SCALABLE SPATIO-TEMPORAL PREDICTION USING SPARSE GAUSSIAN PROCESSES AND GRAPH NEURAL NETWORKS

A Thesis

Presented to

the Faculty of the College of Graduate Studies

Tennessee Technological University

by

Belguutei Ariuntugs

In Partial Fulfillment

of the Requirements of the Degree

MASTER OF SCIENCE

Mathematics

May 2025

AN ABSTRACT OF A THESIS

SCALABLE SPATIO-TEMPORAL PREDICTION USING SPARSE GAUSSIAN PROCESSES AND GRAPH NEURAL NETWORKS

Belguutei Ariuntugs

Master of Science in Mathematics

Spatio-temporal prediction is crucial for understanding and mitigating the effects of geographically and temporally correlated phenomena, such as pollutant dispersion, extreme weather events, and the distribution of insurance claims. As climate change accelerates, accurately modeling these dependencies is increasingly important for policymakers, insurers, and public health officials.

Gaussian Process (GP) models provide strong predictive performance for spatio-temporal processes due to their flexibility in capturing uncertainty. However, they often suffer from high computational costs and scalability limitations when applied to large datasets. This thesis explores the integration of deep learning techniques with traditional spatio-temporal models to improve both prediction accuracy and computational efficiency. Specifically, it investigates the combination of sparse spatio-temporal Gaussian process models with neural network architectures like Graph Neural Networks (GNNs) for modeling spatio-temporal trends. This hybrid approach improves predictive accuracy for complex geographic processes while mitigating computational constraints.

Traditional neural networks and statistical methods typically assume data points are independent, making them less effective for modeling correlated spatio-temporal data. By explicitly incorporating covariance structures into neural networks, this research improves predictive performance in scenarios where spatial and temporal dependencies play a critical role. The proposed framework offers valuable contributions to climate-related risk assessment, disaster response planning, and other applications requiring high-resolution spatio-temporal forecasting.

TABLE OF CONTENTS

CERTIFICATE OF APPROVAL OF THESIS

SCALABLE SPATIO-TEMPORAL PREDICTION USING SPARSE GAUSSIAN PROCESSES AND GRAPH NEURAL NETWORKS

by

Belguutei Ariuntugs

Graduate Advisory Committee:

Kehelwala Dewage Gayan Maduranga, Chair Date

David Smith Date

Padmini Veerapen Date

Approved for the Faculty:

Julie C. Baker, Interim Dean Date
College of Graduate Studies

DEDICATION

To my parents, Ariuntugs and Enkhmurun, and my sister,
Ariunjin—whose endless love, guidance, and encouragement have been
the foundation of my journey.

ACKNOWLEDGMENTS

I express my sincere gratitude to all those who supported me throughout this journey. I am especially grateful to my thesis advisor, Dr. Kehelwala Dewage Gayan Maduranga, for their invaluable guidance, steadfast encouragement, and insightful expertise. I also thank my committee members, Dr. Padmini Veerapen and Dr. David Smith, for their constructive feedback and unwavering support. Lastly, this thesis was prepared in Overleaf and edited with Grammarly and Overleaf Writeful for spelling, grammar, and stylistic consistency. All intellectual content, data analysis, and conclusions presented in this thesis are entirely my own.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Accurately modeling spatio-temporal processes is increasingly critical due to the escalating challenges posed by climate change, profoundly impacting environmental stability, public health, agriculture, infrastructure, and economic sustainability globally. Climate-induced phenomena, including extreme weather events, altered disease patterns, disrupted ecosystems, and economic instability, underscore the necessity for robust and reliable predictive models. Effective modeling enables proactive risk management, efficient resource allocation, and mitigation of potential adverse impacts.

In recent years, climate change has become a major contributor to natural disasters and adverse health impacts. The losses caused by climate-related disasters are becoming increasingly significant, affecting lives and livelihoods around the world. There remains a limited period within which to proactively assess and mitigate the effects of climate change and greenhouse gas emissions. Although Earth's climate has undergone numerous changes throughout its history, current measurements reveal an alarming array of facts about our planet's future. Atmospheric warming has accelerated at an unprecedented rate that has not been seen in the past 10,000 years (NASA, 2024).

Over the last 800,000 years, evidence shows eight cycles of ice ages alternating with warmer periods, with the last ice age ending about 11,700 years ago, according to NASA. Scientists attribute most of these historical changes to minor variations in solar radiation and Earth's orbital parameters. However, this natural variability does not account for the current warming trend. There is a strong scientific consensus that today's spike in temperature is primarily human-induced.

According to the Intergovernmental Panel on Climate Change (Intergovernmental Panel on Climate Change, 2023), by 2020 the global surface temperature had risen by approximately 1.4°C relative to preindustrial levels due to greenhouse gas emissions. Each year brings record-breaking high temperatures globally, accompanied by unprecedented disasters, including blizzards, hurricanes, droughts, and floods.

Historically, vulnerable communities with minimal contributions to climate change have been disproportionately impacted by disasters resulting from post-industrial activities. This adverse impact has drastically increased catastrophic economic losses, particularly in underinsured regions. Climate scientists agree that the frequency and severity of extreme weather events are likely to increase, with potentially enormous economic consequences.

In recent decades, droughts have emerged as the deadliest climate-related hazard, causing over 650,000 deaths globally, while devastating storms have claimed approximately 577,232 lives, according to recent United Nations data (United Nations, 2021). Major floods have led to 58,700 deaths, and extreme temperatures have taken 55,736 lives, according to the United Nations (United Nations, 2021). The IPCC estimates that approximately 3.3 to 3.6 billion people live in regions highly susceptible to climate change's dangers and implications.

Climate change has had substantial impacts on biodiversity and ecosystem stability. Rising temperatures, shifts in weather patterns, and extreme events are forcing many species to migrate or adapt faster than they can. Coral reefs, which support diverse marine life, are particularly vulnerable to ocean warming and acidification, with bleaching events now occurring at alarming frequency (The Royal Society, n.d.). Ecosystem disruptions not only threaten biodiversity,, but also destabilize food security, water quality, and other critical ecosystem servicess to human well-being.

The economic consequences of climate change extend beyond immediate disaster recovery costs. As extreme weather events increase in frequency and intensity, they disrupt agriculture, supply chains, and infrastructure, affecting food security and

global trade (Q. Pan et al., 2022). Vulnerable regions with lower adaptive capacities are expected to bear the heaviest burdens. Adaptation strategies, including investments in climate-resilient infrastructure, the deployment of early warning systems, and the promotion of sustainable agricultural practices, are essential to boost the resilience of communities against climate impacts. However, adaptation alone is not sufficient without significant reductions in emissions to curb future warming.

This compels us to develop effective and scalable predictive tools to assess economic losses across regions and to monitor the spread of harmful substances in areas with limited measurement infrastructures. Although advances in satellite technology have increased our monitoring capabilities, these systems are often costly and impractical for widespread, continuous applications. There is a growing need for tools that allow companies and agencies to assess emerging risks based on observed data and global trends in greenhouse gas emissions, helping them to proactively address the challenges posed by climate change.

Traditional geostatistical methods, notably spatial and spatio-temporal kriging, have served as cornerstones for predicting dependencies using Gaussian Processes (GPs) (Schabenberger and Gotway, 2017). These approaches offer interpretability and rigorous uncertainty quantification but face significant computational limitations with large-scale and high-dimensional datasets, commonly arising from advances in sensor technologies and remote sensing (Zhan and Datta, 2024).

Recent developments in machine learning, particularly deep neural networks (DNNs), offer novel approaches to modeling complex data structures rapidly and efficiently. Specifically, Graph Neural Networks (GNNs) effectively model spatial interactions through graph structures, providing reliable scalability and flexibility in modeling graph-represented data compared to traditional methods (Z. Wu, S. Pan, Long, et al., 2019). Integrating Gaussian process covariance structures with the computational efficiency of GNNs presents an innovative pathway for overcoming these methodological challenges.

This thesis bridges traditional spatial statistical methodologies with modern deep learning approaches, specifically investigating the integration of GNNs and Gaussian process covariance structures to significantly enhance accuracy and computational scalability in modeling spatio-temporal processes. The central research question guiding this thesis is:

> *Can Graph Neural Networks integrated with Gaussian process covariance structures significantly enhance prediction accuracy and computational efficiency in modeling spatio-temporal processes?*

1.2 Deep Neural Networks Overview and Challenges

Deep Neural Networks (DNNs) emerged from the longstanding pursuit of machines that can effortlessly perform tasks humans find natural, such as recognizing objects or interpreting handwriting (Goodfellow et al., 2016). Early programmable computers excelled at mathematical tasks but struggled with intuitive, perception-based problems. Traditional machine learning (ML) alleviated rigid programming by allowing algorithms to learn directly from data. However, approaches like linear/logistic regression, decision trees, and Bayesian networks rely heavily on explicit data representations, limiting their adaptability across diverse datasets.

Within Artificial Intelligence (AI)-a broad field focused on mimicking human cognition-ML stands out for its data-driven approach to improving performance without explicit rules. Deep Learning (DL), a rapidly advancing subset of ML as illustrated in Figure 1, leverages deep neural network architectures that capture highly complex, non-linear relationships. This hierarchical feature-learning capability enables DNNs to build elaborate models from simpler building blocks, making them especially powerful for spatio-temporal modeling.

As illustrated in Figure 2, central to DL is the multilayer perceptron (MLP), formed by stacking multiple layers of simple computational units (neurons). Such neural networks excel at handling large and high-resolution datasets, often measured in

Figure 1

Subfields in Artificial Intelligence Research

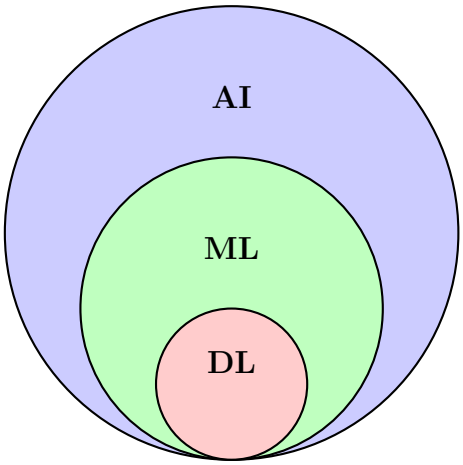
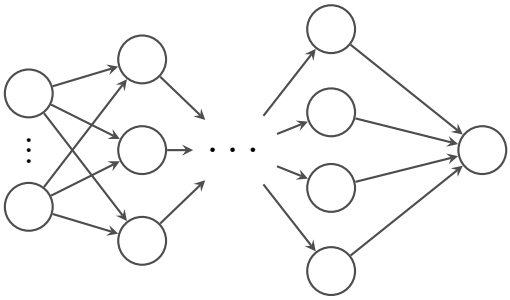


Figure 2

General Deep Neural Networks Architecture



terabytes or petabytes, by harnessing the parallel processing power of general-purpose graphical processing units (GPUs) (W. Chen et al., 2022).

Despite these advantages, applying deep learning to spatio-temporal datasets poses significant challenges. Standard DNN architectures typically assume observations are independent, which contradicts the inherent spatial and temporal correlations prevalent in environmental data (Zhan and Datta, 2024). For example, air pollutant concentrations typically correlate between locations influenced by population density or industrial activities.

Gaussian Processes (GPs) can explicitly model such spatial or temporal dependencies, providing a more nuanced framework for capturing correlation structures. In this thesis, we therefore integrate a modified Graph WaveNet architecture (Z. Wu, S. Pan, Long, et al., 2019) with Nearest-Neighbor Gaussian Process methodologies. This combination efficiently captures complex spatio-temporal patterns and delivers scalable predictions alongside robust uncertainty quantification.

1.3 Traditional Geospatial and Spatio-Temporal Models

Spatial random fields underpin analyses of spatially indexed data, extending naturally to incorporate temporal dimension. Random fields rely primarily on empirical modeling, distinguishing them from statistical mechanics, which employs inductive modeling through physical models described by partial differential equations derived from the underlying principles governing the system (Hristopulos, 2020). Random fields or random functions, generalized forms of random variables indexed over space and time, and model uncertainty with correlated observations, unlike classical statistical approaches that assume independence. These models are widely used in fields such as fluid mechanics, environmental monitoring, mining, and actuarial science.

One of the most important aspects of spatial data is the autocorrelation of observations in space and time. Autocorrelation refers to the correlation of a variable with itself. In particular, observations that are close in space and time tend to be more similar than those that are farther away (Schabenberger and Gotway, 2017). It is imperative to take into account spatial dependence in analysis since it can give rise to heterogeneity in the data which is spatial variability created by underlying complexity (Hristopulos, 2020). Ignoring spatial dependence-induced heterogeneity can result in misleading conclusions and overdispersion, where the observed variance exceeds the variance assumed by the model (Frees et al., 2014).

Traditional spatial data modeling techniques are essential for quantifying spatial structures in climate science, epidemiology, finance, actuarial science, and disaster

mitigation. In actuarial context, it is important to take into account spatial information concerning insurance policies when predicting claims and ratemaking (Frees et al., 2014). There are many cases where accurate modeling of spatially correlated phenomenons is crucial since it can affect real human lives. For example, modeling natural catastrophes for economic damages or modeling frequencies of some diseases in health insurance since it may vary by location (Frees et al., 2014).

Spatio-temporal data can naturally be represented using graph structures and adjacency matrices, forming the core analytical framework of this thesis. Graphs consist of vertices (representing spatial-temporal locations with associated covariates) and edges (indicating relationships or proximity). These relationships are encoded numerically through adjacency matrices, simplifying the representation and analysis of complex spatial and temporal dependencies.

1.4 Research Contributions and Thesis Objectives

The primary contributions and objectives of this thesis include:

1. Developing a scalable, hybrid modeling methodology integrating Graph Neural Networks with Gaussian process covariance structures tailored for complex spatio-temporal data.

2. Evaluating predictive accuracy and computational efficiency of the hybrid approach against traditional spatio-temporal kriging methods.

3. Exploring advancements in covariance structure methods and graph representation learning to enhance model interpretability and performance.

4. Demonstrating applicability through comprehensive case studies simulating realistic environmental and climatic scenarios.

Ultimately, addressing these objectives directly contributes to enhancing our predictive capabilities and improving societal preparedness for climate-induced challenges.

This thesis contributes significantly to the advancement of spatio-temporal modeling methods, equipping researchers and practitioners with innovative, scalable analytical tools for addressing complex challenges posed by climate change.

CHAPTER 2

GEOSTATISTICAL SPACE-TIME MODELS

Spatial data analysis and its temporal extensions play an increasingly important role in making inferences on correlated data that is present in spatial and temporal systems. Oftentimes, empirical data contain information about the attribute of interest and the information about where and when. Spatial and spatio-temporal statistics study collection of random variables that are spatially and temporally indexed such that these variables are highly correlated with each other called autocorrelation. Observations tend to be more similar if they are close to each other in space and in time. These phenomena are, often, very difficult to deterministically modeled.

According to (Hristopulos, 2020) deterministic description of a physical system needs full knowledge of the following:

- Mathematical representations of the natural laws that determine the evolution of the system for example partial differential equations.
- Information about the coefficients of the partial differential equations that can be complex functions of space and time.
- The inputs of the system and the boundary conditions.

If they are known, deterministic mathematical models can be created and solved for the response variables. However, solutions to these equations can be analytically impractical and lead the modeler to numerical approximations.

As a remedy, random fields can be used to model large scale quantities such as spread of atmospheric pollutants, precipitation fields, or frequencies of natural disasters. To account for the inherent randomness in the system, probability distribution functions are used, which results in the involvement of collection of possible

realizations that appear with a frequency coming from the corresponding probability density functions (pdf) in the random field (Hristopulos, 2020).

In view of the impractical costs of dense spatio-temporal monitoring networks, we aim to statistically model the attribute of interest based on limited number of observations in continuous domain of space and time. Such observations are modelled as a partial realizations of a spatio-temporal, typically Gaussian random function (Gneiting et al., 2006).

2.1 Review on Probability Theory

Investigations of many kinds of phenomena can be conducted by repeated experimentation under ideally the same conditions. Each experiment ends with an *outcome* that cannot be predicted with certainty before the experiment but can be described a priori (Hogg et al., 2019). This kind of experiment that can be performed repeatedly under the same condition is called a random experiment and the set of all possible outcomes is referred to as the sample space or all possible states, in our case, denoted as Ω while the elements of the sample space are denoted by lowercase Greek letters like ω.

The subsets of Ω is called events denotes by uppercase Roman letters like A and if the experimental outcome is an element of an event A, then we say that the event A has occurred. After N repeated random experiments, we can then count the number f of times called the frequency that the event A has occurred. The ratio $\frac{f}{N}$ is called the relative frequency of the event A. As the number of attempts N of the random experiment grows large, the relative frequency stabilizes and we associate a number p to the event A which we interpret as the probability of the event A. This number p can then approximate the future relative frequency of event A. We can, also, interpret probability as a rational measure of belief (Hogg et al., 2019).

To construct a mathematical framework for probability and statistics, we must first define a few concepts.

Definition 2.1.1. (σ-algebra of Sets) A collection $\mathcal{A}$ of subsets of a nonempty set Ω is called a **σ-algebra of sets** if:

1. $\emptyset \in \mathcal{A}$.
2. $\forall n \in \mathbb{N},\ \{A_n\} \in \mathcal{A} \Rightarrow \bigcup_{i=1}^{\infty} A_n \in \mathcal{A}$ (Closure under countable unions).
3. $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$ (Closure under complement with respect to $\mathcal{A}$).

Definition 2.1.2. (Set Function) We say that ϕ is a **set function** defined on a σ-algebra $\mathcal{A}$ if ϕ assigns to every $A \in \mathcal{A}$ a real number $\phi(A)$. ϕ is additive if $A, B \in \mathcal{A}$ and $A \cap B = \emptyset$ implies

$$\phi(A \cup B) = \phi(A) + \phi(B),$$

and ϕ is countably additive if $\forall n \in \mathbb{N},\ \{A_n\} \in \mathcal{A}$ and $A_i \cap A_j = \emptyset$ $(i \neq j)$ implies

$$\phi\left(\bigcup_{n=1}^{\infty} A_n\right) = \sum_{n=1}^{\infty} \phi(A_n).$$

Definition 2.1.3. (Probability) Let Ω be a sample space and let $\mathcal{A}$ be the set of events. Let P be a real-valued function defined on $\mathcal{A}$. Then $P(\cdot) : \mathcal{A} \to [0, 1]$ is a **probability set function** if P satisfies the following conditions:

1. $P(A) \geq 0$ for every $A \in \mathcal{A}$.
2. $P(\Omega) = 1$.
3. P is countably additive.

Definition 2.1.4. (Probability Space) The ordered triplet $(\Omega, \mathcal{A}, P)$ is called a **probability space** if the following conditions hold:

1. The sample space Ω is nonempty.

2. The family of events $\mathcal{A}$ is a σ-algebra.

3. The function $P(A)$, for $A \in \mathcal{A}$, is a probability.

Definition 2.1.5. (Random Variable) Consider a random experiment with a sample space Ω. A function X, that assigns to each element $w \in \Omega$ one and only one number $X(w) = x$, is called a **random variable**. The range of X is the set of real numbers $\mathcal{D} = \{x : x = X(w), w \in \Omega\}$.

Definition 2.1.6. (Discrete Random Variable) A random variable X is a **discrete random variable** if its range $\mathcal{D}$ is finite or countable.

Definition 2.1.7. (Continuous Random Variable) A random variable X is a **continuous random variable** if its cdf $F_X(x)$ is a continuous function for all $x \in \mathbb{R}$.

Definition 2.1.8. (Cumulative Distribution Function) Let X be a random variable. Then its **cumulative distribution function** (cdf) is defined by $F_X(x)$, where

$$F_X(x) = P_X((-\infty, x]) = P(\{w \in \Omega : X(w) \leq x\}),$$

which abbreviated as $P(X \leq x)$.

Theorem 2.1.9. *Let X be a random variable with the cdf F_X. Then for $a < b$, $P(a < X \leq b) = F_X(b) - F_X(a)$.*

Proof. Let X be a random variable with the cdf F_X and let $a < b$. This implies that $\{-\infty < X \leq b\} = \{-\infty < X \leq a\} \cup \{a < X \leq b\}$. Since $\{-\infty < X \leq b\}$ is a union of disjoint sets and the probability function P is a set function, $P(\{-\infty < X \leq b\}) = P(\{-\infty < X \leq a\} \cup \{a < X \leq b\}) = P(\{-\infty < X \leq a\}) + P(\{a < X \leq b\})$. Thus, $P(\{a < X \leq b\}) = P(\{-\infty < X \leq b\}) - P(\{-\infty < X \leq a\}) = F_X(b) - F_X(a)$. $\square$

Definition 2.1.10. (Probability Mass Function) Let X be a discrete random variable with range $\mathcal{D}$. The **probability mass function** (pmf) of X is given by

$$p_X(x) = P(X = x), \;\; x \in \mathcal{D}.$$

Definition 2.1.11. (Probability Density Function) If a random variable X can be expressed as

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt,$$

for some function $f_X(x)$, then the function $f_X(x)$ is called a **probability density functions** (pdf) of X. If $f_X(x)$ is continuous, then

$$\frac{d}{dx}F_X(x) = f_X(x).$$

Definition 2.1.12. (Expectation) Let X be a random variable and let $Y = g(X)$ for some function g. If X is a continuous random variable with pdf $f(x)$ and

$$\int_{-\infty}^{\infty} |g(x)|f(x)dx < \infty,$$

then the **expectation** of Y is defined by

$$E[Y] = \int_{-\infty}^{\infty} g(x)f(x)dx.$$

If X is a discrete random variable with pmf $p(x)$ and

$$\sum_{x} |g(x)|p(x) < \infty,$$

then the **expectation** of Y is

$$E[Y] = \sum_{x} g(x)p(x).$$

Definition 2.1.13. (Moment of a Random Variable) Let X be a random variable with pdf $f(x)$ or pmf $p(x)$. Then the m^{th} **moment** of the random variable X is given by

$$E[X^m] = \int_{-\infty}^{\infty} x^m f(x)dx \quad \text{or} \quad \sum_{x} x^m p(x).$$

2.2 Random Fields

The reference (Hristopulos, 2020) defines spatial random fields as "mathematical abstractions that generalize the concept of random processes in d-dimensional spaces." We extend this definition to incorporate the temporal dimension $t \in \mathbb{R}$.

Definition 2.2.1. (Spatio-temporal Random Field (STRF)) A spatio-temporal random field or a spatio-temporal random function

$$\{Z(l;\omega) \in \mathbb{D}_Z \subseteq \mathbb{R};\ l = (\mathbf{s}, t) \in \mathcal{L} = \mathcal{S} \times \mathcal{T} \subset \mathbb{R}^d \times \mathbb{R};\ \omega \in \Omega\}$$

is a mapping from the probability space $(\Omega, \mathcal{A}, P)$ into the space of real numbers.

1. $\mathcal{L}$ is the spatio-temporal domain within which $Z(l;\omega)$ is defined.

2. $Z(l;\omega)$ is indexed in space by $\mathbf{s} \in \mathcal{S} \subset \mathbb{R}^d$ (usually $d = 2$ or 3) and in time by $t \in \mathcal{T} \subset \mathbb{R}$.

3. For each fixed l, $Z(l,\omega)$ is a measurable function of ω.

4. $\mathbb{D}_Z \subset \mathbb{R}$ is the domain where $Z(l;\omega)$ takes values.

5. The sample space $\Omega \neq \emptyset$ is the set of all *possible states or realizations* of the random field.

6. Random field states are functions of the space time coordinate l denoted by $z(l)$ or $z(l;\omega)$.

7. The family $\mathcal{A}$ is the set of all *possible events*, and the function $P(A) \in [0,1]$ assigns a probability to each event $A \in \mathcal{A}$.

This shows that STRF encompasses a host of realizations indexed by ω which we omit to keep the notation simple. Notice that the uppercase Z denotes the random fields and the lowercase z denotes the realizations.

2.2.1 Spatio-temporal Domains

Geostatistical analyses are concerned with being able to do predictions based on a finite number of spatial or temporal locations due to computational limitations or data unavailability due to underdeveloped infrastructures or limited sensors (Gneiting et al., 2006). There can be two different kinds of geostatistical data domains, continuous and discrete.

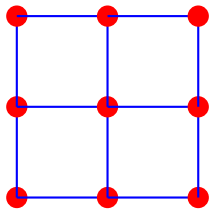
Continuous domain means that the random function $Z(l;\omega)$ can be observed anywhere within $\mathcal{L}$. Between any two sample locations l_i and l_j we can, in principle, place infinite number of other samples (Schabenberger and Gotway, 2017). However, in some cases the random field is not defined on a continuous domain, i.e., the random process is defined over a discrete set of points where the points are also defined by the process (Hristopulos, 2020). This type of spatial data can be observed exhaustively in many applications such as state-wide data providing cancer mortality rates for every county (Schabenberger and Gotway, 2017), public traffic network recording traffic speed over some period of time(Z. Wu, S. Pan, Long, et al., 2019), industrial production chain(Shi et al., 2024), and time evolving social network analysis(Min et al., 2021).

As stated before, our modeling domain is $\mathbb{R}^d \times \mathbb{R}$, where $\mathbb{R}^d$ is for space and $\mathbb{R}$ is for time. Mathematically, we infer that our spatio-temporal domain is simply $\mathbb{R}^d \times \mathbb{R} = \mathbb{R}^{d+1}$ omitting any difference in coordinates. However, in reality, there is a clear distinction between spatial dimensions and temporal dimension. This implies that we must take this into account in our model construction.

Often, spatio-temporal data are observed at fixed temporal increments called lags, and it is sufficient to model the random function on $\mathbb{R}^d \times \mathbb{Z}$ with discrete time steps (Gneiting et al., 2006). Moreover, there are many cases where the spatial random field is not defined everywhere on a continuous domain but only at a set of points in space. It is useful to construct a grid of nodes $\mathcal{G} \in \mathbb{Z}^d$ that represents spatial locations and

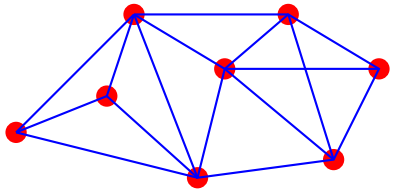
are connected to each other via edges (Hristopulos, 2020). There are two types of grids that arise in applications, a lattice grid and irregular grid.

1. A grid $\mathcal{G}$ is a lattice if the nodes are situated uniformly equidistant from each other on a square grid.



(2.1)

2. A grid $\mathcal{G}$ is irregular if the nodes are scattered non-uniformly with distinct distances from each other.



(2.2)

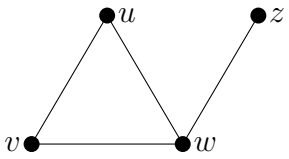
In this thesis, we will consider graph structures to represent spatio-temporal data and it will be used to construct the nearest neighbor set for every location where it is to be estimated. Each fixed node in the graph will be the spatio-temporal random field that contains the realization of the random process. The next section will go over a quick overview of graph theory.

2.2.2 Graph Theory Review

Many entities we consider in mathematics can be represented diagrammatically by means of points and line for example (2.2). The points, in red, that are at the intersection of each line are called *vertices* or *nodes*, the lines, in blue, are called *edges*, and the whole diagram is called a **graph**. However, the intersection of edges are not necessarily a node. Graph structures can represent situations such as road maps, electrical network, or game play between sports teams (Wilson, 1996).

Definition 2.2.2. (Simple Graph) A **simple graph** $\mathcal{G}$ consists of a nonempty finite set $V(\mathcal{G})$ of elements called *vertices* or *nodes*, and finite set $E(\mathcal{G})$ of distinct unordered pairs of distinct elements of $V(\mathcal{G})$ called *edges*. We call $V(\mathcal{G})$ the *vertex set* and $E(\mathcal{G})$ the *edge set* of $\mathcal{G}$. An edge $\{v, w\}$ is said to join the vertices v and w, and is abbreviated to vw.

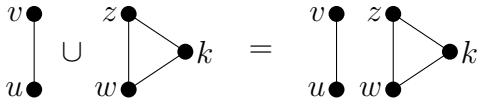
For example, (2.3), is a simple graph $\mathcal{G}$ with vertex set $V(\mathcal{G}) = \{u, v, w, z\}$, and with edge set $E(\mathcal{G}) = \{vu, vw, uw, wz\}$.



(2.3)

Simple graphs have at most only one edge joining any given pair of vertices, however, there are more general graphs with multiple edges joining given two vertices or edges joining a vertex with itself called *loops* (Wilson, 1996). Such general graphs are simply called graphs and we shall restrict ourselves to only simple graphs and call them just graphs.

Definition 2.2.3. (Union of Graphs) Let $\mathcal{G}_1 = (V(\mathcal{G}_1), E(\mathcal{G}_1))$ and $\mathcal{G}_2 = (V(\mathcal{G}_2), E(\mathcal{G}_2))$ be two graphs, where $V(\mathcal{G}_1) \cap V(\mathcal{G}_2) = \emptyset$, then their **union** $\mathcal{G}_1 \cup \mathcal{G}_2$ is a graph with vertex set $V(\mathcal{G}_1) \cup V(\mathcal{G}_2)$ and edge set $E(\mathcal{G}_1) \cup E(\mathcal{G}_2)$.



Definition 2.2.4. (Adjacency) Two vertices v and w of a graph $\mathcal{G}$ are **adjacent** if there is an edge vw joining them, and the vertices v and w are then *incident* with the edge vw. Similarly, two distinct edges e and f are **adjacent** if they have a vertex in common.

Definition 2.2.5. (Complete Graphs) A simple graph in which each pair of distinct vertices are adjacent is a **complete graph**. A complete graph with n vertices are denoted by K_n.

Definition 2.2.6. (Walk) A **walk** from vertices v_0 to v_m in a graph $\mathcal{G}$ is a finite sequence of edges of the form $v_0v_1, v_1v_2, v_2v_3, ..., v_{m-1}v_m$ where any two consecutive edges are adjacent or identical such that the walk determines a sequence of vertices $v_0, v_1, .., v_m$. The first and the last vertices, v_0 and v_m, in the sequence are called the *initial vertex* and the *final vertex*, respectively. The number of edges in a walk is referred to as its *length*.

Definition 2.2.7. (Trail) A walk in which all the edges are distinct is a **trail**.

Definition 2.2.8. (Path) A trail in which all the vertices are distinct, except, possibly, $v_0 = v_m$, is a **path**.

Definition 2.2.9. (Connectedness) A graph $\mathcal{G}$ is **connected** if and only if there is a path between each pair of vertices.

Definition 2.2.10. (Degree of a Vertex) The number of edges incident with a vertex v is the **degree** of v, denoted as $\deg(v)$.

Definition 2.2.11. (Subgraphs) A **subgraph** of a graph $\mathcal{G}$ is a graph, each of whose vertices are the elements of $V(\mathcal{G})$ and each of whose edges are the elements of $E(\mathcal{G})$.

Even though it is interesting to represent a graph by a diagram, it can be limiting if we are to store a large number of spatio-temporal random fields in a computer. Therefore, we will use adjacency matrix representation of graphs.

Definition 2.2.12. (Adjacency Matrix) If $\mathcal{G}$ is a graph with a vertex set $V(\mathcal{G}) = \{v_1, v_2, ..., v_n\}$, its **adjacency matrix A** is the $n \times n$ matrix whose ij^{th} entry is 1 if the vertices v_i and v_j are adjacent.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.4}$$

For example, the graph (2.3) has the adjacency matrix (2.4). However, there can be many ways to represent a graph as matrices depending on the enumeration of its vertices. In (2.4), we have enumerated the vertices of (2.3) as v, u, w, z.

2.3 Exploratory Analysis of Spatial Dependency

By precisely specifying regional structures such as neighborhood, we can go beyond the classical regression models where spatial coordinates are used as categorical covariates with independent and identically distributed assumption. Such improper handling of spatio-temporal coordinates will not represent territorial characteristics such as topography of the region (Frees et al., 2014) and lead to larger error rates (Zhan and Datta, 2024). For example, in insurance applications, claims regarding flood damages or fire damages are not independent, since isolated building may not be affected. The underlying dependence could be due to distance from the nearest cluster of buildings or spatial elevation.

Let $\mathbf{Z} = (z_1, z_2, \ldots, z_n)^T$ be a vector of random variables observed in n distinct locations $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$. There can be many methods of exploring spatio-temporal dependence of the data Z_i, $i = 1, 2, \ldots n$.

1. Graphical illustration

2. Analytical

3. Empirical correlations of spatio-temporal process

2.3.1 Graphical Illustration Method

One of the most instructive first steps in spatio-temporal data analysis is the graphical illustration of the data. The collection of data interpolation methods aims to reconstruct an unknown function or a surface Z from a finite set of discrete data such that this interpolation must use a function f to approximate Z, $f(\mathbf{s}_i) \approx z_i$ for every $i = 1, 2, \ldots, n$ (Wendland, 2004).

To construct a surface of the random field $Z(l; \omega)$, we must consider what kinds of functions to use. In the one-dimensional case with ordered set of data locations, we are interested in continuous function $f : [a, b] \to \mathbb{R}$ with

$$f(s_i) = z_i, \;\; i = 1, 2, \ldots, n.$$

The most natural candidate is that we choose f to be a polynomial p with degree at most $n - 1$ where it is always possible to find a unique interpolation function f (Wendland, 2004).

Definition 2.3.1. (Haar Space) Suppose that the set $\mathcal{S} \subseteq \mathbb{R}^d$ contains at least n points. Let $V \subseteq C(\mathcal{S})$ be an n-dimensional linear space of continuous functions. Then V is called a **Haar space** of dimension n on $\mathcal{S}$ if for any collection of arbitrary distinct points $\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n \in \mathcal{S}$ and arbitrary $z_1, z_2, \ldots, z_n \in \mathbb{R}$ there exists exactly one function $f \in V$ with $f(\mathbf{s}_i) = z_i$ for every $i = 1, 2, \ldots, n$.

Theorem 2.3.2. (Mairhuber-Curtis Theorem) *Suppose that $\mathcal{L} \subseteq \mathbb{R}^d$, $d \geq 2$, contains an interior point. Then there exists no Haar space on $\mathcal{L}$ of dimension $n \geq 2$.*

Unfortunately, according to (Wendland, 2004), it is highly impractical to use polynomials in higher dimensions and due to theorem 2.3.2, finding this unique interpolation function is impossible. As a solution, one may consider using *radial basis functions*. These types of functions are composition of a univariate function with the Euclidean norm. Thus the interpolants are of form

$$f(\mathbf{s}) = \sum_{i=1}^{n} \alpha_i \phi(\|\mathbf{s} - \mathbf{s}_i\|_2) + p(\mathbf{s}), \quad \mathbf{s} \in \mathbb{R}^d,$$

where $\phi : [0, \infty) \to \mathbb{R}$ is a univariate fixed function and $p \in \pi_{m-1}(\mathbb{R}^d)$ is a low-degree $d-$variate polynomial in $\mathbb{R}^d$ such that the coefficients must satisfy

$$\sum_{i=1}^{n} \alpha_i q(\mathbf{s}_i) = 0 \quad \text{for every } q \in \pi_{m-1}(\mathbb{R}^d).$$

Examples of such functions $\phi(\cdot)$ include:

1. Gaussian $\phi(r) = e^{-\alpha r^2}$, $\alpha > 0$.

2. Inverse multiquadric $\phi(r) = \frac{1}{\sqrt{c^2+r^2}}$, $c > 0$.

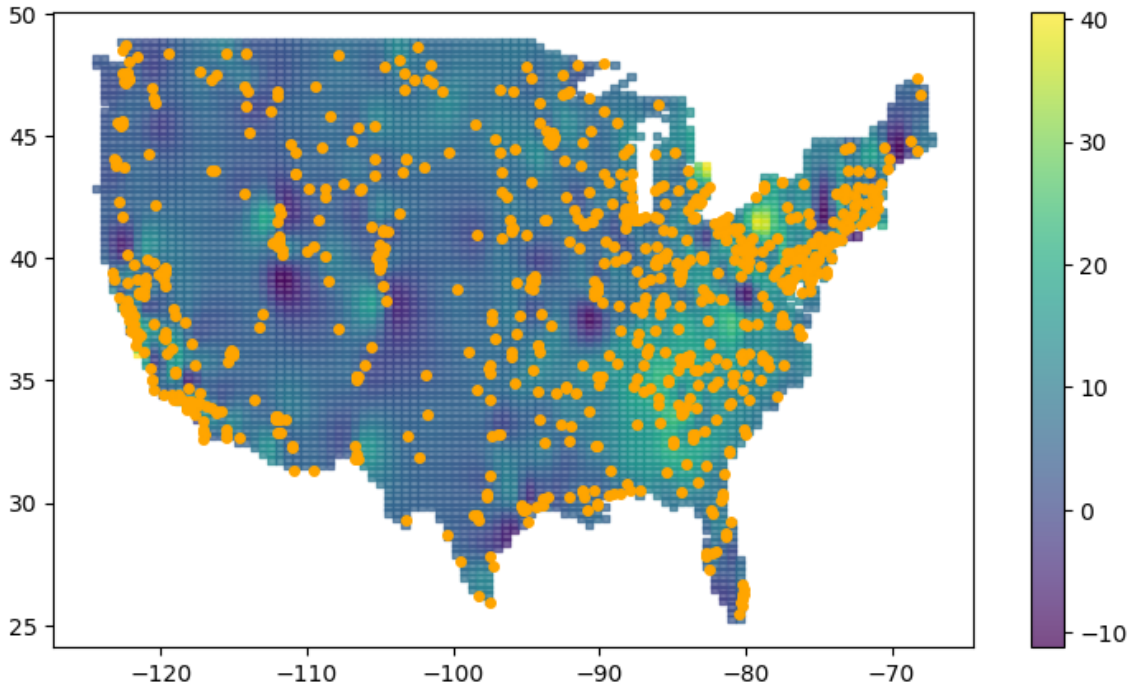
3. Multiquadric $\phi(r) = \sqrt{c^2 + r^2}$, $c > 0$.

As an illustrative example, we will do some exploratory analysis on $PM_2.5$ dataset measured in the continental United States. $PM_{2.5}$ is a fine particulate matter of less than $2.5_{\mu m}$ and is a harmful air pollutant. Its presence is associated with many adverse health complications and diseases such as pulmonary issues and heart attacks (W. Chen et al., 2022) thus requiring the need for high resolution map for its spread and concentration.

PM2.5 data set and its covariates have been obtained from reference (W. Chen et al., 2022) Github repository: https://github.com/aleksada/DeepKriging. We will visually illustrate the data distribution using inverse multiquadric radial basis function interpolation method obtained from (Zhan and Datta, 2024) as initial visual screening method for spatial dependence in figure 3. We can see that there is a clear spatial dependence in the data points demonstrated by colored regions.

Figure 3

Interpolated $PM_{2.5}$ from June 18th, 2022.



Note. Observed $PM_{2.5}$ measurements as spots in orange with smoothed interpolation.

2.3.2 Analytical Exploration of Spatial Dependence

Analytically spatial dependence can be obtained using Morgan's I or Geary's C which are spatial versions of time series analysis methods (Frees et al., 2014). To utilize the two methods, we must construct adjacency matrix encoding neighborhood structure of $\mathbf{Z}$. Let $N[i]$ be the m nearest neighbors of the random variable z_i, $i = 1, 2, \ldots, n$ with some criteria. Then the adjacency matrix $A = (a_{ij}) \in \mathbb{R}^{n \times n}$ is given by

$$a_{ij} = \begin{cases} 1 & \text{for} \quad j \in N[i] \\ 0 & \text{for} \quad j \notin N[i] \end{cases}$$

where the nonzero entries in row i indicates the neighbors of the location i.

Definition 2.3.3. (Moran's I)

$$\rho_I = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}(z_i - \bar{z})(z_j - \bar{z})}{\sum_{i=1}^{n}(z_i - \bar{z})^2}$$

where $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$.

If the data set is spatially independent, then the expectation of Moran's I is $-\frac{1}{n-1}$. When the spatial data is positively associated, $\rho_I > 0$, negative otherwise.

Definition 2.3.4. (Geary's C)

$$\rho_C = \frac{n-1}{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}(z_i - z_j)^2}{\sum_{i=1}^{n}(z_i - \bar{z})^2} \in [0, 2].$$

If the data set is spatially independent, then $\rho_C = 1$. Positive association if $\rho_C < 1$, negative association if $\rho_C > 1$.

2.3.3 Empirical Correlations of Spatio-temporal Process

When assessing spatio-temporal dependence among data points, one way is to plot the empirical correlations of the spatio-temporal random function realizations against the spatio-temporal lags, which is given by, from reference (Montero et al., 2015, August),

$$\hat{C}\left(\mathbf{h}(r), u(k)\right) = \frac{1}{\#N(\mathbf{h}(r), u(k))} \sum_{l_i, l_j \in N(\mathbf{h}(r), u(k))} (Z(l_i) - \bar{Z})(Z(l_j) - \bar{Z}). \tag{2.5}$$

where $\bar{Z} = \frac{1}{n}\sum_i Z(l_i)$ is an estimator of the mean μ of the random field, $N(\mathbf{h}(r), u(k)) = \{l_i, l_j = (\mathbf{s}_i, t_i)(\mathbf{s}_j, t_j) : \|\mathbf{s}_i - \mathbf{s}_j\| \in T(\mathbf{h}(r)), |t_i - t_j| \in T(u(k))\}$, $T(\mathbf{h}(r))$ is the bin size of the spatial lag, $T(u(k))$ is the bin size of the temporal lag, and $\#N(\mathbf{h}(r), u(k))$ is the number of different elements in $N(\mathbf{h}(r), u(k))$, with $r = 1, 2, \ldots, L$ and $k = 1, 2, \ldots, K$.

2.4 Spatio-temporal Prediction

Once the data set in use has been determined to be spatio-temporally dependent, we can proceed with spatio-temporal prediction. Spatio-temporal prediction aims to predict an unknown spatio-temporal random field realization $Z(l_0)$ at an unobserved location l_0. To accomplish this, let $Z(l) : l \in \mathcal{L} \subset \mathbb{R}^d \times \mathbb{R}$ be a spatio-temporal random field and let us assume that realizations of the random field has been recorded at n spatio-temporal locations $\{Z(l_1), Z(l_2), \ldots, Z(l_n)\}$. Our first goal of analysis is to identify the most appropriate model for inference.

Definition 2.4.1. (Stochastic Process) **Stochastic process** is a family of random variables $\{X_t, t \in T\}$ defined on a probability space $(\Omega, \mathcal{A}, P)$ indexed by some set T.

Notice that, for each fixed $t \in T$, X_t is a function $X_t(\cdot)$ on the set Ω and, for each fixed $\omega \in \Omega$, $X_\cdot(\omega)$ is a function on T (Brockwell and Davis, 1991).

Definition 2.4.2. (Realization of a Stochastic Process) The functions $\{X_\cdot(\omega), \omega \in \Omega\}$ on T are known as the realizations of the process $\{X_t, t \in T\}$.

Random field models aim to capture global variability and enhance resolution by accounting for fine-scale local fluctuations, which together define the correlated trend across the spatio-temporal domain. A *trend* in a random field refers to systematic, regular variability within the field. Trends can incorporate both deterministic and stochastic components. Deterministic trends are typically modeled using closed-form expressions, while stochastic trends represent gradual changes that occur within the random field (Hristopulos, 2020).

In modeling spatio-temporal data, the quantity of interest can often be expressed as a superposition of random fields, each characterized by distinct statistical properties. This relationship can be decomposed into the following form:

$$Observation = Trend + Fluctuation + Noise.$$

The fluctuation component captures local variability or stochastic fluctuations around the trend. Unlike purely random noise, fluctuations exhibit dependency on neighboring locations and are governed by specific probability distributions. These fluctuations represent the fine-scale structure inherent in the underlying physical process (Hristopulos, 2020).

By denoting real spatio-temporal data as $Z^*(l;\omega)$, we can distinguish that this is a random field with a noise component. We can then model the random process as

$$Z^*(l;\omega) = Z(l;\omega) + \epsilon(l;\omega) = m_Z(l) + Z'(l;\omega) + \epsilon(l;\omega), \tag{2.6}$$

where the trend $m_Z(l)$ represents large-scale variability which coincides with the expectation of the random field, i.e.,

$$m_Z(l) = E[Z^*(l;\omega)] = E[Z(l;\omega)].$$

The fluctuation term $Z'(l;\omega)$ is the local variability around the trend which we assume to be zero-mean, correlated random field such that

$$E[Z'(l;\omega)] = 0.$$

The noise component $\epsilon(l;\omega)$, also known as the nugget effect, is the uncorrelated fluctuations that are due to random variability such as measurement error or random ambient noise that contaminates the target process such that the noise component cause a discontinuity in the covariance function at the origin (Hristopulos, 2020).

One must consider certain statistical properties when it comes to stochastic spatio-temporal estimators. We would like our estimator to be *unbiased* and *minimum variance*. These two conditions imply that the estimator is precise and has low variability.

Definition 2.4.3. (Unbiasedness) A spatial-temporal prediction method is **unbiased** if the expectation of the estimator at points $l \in \mathcal{L}$ is equal to the true expectation, i.e.,

$$E[\hat{Z}(l;\omega)] = E[Z(l;\omega)].$$

The unbiasedness criterion implies that the spatio-temporal prediction method minimizes the mean squared error (MSE), that is,

$$\text{MSE} = E\Big[\Big(\hat{Z}(l;\omega) - Z(l;\omega)\Big)^2\Big].$$

2.4.1 Spatial Prediction and Kriging

Spatial data analysis includes model estimation, prediction, and simulation. While estimating a suitable model, we aim to find the best parameter values of the model. However, there are nonparametric estimation methods that use deterministic functions such as nearest-neighbor or basis function interpolations, as mentioned in Section 2.3.1. Prediction involves estimation of missing values of the spatial process at unobserved locations. The prediction can be categorized into interpolation, if the prediction location is inside the sampling area, and extrapolation if the prediction point is outside or beyond the sampling area and requires a presence of strong trend in the data. Finally, spatial simulations generate multiple probable scenarios under some conditions (Hristopulos, 2020).

When studying environmental quantities such as pollutions, disasters, and their economic impacts, there could be many locations where we cannot measure the quantities of interest due to financial constraints, inaccessibility of the terrain, or lack of infrastructure. These limitations create the need for accurate, scalable, high-resolution maps that depict the estimates of the random field process for researchers and policy makers to make data-driven decisions. Moreover, spatial prediction methods can be applied to downscaling procedures that aim to artificially enhance the resolution digital images at arbitrary scales (Hristopulos, 2020).

Traditionally, analysis of spatial data has been based on stochastic process models that offer various methods of modeling data on finite collection of locations such that it ensures generalizability in the entire region for inference (Zhan and Datta, 2024).

Especially, Gaussian process (GP) with deterministic component like linear regression that models the mean trend across space and covariance functions that capture the spatial dependence has been the main tool for modelers.

(KARLIN and TAYLOR, 1975) defines Gaussian process as:

Definition 2.4.4. (Gaussian Process) Let T be an abstract set, and $\{X(t) : t \in T\}$ be a stochastic process. We call $\{X(t) : t \in T\}$ a **Gaussian process** if, for every $n = 1, 2, \ldots$ and every finite subset $\{t_1, t_2, \ldots, t_n\}$ of T, the random vector $(X(t_1), X(t_2), \ldots, X(t_n))$ has a multivariate normal distribution. Equivalently, we require for every n that every linear combination

$$\alpha_1 X(t_1) + \cdots + \alpha_n X(t_n), \quad \alpha \in \mathbb{R},$$

have a univariate normal distribution. Every Gaussian process is described uniquely by two parameters, the mean and the covariance function evaluated at two different locations, given respectively by

$$\mu(t) = E[X(t)], \quad t \in T,$$

and

$$C_{XX}(t_i, t_j) = E[(X(t_i) - \mu(t_i))\,(X(t_j) - \mu(t_j))] = E[X(t_i)X(t_j)] - \mu(t_i)\mu(t_j), \;\; t_i, t_j \in T.$$

If $i = j$, then the value of the covariance becomes the variance of the process at the index t_i, i.e.,

$$C_{XX}(t_i, t_i) = E[(X(t_i) - \mu(t_i))^2] = \sigma_X^2(t_i).$$

Sometimes it is useful to model spatial data using the correlation function since the covariance function depends on the variance of the process. This function measures the linear dependence between the random variables.

Definition 2.4.5. (Correlation Function) The correlation function between two distinct indexes is given by

$$\rho_{XX}(t_i, t_j) = \frac{C_{XX}(t_i, t_j)}{\sigma_X(t_i)\sigma_X(t_j)} \in [-1, 1], \quad t_i, t_j \in T.$$

In connection with the spatial predictions, the covariance function decreases as the distance between the two locations increases, but not necessarily monotonically (Hristopulos, 2020). In addition, the correlation function of the spatial random field $\{Z(\mathbf{s}; \omega); \mathbf{s} \in \mathbb{R}^d; \omega \in \Omega\}$ measures the changes in the linear dependence of field values as a function of the distance between two points. If, for two distinct locations $\mathbf{s}_1$ and $\mathbf{s}_2$, the correlation $\rho_{ZZ}(\mathbf{s}_1, \mathbf{s}_2)$ is negative, given that all possible realizations are considered, higher values of the field at $\mathbf{s}_1$ tend to occur with lower values at $\mathbf{s}_2$, vice versa. Conversely, if the correlation between the two locations is positive, then the field values tend to behave similarly at both locations (Hristopulos, 2020).

Another property that we induce on the covariance function is *stationarity* also known as *statistical homogeneity*. This concept is random field equivalence of translation invariance such that it implies the random field is independent of the location in space or time. Stationarity assumption is used to simplify the spatial model structure since more complex models are disadvantaged due to *Occam's razor* principle. There are two types of stationarity that we consider

1. Weak or second order stationarity

2. Strong stationarity.

Definition 2.4.6. (Weak Stationarity) A spatial random field $Z(\mathbf{s}; \omega)$ is called **weakly stationary** if

1. $E[Z(\mathbf{s}; \omega)] = m_Z$ for every $\mathbf{s} \in \mathbb{R}^d$.

2. For every $\mathbf{s}_1, \mathbf{s}_2 \in \mathbb{R}^d$, $C_{ZZ}(\mathbf{s}_1, \mathbf{s}_2) = C_{ZZ}(\mathbf{s}_1 - \mathbf{s}_2) = C_{ZZ}(\mathbf{h})$, where the vector $\mathbf{h} := \mathbf{s}_1 - \mathbf{s}_2$ represents the spatial lag.

Weak stationarity implies that the random field has constant mean and the covariance function is a function of $\mathbf{h}$.

Under the assumption of weak stationarity, two consequences are implied as illustrated in (Hristopulos, 2020):

1. Uniform variance

$$E[\Big(Z(\mathbf{s};\omega) - E[Z(\mathbf{s};\omega)]\Big)^2] = \sigma_Z^2.$$

2. Translation-invariant correlation function

$$\rho_{ZZ}(\mathbf{s}_1, \mathbf{s}_2) = \rho_{ZZ}(\mathbf{h}).$$

(Schabenberger and Gotway, 2017) defines strongly stationary random fields as

Definition 2.4.7. (Strong Stationarity) A spatial random field $Z(\mathbf{s};\omega)$ is called **strongly stationary** if the spatial distribution is invariant under translation of the coordinates, i.e.,

$$P(Z(\mathbf{s}_1;\omega) < z(\mathbf{s}_1), Z(\mathbf{s}_2;\omega) < z(\mathbf{s}_2), \ldots, Z(\mathbf{s}_n;\omega) < z(\mathbf{s}_n)) =$$
$$P(Z(\mathbf{s}_1 + \mathbf{h}) < z(\mathbf{s}_1), Z(\mathbf{s}_2 + \mathbf{h}) < z(\mathbf{s}_2), \ldots, Z(\mathbf{s}_n + \mathbf{h}) < z(\mathbf{s}_n))$$

for every n and $\mathbf{h}$.

2.4.2 Spatial Kriging

In classical geostatistics, the kriging framework emerges from GP assumptions and remains the dominant tool for optimal linear prediction under mean square error loss. The simplest kriging method assumes that the random field is stationary 2.4.6 and with constant mean everywhere. The stationarity assumption implies that the random field has constant variance and the covariance depends only on the lag vector.

Stochastic predictions methods like kriging express predictions at unobserved locations in terms of linear combinations of the sample values. The parameters of

such models are fit by optimizing maximum likelihood or the methods of moments. The maximum likelihood estimation can be defined by

Definition 2.4.8. (Likelihood Function) Let $\{X_1, X_2, \ldots, X_n\}$ be a random sample from a distribution that depends on parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_m)$ with pdf or pmf $f(x; \boldsymbol{\theta})$. Suppose that $\boldsymbol{\theta}$ is restricted to some parameter space Θ. Then, when regarded as a function of $\boldsymbol{\theta}$, the joint pdf or pmf of X is

$$L(\boldsymbol{\theta}; x_i) = \prod_{i=1}^{n} f(x_i|\boldsymbol{\theta}),$$

which is called the **likelihood function** and represents the probability that the specific data is observed if the model with the specified parameter vector is true. The likelihood can be fully specified in terms of the means and the covariance function.

Definition 2.4.9. (Maximum Likelihood Estimation (MLE)) Given that the structure of the model that will be fitted to the data, the optimal parameter $\hat{\boldsymbol{\theta}}$ can be estimated by

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, x_i).$$

However, the method of moments or Weighted Least Squares (WLS) method is based on empirical covariance. Let $\hat{C}(h)$ be the empirical covariance function computed at lag h and let $C(h; \boldsymbol{\theta})$ be a parametric model of the covariance function with parameter vector $\boldsymbol{\theta}$.

The WLS method fits the covariance model to the empirical covariance by minimizing the weighted sum of squared differences:

$$Q(\boldsymbol{\theta}) = \sum_{i=1}^{n} w(h_i) \left[\hat{C}(h_i) - C(h_i; \boldsymbol{\theta})\right]^2,$$

where:

- h_i are the selected lag distances,
- $w(h_i)$ are weights that can be chosen to reflect the reliability or variance of the empirical estimates at each lag.

The parameter estimates $\hat{\boldsymbol{\theta}}$ are then obtained by solving:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}).$$

Generally, stochastic spatial linear prediction claims that the relation

$$\hat{Z}(\mathbf{s};\omega) = \sum_{i=1}^{n} \lambda_i Z(\mathbf{s}_i;\omega) \tag{2.7}$$

holds between the random field at the unobserved location and the observed locations (Hristopulos, 2020). The prediction error of this estimator with respect to the the actual values $Z(\mathbf{s};\omega)$, if given by

$$\epsilon(\mathbf{s};\omega) = Z(\mathbf{s};\omega) - \hat{Z}(\mathbf{s};\omega). \tag{2.8}$$

In spatial kriging, the assumption of constant mean implies that the mean can be obtained theoretically or estimated from the data. However, the constant mean assumption can be relaxed by modeling the mean with regression models.

Permissibility of Valid Covariance Functions When modeling spatial random fields, one must consider some constraints when it comes to using models approximating the covariance between random fields. In spatial and spatio-temporal predictions, covariance models do not have formal governing equations (Hristopulos, 2020). Instead we can choose from known models that satisfies several constraints and we introduce an important theorem that is *Cauchy-Schwartz* inequality expressed in terms of random variables from reference (Hristopulos, 2020).

Theorem 2.4.10. *(Cauchy-Schwartz Inequality) If $X(\omega)$ and $Y(\omega)$ are scalar random variables with finite first and second order moments, then*

$$\sqrt{E\left[X^2(\omega)\right] E\left[Y^2(\omega)\right]} \geq E\left[X(\omega)Y(\omega)\right].$$

The following are necessary but not sufficient conditions for a candidate covariance function $C_{ZZ}(\mathbf{h})$, with a lag vector $\mathbf{h} = \mathbf{s}_1 - \mathbf{s}_2$ between two spatial locations, to be a valid covariance function.

$$\text{Non-negative variance:} \qquad C_{ZZ}(\mathbf{0}) \geq 0. \tag{2.9}$$

$$\text{Symmetry:} \qquad C_{ZZ}(\mathbf{h}) = C_{ZZ}(-\mathbf{h}) \tag{2.10}$$

$$\text{Mode:} \qquad |C_{ZZ}(\mathbf{h})| \leq C_{ZZ}(\mathbf{0}) \tag{2.11}$$

Proof. The non-negativity of variance follows from the fact that the random field is weakly stationary 2.4.6 such that when $\mathbf{h} = 0$, $C_{ZZ}(\mathbf{h}) = \sigma_Z^2$.

From the definition of covariance function in 2.4.4 and from the definition of random process decomposition 2.6, we can infer that under the stationarity assumption

$$\begin{aligned}
C_{ZZ}(\mathbf{s}_1, \mathbf{s}_2) &= E\left[Z(\mathbf{s}_1;\omega)Z(\mathbf{s}_2;\omega)\right] - E\left[Z(\mathbf{s}_1;\omega)\right] E[Z(\mathbf{s}_2;\omega)] \\
&= E\left[\left(m_Z + Z'(\mathbf{s}_1;\omega)\right)\left(m_Z + Z'(\mathbf{s}_2;\omega)\right)\right] - m_Z^2 \\
&= E\left[m_Z^2 + m_Z Z'(\mathbf{s}_1;\omega) + m_Z Z'(\mathbf{s}_2;\omega) + Z'(\mathbf{s}_1;\omega)Z'(\mathbf{s}_2;\omega)\right] - m_Z^2 \\
&= m_Z^2 + m_Z E\left[Z'(\mathbf{s}_1;\omega)\right] + m_Z E\left[Z'(\mathbf{s}_2;\omega)\right] + E\left[Z'(\mathbf{s}_1;\omega)Z'(\mathbf{s}_2;\omega)\right] - m_Z^2 \\
&= E\left[Z'(\mathbf{s}_1;\omega)Z'(\mathbf{s}_2;\omega)\right] = E\left[Z'(\mathbf{s}_2;\omega)Z'(\mathbf{s}_1;\omega)\right]
\end{aligned}$$

by the fact that $E\left[Z'(\mathbf{s};\omega) = 0\right]$.

Notice that $C_{ZZ}(\mathbf{0}) = E\left[(Z(\mathbf{s};\omega) - m_Z)^2\right] = E\left[Z'(\mathbf{s};\omega)^2\right]$. Then, by letting $X(\omega) = Z'(\mathbf{s}_1;\omega)$ and $Y(\omega) = Z'(\mathbf{s}_2;\omega)$, from Cauchy-Schwartz Inequality 2.4.10, we get that

$$\sigma_Z^2 = E\left[Z'(\mathbf{s};\omega)^2\right] \geq \sqrt{E\left[Z'(\mathbf{s}_1;\omega)^2\right] E\left[Z'(\mathbf{s}_2;\omega)^2\right]} \geq \left|E\left[Z'(\mathbf{s}_1;\omega)Z'(\mathbf{s}_2;\omega)\right]\right|.$$

Thus, we get $|C_{ZZ}(\mathbf{h})| \leq C_{ZZ}(\mathbf{0})$. $\square$

Another necessary and sufficient condition that we must consider is that covariance functions must be positive definite functions (Hristopulos, 2020).

Definition 2.4.11. (Positive Definite Functions) Functions $C_{ZZ}(\cdot)$ are **positive definite** if:

1. The are continuous functions.

2. They are symmetric.

3. $\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j E\left[Z'(\mathbf{s}_i;\omega)Z'(\mathbf{s}_j;\omega)\right] \geq 0 \iff \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j C_{ZZ}(\mathbf{s}_i - \mathbf{s}_j) \geq 0$ for every locations $\{\mathbf{s}_i\}_{i=1}^{n}$ and for any set of real-valued coefficients $\{c_i\}_{i=1}^{n}$.

Definition 2.4.12. (Simple Kriging Equation) The simple kriging equation is expressed by

$$\hat{z}(\mathbf{s}) := m_Z + \sum_{i=1}^{n} \lambda_i z_i'^*,$$

where m_Z is the constant mean and $z_i'^*$'s are the fluctuations around the mean at the sampled location $\mathbf{s}_i$ given by

$$z_i'^* = z_i^* - m_Z, \text{ for } i = 1, 2, \ldots, n.$$

Simple kriging is the *best linear unbiased predictor* (BLUP) (W. Chen et al., 2022), since the expectation of the fluctuation term $\sum_{i=1}^{n} \lambda_i z_i'^*$ vanishes by the zero mean assumption. Moreover, the simple kriging variance at the location $\mathbf{s}$ using the covariance function is given by

$$\begin{aligned}\sigma_{SK}^2(\mathbf{s}) &= E\left[\left(\sum_{i=1}^{n} \lambda_i z_i'^* - z'(\mathbf{s})\right)^2\right] \\ &= \sigma_Z^2 - \sum_{\alpha=1}^{n} \lambda_\alpha C_{ZZ}(\mathbf{s}_\alpha - \mathbf{s}).\end{aligned}$$

The simple kriging equation 2.4.12 is expressed in matrix form as

$$\begin{bmatrix} \sigma_Z^2 & C_{ZZ}(\mathbf{s}_1 - \mathbf{s}_2) & \ldots & C_{ZZ}(\mathbf{s}_1 - \mathbf{s}_n) \\ C_{ZZ}(\mathbf{s}_2 - \mathbf{s}_1) & \sigma_Z^2 & \ldots & C_{ZZ}(\mathbf{s}_2 - \mathbf{s}_n) \\ \vdots & \vdots & \vdots & \vdots \\ C_{ZZ}(\mathbf{s}_n - \mathbf{s}_1) & C_{ZZ}(\mathbf{s}_n - \mathbf{s}_2) & \ldots & \sigma_Z^2 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} C_{ZZ}(\mathbf{s}_1 - \mathbf{s}) \\ C_{ZZ}(\mathbf{s}_2 - \mathbf{s}) \\ \vdots \\ C_{ZZ}(\mathbf{s}_n - \mathbf{s}) \end{bmatrix}, \quad (2.12)$$

where $C_{ZZ}(\mathbf{s}_i - \mathbf{s}_j)$ is a covariance between spatial locations $\mathbf{s}_i$ and $\mathbf{s}_j$ for every $i \neq j$, λ_i is a kriging weight of the location $\mathbf{s}_i$ for every $i = 1, 2, \ldots, n$, and $C_{ZZ}(\mathbf{s}_i - \mathbf{s})$ is a cross covariance between a new location $\mathbf{s}$ and a sample location $\mathbf{s}_i$ for every $i = 1, 2 \ldots, n$.

We can express (2.12) more compactly as,

$$\mathbf{C}\boldsymbol{\lambda} = \mathbf{c}_0, \tag{2.13}$$

where $\mathbf{C}$ is the $n \times n$ covariance matrix of the observed data points,

$$[\mathbf{C}]_{i,j} = C_{ZZ}(\mathbf{s}_i - \mathbf{s}_j), \text{ for all } i, j = 1, 2, \ldots, n,$$

$\boldsymbol{\lambda}$ is the vector of kriging weights

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)^T,$$

and $\mathbf{c}_0$ is the $n \times 1$ vector of the covariance function evaluated for all pairs between the prediction and the observed sampling points, i.e.,

$$[\mathbf{c}_0]_i = C_{ZZ}(\mathbf{s}_i - \mathbf{s}), \text{for } i = 1, 2, \ldots, n.$$

A unique solution $\boldsymbol{\lambda} = \mathbf{C}^{-1}\mathbf{c}_0$ exists if $\mathbf{C}$ is invertible such that the *precision matrix* $\mathbf{C}^{-1}$ exists. This is guaranteed since the covariance function $C_{ZZ}(\mathbf{h})$ generating $\mathbf{C}$ is a positive definite function (Hristopulos, 2020). Thus, the kriging equation in matrix formulation is given by

$$\hat{z}(\mathbf{s}) = m_Z + \mathbf{C}^{-\mathbf{1}}\mathbf{c}_0\mathbf{z}'^*, \tag{2.14}$$

and the kriging variance in matrix formulation is given by

$$\sigma^2_{SK}(\mathbf{s}) = \sigma^2_Z - \mathbf{c}_0^T\mathbf{C}^{-1}\mathbf{c}_0 \tag{2.15}$$

Limitations in Spatial Kriging Kriging-based approaches typically require dense covariance-matrix operations such as inversion and multiplication, which become computationally prohibitive at large scales. Efforts to address this computational burden

have prompted a variety of approximation schemes, particularly those leveraging nearest-neighbor methods. For instance, (Datta et al., 2016) introduced the Hierarchical Nearest-Neighbor Gaussian Process (NNGP), which enables large-scale inference by approximating the full covariance structure with a sparse inverse-covariance factorization based on local (nearest-neighbor) conditional dependencies. This allows the floating-point operations—and associated memory overhead—to grow nearly linearly with the number of spatial locations, significantly enhancing scalability while retaining the interpretability and theoretical guarantees of GP-based models.

2.4.3 Spatio-temporal Kriging

In spatio-temporal kriging, the spatio-temporal random field (STRF) $Z(l;\omega) = Z((\mathbf{s},t);\omega)$ predictions rely on the appropriate specifications of the space-time covariance function that depends on the space-time coordinates l_1 and l_2. There could be cases where no further structure exists and the modeler must make simplifying assumptions similar to the spatial kriging like stationarity. Furthermore, a valid stationary covariance function must be positive definite 2.4.11.

Definition 2.4.13. (Stationary Spatio-temporal Random Field) Spatio-temporal random field has *spatially stationary* covariance if $Cov\left(Z((\mathbf{s}_1,t_1);\omega), Z((\mathbf{s}_2,t_2);\omega)\right)$ depends only on the spatial lag vector $\mathbf{h} = \mathbf{s}_1 - \mathbf{s}_2$. STRF has *temporally stationary* covariance if $Cov\left(Z((\mathbf{s}_1,t_1);\omega), Z((\mathbf{s}_2,t_2);\omega)\right)$ depends only on the temporal lag $u = t_1 - t_2$. If the STRF has both the spatial and temporal stationarity, then the STRF is **stationary**. Under stationarity assumption, there exists a function C defnined on $\mathbb{R}^d \times \mathbb{R}$ such that

$$Cov\left(Z((\mathbf{s}_1,t_1);\omega), Z((\mathbf{s}_2,t_2);\omega)\right) = C(\mathbf{s}_1 - \mathbf{s}_2, t_1 - t_2)$$

for every space time coordinates $(\mathbf{s}_1,t_1)$ and $(\mathbf{s}_2,t_2)$ in $\mathbb{R}^d \times \mathbb{R}$.

However, due to the added temporal structure of the STRF, further simplifying assumptions must be made about the covariance function such as separability and full symmetry that allows for computationally efficient estimation.

Definition 2.4.14. (Separable Covariance) Spatio-temporal random field $Z(l;\omega)$ is said to have a **separable covariance** if there exists purely spatial and purely temporal covariance functions C_S and C_T, respectively, such that

$$Cov\left(Z((\mathbf{s}_1,t_1);\omega),Z((\mathbf{s}_2,t_2);\omega)\right)=C_S(\mathbf{s}_1,\mathbf{s}_2)C_T(t_1,t_2)$$

for every space time coordinates $(\mathbf{s}_1,t_1)$ and $(\mathbf{s}_2,t_2)$ in $\mathbb{R}^d\times\mathbb{R}$.

Definition 2.4.15. (Fully Symmetric Covariance) Spatio-temporal random field $Z(l;\omega)$ has **fully symmetric** covariance if

$$Cov\left(Z((\mathbf{s}_1,t_1);\omega),Z((\mathbf{s}_2,t_2);\omega)\right)=Cov\left(Z((\mathbf{s}_1,t_2);\omega),Z((\mathbf{s}_2,t_1);\omega)\right)$$

for every space time coordinates $(\mathbf{s}_1,t_1)$ and $(\mathbf{s}_2,t_2)$ in $\mathbb{R}^d\times\mathbb{R}$.

However, in real world applications, separability and symmetry assumptions are not always appropriate since the underlying process is under other influences. For example, environmental pollutant dispersion can be influenced by air flow and wind direction which results in lack of full symmetry (Gneiting et al., 2006). This requires more general non-stationary, non-separable, and not symmetric covariance functions.

We can fit purely spatial correlation model

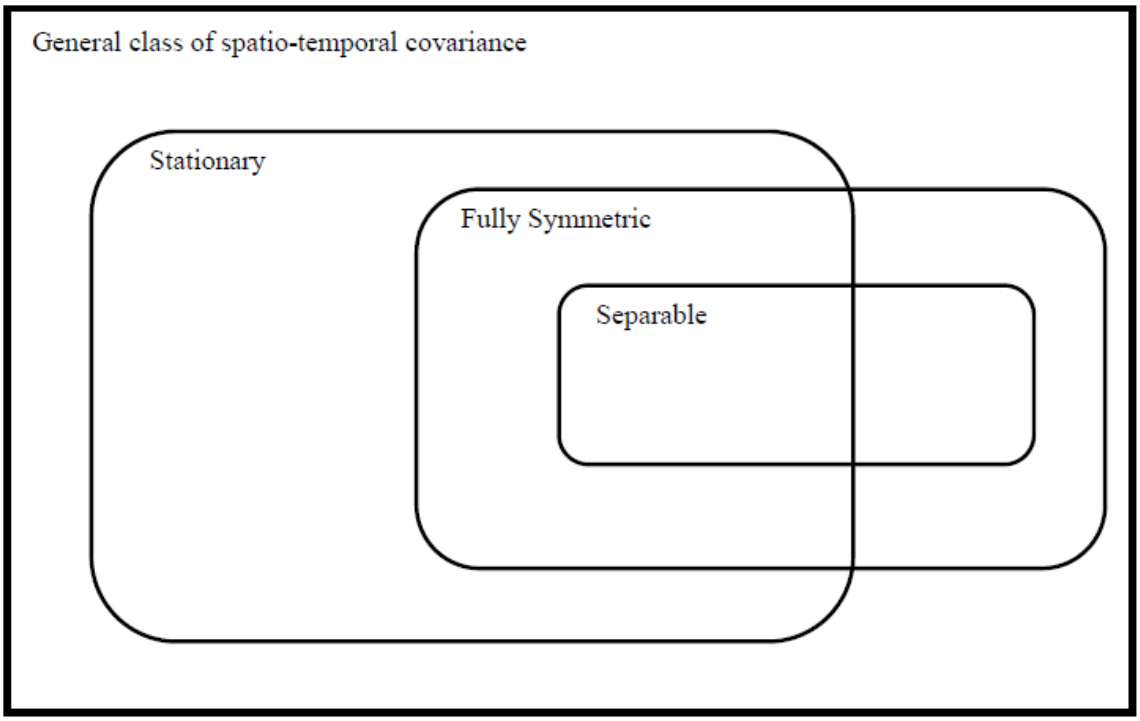
$$C_s(\mathbf{h})=(1-v)exp(-c\|\mathbf{h}\|)+v\delta_{\mathbf{h}=0}, \tag{2.16}$$

where v is the spatial nugget effect and $c\geq 0$ is the scale parameter, to the empirical correlation based on spatial lag using weighted least squares. And we fit purely temporal correlation model of the type

$$C_T(u)=(1+a|u|^{2\alpha})^{-1}, \tag{2.17}$$

Figure 4

(Gneiting et al., 2006) illustrated the relationships between different classes of covariance functions as



where $a \geq 0$ is the temporal scale parameter and $\alpha \in (0, 1]$ is the smoothness parameter. We can fit a separable model

$$C_{SEP}(\mathbf{h}, u) = C_S(\mathbf{h})C_T(u) \tag{2.18}$$

which is simply the product of the fitted purely spatial and temporal models. This model can be embedded into the fully symmetric but generally non-separable correlation function

$$C_{FS}(\mathbf{h}, u) = \frac{1-v}{1+a|u|^{2\alpha}}\Big(\exp\left(-\frac{c\|\mathbf{h}\|}{(1+a|u|^{2\alpha})^{\beta/2}}\right) + \frac{v}{1-v}\delta_{\mathbf{h}=0}\Big), \tag{2.19}$$

where $\beta \in [0, 1]$ is the space-time interaction parameter, fit by weighted least squares with all parameters fixed at the previous estimates. Full WLS optimization over all

correlation parameters simultaneously yields essentially identical estimates (Gneiting et al., 2006).

The simple kriging point predictor or point forecast,

$$\mu_{\mathbf{s},t} = \mathbf{c}_0 \mathbf{C}^{-1} \mathbf{z}_{\mathbf{s},t}, \tag{2.20}$$

where C is the variance covariance matrix of the predictor variables, $\mathbf{c}_0$ is a vector with the covariances between the predictand and the predictor variables, and $\mathbf{z}_{\mathbf{s},t}$ is a vector with the realized values of the predictor variables.

CHAPTER 3

SPATIO-TEMPORAL NEURAL NETWORKS

3.1 Introduction to Neural Networks

Deep feedforward neural networks or multilayer perceptrons (MLPs) are integral components of deep learning (DL), primarily designed to approximate an unknown target function f^*.

Definition 3.1.1. (Feedforward Neural Networks) A feedforward neural network is a parameterized function $f : \mathcal{X} \to \mathcal{Y}$ that maps an input $\mathbf{x}$ (which may be real-valued or encoded categorical data) to an output $\mathbf{y}$ through a series of layers consisting of an input layer, intermediary hidden layers, and an output layer. Each layer $f^{(i)}$ (for $i = 1, 2, \ldots, n$) is typically composed of an affine transformation followed by a non-linear activation function, such that the overall mapping is given by:

$$f(\mathbf{x}) = (f^{(n)} \circ f^{(n-1)} \circ \cdots \circ f^{(1)})(\mathbf{x}).$$

During training, the parameters $\boldsymbol{\theta}$ associated with each layer are iteratively optimized (often via gradient descent and backpropagation) to approximate a target function f^*. The network is represented as a directed acyclic graph, ensuring a feedforward structure without cycles. A network is generally termed *deep* if it contains multiple hidden layers, typically more than one or two, depending on the context.

In deep learning terminology, each individual layer is constructed by multiple computational units referred to as **neurons** or **units**. These units perform linear transformations on their inputs, typically described by weighted summations with additional biases, followed by the application of nonlinear functions called **activation functions**. The role of activation functions is pivotal, introducing necessary nonlinearities that allow neural networks to capture intricate, non-linear patterns present

in data. Frequently employed activation functions include the *sigmoid* (McCulloch and Pitts, 1943), *hyperbolic tangent (tanh)* (LeCun et al., 1998), *rectified linear unit (ReLU)* (Glorot et al., 2011), *leaky ReLU* (Maas et al., 2013), *exponential linear units (ELU)* (Clevert et al., 2016), and *softplus* (Dugas et al., 2001), each chosen according to specific modeling needs and computational considerations.

The intermediate layers between the input and output layers are known as **hidden layers**. These layers are termed "hidden" since their internal representations are typically not directly interpretable and are processed internally within the network. The number of neurons within each hidden layer defines the **width** of the model. Networks with wider layers, having numerous neurons, generally have higher representational capacity, enabling them to model complex relationships. However, wider networks may also be prone to overfitting, as they can memorize training data instead of learning generalizable patterns. In contrast, networks with fewer neurons per layer are simpler, computationally more efficient, but might underfit, failing to adequately capture the complexity inherent in the data.

Neural networks can handle nonlinear modeling through transformations $\phi(\mathbf{x})$ applied to the input features. Such transformations may take various forms, including polynomial expansions, radial basis functions (RBF), Fourier expansions, or sophisticated learned transformations via hidden layers in deep architectures. These transformations enhance the capability of networks, allowing the capturing of complex and intricate data relationships that linear methods alone cannot model effectively.

To effectively learn the parameters $\boldsymbol{\theta}$, neural networks require substantial datasets composed of numerous input-output pairs $(\mathbf{x}, \mathbf{y})$. This rich data provides the essential context from which the network can generalize and infer patterns, ultimately approximating the desired function f^*.

The training process commonly employs the **backpropagation algorithm**, a fundamental computational mechanism in neural network learning. Backpropagation computes gradients of a defined loss function, measuring the discrepancy between

network predictions and actual target values. These gradients are calculated systematically using the chain rule of differentiation, propagated backward through the network from the output layer toward the input layer. The resulting gradients are then utilized in conjunction with optimization algorithms such as stochastic gradient descent (SGD) (Robbins and Monro, 1951), RMSProp (Tieleman and Hinton, 2012), or Adam (Kingma and Ba, 2015) to iteratively adjust network parameters $\boldsymbol{\theta}$, thus progressively minimizing the loss function and improving model accuracy.

Overall, the combination of appropriate architecture selection (depth and width), choice of activation functions, sophisticated input transformations, and efficient training via backpropagation collectively empower deep feedforward neural networks to robustly approximate complex, real-world functions across diverse applications.

3.1.1 Neural Networks Architecture

One of the most important things to consider when building a deep learning (DL) algorithm is determining the architecture of the model. Architecture refers to the overall structure of the network where it involves deciding how many units, the number of layers, and how they are connected to each other. The most common neural network architectures arrange layers in a chain-like structure, where each layer applies a function to the output of the previous layer. The first layer is given by

$$\mathbf{h}^{(1)} = g^{(1)}(\mathbf{W}^{(1)T}\mathbf{x} + \mathbf{b}^{(1)}), \tag{3.1}$$

the subsequent layers are given by

$$\mathbf{h}^{(l)} = g^{(l)}(\mathbf{W}^{(l)T}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}) \text{ for every } i = 2, 3, \ldots, L, \tag{3.2}$$

and the final output layer is given by

$$\mathbf{O}^{(L+1)} = g^{(L+1)}(\mathbf{W}^{(L+1)T}\mathbf{h}^{(L)} + \mathbf{b}^{(L+1)})), \tag{3.3}$$

where $\mathbf{h}^{(l)}$ is the vector of p_l nodes, $\mathbf{W}^{(l)T}$ is the $p_l \times p_l$ weight matrix, $\mathbf{b}^{(l)}$ is the vector of biases, and $g^l(\cdot)$ is the known linear or non-linear activation (link) function

that is applied to each layer component-wise, e.g., sigmoid or ReLU. The final layer gives the modeled mean response, $\mathbf{O}^{(L+1)} = f_{NN}(\mathbf{x}) = E[\mathbf{y}]$ (Goodfellow et al., 2016; Zhan and Datta, 2024).

The unknown weights or parameters are estimated typically using backpropagation based on the ordinary least squares (OLS) loss, i.e.,

$$L(f(\mathbf{x}), \mathbf{y}) = \sum_{i=1}^{n} (y_i - f(x_i))^2. \tag{3.4}$$

Since the estimation of parameters for large data sets are computationally expensive, one applies mini-batching where the data are split into smaller disjoint sets called mini-batches, and cycles among the mini-batches for each iteration or epochs (Zhan and Datta, 2024).

Training aims to identify a neural network function that best approximates the true underlying process by minimizing the prediction error across the training data. Specifically, the goal is to find the optimal predictor $\hat{f}^{\text{opt}}(\mathbf{x})$ that approximates the true process $f(\mathbf{x})$ through a neural network model:

$$\hat{f}_{NN}^{\text{opt}}(\mathbf{x}) = \underset{f_{NN} \in \mathcal{F}}{\arg\min}\, L(f_{NN}(\mathbf{x}), \mathbf{y}), \tag{3.5}$$

where $f_{NN}(\cdot) \in \mathcal{F}$ denotes a function from the family of neural networks, and $\mathbf{x}$ and $\mathbf{y}$ are the predictor and response variables, respectively. In practice, finding the exact optimal neural network predictor $f_{NN}^{\text{opt}}(\mathbf{x})$ is infeasible, as the true data-generating process $f(\mathbf{x})$ is typically unknown and may involve noise or latent structure (W. Chen et al., 2022).

3.1.2 Gradient-Based Learning

Neural network (NN) design and training follow many of the principles used in other machine learning and regression models. In particular, they often rely on **gradient descent** methods for optimization. The key difference between NNs and many other models is that the **nonlinear activation functions** in NNs make the overall loss function *nonconvex* (Goodfellow et al., 2016). As a result, training a neural network

involves finding parameters that (approximately) minimize a nonconvex objective, which generally has no global convergence guarantees and is very sensitive to the initial parameter values.

Forward and Backward Passes. Training proceeds iteratively on mini-batches of data. In each iteration, or *forward pass*, an input vector $\mathbf{x}$ is fed through the layers of the network, producing an output prediction $\hat{f}(\mathbf{x}) = \hat{\mathbf{y}}$.

A scalar-valued loss, such as the mean squared error or cross-entropy, $L\Big(\hat{\mathbf{y}}, \mathbf{y}\Big)$, is then computed by comparing the prediction $\hat{\mathbf{y}}$ with the true label $\mathbf{y}$. If we represent the NN as $f_{NN}\Big(\mathbf{x}; \boldsymbol{\theta}\Big)$ with parameters $\boldsymbol{\theta}$ (weights and biases), the training objective is

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{n=1}^{N} L\Big(f_{NN}(\mathbf{x}_n; \boldsymbol{\theta}), \mathbf{y}_n\Big). \tag{3.6}$$

To adjust $\boldsymbol{\theta}$ and drive the loss down, the network uses the *back-propagation* algorithm, which applies the **chain rule** to compute the gradient

$$\nabla_{\boldsymbol{\theta}} L\Big(f_{NN}(\mathbf{x}; \boldsymbol{\theta}), \mathbf{y}\Big)$$

efficiently. Backpropagation efficiently computes gradients by applying the chain rule layer by layer, collecting partial derivatives with respect to all parameters. These derivatives can then be used to update the weights and biases in the direction that reduces the loss.

Definition 3.1.2. (Chain Rule of Multivariate Functions) Let $\mathbf{x} \in \mathbb{R}^m$, $\mathbf{y} \in \mathbb{R}^n$, $g : \mathbb{R}^m \to \mathbb{R}^n$, and $f : \mathbb{R}^n \to \mathbb{R}$. Suppose that $\mathbf{y} = g(\mathbf{x})$ and $z = f(\mathbf{y})$. Then

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j} \frac{\partial y_j}{\partial x_i}.$$

In vector notation,

$$\nabla_{\mathbf{x}} z = \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)^T \nabla_{\mathbf{y}} z,$$

where $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ is the $n \times m$ Jacobian matrix of g.

Gradient Descent and Learning Rates. Once the gradient is obtained, the parameters are updated according to a rule of the form

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} L\Big(f_{NN}(\mathbf{x}; \boldsymbol{\theta}), \mathbf{y}\Big),$$

where η is the **learning rate**. This hyperparameter controls the size of each step in parameter space:

- **Large learning rate:** The updates are significant, which may help the model escape shallow local minima, but can also cause divergence or oscillations around minima if the step size is too big.
- **Small learning rate:** The updates are more precise, but convergence can be slow and the model may get trapped in poor local minima or saddle points.

Choosing an appropriate learning rate is crucial for stable and efficient training. In practice, strategies such as *learning rate decay* or adaptive learning rate optimizers (e.g., Adam, RMSProp) are used to improve performance.

Regularization. Because neural networks typically contain large numbers of parameters, they are prone to overfitting. To combat this, **regularization** techniques are incorporated into the training process to encourage simpler models that generalize better. Common regularization methods include:

- **Weight decay (L2 regularization):** Adds a penalty term $\lambda\|\boldsymbol{\theta}\|^2$ to the loss function, shrinking weights towards zero.
- **L1 regularization:** Encourages sparsity by adding $\lambda\|\boldsymbol{\theta}\|_1$.
- **Dropout:** Randomly "drops" certain neuron outputs during training to prevent co-adaptation of features.
- **Early stopping:** Monitors performance on a validation set and halts training when overfitting is detected.

Overall, neural networks learn by iteratively applying *forward propagation* to compute predictions and a loss, then using the *back-propagation* algorithm to efficiently compute gradients of that loss with respect to all parameters. These gradients are used in a gradient-descent-based update rule, controlled by a learning rate, and typically combined with regularization methods to improve generalization and prevent overfitting. Although there is no guarantee of finding a global minimum of the nonconvex loss function, these procedures have proven remarkably successful in practice for many real-world applications.

3.2 Graph Neural Networks

Conventional deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have achieved substantial successes primarily in Euclidean data-based tasks involving structured data representations like images, text, and sequential data. However, numerous applications naturally possess or are optimally represented with graph structures. Examples include social networks, recommendation systems, drug discovery, citation networks, and even computer vision tasks where relationships among objects are crucial.

Graph structures represent complex and non-Euclidean relationships effectively, capturing pairwise interactions among nodes and their intrinsic interdependencies. Graph Neural Networks (GNNs) have emerged as the most successful approach in handling these graph-structured data due to their capability to model complex relational information by encoding underlying symmetries and interactions among heterogeneous entities (L. Wu et al., 2022).

Graph Neural Networks specifically operate by iteratively updating node representations by aggregating information from neighboring nodes and past node representations. This iterative aggregation and update process, commonly referred to as message passing, allows GNNs to capture rich local and global structural information inherent in graph data. GNNs' expressive power and their ability to generalize conventional deep

learning paradigms like convolution and recurrence make them a powerful framework for a variety of learning tasks (L. Wu et al., 2022).

As a motivating example, consider predicting the function of a protein in a biological network. Each protein is represented as a node, and edges represent interactions between proteins. GNNs can learn to aggregate information from each protein's neighbors to improve prediction accuracy by capturing the underlying structure of the network.

The general framework of GNN defines a graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ with $N = |V(\mathcal{G})|$ nodes. Denote its adjacency matrix by $\mathbf{A} \in \mathbb{R}^{N \times N}$, and the node feature matrix by $\mathbf{X} \in \mathbb{R}^{N \times q}$. A GNN layer produces new node embeddings $\mathbf{H} \in \mathbb{R}^{N \times k}$ by aggregating and transforming each node's own features and the features of its neighbors.

A widely used definition of the graph convolution layer is given by:

$$\mathbf{H} \;=\; \sigma\Big(\tilde{\mathbf{D}}^{-\frac{1}{2}}\,\tilde{\mathbf{A}}\,\tilde{\mathbf{D}}^{-\frac{1}{2}}\,\mathbf{X}\,\mathbf{W}\Big), \tag{3.7}$$

where $\tilde{\mathbf{A}} \;=\; \mathbf{A}+\mathbf{I}_N, \quad \tilde{\mathbf{D}}_{ii} \;=\; \sum_j \tilde{\mathbf{A}}_{ij}$ is the diagonal degree matrix 2.2.10, $\mathbf{W} \in \mathbb{R}^{q \times k}$ is a learnable weight matrix, and $\sigma(\cdot)$ is a nonlinear activation (e.g., ReLU). Adding the identity matrix $\mathbf{I}_N$ ensures that each node has a self-loop, and normalizing by reciprocal square root of the degree matrix $\tilde{\mathbf{D}}^{-\frac{1}{2}}$ enforces consistent scaling of aggregated neighbor features (Nguyen, 2022). This normalization mitigates the issue of degrees of explosion or vanishing nodes, ensuring a stable propagation of messages between layers.

In a more general message passing view, each node v updates its embedding by collecting information from its neighborhood $\mathcal{N}(v)$:

$$\mathbf{h}_v^{(l+1)} \;=\; \gamma\Big(\{\mathbf{h}_u^{(l)} : u \in N(v)\}\Big) \;\oplus\; \mathbf{h}_v^{(l)},$$

where:

- $\gamma(\cdot)$ is a permutation-invariant function that aggregates features from the neighboring nodes of v such as mean, sum, or a learnable attention-based mechanism.

- $\oplus$ is an operation (e.g., concatenation, sum, or another function) to combine the aggregated neighbor messages with the node's current hidden state.

Extensions and Properties

- **Multi-head attention (Graph Attention Networks)**: Instead of using a fixed adjacency matrix, attention weights between pairs of nodes are dynamically learned. This allows the model to attend more strongly to the most relevant neighbors.

- **Graph Convolutional Networks (GCN)**: They often rely on the normalized adjacency matrix for stable training and can stack multiple layers to capture higher-order interactions, leading to richer node or graph-level representations.

- **Applications**: Node-level tasks (classification, regression), edge-level tasks (link prediction), and graph-level tasks (classification, regression, clustering) can all leverage GNNs with appropriate architecture choices and pooling strategies (Wiltschko, n.d.; Z. Wu, S. Pan, F. Chen, et al., 2021).

Graph-level Tasks Graph-level tasks involve predicting properties or labels for the entire graph. To achieve this, graph-level representations are derived through graph pooling layers, which condense node-level representations learned from convolutional layers into a unified embedding. These representations summarize the essential characteristics of the input graph, allowing accurate predictions about properties such as molecular toxicity, bioactivity, or graph categorization tasks like determining the nature of citation networks (L. Wu et al., 2022; Z. Wu, S. Pan, F. Chen, et al., 2021). Techniques used for graph-level representations typically involve simple aggregation

methods (mean, sum, or max pooling) or advanced methods like attention-based and hierarchical clustering-based pooling (L. Wu et al., 2022).

Node-level Tasks Node-level tasks focus on classifying individual nodes within a graph or predicting node-specific properties. Examples include predicting user roles in social networks, labeling nodes in citation graphs, or assessing attributes of biological entities in molecular graphs. Node representations are learned directly by aggregating information from immediate neighbors and possibly multi-hop neighborhoods across iterative layers of GNN architectures. Techniques like GraphSage and Graph Attention Networks (GAT) employ sampling and attention mechanisms respectively to weigh the importance of different neighboring nodes dynamically (Z. Wu, S. Pan, F. Chen, et al., 2021).

Edge-level Tasks Edge-level tasks aim at predicting relationships or interactions between pairs of nodes, such as inferring missing links in a network or classifying edge types in heterogeneous graphs. This involves deriving relational representations that explicitly encode pairwise interactions. Typically, these tasks start with fully connected complete graphs and iteratively prune edges to obtain sparse graph structures based on predicted edge weights or connection strengths (L. Wu et al., 2022; Z. Wu, S. Pan, F. Chen, et al., 2021).

Technical Considerations and Limitations GNNs face several technical challenges including over-smoothing, scalability to large-scale graphs, interpretability, and adversarial robustness. Over-smoothing occurs when node representations become indistinguishable after multiple iterations of message passing. Recent approaches mitigate this by employing techniques such as residual connections, dropout mechanisms, and normalization layers (L. Wu et al., 2022).

Scalability remains a concern due to the significant memory requirements of traditional GNN methods. Recent developments in scalable GNNs employ node-wise sampling, layer-wise sampling, and graph-wise sampling strategies to manage computational resources efficiently (L. Wu et al., 2022).

Interpretability and adversarial robustness of GNNs are critical for real-world deployment, particularly in sensitive applications like drug discovery and social networks. Methods leveraging white-box approaches (gradient-based explanations) and black-box approximation strategies (interpretable surrogate models) have been proposed to enhance interpretability, while adversarial training and certified robustness techniques help bolster GNN reliability against malicious perturbations (L. Wu et al., 2022).

3.3 Related Works on Spatio-Temporal Neural Networks

In parallel to advancements in Gaussian processes, neural networks and deep learning architectures have gained traction for spatial prediction, precisely because they can capture complex, nonlinear relationships that may be missed by linear or stationarity assumptions. The growing interest in combining deep neural networks with spatial and spatio-temporal processes has motivated several notable contributions. However, a key limitation of many early attempts at neural-network-based spatial prediction was the lack of an explicit spatial covariance component.

(Wang et al., 2019) propose a Nearest-Neighbor Neural Network (4N) abbreviated as 4N process that preserves the notion of local dependence from Gaussian processes but embeds flexible nonlinear predictors. By reusing neighbor sets and partial correlation ideas from NNGP, the 4N process allows for a fully nonparametric link to the nearest-neighbor kriging concept. The 4N model extends the link function $f(\cdot)$ used in kriging by replacing the linear predictor with an MLP. So that the (4N) model

$$Y_i = f(\mathbf{X}_i) + \epsilon_i$$

uses $n \times p$ covariate matrix $\mathbf{X} = (\mathbf{X}_1^T, \mathbf{X}_2^T, \ldots, \mathbf{X}_n^T)^T$ where $\mathbf{X}_i,\ i = 1, 2, \ldots, n$ is associated with corresponding $Y_{\mathcal{N}_i}$ response based on the nearest neighbor set to $s_i \in \mathbb{R}^d$. Key contribution is that they use the kriging predictions as the training set for the MLP and have achieved greater performance than the NNGP model alone.

Similarly, (Zhan and Datta, 2024) develop a framework for neural networks in geostatistics that treats the linear mean model in a Gaussian process as a nonlinear neural network. Their approach, called NN-GLS, preserves the Gaussian process interpretation—thus allowing standard kriging, parameter estimation, and spatial uncertainty quantification—while swapping out the usual linear regression form for a deep neural network. However, due to Gauss-Markov theorem, Generalized Least Squares (GLS) loss, given by

$$L_n(f) = \frac{1}{n}(\mathbf{Y} - \mathbf{f}(\mathbf{X})^T \mathbf{Q}(\mathbf{Y} - \mathbf{f}(\mathbf{X}))), \tag{3.8}$$

is more efficient for parameter estimation of models estimating dependent data. Here $\mathbf{Q}$ is the working precision matrix that is equivalent to the inverse of the covariance matrix $\mathbf{C}$. However, the use of GLS loss introduces multiple computational issues for both mini-batching and backpropagation techniques for neural networks predictions since GLS loss is not additive over the data units. This requires a reliable method that allows one to transform the GLS loss to Ordinary Least Squares (3.4).

The framework that they have proposed is to combine the nearest neighbor kriging with neural networks by taking the kriging weights to obtain decorrelated responses from the neural networks predictions, thus avoiding the need for inverting $N[i] \times N[i]$ nearest neighbor covariance matrix. They have shown that the decorrelation process becomes graph convolution in the nearest neighbor adjacency matrix. By exploiting a graph-neural-network view of the nearest-neighbor precision matrix, they demonstrate how mini-batching and backpropagation can be done efficiently, even for large point-referenced datasets.

(W. Chen et al., 2022) offers a unifying perspective by embedding spatial coordinates into a set of basis functions and then feeding them into a deep neural network. Neural Networks models require a large number of data points and covariates to be able to perform effectively. However, geospatial data often have constrained observations and relevant covariate information. By embedding spatial coordinates into a set of basis functions, one can obtain arbitrarily scalable additional covariates, since we can use any number of basis functions. This approach retains the interpretability of kriging while reaping the expressive benefits of deep learning, thereby improving predictive performance for non-Gaussian and non-stationary spatial data.

As neural network models grow increasingly complex, the need for interpretability has become a central concern, particularly in high-stakes applications where transparency and trust are essential. Beyond "pure" spatial settings, these nearest-neighbor and neural approaches have also proven fruitful in spatio-temporal data contexts—such as traffic flow predictions, crime rate prediction, industrial process optimization, and social network analysis—where both time and location introduce dependence structure. In (Tang et al., 2023), the authors develop a framework to identify the most influential spatial and temporal subgraphs underlying the predictions of a spatio-temporal GNN, combining a graph information bottleneck objective with position-aware attention mechanisms. They have developed unified spatio-temporal graph (STG) encoder and decoder that generate explainable and generalizable STG representations. This line of work responds to the rising demand for transparent models—especially in urban analytics and resource allocation scenarios—where interpretability is central to user trust and actionable insights.

Complementing these developments is (Z. Wu, S. Pan, Long, et al., 2019), which highlights the advantages of convolution-based models for spatio-temporal graph prediction. By learning an adaptive adjacency matrix to uncover hidden spatial dependencies, and by using dilated causal convolutions, Graph WaveNet captures both long-range temporal patterns and flexible spatial interactions. These ideas align

well with recent neural approaches that handle large-scale sensor networks in traffic forecasting, ecology, and other real-world spatial applications.

The core idea remains to circumvent expensive matrix factorizations (often $O(n^3)$) in naive form) by limiting conditional dependencies to local neighbors in space and/or time, then embedding additional nonlinearities via neural networks.

Collectively, these lines of research illustrate how nearest-neighbor strategies serve as an elegant mechanism to preserve much of the interpretability and uncertainty quantification advantages of Gaussian processes while sharply reducing the computational load. Recent extensions that incorporate neural-network-based function approximators add further flexibility for capturing non-stationary, non-Gaussian, or highly complex spatial (and spatio-temporal) structure. A unifying perspective across these approaches is that nearest-neighbor models serve as fundamental building blocks for spatial and spatio-temporal modeling. When combined with flexible neural network architectures, these models gain the capacity to capture complex, non-stationary, and nonlinear spatial dependencies, while preserving the interpretability and computational advantages of localized approximation.

CHAPTER 4

METHODOLOGY AND RESULTS

In this chapter, we develop a hybrid framework that integrates spatio-temporal kriging with a Graph Neural Network (GNN) to interpolate and forecast air quality measurements (PM_{10}). The approach leverages both classical geostatistical techniques and modern deep learning methods. The core idea is to leverage kriging predictions as input features that are skip-connected to the output layer of a GNN, thereby enriching the model with spatial statistical structure. We have used Python packages Torch, Pandas, and Numpy to implement the method on Tennessee Technological University High Performance Computing Center. All codes can be found in the GitHub repository: https://github.com/Belguutei-Ariuntugs/Scalable-Spatio-Temporal-Prediction-Using-Sparse-Gaussian-Processes-and-Graph-Neural-Networks

To estimate the parameters of the covariance function, we employ a Weighted Least Squares (WLS) approach based on the empirical covariance. We first fit the purely spatial and purely temporal covariance components independently, and then estimate the space-time interaction parameter β, conditioning on the fixed spatial and temporal covariance parameters.

4.1 PM_{10} Data Set Description

The dataset used is taken from The European air quality database (The European Environmental Agency, 2024) and further discussed in (Gräler et al., 2016). We focus on PM_{10} particulate matter with a diameter less than $10\,\mu$m measured at rural background stations from 2005 to 2006 in Germany. (Gräler et al., 2016) found that only background stations should be considered rather than urban or traffic stations, as the underlying processes in these environments are fundamentally different.

PM is defined as a "mixture of solid particles and liquid droplets found in the air. Some particles, such as dust, dirt, soot, or smoke, are large or dark enough to be seen with the naked eye. Others are so small they can only be detected using an electron microscope" (US Environmental Protection Agency, 2024). PM_{10} refers to microscopic solid particles or liquid droplets small enough to be inhaled, posing significant health risks. Particles measuring less than 10 micrometers can penetrate deep into the lungs, and some might even enter the bloodstream. These particles vary in size, shape, and chemical composition, and originate both from direct sources (e.g., construction activities, unpaved roads, agricultural fields, smokestacks, or wildfires) and from secondary formation through complex chemical reactions involving pollutants such as sulfur dioxide and nitrogen oxides emitted by power plants, industrial processes, and vehicles.

Data Preprocessing The raw dataset undergoes several preprocessing steps to ensure consistency and proper scaling, which are essential for both the spatio-temporal kriging and Graph Neural Network (GNN) analyses. The preprocessing steps are as follows:

1. **Data Loading and Initial Processing:** The dataset is imported from an Excel file. The time column is converted into a `datetime` format, and a unique node identifier is created by concatenating the spatial coordinates to uniquely distinguish different measurement locations.

2. **Handling Covariates:** The routine checks for the presence of extra covariates such as **station altitude** and **annual mean PM10** and missing values are handled by filling them with the mean.

3. **Time Series Pivoting and Grouping:** A pivot table is constructed where each row corresponds to a unique node and each column represents a time instance of PM_{10} measurements. Missing values in the time series are imputed

using the overall mean of the data, ensuring that each node has a complete time series.

4. **Extraction of Static Node Features:** Static features—including spatial coordinates and additional covariates—are extracted by grouping the data by node and selecting the first occurrence of each feature. These features are converted from a Pandas DataFrame to a NumPy array and then to a PyTorch tensor for compatibility with the GNN model.

5. **Coordinate Conversion:** To ensure compatibility with the spatio-temporal covariance function used in kriging, spatial coordinates (originally in meters) are converted to kilometers. Time values are also standardized by converting them into days relative to a earliest observation.

4.2 Neighbor Selection for Kriging

We define the set of n observed realizations from the random field as $S = \{z_1, z_2, \ldots, z_n\}$. This set is partitioned into training and test subsets:

$$\mathscr{T} = \{z_1, z_2, \ldots, z_r\} \quad \text{and} \quad \mathscr{E} = \{z_{r+1}, z_{r+2}, \ldots, z_n\},$$

where $r = \lfloor 0.8n \rfloor$ denotes the number of training samples and the remaining $n - r$ samples form the test set.

For any new spatio-temporal location $(\mathbf{s}^*, t^*) = \ell^* \in \mathcal{L} \subset \mathbb{R}^d \times \mathbb{R}$, we consider two neighbor selection strategies for kriging.

1. **Overall Covariance Ranking:** For each prediction point $(\mathbf{s}^*, t^*)$, we compute the covariance between $(\mathbf{s}^*, t^*)$ and every training observation using the specified non-separable spatio-temporal covariance function. The m observations with the highest covariance values are then selected. In our experiments, we set $m = 36$, resulting in the construction of a 36×36 covariance matrix $\mathbf{C}$ for use in the kriging prediction.

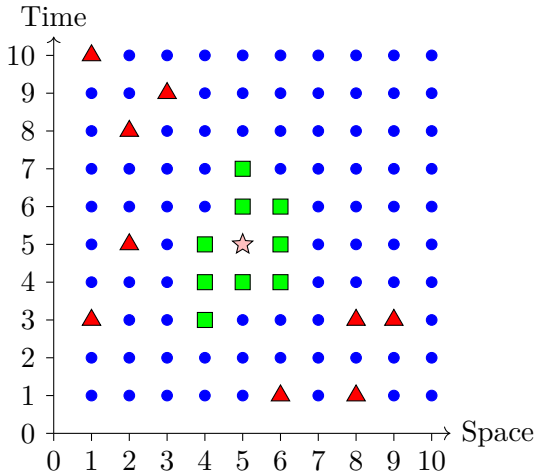
2. **Combined Spatial and Temporal Selection:** Alternatively, we select neighbors by considering spatial and temporal proximity separately. For a prediction point $(\mathbf{s}^*, t^*)$, let $q = \sqrt{m}$ (e.g., $q = 6$ when $m = 36$). We first identify the q nearest neighbors based on spatial distance and the q nearest neighbors based on temporal difference. The union of these two candidate sets forms the preliminary neighborhood:

$$N[\ell^*] = \{\ell'_1, \ell'_2, \ldots, \ell'_k\},$$

where $k \leq 2q$ denotes the number of unique neighbors. If $k < m$, additional neighbors are selected based on overall covariance ranking to reach m observations. Conversely, if $k > m$, the set is pruned by retaining only the m observations with the highest covariance values.

Figure 5

10×10 *Spatio-temporal data set with all triangular red points are the test points and circular blue points are training set for prediction. For the star shaped point (5,5) in pink, square green points are the 9 nearest neighbors set* $N[(5,5)]$ *evaluated by the covariance function.*



4.3 Non-separable Spatio-temporal Covariance for Kriging

We adopt a fully symmetric, yet generally non-separable, spatio-temporal covariance function:

$$C_{FS}(\mathbf{h}, u) = \frac{1-v}{1+a|u|^{2\alpha}} \left(\exp\left(-\frac{c\|\mathbf{h}\|}{(1+a|u|^{2\alpha})^{\beta/2}} \right) + \frac{v}{1-v} \delta_{\mathbf{h}=0} \right), \tag{4.1}$$

as introduced in (2.19), where $\delta_{\mathbf{h}=0}$ is the Dirac delta function accounting for the spatial nugget effect. By WLS, the parameters of the covariance function were estimated to be

1. Nugget: $v = 0.0977$.
2. Spatial scale: $c = 0.0013$.
3. Temporal scale: $a = 0.493014$.
4. Temporal smoothness: $\alpha = 0.874308$.
5. Space-time interaction: $\beta = 1.0$.

We construct the m nearest-neighbor covariance matrix $\mathbf{C}_{N[l^*],N[l^*]}$ of size $m \times m$ and the vector $\mathbf{c}_0$ of cross-covariances:

$$(\mathbf{C}_{N[l^*],N[l^*]})_{ij} = C_{FS}(\mathbf{s}'_i - \mathbf{s}'_j, t'_i - t'_j) = C_{FS}(\mathbf{h}', u') \tag{4.2}$$

$$(\mathbf{c}_0)_i = C_{FS}(\mathbf{s}'_i - \mathbf{s}^*, t'_i - t^*) = C_{FS}(\mathbf{h}^\star, u^\star), \tag{4.3}$$

for every $(\mathbf{s}'_i, t'_i), (\mathbf{s}'_j, t'_j) \in N[\ell^*]$

Solving the linear system $\mathbf{C}\boldsymbol{\omega} = \mathbf{c}_0$ via Cholesky factorization yields the kriging weights $\boldsymbol{\omega}$, leading to the spatio-temporal kriging predictor:

$$\hat{z}_* = \sum_{i=1}^{m} \omega_i z_i \quad \text{where} \quad \boldsymbol{\omega} = \mathbf{C}^{-1}\mathbf{c}_0.$$

This nearest neighbor–based spatio-temporal kriging approach substantially reduces the computational burden. While conventional spatio-temporal kriging requires $O(n^3)$ operations, restricting the analysis to $m \ll n$ nearest neighbors lowers the complexity

to $O(nm^3)$. For instance, if $n = 23231$ and $m = 25$, the computational cost decreases from $O(23231^3)$ to $O(23231 \cdot 25^3)$.

4.4 Graph Neural Network Forecaster

In this thesis, the GNN implementation was inspired by Graph WaveNet done by (Z. Wu, S. Pan, Long, et al., 2019). Given node features $\mathbf{X} \in \mathbb{R}^{N \times r}$, where r is the number of covariates, and a learned adjacency matrix $\mathbf{A}$, we employ a GNN architecture such as a GCN or GCN-like layer. A standard GCN layer is defined as:

$$\mathbf{Z} \;=\; \sigma\Big(\tilde{\mathbf{D}}^{-1/2}\,\tilde{\mathbf{A}}\,\tilde{\mathbf{D}}^{-1/2}\,\mathbf{X}\,\mathbf{W}\Big), \tag{4.4}$$

where:

- $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ adds self-loops to the graph;
- $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$ is the corresponding degree matrix; and
- $\sigma(\cdot)$ is a non-linear activation function (e.g., ReLU).

Stacking multiple such layers allows the model to capture higher-order spatial dependencies by aggregating information from multi-hop neighborhoods.

4.5 Dynamic Adjacency Matrix Computation via Pretrained Covariance Function

To couple kriging with a GNN, we compute a *dynamic adjacency matrix* $\mathbf{A} \in \mathbb{R}^{N \times N}$ based on a spatial covariance subcomponent:

$$\mathbf{A}_{ij} \;=\; \max\Big(0,\; (1 - v)\,\exp(-c\,\|\mathbf{s}_i - \mathbf{s}_j\|)\Big), \tag{4.5}$$

where $\|\cdot\|$ denotes the Euclidean distance (in kilometers) between locations $\mathbf{s}_i$ and $\mathbf{s}_j$. To promote sparsity, we retain only the top-k strongest edges for each node, resulting in a sparse adjacency matrix and an associated edge set $\{(i, j)\}$ used in the graph-based neural layer.

4.5.1 Temporal Convolution and Residual Stacking

We handle temporal sequences of length T_{in} using one-dimensional (1D) dilated convolutions or standard temporal CNN layers. The input sequence of each node, $\mathbf{x}_n$, is processed as:

$$\mathbf{x}_n \;\mapsto\; \text{Conv1D}_{\text{temp}}\Big(\mathbf{x}_n\Big) \;\to\; \text{BatchNorm} \;\to\; \text{ReLU} \;\to\; \cdots$$

with residual connections. After applying temporal pooling or selecting the final hidden states, each node is represented by a hidden embedding, which is then passed to the spatial GCN layers.

Hybrid Kriging Skip Connection

The model outputs a per-node multi-step forecast:

$$\hat{y}_{\text{GNN}} \;\in\; \mathbb{R}^{N\times H}$$

for horizon H. We augment this with the kriging prediction $\hat{y}_{\text{krig}}$ to obtain:

$$\hat{y}_{\text{final}} \;=\; \hat{y}_{\text{GNN}} \;+\; \hat{y}_{\text{krig}},$$

treating $\hat{y}_{\text{krig}}$ as a "skip connection" that conveys classical geostatistical structure.

4.5.2 Loss Function and Training

The model is trained by minimizing a smooth loss (e.g. Huber or L_1) between the final forecast $\hat{y}_{\text{final}}$ and the true target y:

$$\mathcal{L}(\theta) \;=\; \text{SmoothL1}\Big(\hat{y}_{\text{final}}(\theta),\; y\Big), \tag{4.6}$$

with respect to the GNN parameters θ. Covariance parameters (v, c, a, α, β) may be fixed or fit offline. The adjacency $\mathbf{A}$ and the nearest-neighbor sets are dynamically recomputed using the spatial covariance subcomponent.

Evaluation Metrics. We assess model performance using several standard error metrics:

- **MSE** (Mean Squared Error): Measures the average of the squared differences between predicted and true values.
- **RMSE** (Root Mean Squared Error): The square root of MSE; interpretable in the same units as the original data.
- **MAE** (Mean Absolute Error): Measures the average absolute difference between predictions and true values.
- **medAE** (Median Absolute Error): The median of the absolute differences; more robust to outliers than MAE.
- **MASE** (Mean Absolute Scaled Error): A scale-free error metric that allows comparison across datasets; values greater than 1 indicate worse performance than naive forecasts.
- **SMAPE** (Symmetric Mean Absolute Percentage Error): A percentage-based error metric that symmetrically penalizes over- and under-predictions.

Table 1

Forecasting performance of the hybrid kriging-GNN model using 36 nearest neighbors for extrapolation tasks. Lower error metrics across all horizons indicate improved predictive accuracy over classical kriging.

Forecast Length	MSE	RMSE	MAE	medAE	MASE	SMAPE
1	42.1608	6.4931	4.7806	3.9233	1.9507	26.864 9%
2	42.5130	6.5202	4.7693	3.8523	1.9463	26.2732%
4	42.3762	6.5097	4.7612	3.8670	1.9442	26.3850%
5	42.5394	6.5222	4.8273	4.0728	1.9732	27.794 4%

Table 2

Baseline kriging model performance using 36 nearest neighbors for extrapolation. Compared to the hybrid model, kriging exhibits significantly higher error metrics, highlighting the benefit of GNN augmentation.

Forecast Length	MSE	RMSE	MAE	medAE	MASE	SMAPE
1	149.2285	12.2159	10.7608	10.8153	4.3910	91. 7002%
2	149.2166	12.2154	10.7595	10.8191	4.3909	91.6214%
4	148.9959	12.2064	10.7517	10.8191	4.3902	91.4739%
5	148.8269	12.1995	10.7456	10.8178	4.3923	91. 3940%

Table 3

Forecasting performance of the hybrid model with 25 nearest neighbors. Error metrics remain stable across horizons and generally outperform kriging.

Forecast Length	MSE	RMSE	MAE	medAE	MASE	SMAPE
1	42.4190	6.5130	4.7765	3.8781	1.9491	26.4695%
2	44.7660	6.6907	4.8515	3.6696	1.9799	25.6772%
4	42.4326	6.5140	4.8001	3.9741	1.9600	27.1710%
5	42.4072	6.5121	4.7827	3.9185	1.9550	26.786 8%

Table 4

Kriging baseline performance with 25 nearest neighbors. Compared to hybrid models, error remains considerably higher for all forecast lengths.

Forecast Length	MSE	RMSE	MAE	medAE	MASE	SMAPE
1	150.3225	12.2606	10.8052	10.8576	4.4091	92.3119%
2	150.3123	12.2602	10.8040	10.8595	4.4091	92.2337%
4	150.0949	12.2513	10.7963	10.8595	4.4084	92.0878%
5	149.9274	12.2445	10.7902	10.8592	4.4106	92. 0087%

Table 5

Interpolation performance for the hybrid model using with 9, 16, 25, and 36 nearest neighbors. Error metrics are significantly lower, demonstrating the model's capacity for fine-scale predictions within the training region.

Neighbors	MSE	RMSE	MAE	medAE	MASE	SMAPE
9	3.8264	1.9561	1.1904	0.8384	0.2212	8.6034%
16	4.2683	2.0660	1.4096	1.1454	0.2619	9.8488%
25	3.4768	1.8646	1.0545	0.7223	0.1959	7.6408%
36	3.9214	1.9803	1.1273	0.7696	0.2094	8.3504%

Table 6

Kriging interpolation performance with 9, 16, 25, and 36 neighbors. Error metrics are substantially higher than the hybrid model, particularly in MAE and SMAPE.

Neighbors	MSE	RMSE	MAE	medAE	MASE	SMAPE
9	385.6802	19.6387	16.8855	15.4978	3.1370	193.9810%
16	385.5786	19.6362	16.8822	15.4758	3.1364	193.8963%
25	385.4973	19.6341	16.8787	15.4752	3.1357	193.7600%
36	385.3673	19.6308	16.8748	15.4647	3.1350	193.5714%

4.6 Conclusion and Future Directions

In this work, we have introduced a novel hybrid framework that combines spatio-temporal kriging with a Graph Neural Network (GNN) to enhance the interpolation and forecasting of PM_{10} air quality measurements. By leveraging the strengths of both classical geostatistical methods and modern deep learning techniques, our approach effectively captures complex spatio-temporal dependencies that are not fully addressed by traditional methods.

The experimental results presented in this thesis demonstrate that our hybrid kriging-GNN model consistently outperforms the baseline kriging approach. Specifically, the hybrid model yields significantly lower error metrics across multiple forecast horizons, as evidenced by improved value of MSE, RMSE, MAE, medAE, MASE, and SMAPE (see Tables 1–5). The integration of kriging predictions as a skip connection within the GNN architecture enables the model to incorporate spatial statistical structure while simultaneously learning complex temporal dynamics. Additionally, the dynamic adjacency matrix computed from the pretrained covariance function further enhances the model's ability to capture spatial relationships among monitoring stations.

4.6.1 Future Directions

While the proposed methodology shows considerable promise, several avenues for future research remain:

1. **Alternative Covariance Models:** Future work could explore different covariance functions, including non-stationary, anisotropic, or locally adaptive models. Such models may more accurately capture the underlying spatio-temporal variability and further improve prediction performance.

2. **Advanced Nearest Neighbor Selection:** Our current strategy for neighbor selection combines spatial and temporal proximity. Future research could

investigate adaptive or machine learning-based methods to identify the most relevant neighbors, thus optimizing the kriging component.

3. **Integration of Additional Data Sources:** Incorporating supplementary exogenous variables—such as meteorological data or traffic information—could enhance the forecasting capabilities of the hybrid model. A richer set of input features would allow for more nuanced modeling of the factors affecting air quality.

4. **Extension to Other Environmental Applications:** The hybrid framework developed in this thesis is generalizable and can be applied to other spatio-temporal datasets. Future studies could extend this approach to different types of pollutants or environmental phenomena, such as disease spread, economic impacts, or the consequences of natural disasters, thereby validating its versatility and robustness.

In summary, the proposed hybrid kriging-GNN model provides a powerful tool for spatio-temporal interpolation and forecasting of air quality data, significantly outperforming traditional kriging methods. The promising results encourage further exploration into alternative covariance models and refined neighbor selection strategies, which could lead to even greater improvements in predictive accuracy and broader applicability across environmental and spatio-temporal modeling tasks. As climate change accelerates, accurate prediction methods for the dispersion of pollutants are critical to identifying major pollution sources and regions disproportionately affected by its devastating impacts. This work can assist researchers and policymakers in creating accurate, high-resolution maps that support preparedness and mitigation efforts. Moreover, such maps are essential for insurance agencies and companies to redirect investments away from harmful activities and make informed, data-driven decisions.

REFERENCES

[1] Brockwell, P., & Davis, R. (1991). *Time series: Theory and methods: Theory and methods*. Springer New York. https://books.google.com/books?id=ZW_ThhYQiXIC

[2] Chen, W., Li, Y., Reich, B. J., & Sun, Y. (2022). Deepkriging: Spatially dependent deep neural networks for spatial prediction. https://arxiv.org/abs/2007.11972

[3] Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (elus). *Proceedings of the International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1511.07289

[4] Datta, A., Banerjee, S., Finley, A. O., & Gelfand, A. E. (2016). Hierarchical nearest-neighbor gaussian process models for large geostatistical datasets. https://arxiv.org/abs/1406.7343

[5] Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., & Garcia, R. (2001). Incorporating second-order functional knowledge for better option pricing. *Advances in Neural Information Processing Systems (NeurIPS)*, 472–478. https://proceedings.neurips.cc/paper_files/paper/2000/hash/8f0e8b8f2a79361e3548c1851f985d67-Abstract.html

[6] Frees, E. W., Meyers, G., & Derrig, R. A. (Eds.). (2014). *Predictive modeling applications in actuarial science*. Cambridge University Press.

[7] Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 315–323. http://proceedings.mlr.press/v15/glorot11a.html

[8] Gneiting, T., Genton, M., & Guttorp, P. (2006). Geostatistical space-time models, stationarity, separability and full symmetry. *Statistical Methods for Spatio-temporal Systems (Monographs on Statistics and Applied Probability), 107*. https://doi.org/10.1201/9781420011050.ch4

[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

[10] Gräler, B., Pebesma, E., & Heuvelink, G. (2016). Spatio-temporal interpolation using gstat. *The R Journal*, *8*(1), 204–218. https://journal.r-project.org/archive/2016/RJ-2016-014/index.html

[11] Hogg, R., McKean, J., & Craig, A. (2019). *Introduction to mathematical statistics* (8th ed.) [Eighth edition]. Pearson.

[12] Hristopulos, D. (2020). *Random fields for spatial data modeling: A primer for scientists and engineers*. Springer Netherlands. https://books.google.com/books?id=wivRDwAAQBAJ

[13] Intergovernmental Panel on Climate Change. (2023). Ipcc ar6 synthesis report: Summary for policymakers [Accessed: 11-13-2024].

[14] KARLIN, S., & TAYLOR, H. M. (1975). *A first course in stochastic processes* (2nd ed.) [Second edition]. Academic Press. https://doi.org/https://doi.org/10.1016/B978-0-08-057041-9.50004-0

[15] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1412.6980

[16] LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (1998). Efficient backprop, 9–50. https://doi.org/10.1007/3-540-49430-8_2

[17] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. ICML*. https://web.stanford.edu/~awni/papers/relu_hybrid_icml2013_final.pdf

[18] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*, 115–133. https://doi.org/10.1007/BF02478259

[19] Min, S., Gao, Z., Peng, J., Wang, L., Qin, K., & Fang, B. (2021). Stgsn — a spatial–temporal graph neural network framework for time-evolving social networks. *Knowledge-Based Systems*, *214*, 106746. https://doi.org/https://doi.org/10.1016/j.knosys.2021.106746

[20] Montero, J., Fernández-Avilés, G., & Mateu, J. (2015, August). *Spatial and spatio-temporal geostatistical modeling and kriging*. John Wiley; Sons, Ltd. https://doi.org/10.1002/9781118762387

[21] NASA. (2024). Evidence | climate change: Vital signs of the planet [Accessed: 11-13-2024].

[22] Nguyen, D. (2022). *Some mathematical perspectives of graph neural networks* [Master's thesis, University of Waterloo].

[23] Pan, Q., Porth, L., & Li, H. (2022). Assessing the effectiveness of the actuaries climate index for estimating the impact of extreme weather on crop yield and insurance applications. *Sustainability*, *14*(11). https://doi.org/10.3390/su14116916

[24] Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, *22*(3), 400–407. https://doi.org/10.1214/aoms/1177729586

[25] Schabenberger, O., & Gotway, A. C. (2017). *Statistical methods for spatial data analysis*. Chapman; Hall/CRC.

[26] Shi, G., Pan, S., Zou, R., & Yu, A. (2024). Stgnets: A spatial–temporal graph neural network for energy consumption prediction in cement industrial manufacturing processes. *Powder Technology*, *434*, 119280. https://doi.org/https://doi.org/10.1016/j.powtec.2023.119280

[27] Tang, J., Xia, L., & Huang, C. (2023). Explainable spatio-temporal graph neural networks. https://arxiv.org/abs/2310.17149

[28] The European Environmental Agency. (2024). Pm10 european air quality data (interpolated data) [Accessed 04-01-2025].

[29] The Royal Society. (n.d.). How does climate change affect biodiversity? [Accessed 03-13-2025].

[30] Tieleman, T., & Hinton, G. (2012). Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent magnitude [Lecture slides from Neural Networks for Machine Learning course, University of Toronto].

[31] United Nations. (2021). Water-related disasters dominate past 50 years [Accessed 03-13-2025]. https://www.unwater.org/news/water-related-disasters-dominate-past-50-years

[32] US Environmental Protection Agency. (2024). Particulate matter (pm) basics [Accessed 04-01-2025].

[33] Wang, H., Guan, Y., & Reich, B. J. (2019). Nearest-neighbor neural networks for geostatistics. https://arxiv.org/abs/1903.12125

[34] Wendland, H. (2004). *Scattered data approximation*. Cambridge University Press.

[35] Wilson, R. J. (1996). *Introduction to graph theory* (4th ed.) [Fourth edition]. Addison Wesley.

[36] Wiltschko, A. B. (n.d.). A Gentle Introduction to Graph Neural Networks — distill.pub [Accessed 03-21-2025].

[37] Wu, L., Cui, P., Pei, J., Zhao, L., & Song, L. (2022). Graph neural networks. In L. Wu, P. Cui, J. Pei, & L. Zhao (Eds.), *Graph neural networks: Foundations, frontiers, and applications* (pp. 27–37). Springer Singapore.

[38] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(1), 4–24. https://doi.org/10.1109/tnnls.2020.2978386

[39] Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph wavenet for deep spatial-temporal graph modeling. https://arxiv.org/abs/1906.00121

[40] Zhan, W., & Datta, A. (2024). Neural networks for geospatial data. https://arxiv.org/abs/2304.09157

VITA

Belguutei Ariuntugs was born on March 11, 2000, in Ulaanbaatar, Mongolia. He completed his secondary education at the 84th Secondary School in June 2017 and enrolled at Tennessee Technological University in August 2018. He earned a Bachelor of Science Degree in Mathematics in May 2023 and is currently pursuing a Master of Science Degree in Mathematics, which is expected to be conferred in May 2025.