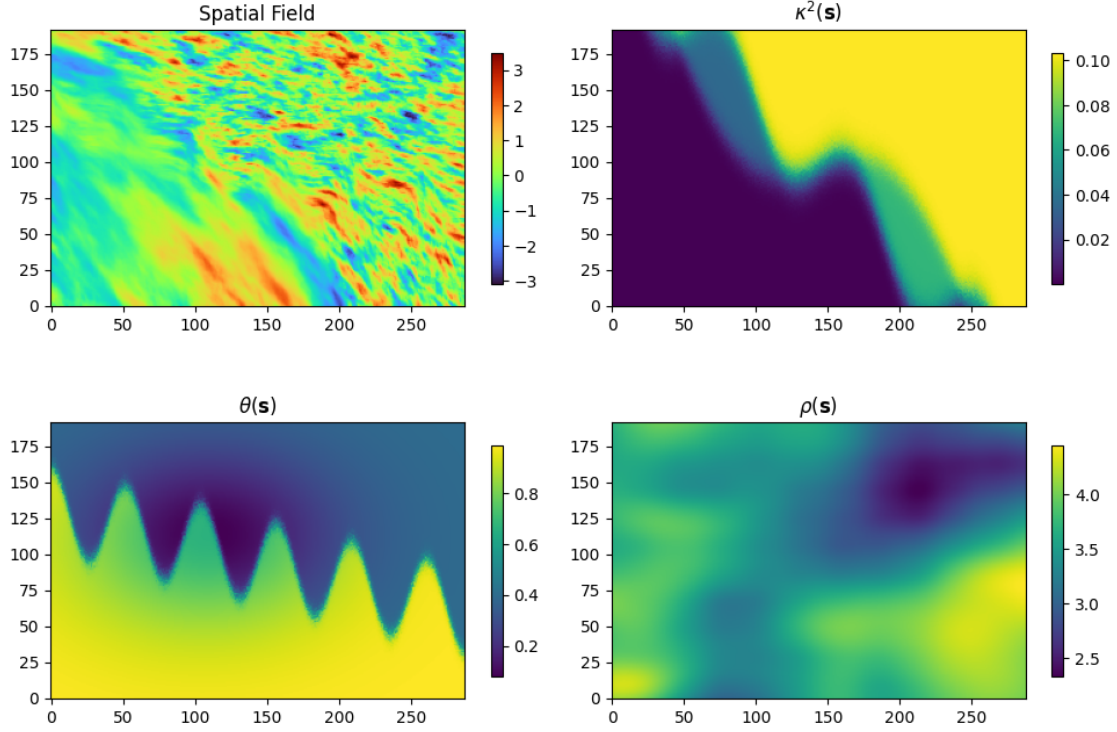


Figure B.1 The first replicate in a training sample (*top-left*), and the accompanying parameter fields that were used to generate it (*remaining*). In this instance, a Double Coastline pattern was used for $\kappa^2(\mathbf{s})$, $\theta(\mathbf{s})$ is the result of stacked Coastline and Double Bump patterns, and $\rho(\mathbf{s})$ is created with a GP-Based pattern.

B.4 Further Experiments with Simulated Data

In this section, Table B.1 contains the results of experimenting with different positional embeddings for STUN and ViT. Table B.2 and Table B.3 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 B.1 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 B.2 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 B.3 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

B.5 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 [3], 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 B.4. 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 B.4 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}

APPENDIX C

MODELING ITALIAN AIR QUALITY

C.1 Change of support

The latent process is defined on the continuous domain $\mathcal{S} \subset \mathbb{R}^2$:

$$y_t(\mathbf{s}) = \mathbf{x}_t(\mathbf{s})^\top \boldsymbol{\beta}_t + \alpha_t y_{t-1}(\mathbf{s}) + g_t(\mathbf{s}), \quad \mathbf{s} \in \mathcal{S}, \quad (\text{C.1})$$

where $g_t(\mathbf{s}) \sim GP(0, C_{g_t}(\mathbf{s}, \mathbf{s}'))$ and $C_{g_t}(\cdot, \cdot)$ is anisotropic and nonstationary. Observations from the monitoring stations are considered as realizations of the point-level Gaussian process $z_t^{(1)}(\cdot)$ defined as

$$z_t^{(1)}(\mathbf{s}) \equiv y_t(\mathbf{s}) + \varepsilon_t^{(1)}(\mathbf{s}), \quad (\text{C.2})$$

where $\varepsilon_t^{(1)}(\mathbf{s}) \sim \mathcal{N}(0, \tau_t^2)$, $\tau_t^2 > 0$. Data from the chemical transport model are considered as realizations of an areal process representing imperfect measurements of areal averages of the latent process,

$$z_{i,t}^{(2)} \equiv \bar{y}_{i,t} + \varepsilon_{i,t}^{(2)}, \quad (\text{C.3})$$

where $\varepsilon_{i,t}^{(2)} \sim \mathcal{N}(0, \xi_t^2)$, $\xi_t^2 > 0$ and $\bar{y}_{i,t}$ represents the exact areal averages of the latent process in $A_i \subset \mathcal{S}$ with $\bigcup_{i=1}^N A_i \equiv \mathcal{S}$. The collection $\{A_i\}_{i=1}^N$ induces a discretised spatial support $\mathcal{S}^d$ that represents the tiling of the domain. Every component of the latent process is also defined on this spatial support as

$$\bar{y}_{i,t} = \frac{1}{|A_i|} \int_{A_i} y_t(\mathbf{s}) d\mathbf{s}, \quad (\text{C.4})$$

$$\bar{\mathbf{x}}_{i,t} = \frac{1}{|A_i|} \int_{A_i} \mathbf{x}_t(\mathbf{s}) d\mathbf{s}, \quad (\text{C.5})$$

$$\bar{g}_{i,t} = \frac{1}{|A_i|} \int_{A_i} g_t(\mathbf{s}) d\mathbf{s}. \quad (\text{C.6})$$

The collection of these components on the limited set of tilings produces the vectors $\bar{\mathbf{y}}_t = (\bar{y}_{1,t}, \dots, \bar{y}_{N,t})^\top$ and $\bar{\mathbf{g}}_t = (\bar{g}_{1,t}, \dots, \bar{g}_{N,t})^\top$ where both $\bar{\mathbf{y}}_t$ and $\bar{\mathbf{g}}_t \in \mathbb{R}^N$. Covariates can be rearranged as a matrix $\bar{\mathbf{X}}_t = (\bar{\mathbf{x}}_{1,t}, \dots, \bar{\mathbf{x}}_{N,t}) \in \mathbb{R}^{p \times N}$ with p covariates. Thus, the areal latent process can be completely defined on the discretised spatial support $\mathcal{S}^d$ as

$$\bar{\mathbf{y}}_t = \bar{\mathbf{X}}_t^\top \boldsymbol{\beta}_t + \alpha_t \bar{\mathbf{y}}_{t-1} + \bar{\mathbf{g}}_t, \quad (\text{C.7})$$

where $\bar{\mathbf{g}}_t \sim \mathcal{N}(0, \Sigma_{\bar{g}_t})$. The elements of the matrix $\Sigma_{\bar{g}_t}$ correspond to the covariance between all pairs $\bar{g}_{i,t}$ and $\bar{g}_{j,t}$, calculated as

$$(\Sigma_{\bar{g}_t})_{i,j} = \text{COV}(\bar{g}_{i,t}, \bar{g}_{j,t}) = \frac{1}{|A_i||A_j|} \int_{A_i \times A_j} C_{g_t}(\mathbf{s}, \mathbf{s}') d\mathbf{s} d\mathbf{s}'. \quad (\text{C.8})$$

The different correlation structure between $C_{g_t}(\mathbf{s}, \mathbf{s}')$ and $\Sigma_{\bar{g}_t}$ depends on the spatial resolution of the tiling, as discussed below. Considering the i -th region A_i and substituting in the areal model (C.3), the process of areal data becomes

$$z_{i,t}^{(2)} = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t \bar{y}_{i,t-1} + \bar{g}_{i,t} + \varepsilon_{i,t}^{(2)}, \quad (\text{C.9})$$

$$z_{i,t}^{(2)} = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t (z_{i,t-1}^{(2)} - \varepsilon_{i,t-1}^{(2)}) + \bar{g}_{i,t} + \varepsilon_{i,t}^{(2)}, \quad (\text{C.10})$$

$$z_{i,t}^{(2)} = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \bar{g}_{i,t} + \varepsilon_{i,t}^{(2)} - \alpha_t \varepsilon_{i,t-1}^{(2)}, \quad (\text{C.11})$$

$$z_{i,t}^{(2)} = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \eta_{i,t}^{(2)}, \quad (\text{C.12})$$

with

$$\eta_{i,t}^{(2)} = \bar{g}_{i,t} + \varepsilon_{i,t}^{(2)} - \alpha_t \varepsilon_{i,t-1}^{(2)}. \quad (\text{C.13})$$

Here, $\eta_{i,t}^{(2)}$ captures the spatial component and the error term, in particular the collection $\boldsymbol{\eta}_t^{(2)} = (\eta_{1,t}^{(2)}, \dots, \eta_{N,t}^{(2)})^\top \sim \mathcal{N}(0, \Sigma_{\eta_t}^{(2)})$ and $\Sigma_{\eta_t}^{(2)} = \Sigma_{\bar{g}_t} + I(\alpha_t^2 \xi_{t-1}^2 + \xi_t^2)$. From $\boldsymbol{\eta}_t^{(2)}$, we use the STUN estimator to retrieve estimates of parameters for the covariance matrix $\Sigma_{\bar{g}_t}$. These parameters are then used to inform the point-level model (Equation C.2). Despite the temporal correlation in the error term being induced by the first-order moving average, this is not modeled explicitly, as the analysis is performed one day at a time. Moreover, the conditional variance $\text{VAR}(\eta_t^{(2)} | \eta_{t-1}^{(2)})$ is less than or equal to the marginal variance; therefore, considering the marginal variance does not introduce substantial error. Regarding the connection between $\Sigma_{\bar{g}_t}$ and C_{g_t} , for sufficiently small areal units $|A_i|$, the induced covariance $\Sigma_{\bar{g}_t}$ closely approximates the point-level covariance C_{g_t} . However, as the support size increases, $\Sigma_{\bar{g}_t}$ progressively smooths out the short-range dependence structure. Therefore, when moving from areal data to point-level data, we include a term that captures the local variability that is smoothed-out by the averages. This term is assumed to follow a Gaussian process with short-range spatial dependence. Specifically, the areal averages $\bar{y}_{i,t}$ and the point process $y_t(\mathbf{s})$ can be related as

$$y_t(\mathbf{s}) = \bar{y}_{i,t} + \psi_t(\mathbf{s}), \quad \mathbf{s} \in A_i, \quad (\text{C.14})$$

where $\psi_t(\mathbf{s}) \sim GP(0, C_{\psi_t}(\mathbf{s}, \mathbf{s}'))$ and C_{ψ_t} is a valid covariance function where $C_{\psi_t} = f(C_{g_t})$. We now derive an explicit connection between the station measurements $z_t^{(1)}(\mathbf{s})$ and the CTM areal data $z_{i,t}^{(2)}$.

Substituting Equation C.14 into the point-level model (Equation C.2), where i indexes the grid cell A_i containing location $\mathbf{s}$, gives

$$z_t^{(1)}(\mathbf{s}) = \bar{y}_{i,t} + \psi_t(\mathbf{s}) + \varepsilon_t(\mathbf{s}), \quad \mathbf{s} \in \mathcal{S} \quad (\text{C.15})$$

$$z_t^{(1)}(\mathbf{s}) = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t \bar{y}_{i,t-1} + \bar{g}_{i,t} + \psi_t(\mathbf{s}) + \varepsilon_t^{(1)}(\mathbf{s}), \quad (\text{C.16})$$

$$z_t^{(1)}(\mathbf{s}) = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t(z_{i,t-1}^{(2)} - \varepsilon_{i,t-1}^{(2)}) + \bar{g}_{i,t} + \psi_t(\mathbf{s}) + \varepsilon_t^{(1)}(\mathbf{s}), \quad (\text{C.17})$$

$$z_t^{(1)}(\mathbf{s}) = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \bar{g}_{i,t} - \alpha_t \varepsilon_{i,t-1}^{(2)} + \psi_t(\mathbf{s}) + \varepsilon_t^{(1)}(\mathbf{s}). \quad (\text{C.18})$$

For ease of exposition, let

$$\eta_t^{(1)}(\mathbf{s}) = \bar{g}_{i,t} + \varepsilon_t^{(1)}(\mathbf{s}) - \alpha_t \varepsilon_{i,t-1}^{(2)} + \psi_t(\mathbf{s}), \quad (\text{C.19})$$

where $\eta_t^{(1)}(\mathbf{s}) \sim GP(0, C_{\eta_t}^{(1)})$. The covariance function $C_{\eta_t}^{(1)}(\mathbf{s}, \mathbf{s}')$ takes different forms depending on the relative locations of $\mathbf{s}$ and $\mathbf{s}'$:

- if $\mathbf{s} = \mathbf{s}' \in A_i$,

$$C_{\eta_t}^{(1)}(\mathbf{s}, \mathbf{s}) = (\Sigma_{\bar{g}_t})_{i,i} + \alpha_t^2 \xi_{t-1}^2 + C_{\psi_t}(\mathbf{s}, \mathbf{s}) + \tau_t^2, \quad (\text{C.20})$$

- if $\mathbf{s} \neq \mathbf{s}' \in A_i$,

$$C_{\eta_t}^{(1)}(\mathbf{s}, \mathbf{s}') = (\Sigma_{\bar{g}_t})_{i,i} + \alpha_t^2 \xi_{t-1}^2 + C_{\psi_t}(\mathbf{s}, \mathbf{s}'), \quad (\text{C.21})$$

- if $\mathbf{s} \in A_i$ and $\mathbf{s}' \in A_j$,

$$C_{\eta_t}^{(1)}(\mathbf{s}, \mathbf{s}') = (\Sigma_{\bar{g}_t})_{i,j} + C_{\psi_t}(\mathbf{s}, \mathbf{s}'), \quad (\text{C.22})$$

where $(\Sigma_{\bar{g}_t})_{i,j}$ indicates the i, j -th element of the matrix $\Sigma_{\bar{g}_t}$. Introducing the term $\eta_t^{(1)}(\mathbf{s})$ allows us to write the point-level model as

$$z_t^{(1)}(\mathbf{s}) = \bar{\mathbf{x}}_{i,t}^\top \boldsymbol{\beta}_t + \alpha_t z_{i,t-1}^{(2)} + \eta_t^{(1)}(\mathbf{s}). \quad (\text{C.23})$$

To summarize, both the point-level covariance function $C_{\eta_t}^{(1)}(\cdot, \cdot)$ and the areal covariance matrix $\Sigma_{\eta_t}^{(2)}$ encode the same large-scale spatial structure determined by $\Sigma_{\bar{g}_t}$. For sufficiently small areal units $|A_i|$, the measurement errors are approximately equal ($\tau_t^2 \simeq \xi_t^2$) and the sub-grid component $\psi_t(\mathbf{s})$ is negligible ($C_{\psi_t}(\cdot, \cdot) \simeq 0$), so that $C_{\eta_t}^{(1)}(\cdot, \cdot)$ and $\Sigma_{\eta_t}^{(2)}$ are almost equivalent. In the case of larger $|A_i|$, as is assumed in our application, $\Sigma_{\eta_t}^{(2)}$ does not include the short-range structure from C_{ψ_t} . We correct for this by introducing an additional range parameter estimated from the station data, which has the overall effect of reducing the correlation range.

C.2 Neural Estimator Details

The STUN (Spatial TransUNet) neural estimator was first introduced in [129], and is based on the TransUNet architecture of Chen et al. [60]. STUN combines a symmetric, UNet-style encoder–decoder with a Vision Transformer bottleneck. The encoder (respectively decoder) consists of a series of convolutional blocks that progressively downsample (respectively upsample) the spatial resolution, while the bottleneck

applies multi-head self-attention across patches of the image to capture long-range dependencies. Skip connections between corresponding encoder and decoder layers preserve fine-scale spatial detail throughout the network.

The network is trained with a batch size of $b = 64$, so each training input has shape $[b, r, h_A, w_A] = [64, 30, 128, 128]$, with corresponding target parameter images of shape $[64, 3, 128, 128]$. We use the AdamW optimizer [57] with a step-wise learning rate schedule and early stopping triggered after 10 epochs without improvement on the validation set, training for a maximum of 200 epochs. The loss function is mean squared error (MSE), computed on standardized parameter values within the training loop to avoid scale imbalance across the three parameter channels. Data augmentation is restricted to spatial translation and random field negation, as other common augmentation techniques (such as rotation) often do not preserve the statistical properties of the image.

STUN is implemented in PyTorch [161] and trained on a single NVIDIA RTX A6000 GPU. Synthetic data generation is performed in R on a laptop with a 2.20 GHz Intel Core i9-14900HX processor and 32 GB RAM, using the `LatticeKrig` package [3] to simulate spatial fields. The complete synthetic dataset—comprising 20,000 training pairs, each with $r = 30$ replicates—required approximately 16 hours to generate and occupies 80 GB of storage. Data are compressed and stored in the HDF5 format to allow easy access using both R and Python.

C.3 Additional Cross Validation Results

To further examine the reliability of the nonstationary model, Figure C.1 shows the station-wise spatial distribution of weighted RMSE, coverage, and prediction interval widths. The coverage is well-calibrated and spatially homogeneous, lending confidence to the nominal 95% intervals despite the large number of days and stations contributing to the aggregate PICP. Prediction interval widths and RMSE are consistent with the complexity of terrain and regions with fewer stations. Overall, predictive performance is strong across the entire domain, with the exception of a small number of outlier stations situated in atypical environments, where local emission sources (such as road tunnels) produce concentration patterns that are not well represented by the correlation structure of the model.

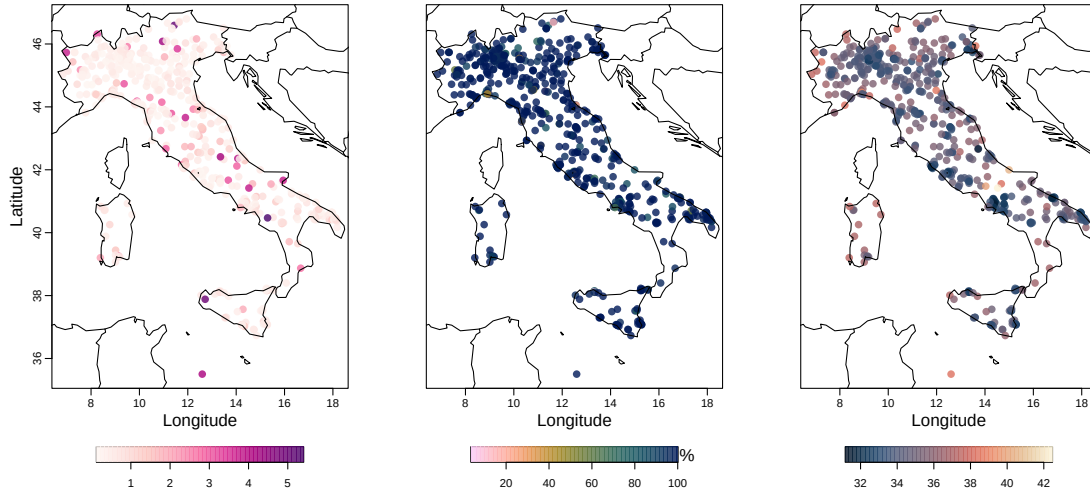


Figure C.1 Station-level cross-validation metrics for the nonstationary (adjusted) model, aggregated over all 349 days. From left to right: RMSE weighted by the local annual mean NO₂ concentration, 95% prediction interval coverage probability, and mean prediction interval width ($\mu\text{g}/\text{m}^3$).

APPENDIX D

COPYRIGHT PERMISSIONS

Permissions to include previously published materials in this thesis are provided below.

1. Copyright permission from Wiley, the publisher of *Stat*, is not required for the inclusion of Chapter 3, as the corresponding author retains the right to reuse the full text of their published article in a thesis or dissertation under Wiley's author reuse policy. The original publication is cited according to the policy.
2. Copyright permission from PMLR, the publisher of the AISTATS proceedings, is not required for the inclusion of Chapter 5, as the corresponding author retains full ownership of copyright in the article under the PMLR publication agreement and the article is published under a Creative Commons Attribution 4.0 International (CC BY 4.0) license. The publication is cited and linked according to the policy.
3. Copyright permission is not required for the inclusion of Chapter 6, as the work has been submitted but not yet published as of the submission of this thesis, and the authors therefore retain all rights in the manuscript.