NOVEL LEARNING ALGORITHMS FOR MINING GEOSPATIAL DATA

By

Shuai Yuan

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2017

ABSTRACT

NOVEL LEARNING ALGORITHMS FOR MINING GEOSPATIAL DATA

By

Shuai Yuan

Geospatial data have a wide range of applicability in many disciplines, including environmental science, urban planning, healthcare, and public administration. The proliferation of such data in recent years have presented opportunities to develop novel data mining algorithms for modeling and extracting useful patterns from the data. However, there are many practical issues remain that must be addressed before the algorithms can be successfully applied to real-world problems. First, the algorithms must be able to incorporate spatial relationships and other domain constraints defined by the problem. Second, the algorithms must be able to handle missing values, which are common in many geospatial data sets. In particular, the models constructed by the algorithms may need to be extrapolated to locations with no observation data. Another challenge is to adequately capture the nonlinear relationship between the predictor and response variables of the geospatial data. Accurate modeling of such relationship is not only a challenge, it is also computationally expensive. Finally, the variables may interact at different spatial scales, making it necessary to develop models that can handle multi-scale relationships present in the geospatial data.

This thesis presents the novel algorithms I have developed to overcome the practical challenges of applying data mining to geospatial datasets. Specifically, the algorithms will be applied to both supervised and unsupervised learning problems such as cluster analysis and spatial prediction. While the algorithms are mostly evaluated on datasets from the ecology domain, they are generally applicable to other geospatial datasets with similar characteristics.

First, a spatially constrained spectral clustering algorithm is developed for geospatial data. The algorithm provides a flexible way to incorporate spatial constraints into the spectral clustering formulation in order to create regions that are spatially contiguous and

homogeneous. It can also be extended to a hierarchical clustering setting, enabling the creation of fine-scale regions that are nested wholly within broader-scale regions. Experimental results suggest that the nested regions created using the proposed approach are more balanced in terms of their sizes compared to the regions found using traditional hierarchical clustering methods.

Second, a supervised hash-based feature learning algorithm is proposed for modeling nonlinear relationships in incomplete geospatial data. The proposed algorithm can simultaneously infer missing values while learning a small set of discriminative, nonlinear features of the geospatial data. The efficacy of the algorithm is demonstrated using synthetic and real-world datasets. Empirical results show that the algorithm is more effective than the standard approach of imputing the missing values before applying nonlinear feature learning in more than 75% of the datasets evaluated in the study.

Third, a multi-task learning framework is developed for modeling multiple response variables in geospatial data. Instead of training the local models independently for each response variable at each location, the framework simultaneously fits the local models for all response variables by optimizing a joint objective function with trace-norm regularization. The framework also leverages the spatial autocorrelation between locations as well as the inherent correlation between response variables to improve prediction accuracy.

Finally, a multi-level, multi-task learning framework is proposed to effectively train predictive models from nested geospatial data containing predictor variables measured at multiple spatial scales. The framework enables distinct models to be developed for each coarse-scale region using both its fine-level and coarse-level features. It also allows information to be shared among the models through a common set of latent features. Empirical results show that such information sharing helps to create more robust models especially for regions with limited or no training data. Another advantage of using the multi-level, multi-task learning framework is that it can automatically identify potential cross-scale interactions between the regional and local variables.

ACKNOWLEDGEMENTS

I spent five years in the department of computer science and engineering. I would not say time flies as they were long and tough five years. Lots of things happened, including not only the exciting ones but also the awful ones. Some things were pushing me down and made me want to quit, but there were also lots of things that pushing me up and encouraged me a lot. During these years of study, I received numerous help from advisor, collaborators, families and friends.

First, I would like to express my sincere gratitude towards my advisor Professor Pang-Ning Tan, who has been a great advisor and mentor over the past years, without whom, I would not be able to finish my PhD research. I would like to thank him for his patience, encouragement, and immense knowledge. I could not have imagined having a better mentor for my PhD study. Five years ago I was first admitted as a master student in the program. As a master student, I was lucky enough to be exposed to lots of research opportunities. Professor Tan encouraged me to join the weekly group meeting where we discussed the newest research papers from top conferences. I was also involved in an interdisciplinary project later that semester, where I got a chance to learn how data mining can be applied to solve real scientific questions. I enjoyed doing research in this group and all these exciting experience lead to a PhD program. Professor Tan is knowledgeable and a prominent expert in his domain and he is also a great educator. I still remember those many times, he was so patient to explain the very details to me when I am confused with a concept. I admire him as an inspiring researcher and a great educator.

I would also like to thank everyone in the CSI-limnology group, an interdisciplinary group of researchers working on lake multi-scaled geospatial and temporal database. Thanks to the great leadership of the group, we had well organized month meeting, annual workshops and sub-manuscript discussions. Everyone in the group is patient and helpful when I have questions and they are passionate about what we did and what we can do. Special thanks

to Patricia Soranno, Kendra Spence Cheruvelil, Sarah M. Collins, Emi Fergus, Nicholas K. Skaff and Tyler Wagner, without whom it would be impossible to conduct the research. A very special gratitude also goes to the National Science Foundation for providing the funding for my research under grant #EF-1065786.

My sincere thanks also goes to my PhD committee members Professor Xiaoming Liu, Professor Jiayu Zhou and Professor Patricia Soranno for their insightful comments and guidance regarding the dissertation.

I would also like to thank my labmates and friends for all the great discussion, the help and all the fun we have had for the past five years. I would like to thank my adorable puppy Elsa for the undivided attention and company.

Last but not the least, I would like to thank my family. During each phrase of my life, my parents always give their unconditional love and support.

PhD study is five years of my life that I will never forget. I'm ready to start a new chapter of my life and wish all the best to the people I met here. Go spartan!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

For the past decade, due to advances in remote sensing, GPS technology and the proliferation of web-based geographical data sharing services, more and more geospatial data have become available. Geospatial data can be defined as data with geographical information, such as latitude, longitude, address, and zip code [117]. Traditional geospatial data include maps of cities, lakes, and watersheds. Modern geospatial data are more ubiquitous and heterogeneous. Examples include photos captured by smartphones with their location service enabled, tweets posted by users tagged with their address information, and the real-time latitude and longitude information recorded by ride-sharing apps.

Geospatial data contains valuable information that can be used for various applications, such as environmental science, healthcare, urban planning, and business management. For example, real time traffic data can be used to predict traffic congestion, historical criminal data can be used to predict the locations of future crime events, and satellite images of an area can be used to monitor land cover changes. Because of their wide range applicability, this has led to the growing amount of interest in developing data mining and machine learning techniques to analyze the geospatial data [113, 25, 50, 47, 82, 45, 103]. Such techniques including regression, classification and clustering. Regression analysis is the process for estimating the statistical relationship among variables, specifically how a change in a continuous-valued dependent variable is associated with a change in the independent variables. For example Bertazzon et al. [11] applied a land-use regression (LUR) model to estimate the risks for air pollution. Classification techniques can be used to assign each spatial object to its predefined category [112]. For example, Thil and Wheeler [113] applied a decision tree classifier to predict shopping destination of participants living in a metropolitan area while in [25], Cleve et al. employed a combination of fuzzy and nearest neighbor classification technique to recognize objects in satellite imagery. Clustering is another widely used

data mining technique for analyzing geospatial data. The goal of clustering is to partition the spatial objects into groups in such a way that objects belonging to the same group are more similar (i.e., “closer") to each other compared to those belonging to other groups. Clustering techniques have been applied to various geospatial datasets for region delineation [50, 47] as well as to detect threats to the forest ecosystem from insects, diseases, and other agents [82]. Other techniques such as association rule mining are also applicable to the geospatial data. For example, Han et al. [45] applied association rule mining to discover spatial relationships between large towns and nearby lakes or highways.

1.1 Challenges and Motivation

While the large amount of geospatial data and their wide range of applicability have led to the development of numerous data mining algorithms, there are many practical issues remain that need to be addressed. First, the algorithms must be able to take into account spatial autocorrelation and other spatial constraints defined by the domain. For example, it is known that nearby geospatial objects tend to be more similar to each other compare to objects located far away from each other. This suggests that spatial autocorrelation must be incorporated when analyzing geospatial data in order to produce more realistic and useful results. For example, in cluster analysis for region delineation, spatial contiguity of the regions is a desirable criterion as the contiguous area of land is useful for research, policy, and management purposes [67]. However, many traditional clustering methods are not designed to produce spatially contiguous clusters. For example, Figure 1.1 shows the result of applying k-means clustering to lake ecology data, where each region (cluster) is represented with a different color on the map. Although some regions appear to be spatially contiguous, many of them are geographically disconnected. Alternative clustering algorithms are therefore needed that can effectively incorporate spatial autocorrelation for applications that require spatially contiguous regions.

Second, the algorithm should be able to handle missing values, which are commonly

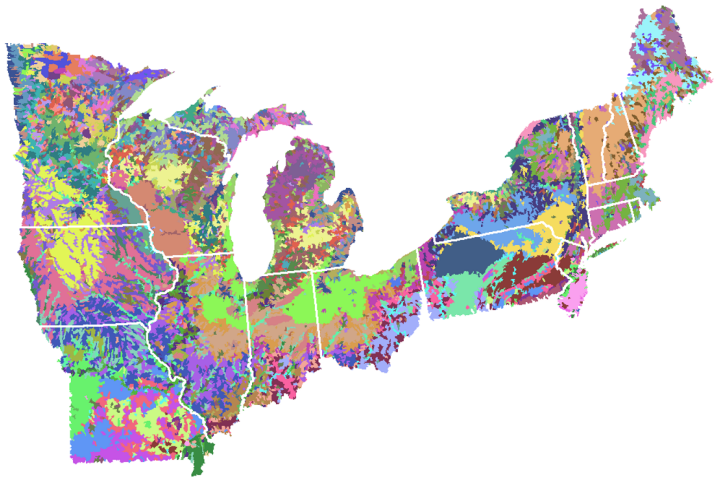


Figure 1.1: K-means clustering applied on geospatial features.

found in many real-world geospatial datasets. For example, real-time data from GPS devices have been increasingly used to monitor the physical activities of human subjects. However, the signal lapses that are inherent in GPS often lead to incomplete traces for applying more advanced analytical techniques [79]. Another example where missing values are often present is in lake ecology data (see the LAGOS-NE database to be described in Section 1.2). Measurements of lake water quality variables would be missing if they were not sampled at a given lake on a particular day. Although there are several standard techniques for handling missing values, they are mostly applied during the preprocessing step. For example, a simple strategy is to employ the complete case approach, where all incomplete data points will be discarded. This strategy is not effective if there are very few data points without missing values. Another common strategy is to impute the missing value with the average value of the corresponding attribute. However, this strategy will fail when the data are skewed or not centered at the mean value. More importantly, since these strategies are implemented as a data cleaning step, they are oblivious to the needs and characteristics of subsequent data mining algorithms to be applied to the preprocessed data.

The third challenge is to accurately capture the nonlinear relationship between the geospatial predictors and response variables. Previous studies have showed that many

geospatial relationships are nonlinear [86]. For example, Shaker and McCauley et al. [77] showed the relationship between nutrients and chlorophyll among lakes follow a sigmoidal function. Ehlinger [102] showed that the relationship between aquatic ecological conditions and landscape features in Southern Wisconsin is nonlinear. However, modeling this nonlinear relationship is not only difficult, it is also computationally expensive.

Finally, the geospatial variables may exist at multiple spatial scales and variables from different scales may interact with each other. For instance, previous work in lake ecology found evidence of cross-scale interactions between geospatial driver variables quantified at local and regional spatial scales for predicting lake nutrients [110]. Designing algorithms that can handle multi-scale relationships in geospatial data sets is another challenge that must be addressed in order to construct more robust predictive models.

In this thesis, I present four novel learning algorithms for mining geospatial data. The algorithms were applied mostly to the ecology domain. This domain is chosen as the testbed for my research as it has many applications, such as regionalization and spatial prediction, that may benefit from the deployment of the algorithms developed in this study. As proof of concept, the proposed algorithms will be applied to a newly integrated lakes ecology database named LAGOS-NE (LAke multi-scaled GeOSpatial and temporal database) [109]. LAGOS-NE consists of lake ecology data gathered from multiple sources, covering a wide range of spatial scales. Although the proposed algorithms are mostly evaluated on LAGOS-NE, they are generally applicable to other geospatial datasets with similar characteristics.

1.2 Lake Ecology Database

LAGOS-NE is a multi-scaled geospatial temporal database integrated from disparate data sources. The database covers a study region spanning across 17 states in the Upper Midwestern and Northeastern part of the United States, as shown in Figure 1.2. The database is divided into two modules, one containing measurements of lake water quality such as total phosphorus, total nitrogen and water clarity while the other include climate,

landscape, and other characteristics of the lakes (including lake location, lake area, and freshwater connectivity).

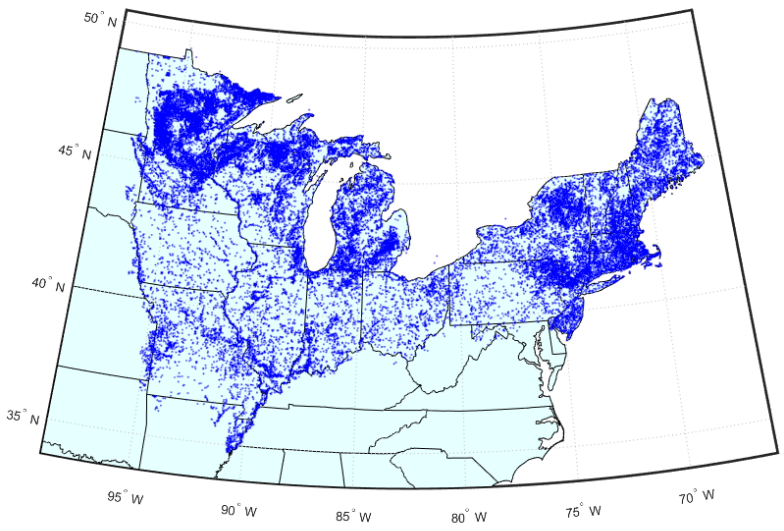


Figure 1.2: The study region of LAGOS-NE database. The blue dots represent lakes with area that are greater than 4 ha. The study region covers 17 states in the Northeastern and upper Midwest parts of the US.

LAGOS-NE is chosen for this study due to several reasons. First, it is a rich geospatial dataset, containing variables measured at different spatial scales. Both the volume and variety of the data are large, making it a challenging dataset for applying conventional data mining algorithms. The variables in the database also exhibit many of the properties described in Section 1.1, including spatial autocorrelation, missing values, nonlinearity, etc., thus providing opportunities for developing novel data mining algorithms. Finally, even though the database was created by integrating data from multiple sources, rigorous quality assurance/quality control (QAQC) procedures have been performed to produce a standardized database with consistent format and convention [109].

1.3 Thesis Contributions

This section summarizes the technical contributions of this thesis in addressing the challenges of applying data mining to geospatial datasets. The proposed data mining algorithms

will be applied to various unsupervised and supervised learning problems such as cluster analysis and regression. The specific contributions of this thesis are as follows:

- A spatially constrained spectral clustering algorithm is developed for the region delineation problem. Spectral clustering is a clustering algorithm that partitions a collection of data objects into smaller groups by performing eigen-decomposition of their feature similarity matrix. For geospatial data, the approach can be used to split the geographical landscape units into smaller regions or zones. However, similar to k-means, existing spectral clustering formulation is not designed to produce regions that are spatially contiguous. The proposed algorithm introduces a flexible way to incorporate spatial relationships into the spectral clustering formulation. The resulting regions created by the proposed algorithm were found to be spatially contiguous and relatively more homogeneous compared to existing constrained clustering methods. The formulation can be further extended to create regions that are nested wholly within broader-scale regions.

- A supervised hash-based feature learning algorithm is proposed for modeling non-linear relationships in incomplete geospatial data. The proposed algorithm simultaneously infers the missing feature values and learns a set of nonlinear hash-based features from the incomplete data. The learned hash-based features have the following properties: (i) complete (i.e., have no missing values), (ii) lower dimensionality than the original data, (iii) incorporates supervised information to learn the features, (iv) enables a linear model to be trained to capture nonlinear relationship between the predictor and response variables.

- A multi-task learning framework is proposed for the joint modeling of geospatial data with multiple response variables. Instead of training the local model for each response variable at each location independently, the framework simultaneously fits the local models for every response variable at all locations by optimizing a joint objective func-

tion with trace-norm regularization. The proposed framework enhances the prediction performance of the local models by incorporating spatial autocorrelation between the different locations as well as correlations between response variables into its unified formulation.

- A multi-level multi-task learning framework is developed for spatial prediction from nested geospatial data. The framework trains an independent model for each region using both its fine-level and coarse-level features. It also allows information to be shared among the various models through their common set of latent vectors. In addition, the framework can automatically identify potential cross-scale interactions between fine-level and coarse-level variables of the geospatial data.

1.4 Roadmap

The remainder of the thesis is organized as follows: Chapter 2 presents a literature review on existing techniques related to this research. Chapter 3 describes the proposed spatially constrained spectral clustering approach for region delineation. Chapter 4 introduces the hash-based feature learning algorithm for incomplete geospatial data. Chapter 5 presents a multi-task learning approach for modeling geospatial data with multiple response variables. Chapter 6 discusses the proposed multi-level multi-task learning approach for handling nested geospatial data. Finally Chapter 7 concludes the work.

The findings from three of the chapters in the thesis have been accepted for publication at various data mining conferences. Chapter 3 is based on the materials taken from Shuai et al. [146], which was accepted for oral presentation at the 2015 IEEE International Conference Data Science and Advanced Analytics Special Session on Environmental and Geo-spatial Data Analytics. Chapter 4 is taken from a conference paper presented at the 2017 SIAM International Conference on Data Mining [145]. Chapter 6 is adopted from a manuscript that was recently accepted for the 2017 IEEE International Conference on Data Mining.

CHAPTER 2

LITERATURE REVIEW

This chapter presents an overview of previous work related to this study. Section 2.1 introduces the general problem of mining geospatial data. Section 2.2 provides an overview of feature learning and hash-based methods. Section 2.3 presents the previous work on handling incomplete data. Section 2.4 presents the existing work on multi-task learning while Section 2.5 describes the previous work on constrained spectral clustering.

2.1 Mining Geospatial data

Geospatial data are abound, collected from various sources including field sampling, satellite imagery, GPS and geotagging services. This type of data has unique properties that distinguish it from others. First, it is geo-referenced, i.e., containing location information such as latitude, longitude, height/elevation, address, or zip code. Second, in addition to their location information, the geospatial data often contain non-spatial features characterizing properties associated with the locations. These non-spatial features typically exhibit some spatial patterns, which leads to the spatial autocorrelation associated with the geospatial data. Third, the geospatial data may contain variables defined at multiple spatial scales. These variables may interact at different characteristic spatial scales to produce the non-linear relationships observed in the data [110]. Due to these unique characteristics, special consideration and techniques are needed for mining geospatial data.

Classification is a data mining technique that seeks to categorize data objects into their corresponding class labels [112]. For geospatial data, the classification technique must consider not only the relationship between the predictor and response variables, but also their spatial dependencies as previous studies have shown that the classification performance can be improved by incorporating such dependencies into the models [46, 100, 40, 81]. There are many applications where classification can be applied to the geospatial data. For example,

Thil and Wheeler [113] used a top-down spatial decision tree that based on information gain to predict travel destination choices. Cleve et al. [25] used classification algorithm to recognize land-use and land-cover categories from the 3-band aerial imagery. Hollister et al. [49] applied random forests to the land use, land cover and lake morphometry data, to predict the lake trophic state in order to monitor the ecosystem condition.

Clustering is another popular data mining technique for geospatial data. For example, it can be applied to create ecological regions of the landscape based on land cover and land use features. Early works such as Openshaw [89] proposed a two-step approach, in which a conventional clustering method was applied followed by a cluster refinement step to separate clusters that were not geographically connected. Host et al. [50] performed hierarchical k-means clustering on monthly temperature and precipitation data across northwestern Wisconsin to identify clusters(regions) with similar seasonal climatic patterns. Hargrove et al. [47] applied k-means clustering on elevation, climatic, and edaphic factors to generate regions. Besides regionalization, clustering techniques can also be used for threat identification. For example, Mills et al. [82] applied k-means clustering to satellite images and analyzed the transition distance between cluster assignment between any two years in order to identify threats to the forest ecosystem.

2.2 Feature Learning

Feature learning [10] is the task of extracting an alternative feature representation of a given data set. Feature learning methods can help improve the performance of predictive models in many ways, such as reducing the dimensionality of large-scale data to improve its training efficiency and removing noise, redundant, or irrelevant features that are present in the data. Classical methods such as principal component analysis [90] were developed to create features that preserve variability in the original data. These methods are mostly unsupervised, and thus, provide no guarantee about the usefulness of their extracted features for predictive modeling tasks. More recently, methods such as stacked autoencoders [14, 147]

and deep learning [61, 149] have been proposed to create hierarchical features from the data. Although such methods have been successfully applied to applications such as image classification [61], they are expensive to train and require considerable human efforts to design the right architecture for a given prediction problem.

Hashing is another feature learning technique that has attracted considerable attention in recent years. The goal of hash-based feature learning is to transform the data into easily computable features that preserve the underlying properties of the data. For example, the Min-hash method [15] was designed to create features that preserve the Jaccard similarity between instances. Random hyperplane based hashing were introduced in [20] to preserve cosine similarity. These features can help improve the efficiency of processing similarity search queries in large data sets. Hash-based methods have also been developed for other similarity measures [131, 85]. In general a hash function is defined as $y = h(x)$, where y is called the hash code or signature and $h(\cdot)$ is the hash function.

Wang et al. [126] divided hashing methods into two categories: one that does not explore the data distribution while the other that learns the hash function based on the distribution of the data. The former category includes locality sensitive hashing (LSH), which is a family of hashing techniques that map instances with similar features into the same hash code with higher probability [52]. LSH can be viewed as a probabilistic similarity-preserving dimension reduction method. Many variants of LSH have been designed to provide an unbiased set of features that preserve certain distance metrics [15, 20, 131, 85]. This includes the Min-hash [15] and random hyperplane [20] methods described in the previous paragraph.

Unlike the data independent hashing methods described in previous paragraph, the second category encompasses techniques that learn hash functions from a given input dataset. There are three elements that must be considered in this learning to hash scheme [126]: distance metric, hash function and optimization criterion. For example, Weiss et al. [131] proposed an algorithm known as spectral hashing to learn a binary code representation of data such that the Hamming distance between data instances in terms of their hash

code correlates with the similarities of their features. The hash function used is given by $h(\mathbf{x}) = \text{sgn}(\sin(\frac{\pi}{2} + \gamma \mathbf{w}^T \mathbf{x}))$. Let $\mathbf{y}_i \in \mathbf{R}^K$ be the hash code of length K for the i^{th} data instance and $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_n]$. The spectral hashing algorithm is designed to solve the following optimization problem:

$$\begin{aligned} \min_{\mathbf{Y}} \quad & \text{Trace } (\mathbf{Y}(\mathbf{D} - \mathbf{W})\mathbf{Y}^T) \\ s.t. \quad & \mathbf{Y1} = 0, \quad \mathbf{Y}\mathbf{Y}^T = \mathbf{I}, \quad y_{im} \in \{-1, 1\} \end{aligned}$$

where $\mathbf{W}$ is a similarity matrix with $\mathbf{W}(i,j) = \exp(- \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2 / \epsilon^2)$ and $\mathbf{D}$ is a diagonal matrix, where the diagonal elements are the corresponding row sum of $\mathbf{W}$.

Kulis and Darrell [62] proposed an optimization scheme that minimizes the difference between the Euclidean distance in the original feature space and the Hamming distance in the hash code feature space. Specifically, for a given input data set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$, the objective function of their optimization is given as follows:

$$\min_{w} \sum_{(i,j) \in N} [d_M(\mathbf{x}_i, \mathbf{x}_j) - d_R(\mathbf{x}_i, \mathbf{x}_j)]^2$$

where $d_M(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2} \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2$, $d_R(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{K} \sum_{k=1}^{K} (h_k(\mathbf{x}_i) - h_k(\mathbf{x}_j))^2$ and $h_p(x) = \text{sgn}(\sum_q \mathbf{W}_{pq} \kappa(x_{pq}, x))$. In this formulation, K is the number of hash functions, $\kappa(x_i, x_j)$ the kernel function over the data and $\mathbf{W}$ is the weight matrix to be learnt.

2.3 Incomplete Geospatial Data

Missing value is a common problem encountered when dealing with real world datasets. A missing value means there is no data value stored for a given feature. There are different types of missing values—missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAM) [70]. Some of the popular techniques for dealing with missing values include: (1) using only the complete cases by discarding instances with missing values; (2) imputing the missing values within each feature by their respective average feature values. In addition, model-based approaches have been developed as well, such as those based

on maximum likelihood, Bayesian, and multiple imputation methods [69]. However, most of these techniques are parametric methods, which assume that the data follow standard probability distributions such as Gaussian.

More recently, matrix completion has emerged as an increasingly popular technique for handling missing values in an input data matrix. Matrix completion employs an optimization scheme for completing a partially observed matrix under the assumption that the original matrix has a low rank. Specifically, the matrix completion approach can be cast into the following constraint rank minimization problem:

$$\begin{aligned} &\min\ \text{rank}(X) \\ &s.t.\ X_{ij} = M_{ij}, (i,j) \in \Omega \end{aligned} \tag{2.1}$$

where M is the data matrix we wish to recover and X is a low rank approximation of the matrix M. Ω is the set of indices for the non-missing elements of M. Candès and Recht [18] proved that most low rank matrices can be effectively recovered when the number of observed entries m satisfies the following inequality:

$$m > Cn^{1.2}rlog(n),$$

where n is the dimension of an $n \times n$ matrix M, r is the desired rank of the matrix X and C is a positive constant. Since solving the rank minimization problem is computationally NP-hard, an alternative way is to approximate the rank with the nuclear norm and solve the convex relaxation problem shown in Equation (2.2) using semidefinite programming.

$$\begin{aligned} &\min\ \| X \|_* \\ &s.t.\ X_{ij} = M_{ij}, (i,j) \in \Omega \end{aligned} \tag{2.2}$$

Cai et al.[17] proposed a singular value thresholding (SVT) algorithm for solving the optimization problem shown in Equation (2.3) by reformulating the objective function as follows:

$$\begin{aligned} &\min\ \tau \| X \|_* + \frac{1}{2} \| X \|_F^2 \\ &s.t.\ X_{ij} = M_{ij}, (i,j) \in \Omega \end{aligned} \tag{2.3}$$

They showed that the solution of the constrained nuclear norm minimization problem in Equation (2.3) converges to the solution of Equation (2.2) when τ approaches infinity.

A faster algorithm based on accelerated proximal gradient method have been proposed to solve least square minimization problems with nuclear norm regularization [115, 56]. For matrix completion, instead of adding the equality constraint $X_{ij} = M_{ij}$ as in Equations (2.2) and (2.3), a least square loss term can be added to the objective function. The matrix completion formulation can be cast into the following objective function:

$$\min_{X} \ \frac{1}{2} \parallel \mathcal{A}(X) - b \parallel_2^2 + \mu \parallel X \parallel_* \tag{2.4}$$

where $\mathcal{A} : \mathcal{R}^{m \times n} \rightarrow \mathcal{R}^p$ is a linear map that extracts the non-missing entries of its input matrix and μ is a predefined parameter.

2.4 Multi-task Learning

Multi-task learning (MTL) is a machine learning approach for solving multiple related prediction problems at the same time by capturing their shared information [19]. The rationale for using MTL is that by incorporating the relationship between different tasks, one can improve the learning ability. For the past decades, MTL has been successfully used in many learning schemes including regression[137, 151], clustering [32] and classification[142, 140]. MTL can be applied to a wide range of applications, such as disease progression prediction [151], Web image and video search [129], as well as web page categorization [22].

The simplest way to integrate different kinds of task relationship is by introducing a regularization term into the learning formulation. Since the assumption of how the tasks are related are different for different applications, this has led to the design of different regularization terms. For example, one simple assumption is that the task parameters are close to each other. This simple intuitive assumption leads to the regularization term based on the mean value of all the task parameters [33]. Another common assumption is that the model parameters shared a low-rank representation. Since the rank of the model parameter matrix is hard to optimize, it can be approximated by using the trace norm regularization

instead of minimizing the matrix rank directly. For example, Chen et al. [23] proposed a robust MTL algorithm that can learn multiple tasks simultaneously while identifying the irrelevant tasks. Argyriou et al. [2] generalized the single task L1 norm and proposed a method to learn the sparse representation shared across different models. Kumar et al.[63] assumed that each model is a linear combination of a finite set of base models.

Since many geospatial datasets involve predictions at multiple locations, they can be naturally cast into a multi-task learning framework. For example, Xu et al. [138] presented a multi-task learning formulation for predicting monthly precipitation at 37 weather stations in Canada based on geospatial data. Xu et al. [139] proposed a weighted incremental multi-task learning algorithm that simultaneously identifies the latent factors through supervised tensor decomposition and learns spatial temporal models. Zhou et al. [151] proposed a MTL learning framework for predicting the disease progression. The proposed framework consider prediction at each time stamp as a task and task relatedness is captured by a temporal group lasso regularizer. Zhao et al. [148] builded predictive models for spatio-temporal event forecasting. The proposed MTL model improved the predicting performance by utilizing both static features and dynamic features generated from a multi-task feature learning framework, and using the shared information between different locations.

2.5 Constrained Spectral Clustering

Constrained clustering is a semi-supervised learning algorithm that utilizes side information from the domain to improve clustering performance. Based on the domain knowledge, experts can explicitly specify which pair of data instances must be in the same cluster (i.e., Must-link(ML) constraints) and which pair of instances must not be in the same cluster (i,e, Cannot-link(CL) constraints). For example, a constrained K-means clustering approach was developed in [125]. Shi et al. [105] proposed a constrained co-clustering algorithm that takes into account of both feature similarity and the ML and CL constraints. De Bie et al. [12] proposed a method that constrains the eigenspace for which the cluster membership vector

is projected. Coleman et al. [26] extended the approach in [12] to handle the situation in which some of constraints are potentially inconsistent with each other.

In this thesis, we focus on constrained spectral clustering. Spectral clustering [104, 75] is a well-known clustering method that uses the eigenvector spectrum of a feature similarity matrix to find the underlying clusters of a given data set. Advantages of using spectral clustering include its flexibility in terms of incorporating diverse types of similarity functions, superiority of its clustering solution compared to the traditional k-means algorithm [84], and its well-established theoretical properties (including the consistency [120] and convergence [121] guarantees of the algorithm). There are two popular ways to incorporate constraints into the spectral clustering framework. The first category encompasses methods that directly alter the graph Laplacian matrix. For example, Kamvar et al. [57] employed a Gaussian kernel as their similarity matrix and considered a binary constraint matrix where ML constraints are assigned as 1 and CL constraints are designated as 0. Kawale and Boley [13] proposed adding an ℓ_1-regularizer that penalizes constraint violations to the graph cut objective function. However, the proposed method was designed for constrained spectral clustering with only 2 clusters.Another way to alter the matrix is by performing a weighted sum between the feature similarity matrix and the adjacency matrix of the constraint graph. The modified graph Laplacian is given by a convex combination of the original graph Laplacian and the Laplacian induced by the constraint matrix. The second category of approaches for incorporating domain constraints is by restricting the feasible solution set of the spectral clustering algorithm. For example, Wang and Davidson [128] proposed an algorithm that optimizes the objective function of spectral clustering while adding a constraint that the ML and CL must be satisfied more than a predefined threshold.

CHAPTER 3

SPATIALLY CONSTRAINED SPECTRAL CLUSTERING FOR REGIONALIZATION

3.1 Introduction

A regionalization framework delineates the geographical landscape into spatially contiguous, homogeneous units known as regions or zones. Regionalizations are important because they provide the spatial framework used in many disciplines, including landscape ecology, environmental science, and economics, as well as for applications such as public policy and natural resources management [24, 73, 39, 76]. For example, the hierarchical system of hydrologic units described in [101] provides a standardized regionalization framework that has been widely used in water resource and land use studies [31]. Abell et al. [1] have also developed a global biogeographic regionalization framework that serves as a useful tool for studying biodiversity in freshwater systems and for conservation planning efforts.

McMahon et al. [78] divide existing multivariate regionalization methods into two categories, qualitative and quantitative. For qualitative methods, regions with similar landscape characteristics are delineated by experts from multiple maps of different geographic features using manual visual interpretation [4, 88]. For quantitative methods, clustering approaches such as k-means and hierarchical clustering [50, 47, 55] are used to partition the geographical area into smaller regions. Although quantitative clustering approaches provide a more systematic and reproducible way to identify regions compared to qualitative approaches, one potential limitation of existing clustering methods is that the regions created may not be spatially contiguous. Region contiguity is a desirable criterion for many applications that treat regions as individual entities for purposes including research, policy, and management (e.g., site-specific management in precision agriculture [67]). Therefore, alternative methods are needed that can effectively cluster similar areas based on multiple mapped variables, but

have the added constraint of being spatially contiguous.

In the preliminary version of this work [146], we presented a spatially constrained spectral clustering framework that uses a truncated exponential kernel [58] to produce spatially contiguous and homogeneous regions [57, 13, 105, 128]. In this chapter, we extend the formulation to create hierarchical regions, where fine-scaled regions can be nested within broad-scale regions. Creating such nested regions is extremely useful for many applications because hierarchical structure is often held up as a fundamental feature of both the natural world and complex systems (as reviewed in [135]). In fact, the world?s biomes and ecological regions have often been delineated in a nested hierarchical structure [5, 1]. Constrained versions of hierarchical clustering techniques [83] such as single-link [66], complete-link [95], UPGMA [60, 98], and Ward's method [53, 134] have often been used to create such nested regions. However, as will be shown in this study, the regions generated by such methods tend to be highly imbalanced in terms of their sizes, and thus, are not as suitable for many applications, including resource planning and management.

We use a recursive bisection approach to extend our formulation in [146] to hierarchical clustering. Our top-down approach for creating nested regions is different from the bottom-up approach commonly used by existing methods [53, 95, 60, 98, 134]. Using three criteria for region evaluation—landscape homogeneity, region contiguity, and region size—our experimental results suggest that the proposed framework outperforms three other constrained hierarchical clustering methods in 2 out of the 3 criteria. For example, it consistently produces regions that are more homogeneous and balanced in region size compared to the spatially constrained complete-link [95] and UPGMA [60, 98] algorithms. Our proposed algorithm also outperforms the constrained version of Ward's method [134] in terms of producing regions that are spatially contiguous and approximately uniform in size. Finally, although the spatially constrained single link method [66] is also capable of producing regions that are homogeneous and contiguous, it tends to create one or two very large regions that cover the majority of the landscape area. An ad-hoc parameter for maximum region size is needed by

the spatially constrained single link method to prevent the formation of such large regions [96]. Tuning this parameter is cumbersome as it must be done at every level of the hierarchy since the maximum region size depends on the number of regions. Our proposed hierarchical method does not have such a problem because its objective function, which is based on the normalized cut criterion [104] used in spectral clustering, is inherently biased towards producing more uniformly-sized clusters.

The remainder of this chapter is organized as follows. Section 3.2 reviews previous work on the development of regionalization frameworks, constrained clustering, and hierarchical clustering methods. Section 3.3 formalizes the region delineation problem and presents an overview of spectral clustering. Section 3.4 describes the different ways in which spatial constraints can be incorporated into the spectral clustering framework. It also presents the partitional and hierarchical implementations of our proposed spatially constrained spectral clustering framework. Section 3.5 describes the application of spatially constrained spectral clustering algorithms to the region delineation problem. Section 3.6 concludes with a summary of the results of this study.

3.2 Related Work

Region delineation has traditionally been studied as a spatial clustering [44] problem. Duque et al. [31] classified the existing data-driven approaches into two categories. The first category does not require explicit representation and incorporation of spatial constraints into the clustering procedure. Instead, the constraints are satisfied by post-processing the clusters or optimizing other related criteria. For example, Openshaw [89] applied a conventional clustering method followed by a cluster refinement step to split clusters that contained geographically disconnected patches. The second category of methods explicitly incorporates spatial constraints into the clustering algorithm [31]. Examples of such methods include adapted hierarchical clustering, exact optimization methods, and graph theory based methods. This second category also encompasses the constrained clustering methods [3, 125, 29]

developed in the fields of data mining and machine learning.

Constrained clustering [7, 125] is a semi-supervised learning approach that uses the domain information provided by users to improve clustering results. The domain information is typically provided as must-link (ML) and cannot-link (CL) constraints to be satisfied by the clustering solution. ML constraints restrict the pairs of data points that must be assigned to the same cluster, whereas CL constraints specify the pairs of points that must be assigned to different clusters. For example, Kamvar et al. [57] uses the ML and CL constraints to define the affinity matrix of the data. Shi et al. [105] proposed a constrained co-clustering method that considers both the similarity of features as well as the ML and CL constraints. All of these methods were designed to manipulate the graph Laplacian matrix using the domain constraints available. There has also been growing interest in developing constrained-based approaches for spectral clustering [57, 13, 105, 128, 28]. For example, De Bie et al. [12] developed an approach that restricts the eigenspace for which the cluster membership vector is projected. Wang and Davison [128] proposed a constrained spectral clustering method that considers real-valued constraints and imposed a threshold on the minimum amount of constraints that must be satisfied by the feasible solution. However, none of these constrained spectral clustering methods were designed for the region delineation problem. The framework presented in our previous work [146] employs a Hadamard product to combine the feature similarity matrix with spatial contiguity constraints, which is similar to the approach used in Craddock et al. [28] for generating an ROI atlas of the human brain using fMRI data. However, unlike the approach used in [28], we consider a truncated exponential kernel to relax the spatial neighborhood constraints and perform extensive experiments comparing the framework to various constrained spectral clustering algorithms.

Current constrained spectral clustering algorithms have also focused primarily on partitional clustering. They require the number of clusters to be specified *a priori*. In contrast, hierarchical methods generate a nested set of clustering for every possible number of clusters. The hierarchy of clusters, also known as a dendrogram, can be created either in a top-down

(i.e., divisive hierarchical clustering) or bottom-up (i.e., agglomerative hierarchical clustering) fashion [112, 55, 54]. Some of the widely used agglomerative hierarchical clustering algorithms include single link [53], complete link [95], group average (UPGMA) [60, 98], and Ward's method [55, 134] whereas examples of divisive hierarchical clustering algorithms include minimum spanning tree [42] and bisecting k-means [99]. Current approaches for creating nested regions are mostly based on different variations of agglomerative hierarchical clustering. Each of these variations has its own strengths and limitations [83]. For example, the single-link method can identify irregular shaped clusters but is highly sensitive to noise [112]. In contrast, the Ward's method can minimize the cluster variance but is susceptible to the inversion problem [83]. Unfortunately, many of these agglomerative methods can produce highly imbalanced sizes of regions, which is not desirable for many applications [6].

3.3 Preliminaries

This section formalizes region delineation as a constrained clustering problem and presents a brief overview of spectral clustering and its constrained-based methods.

3.3.1 Region Delineation as Constrained Clustering Problem

Consider a data set $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{s}_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is a d-dimensional vector of landscape features associated with the geo-referenced spatial unit $\mathbf{s}_i \in \mathbb{R}^2$. Let $\mathcal{R} = \{1, 2, \cdots, k\}$ denote the set of region identifiers, where k is the number of regions, and $\mathcal{C} = \{(\mathbf{s}_i, \mathbf{s}_j, \mathbf{C}_{ij})\}$ denote the set of spatial constraints. For region delineation, we consider only ML constraints and represent them using a constraint matrix $\mathbf{C}$ defined as follows:

$$\mathbf{C}_{ij} = \begin{cases} 1 & \text{if } \mathbf{s}_i \text{ and } \mathbf{s}_j \text{ are spatially adjacent,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

The goal of region delineation is to learn a partition function $\mathcal{V}$ that maps each spatial unit $\mathbf{s}_i$ to its corresponding region identifier $r_i \in \mathcal{R}$ in such a way that (1) maximizes the similarity

between the spatial units in each region and (2) minimizes the constraint violations in the set $\mathcal{C}$.

3.3.2 Spectral Clustering

Spectral clustering is a class of partitional clustering algorithms that relies on the eigen-decomposition of an input affinity (similarity) matrix $\mathbf{S}$ to determine the underlying clusters of the data set. Let $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ be a set of points to be clustered. To apply spectral clustering, we first compute an affinity matrix $\mathbf{S}$ between every pair of data points. The affinity matrix is used to construct an undirected weighted graph $\mathcal{G} = (V, E)$, where V is the set of vertices (one for each data point) and E is the set of edges between pairs of vertices. The weight of each edge is given by the affinity between the corresponding pair of data points. The Laplacian matrix of the graph is defined as $\mathbf{L} = \mathbf{D} - \mathbf{S}$, where $\mathbf{D}$ is a diagonal matrix whose diagonal elements correspond to $\mathbf{D}_{ii} = \sum_j \mathbf{S}_{ij}$. The goal of spectral clustering is to create a set of partitions on the graph $\mathcal{G}$ in such a way that minimizes the graph cut while maintaining a balanced size of the cluster partitions [75].

The spectral clustering solution can be found by solving the following optimization problem [75]:

$$\arg\min_r r^T \mathbf{L} r \text{ s.t. } r^T \mathbf{D} r = \sum_i \mathbf{D}_{ii},\ \mathbb{1}^T \mathbf{D} r = \mathbf{0} \tag{3.2}$$

where $\mathbb{1}$ and $\mathbf{0}$ are vectors whose elements are all 1s and 0s, respectively. The solution for r is obtained by solving the following generalized eigenvalue problem: $\mathbf{L}r = \lambda \mathbf{D}r$. To obtain k clusters, we first extract the top k generalized eigenvectors and apply a standard clustering algorithm such as k-means to the data matrix generated from the eigenvectors.

3.3.3 Constrained Spectral Clustering

Current methods for incorporating constraints into spectral clustering algorithms can be divided into two categories. The first category encompasses methods that directly alter the

graph Laplacian matrix, e.g., by applying a weighted sum between the feature similarity matrix $\mathbf{S}$ and the constraint matrix $\mathbf{C}$ given in Equation (3.1):

$$\text{Weighted sum:} \quad \mathbf{S}^{\text{total}}(\delta) = (1-\delta)\mathbf{S} + \delta\mathbf{C}, \tag{3.3}$$

$\delta \in [0, 1]$ is a parameter that controls the trade-off between maximizing cluster homogeneity and preserving the constraints of the data. When δ approaches zero, the clustering solution is more biased towards maximizing the feature similarity whereas when δ approaches one, it is more biased towards preserving the constraints.

Let $\mathbf{D}$ and $\mathbf{D}^{(c)}$ be the diagonal matrices constructed from the feature similarity matrix ($\mathbf{S}$) and constraint matrix ($\mathbf{C}$) in the following way:

$$\mathbf{D}_{ii} = \sum_j \mathbf{S}_{ij}, \quad \mathbf{D}^{(c)}_{ii} = \sum_j \mathbf{C}_{ij}.$$

Using Equation (3.3), it can be shown that the modified graph Laplacian is given by a convex combination of the graph Laplacian for the feature similarity matrix and the graph Laplacian for the constraint matrix, i.e.,

$$\begin{aligned} \mathbf{L}^{\text{total}} &= \mathbf{D}^{\text{total}} - \mathbf{S}^{\text{total}} \\ &= (1-\delta)(\mathbf{D} - \mathbf{S}) + \delta(\mathbf{D}_c - \mathbf{C}) \end{aligned} \tag{3.4}$$

The weighted sum approach described above is a special case of the spectral constraint modeling (SCM) algorithm proposed by Shi et al. [105]. The altered graph Laplacian can be substituted into Equation (3.2), which in turn, allows us to apply existing spectral clustering algorithm to identify the regions.

$$\begin{aligned} \text{SCM:} \qquad & \arg\min_{r \in \mathbb{R}^N} r^T \mathbf{L}^{\text{total}} r \\ \text{s.t.} \qquad & r^T \mathbf{D}^{\text{total}} r = \sum_i \mathbf{D}^{\text{total}}_{ii}, \; \mathbb{1}^T \mathbf{D}^{\text{total}} r = \mathbf{0}. \end{aligned} \tag{3.5}$$

The second category of approaches for incorporating domain constraints is to alter the feasible solution set of the spectral clustering algorithm. For example, Wang and Davidson

[128] proposed the CSP algorithm, which optimizes the following objective function.

$$\begin{aligned} \text{CSP:} \quad & \arg\min_{r \in \mathbb{R}^N} r^T \bar{\mathbf{L}} r \\ \text{s.t.} \quad & r^T \bar{\mathbf{C}} r \geq \alpha,\ r^T r = vol(\mathcal{G}),\ r \neq \mathbf{D}^{1/2} 1, \end{aligned} \tag{3.6}$$

where $\bar{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ and $\bar{\mathbf{C}} = \mathbf{D}_c^{-1/2}\mathbf{C}\mathbf{D}_c^{-1/2}$ are the normalized graph Laplacian and normalized constraint matrix, respectively. The threshold α gives a lower bound on the amount of constraints in $\mathbf{C}$ that must be satisfied by the clustering solution. Instead of setting the parameter for α, Wang and Davison [128] requires users to specify a related parameter β, which was shown to be a lower bound for α.

3.4 Spatially Constrained Spectral Clustering

In this section, we describe the various ways to represent spatial contiguity constraints and to incorporate them into the spectral clustering framework.

3.4.1 Kernel Representation of Spatial Contiguity Constraints

For constrained spectral clustering, we can define a corresponding constraint graph $\mathcal{G}_C = (V, E_C)$, where V is the set of data points and E_C is the set of edges whose weights are defined as follows:

$$E_{ij} = \begin{cases} 1, & (v_i, v_j) \text{ is a ML edge;} \\ -1, & (v_i, v_j) \text{ is a CL edge;} \\ 0, & \text{otherwise.} \end{cases} \tag{3.7}$$

For region delineation, the vertices of the constraint graph correspond to the set of spatial units to be clustered, while the ML edges correspond to pairs of spatial units that are adjacent to each other. It is also possible to define a CL edge between every pair of spatial units that are either located too far away from each other or are obstructed by certain barriers (e.g., large bodies of water) that make them unreasonable for assignment to the same region. However, since the number of CL edges tends to grow almost quadratically

with increasing number of points, this severely affects the runtime of spectral clustering algorithm. Furthermore, the ML edges are often sufficient to provide guidance on how to form spatially contiguous regions. For these reasons, we consider constraint graphs that have ML edges only in this chapter. Let $\mathbf{C}$ denote the adjacency matrix representation of the edge set E_C.

A constrained spectral clustering algorithm is designed to produce solutions that are consistent with the constraints imposed by $\mathcal{G}_C$. Unfortunately, for region delineation, it may not be sufficient to use the adjacency information between neighboring spatial units to control the trade-off between spatial contiguity and landscape homogeneity of the regions. To improve its flexibility, we introduce a *spatially constrained kernel matrix*, $\mathbf{S}_c$. The simplest form of the kernel would be a linear kernel, which is defined as follows:

$$\text{Linear Kernel:} \qquad \mathbf{S}_c^{\text{linear}} = \mathbf{C} \tag{3.8}$$

More generally, we can define an exponential kernel [58] on the adjacency matrix $\mathbf{C}$ as follows.

$$\text{Exponential Kernel:} \quad \mathbf{S}_c^{\text{exp}} = e^{\mathbf{C}} = \mathbb{I} + \mathbf{C} + \frac{1}{2!}\mathbf{C}^2 + \frac{1}{3!}\mathbf{C}^3 + \cdots = \sum_{k=0}^{\infty} \frac{\mathbf{C}^k}{k!} \tag{3.9}$$

where $\mathbb{I}$ is the identity matrix. Since we consider only ML constraints, the k-th power of the adjacency matrix $\mathbf{C}$ represents the number of ML paths of length k that exist between every pair of vertices. An ML path between vertices (v_i, v_j) refers to a sequence of ML edges $e_1, e_2, \cdots, e_m$ such that the initial vertex of e_1 is v_i and the terminal vertex of e_m is v_j. It can be shown that $\mathbf{S}_c^{\text{exp}}$ is a symmetric, positive semi-definite matrix, and thus, is a valid kernel [58]. Furthermore, as the diameter of the constraint graph is finite, we also consider a truncated version of the exponential kernel:

$$\text{Truncated Exponential :} \quad \mathbf{S}_c^{\text{trunc}}(\delta) \equiv \sum_{k=0}^{\delta} \frac{\mathbf{C}^k}{k!} \tag{3.10}$$

where the parameter δ controls the ML neighborhood size of a vertex. The ML neighborhood specifies the set of vertices that should be in the same region as the vertex under consideration. As an example, consider the graph shown in Figure 3.1. When $\delta = 1$, the ML

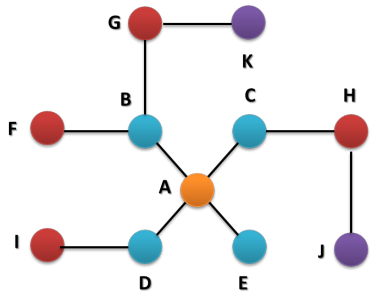


Figure 3.1: An illustration of spatial contiguity constraint.

neighborhood for vertex A corresponds to its immediate neighbors, B, C, D and E. When $\delta = 2$, the ML neighborhood of vertex A is expanded to include vertices that are located within a path of length 2 or less from A, i.e., B, C, D, E, F, G, H and I. When $\delta = 3$, the ML neighborhood for vertex A includes all of the vertices in the graph. Note that each term in the summation given in Equation (3.9) is normalized by the path length; therefore, a vertex that is located further away from a given vertex has less influence as compared to a nearer vertex.

Finally, the truncated exponential kernel matrix can be binarized so that it can be interpreted as an adjacency matrix for an expanded constraint graph, whose ML neighborhood size is given by the parameter δ.

$$\text{Binarized Truncated Exponential Kernel :} \qquad \mathbf{S}_c^{\text{bin}}(\delta) \equiv \mathbf{I}\left[\sum_{k=0}^{\delta} \mathbf{C}^k > 0\right] \tag{3.11}$$

where $\mathbf{I}[\cdot]$ is an indicator function whose value is equal to 1 if its argument is true and 0 otherwise. Both the truncated and binarized truncated exponential kernels allow us to vary the degree to which the original constraint graph should be satisfied. As δ increases, the constraint satisfaction becomes more relaxed. Ultimately, when δ is greater than or equal to the diameter of the graph, $\mathbf{S}_c^{\text{bin}}$ reduces to a matrix of all 1s, which is equivalent to ignoring the spatial contiguity constraints.

3.4.2 Hadamard Product Graph Laplacian

We now describe our approach for incorporating the spatially constrained kernel matrix $\mathbf{S}_c$ into the spectral clustering formulation. Instead of using the weighted sum approach given in Equation (3.3), we consider a Hadamard product approach to combine $\mathbf{S}_c$ with the feature similarity matrix $\mathbf{S}$:

$$\text{Hadamard Product:} \qquad \mathbf{S}^{\text{total}}(\delta) = \mathbf{S} \circ \mathbf{S}_c(\delta), \tag{3.12}$$

where $\mathbf{S}_c(\delta)$ corresponds to either the truncated exponential kernel (Equation (3.10)) or the binarized truncated exponential kernel (Equation (3.11)).

There are several advantages to using a Hadamard product approach to combine the matrices. First, unlike the weighted sum approach, it discourages spatial units that are located far away from each other from being assigned to the same cluster even though their feature similarity is high. Second, it produces a sparser kernel matrix, which is advantageous for large-scale graph analysis. Finally, it gives more flexibility to the users to specify the level of constraints that must be preserved by tuning the parameter δ, which controls the ML neighborhood size of the constraint graph.

Let $\mathbf{D}_{ii}^{\text{total}} = \sum_j [\mathbf{S} \circ \mathbf{S}_{(c)}(\delta)]_{ij}$ be elements of a diagonal matrix computed from $\mathbf{S}^{\text{total}}$. The Hadamard product graph Laplacian is given by $\mathbf{L}^{\text{total}} = \mathbf{D}^{\text{total}} - \mathbf{S} \circ \mathbf{S}_c(\delta)$. The modified graph Laplacian can be substituted into Equation (3.2) and solved using the generalized eigenvalue approach to identify the regions.

3.4.3 Partitional Spatially-Constrained Spectral Clustering Algorithm

Algorithm 1 presents a high-level overview of our partitional clustering approach. First, a feature similarity matrix is created by applying the Gaussian radial basis function kernel, $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2})$ to the feature set of the spatial units. The spatially constrained kernel matrix $\mathbf{S}_c$ is then computed from the constraint matrix $\mathbf{C}$, where $\mathbf{C}_{ij} = 1$ if $(\mathbf{s}_i, \mathbf{s}_j)$ is a ML edge and 0 otherwise. Note that if the truncated exponential kernel is used to represent

Algorithm 1 Partitional Spatially-Constrained Spectral Clustering.

Input:
$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{s}_1), (\mathbf{x}_2, \mathbf{s}_2), ..., (\mathbf{x}_N, \mathbf{s}_N)\}$
$\mathbf{C} \in R^{N \times N}$: spatial constraint matrix.
k: number of clusters.
δ: neighborhood size.
Output:
$\mathcal{R} = \{R_1, R_2, ..., R_k\}$ (set of regions).

1. Create similarity matrix $\mathbf{S}$ from $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$.
2. Compute the spatially constrained kernel matrix, $\mathbf{S}_c(\delta)$.
3. Compute the combined kernel $\mathbf{S}^{\text{total}}$ based on $\mathbf{S}$ and $\mathbf{S}_c$.
4. Compute $\mathbf{D}^{\text{total}}$ and $\mathbf{L}^{\text{total}}$.
5. Solve the generalized eigenvalue problem $\mathbf{L}^{\text{total}} r = \lambda \mathbf{D}^{\text{total}} r$. Create matrix $\mathbf{X}_r = [r_1 r_2 \cdots r_k]$ from the top-k eigenvectors.
6. $\mathcal{R} \leftarrow$ `k-means(`$\mathbf{X}_r$`,`k`)`

the spatially constrained kernel matrix, we termed the approach as a spatially-constrained spectral clustering (SSC) algorithm. However, if the binarized truncated exponential kernel is used, the approach is known as a binarized spatially-constrained spectral clustering (BSSC).

Once the combined graph Laplacian, $\mathbf{L}^{\text{total}}$ is found, we extracted the first k eigenvectors as the low rank approximation of the combined kernel matrices. We then applied k-means clustering to partition the data into its respective regions. Note that the partitional clustering framework shown in Algorithm 1 is also applicable to the SCM and CSP algorithms, by setting their corresponding graph Laplacian, $\mathbf{L}^{\text{total}}$ and diagonal matrix, $\mathbf{D}^{\text{total}}$. The computational complexity of the spatially constrained spectral clustering is equivalent to the standard spectral clustering algorithm, which is $O(N^3)$.

3.4.4 Hierarchical Spatially-Constrained Spectral Clustering Algorithm

The formulation described in the previous section can be extended to hierarchical clustering by using a recursive bisection approach. Specifically, the algorithm will iteratively identify the least homogeneous region to be split into two smaller subregions until every subregion contains only a single spatial unit.

Algorithm 2 Hierarchical Spatially-Constrained Spectral Clustering (HSSC).

Input:
$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{s}_1), (\mathbf{x}_2, \mathbf{s}_2), ..., (\mathbf{x}_N, \mathbf{s}_N)\}$
$\mathbf{C} \in R^{N \times N}$: spatial constraint matrix.
δ: neighborhood size.
Output:
$\mathcal{R} = \{R_1, R_2, ..., R_k\}$ (set of regions).

1. Create similarity matrix $\mathbf{S}$ from $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$.
2. Compute the spatially constrained kernel matrix, $\mathbf{S}_c(\delta)$.
3. Compute the combined kernel $\mathbf{S}^{\text{total}}$ based on $\mathbf{S}$ and $\mathbf{S}_c$.
4. Compute $\mathbf{D}^{\text{total}}$ and $\mathbf{L}^{\text{total}}$.
5. Initialize R_1 as the cluster containing all N spatial units.
6. for $k = 2$ to N **do**
6a. $C^* = \text{choose}(R_{k-1})$
6b. $R_k \leftarrow R_{k-1} - C^*$
6c. $(C_1, C_2) \leftarrow \text{SSC}(\mathcal{D}_{C^*}, 2)$
6d. $R_k \leftarrow R_{k-1} \cup \{C_1, C_2)$.

The pseudocode of the proposed algorithm is shown in Algorithm 2. First, the feature similarity matrix $\mathbf{S}$ is computed using the Gaussian RBF kernel function. Next, the spatial constraint matrix $\mathbf{S}_c(\delta)$ is created using Equation 3.11. The algorithm will then compute the combined kernel $\mathbf{S}^{\text{total}}$ and its corresponding graph Laplacian matrix $\mathbf{L}^{\text{total}}$, similar to the approach described in Section 3.3.3. The algorithm initially assigns all the data points to a single cluster. It then recursively partitions the data until k clusters are obtained, as shown in lines 6a-6d in Algorithm 2. Let R_{k-1} be the set of clusters found after $k-1$ iterations. On line 6a, the algorithm chooses the cluster $\mathcal{C}^k \in R_{k-1}$ with the worst sum of square within errors (SSW) to be split into two smaller clusters, C_1 and C_2 (line 6c). One advantage of using our top-down recursive partitioning approach is that neither the feature similarity nor the spatial constraint matrix have to be updated at each iteration unlike the bottom-up hierarchical clustering, which requires us to re-compute the modified feature similarity and constraint matrices each time a pair of clusters is merged.

3.5 Application to Region Delineation

To evaluate the effectiveness of constrained spectral clustering for region delineation, we conducted a case study on a large-scale terrestrial ecology data set. The results of the case study are presented in this section.

3.5.1 Data set

The constrained spectral clustering methods were assessed using geospatial data from the LAGOS-NE_{GEO} [109] database. The database contains landscape characterization features measured at multiple spatial scales with a spatial extent that covers a land area spanning 17 U.S. states. The land area was divided into smaller hydrologic units (HUs), identified by their 12-Digit Hydrologic Unit Code [101]. Our goal was to develop a regionalization system for the landscape by aggregating the 20,257 HUs into coarser regions. We selected 28 terrestrial landscape variables and performed experiments on three study areas—Michigan, Iowa, and Minnesota. When the values for a landscape variable was always zero, we removed that variable before applying the clustering methods. The number of HUs to be clustered in each study region, as well as number of landscape variables for each, are summarized in Table 4.2.

Table 3.1: Summary statistics of the data set.

Study Area	# HUs	# landscape variables	# PCA components	Diameter of constraint graph
Michigan	1,796	17	10	41
Iowa	1,605	19	12	43
Minnesota	2,306	19	11	57

The data set was further preprocessed before applying the constrained clustering algorithms. First, each variable was standardized to have a mean value of zero and variance of one. Since some of the landscape variables were highly correlated, we applied principal component analysis to reduce the number of features, keeping only the principal components

that collectively explained at least 85% of the total variance. The principal component scores were then used to calculate a feature similarity matrix for all pairs of HUs in each study area. The ML edges for the constraint graph were determined based on whether the polygons for two HUs were adjacent to each other.

3.5.2 Baseline Methods

For partitional-based constrained clustering, we compared our algorithms, SSC and BSSC, against three competing baseline methods. The first baseline, called SCM [105], uses a weighted sum approach (Equation (3.3)) to combine the feature similarity matrix $\mathbf{S}$ with the adjacency matrix $\mathbf{C}$ of the constraint graph. The algorithm has a parameter $\delta \in [0, 1]$ that controls whether the clustering should favor homogeneity or spatial contiguity of the regions. When δ approaches 0, the algorithm is biased towards maximizing the similarity of features in the regions whereas when δ approaches 1, it is biased towards producing more contiguous regions.

The second baseline method, called CSP [128], uses the spatial constraints to restrict the feasible set of the clustering solution (Equation (3.6)). As noted in Section 3.3.3, the algorithm has a parameter β that gives a lower bound on the proportion of constraints that must be satisfied by the clustering solution. Furthermore, $\beta < \lambda_{\max} vol(\mathcal{G})$ to ensure the existence of a feasible solution [128]. Instead of using β, we define an equivalent tuning parameter $\delta = \beta/[\lambda_{max} vol(\mathcal{G})]$ so that its upper bound, which is equal to 1, is consistent with the upper bound for other algorithms evaluated in this study.

The third baseline is a spatially constrained clustering method proposed recently in the ecology literature by Miele et al. [80]. It uses a stochastic model to represent nodes and links in a spatial ecological network. The cluster membership of each node is assumed to follow a multinomial distribution. Spatial constraints are introduced as a regularization penalty in the maximum likelihood estimation of the model parameters. The algorithm is implemented as part of the Geoclust R package. We denote the model-based method as MB

in the remainder of this chapter.

For hierarchical clustering, we compare our proposed HSSC algorithm against the space-constrained clustering method described in [66]. The method is similar to traditional agglomerative hierarchical clustering, except it applies a Hadamard product between the feature similarity matrix $\mathbf{S}$ with the spatial constraint matrix $\mathbf{S}_c$ to generate a combined similarity matrix $\mathbf{S}^{\text{total}}$. This is identical to the approach used in HSSC. The agglomerative clustering algorithm initially assigns each spatial unit to be in its own cluster (region). It then merges the two clusters with the highest similarity value in $\mathbf{S}^{\text{total}}$. Both the feature similarity matrix $\mathbf{S}$ and the spatial constraint matrix $\mathbf{S}_c$ are then updated accordingly. The update for $\mathbf{S}$ depends on how the similarity between two clusters is computed. Among the popular approaches that have been used to update $\mathbf{S}$ include single link [106], complete link [111], group average (UPGMA) [108], and the Ward's method [130]. The adjacency matrix $\mathbf{C}$ is updated based on whether there is a path from any point in one cluster to any point in the other cluster and the constrained similarity matrix $\mathbf{S_c}$ is updated based on Equation 3.10 with a predefined δ.

We implemented SCM, SSC, BSSC, HSSC and the spatially constrained agglomerative hierarchical clustering (single link, complete link, UPGMA, Ward's method) in Matlab. For CSP and MB, we downloaded their software from the links provided by the authors[1].

3.5.3 Evaluation Metrics

We evaluated the performance of the algorithms based on three criteria: homogeneity, spatial contiguity, and region size. To determine whether the regions were ecologically homogeneous, we computed their within-cluster sum-of-square error (SSW) [112]:

$$\text{SSW} = \sum_{i=1}^{k} \sum_{x \in C_i} dist(\mu_i, x)^2 \tag{3.13}$$

[1]CSP was obtained from `https://github.com/gnaixgnaw/CSP` whereas MB was downloaded from `http://lbbe.univ-lyon1.fr/Download-5012.html?lang=fr`.

where μ_i is the centroid of the cluster C_i. The lower SSW is, the more homogeneous are the spatial units within the regions.

The second criteria assesses the spatial contiguity of the resulting regions. We consider two metrics for this evaluation. The first metric computes the percentage of ML constraints preserved within the regions:

$$\text{PctML} = \frac{\#\text{ ML edges within discovered regions}}{\text{Total }\#\text{ of ML edges}} \tag{3.14}$$

The second metric corresponds to a relative contiguity metric proposed in the ecology literature by Wu and Murray [136]. The metric takes into consideration both the within patch contiguity (ϕ) and between patch contiguity (ν):

$$c = \frac{\phi + \nu}{\Omega} \tag{3.15}$$

where

$$\begin{aligned} \phi &= \sum_{i=1}^{k} \left(\frac{N_i(N_i-1)}{2}\right), \\ \nu &= \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1, j\neq i}^{k} \left(\frac{N_i N_j}{l_{ij}^{\gamma}}\right) \\ \Omega &= \frac{(\sum_{i=1}^{k} N_i)(\sum_{i=1}^{k} N_i - 1)}{2} \end{aligned} \tag{3.16}$$

In the preceding formula, k is the number of regions and N_i is the number of spatial units assigned to the i-th region. l_{ij} denote the minimum spanning tree path length between regions i and j while γ is a distance decay parameter. Since the metric is normalized by the total number of possible edges in a complete graph (Ω), it ranges between 0 and 1.

Although spatial contiguity is a desirable criterion, it may lead to highly imbalanced regions [83]. For example, an algorithm that creates one very large region along with many smaller but contiguous regions will likely have a high contiguity value. Previous studies [30, 6] have shown the importance of maintaining a more balanced cluster sizes to ensure good clustering performance. Thus, given a set of k clusters with their corresponding cluster

sizes, $n_1, n_2, ..., n_k$, we define a metric, *Cbalance*, based on the normalized geometric mean of the cluster sizes:

$$Cbalance = \frac{k}{N}\left[n_1 \times n_2 \times ... \times n_k\right]^{\frac{1}{k}}, \tag{3.17}$$

where N is the total number of data points and k is the number of clusters. The metric ranges from 0 to 1 and the larger the value, the more balanced are the cluster sizes.

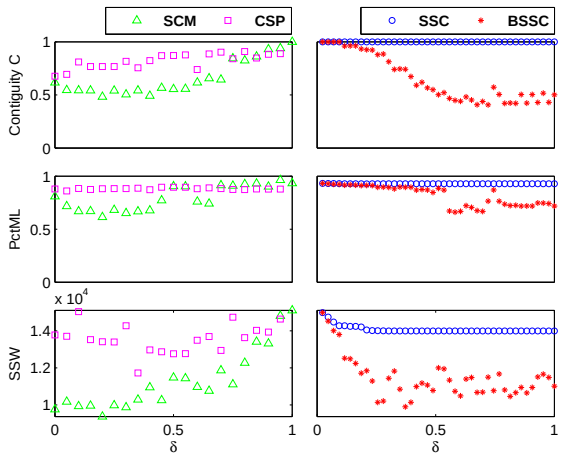
3.5.4 Results and Discussion

This section presents the results of applying various clustering algorithms to the terrestrial ecology data.

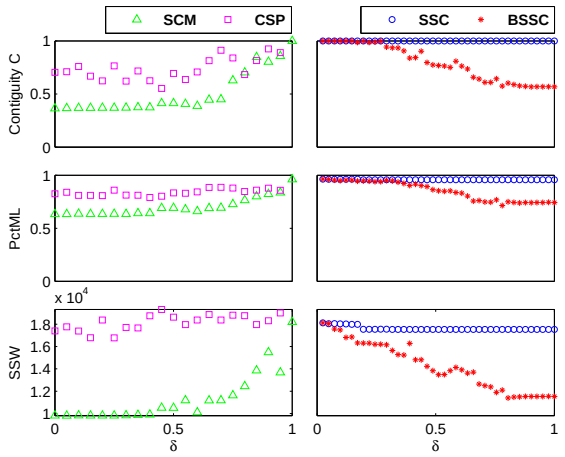
3.5.4.1 Tradeoff between Homogeneity and Spatial Contiguity

We first analyze the trade-off between landscape homogeneity and spatial contiguity of the regions by comparing the results for four partitional constrained spectral clustering algorithms: SCM, CSP, SSC, and BSSC. The number of clusters was set to 10. As each algorithm has a parameter δ that determines whether the clustering should be more biased towards increasing the within-cluster similarity or preserving the ML constraints, we varied the parameter and assessed their performance using the metrics described in Section 3.5.3. The δ parameter for SSC and BSSC has been re-scaled to a range between 0 and 1 by dividing the ML neighborhood size with the diameter of the constraint graph.

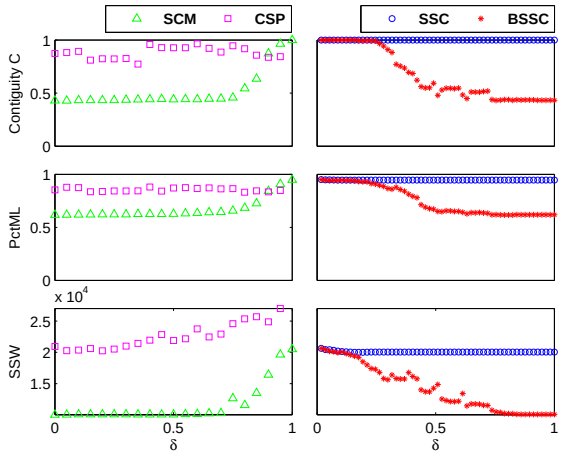
The results are shown in Figure 3.2. Observe that the contiguity score (c and PctML) for SCM increases rapidly as δ becomes closer to 1. This is because increasing δ would bias the algorithms towards preserving the spatial constraints. A similar increasing trend was also observed for CSP, especially in Iowa and Michigan, though the increase is not as sharp as SCM. In contrast, the contiguity scores would decrease for BSSC as δ increases because it creates more new ML edges involving spatial units that are not adjacent to each other. For SSC, the contiguity scores do not appear to change by much as δ increases. This is



(a) Iowa



(b) Michigan



(c) Minnesota

Figure 3.2: Comparison between various constrained spectral clustering algorithms in terms of their landscape homogeneity (SSW) and spatial contiguity (PctML and c). The horizontal axis in the plots corresponds to the parameter value δ.

because the weight $1/k!$ associated with each path of length k decreases rapidly to zero as k increases. As a consequence, the ML neighborhood size for SSC grows until it reaches a maximum size by which increasing δ will not significantly alter the constraint graph. Thus, SSC is less sensitive to parameter tuning compared to BSSC. Figure 3.2 also shows there is generally an increasing trend in SSW for SCM and CSP as δ increases. For SSC, the SSW values do not appear to change significantly with increasing δ whereas for BSSC, the SSW curve decreases monotonically as the neighborhood size increases.

The results of this study showed that the trade-off between landscape homogeneity and spatial contiguity varies among the constrained spectral clustering algorithms. For CSP and SSC, the parameters provided by the algorithms do not allow us to achieve the full range of SSW and contiguity scores. Although these algorithms can produce regions with high contiguity scores, their SSW values were also very high. In contrast, with careful parameter tuning, SCM and BSSC can produce regions with significantly lower SSW compared to CSP and SSC. Observe that the slopes of the curves are steeper near $\delta = 1$ for SCM, which suggests that decreasing δ below 1 would lead to a dramatic reduction in the contiguity score and SSW of the regions. This makes it harder for SCM to produce regions that are both spatially contiguous and homogeneous. In contrast, the curves for the contiguity scores of BSSC are flatter near $\delta = 0$. This enables the BSSC algorithm to produce regions with homogeneous landscape features yet are still spatially contiguous.

3.5.4.2 Performance Comparison for Partitional based Constrained Clustering

In this experiment, we set the number of clusters to 10 and selected the δ parameter that gives the highest contiguity score for each constrained spectral clustering method. If there are more than one parameter values that achieve the highest contiguity score, we chose the one with lowest SSW. For MB, since the Geoclust R package did not support parameter tuning by users, we applied the algorithm using its default setting.

Table 3.2 summarizes the results of our analysis. SCM, SSC, and BSSC can be tuned to

Table 3.2: Performance comparison among various partitional spatially-constrained clustering algorithms with the number of clusters set to 10.

States	Method	PctML	c	SSW	Cbalance
IA	SCM	93.26%	1.00	15104	0.95
	CSP	87.37%	0.91	13628	0.19
	MB	89.95%	0.69	18997	0.34
	SSC	92.83%	1.00	13993	0.95
	BSSC	92.40%	1.00	14001	0.94
MI	SCM	96.08%	1.00	18200	0.85
	CSP	87.81%	0.92	18307	0.44
	MB	88.76%	0.65	16091	0.91
	SSC	95.69%	1.00	17534	0.92
	BSSC	94.92%	1.00	17485	0.93
MN	SCM	94.78%	1.00	20506	0.91
	CSP	86.62%	0.96	23755	0.69
	MB	88.96%	0.64	20400	0.67
	SSC	94.57%	1.00	19998	0.93
	BSSC	94.12%	1.00	19594	0.91

produce regions that are fully contiguous ($c = 1$). The SSW for BSSC and SSC are consistently better than SCM. These results clearly showed the advantage of using a Hadamard product approach instead of a weighted sum approach to integrate spatial constraints into the feature similarity matrix. The limitation of using a weighted sum approach can be explained as follows. Since the highest contiguity score is achieved by setting $\delta = 1$, the clustering solution of SCM is equivalent to applying spectral clustering on the constraint graph only, without considering the feature similarity. If we reduce the parameter value to, say $\delta = 0.95$, its contiguity score decreases sharply (see Figure 3.2) while its SSW value is still worse than BSSC. The weighted sum approach has poor SSW because it significantly alters the feature similarity matrix.

To illustrate the limitation of using the weighted sum approach, consider the toy example shown in Figure 3.3. Assume there are 4 data points: A, B, C and D, that need to be clustered. A sample of their pairwise similarity values is shown in Table 3.3.

Although the A-B pair has a significantly lower similarity value than C-D, the weighted sum approach inflates the similarity significantly (assuming $\delta = 0.95$) which makes it overall

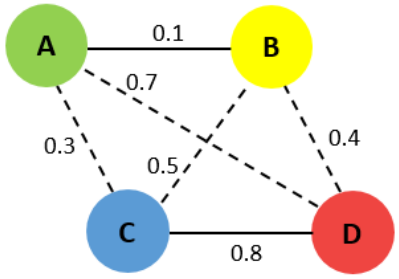


Figure 3.3: A toy example illustrating the advantage of using Hadamard product for combining constraints with feature similarity. Each labeled node is a data point, with a solid line representing a must-link constraint between two points and a dashed line representing the absence of such constraint. The weight of each edge denotes the feature similarity between two data points.

Table 3.3: A toy example illustrating the advantage of using Hadamard product for combining constraints.

Pairs	Feature Similarity	ML Constraint	Weighted Sum	Hadamard Product
A-B	0.1	1	0.955	0.1
B-C	0.5	0	0.025	0
C-D	0.8	1	0.990	0.8

similarity to be comparable to C-D. In contrast, the Hadamard product approach simply zeros out the similarity of pairs that do not have ML edges, and thus, will not artificially inflate the similarities of pairs with ML edges.

Furthermore, since the feature similarity is computed using Gaussian radial basis function (see Section 3.4.3), the resulting matrix $\mathbf{S}$ for the weighted sum approach is still dense after incorporating the spatial constraints. Unless $\delta = 1$, the weighted sum approach will not prevent spatial units that are located far from each other from being placed into the same region. For example, consider the regions found by the weighted sum approach for Iowa, as shown in Figure 3.4. Although the regions appear to be spatially contiguous, they are not compact and have varying sizes. In fact, most of the spatial units were assigned to the same region when $\delta = 0.95$. Even at the lower δ threshold, its SSW (14805) is still the worse than the SSW for our framework and CSP.

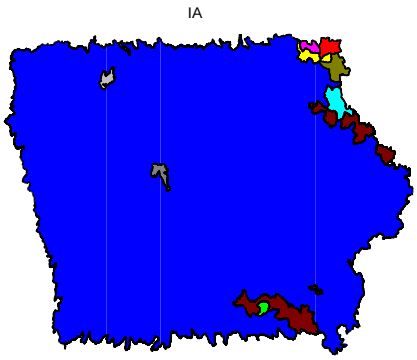
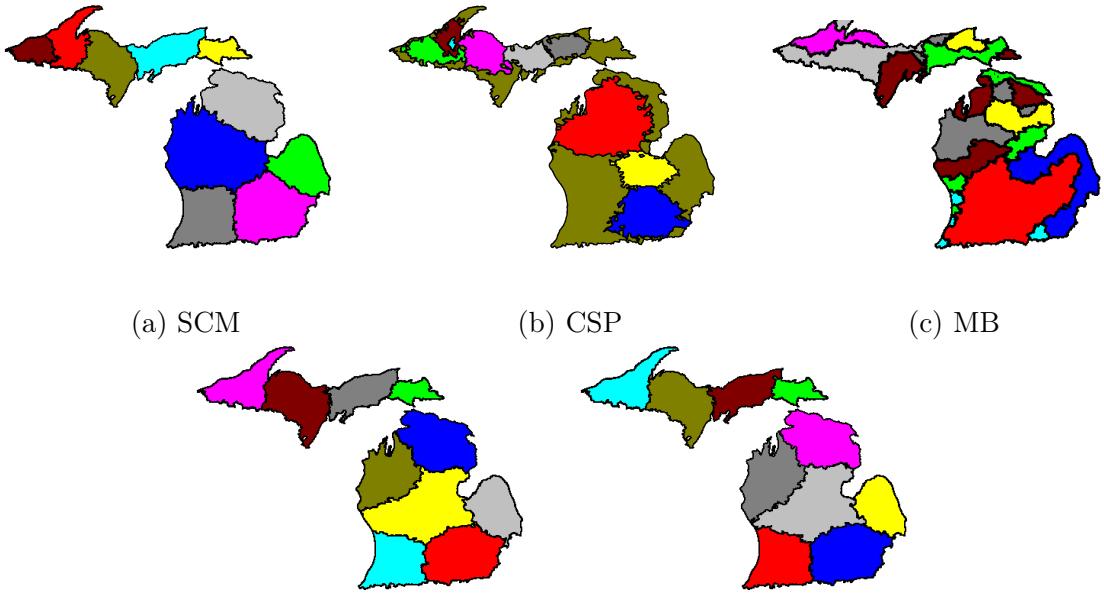


Figure 3.4: Regions for Iowa created by the SCM algorithm using the weighted sum approach (with $\delta = 0.95$).

The contiguity scores for MB are worse than other constrained clustering methods. Nevertheless, it preserves at least 88% of the ML edges within the regions. Except for Michigan, its SSW values are also worse than other methods. In contrast, CSP has the lowest contiguity score among all the constrained spectral clustering methods. Except for Iowa, its SSW values are also among the worst. The limitation of CSP [128] is a consequence of the parameter used to control its spatial contiguity. As shown in Equation (3.6), the level of spatial constraints satisfied by the clustering solution depends on the parameter α. However, instead of directly tuning α, the authors suggested to vary another parameter, β, which was shown to be an upper bound of α. The results of our case study showed that increasing the value of β does not necessarily imply an increase in α. To illustrate this point, we randomly generated a constraint graph that has nine vertices with a randomly generated feature similarity matrix. Assuming the number of clusters is equal to 2, we ran the CSP algorithm with different parameter settings and plotted their values of α and β in Figure 3.6. Although this figure shows that the value of β (blue diamond) is a lower bound of α (red circle), the bound is so loose that it can not guarantee that increasing β will increase α. In fact, the figure on the right shows that α is not a monotonically increasing function of β. This is why controlling its parameter value will not always guarantee that the regions will be contiguous even when $\delta = 1$ (unlike SCM and the Hadamard product approaches).

In terms of the Cbalance measure, our results suggest that SCM, SSC, and BSSC achieve



(a) SCM (b) CSP (c) MB

(d) SSC (e) BSSC

Figure 3.5: Regions created by the SCM, CSP, MB, SSC and BSSC algorithms for the state of Michigan.

the highest cluster balance for all three states. This can be verified by examining the regions generated by all the competing algorithms for the the state of Michigan, as shown in Figure 3.5[2] As can be seen from the figure, the regions produced by SCM, SSC, and BSSC are more compact and uniform in size compared to CSP and MB. However, the SSW for SCM is worse than the SSW for our proposed SSC and BSSC algorithms. This is not surprising as SCM cannot produce contiguous clusters unless δ is very close to 1. If δ is lowered slightly to 0.95, the regions changed significantly, as shown in Figure 3.4. By setting δ close to 1, SCM will focus only on preserving the spatial constraints, and thus, has worse cluster homogeneity compared to our algorithms. Thus, our results clearly show the benefits of applying BSSC to develop homogeneous and spatially contiguous regions compared to other baseline algorithms. These results hold true even when the number of regions is varied. A comparison of the results for different number of clusters can be found in our earlier work [146].

[2]The corresponding maps of regions for other states can be found in [146].

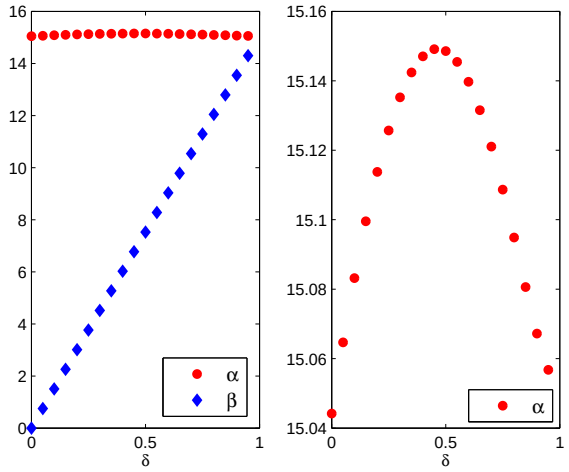


Figure 3.6: The relationship between α and β parameter values for the CSP algorithm when applied to a synthetic graph data.

3.5.4.3 Performance Comparison for Hierarchical-based Constrained Clustering

In this section, we compared our proposed HSSC algorithm against the spatially-constrained agglomerative clustering methods for constructing nested regions. Note that all of the algorithms apply a Hadamard product to combine the constrained matrix $\mathbf{S}_c$ (for a given δ) with the feature similarity matrix $\mathbf{S}$ to generate the combined matrix $\mathbf{S}^{\text{total}}$ before applying hierarchical clustering. For the spatially-constrained agglomerative hierarchical clustering methods, the regions are iteratively merged starting from the initial $\mathbf{S}^{\text{total}}$.

For a fair comparison, we set $\delta = 1$ for all the methods. The results for $k = 10$ are summarized in Table 3.4. In terms of region contiguity, observe that all the methods can achieve $c = 1$. However, the spatially constrained complete link and UPGMA algorithms produce the highest PctML values while the Ward's method produces the lowest value. The PctML for our proposed HSSC algorithm is still relatively high and comparable to its non-hierarchical counterparts, SSC and BSSC (see Table 3.2). Despite their high spatial contiguity, both the spatially-constrained complete link and UPGMA methods have the worst SSW compared to other methods. Worst still, their Cbalance values are close to 0, suggesting that the sizes of their regions are highly imbalanced. This can be seen from the maps shown

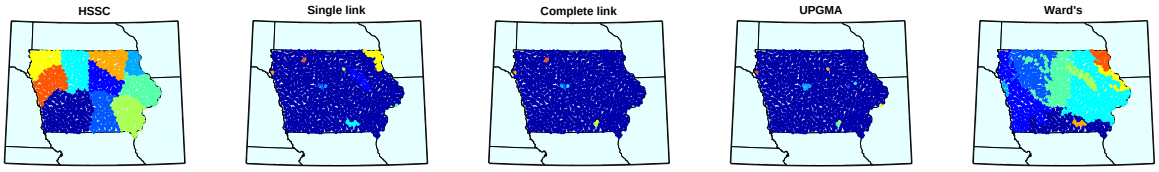
Table 3.4: Performance comparison among various hierarchical spatially-constrained clustering algorithms with $\delta = 1$ and the number of clusters set to 10.

States	Method	PctML	c	SSW	Cbalance
IA	HSSC	92.53 %	1	15080	0.92
	Single link	96.02 %	1	14191	0.04
	Complete link	98.75 %	1	18309	0.02
	UPGMA	98.45 %	1	18227	0.02
	Ward's	84.96 %	1	9281	0.48
MI	HSSC	95.41 %	1	17420	0.86
	Single link	95.31 %	1	16575	0.08
	Complete link	98.72 %	1	20083	0.03
	UPGMA	98.33 %	1	19441	0.04
	Ward's	86.28 %	1	14047	0.79
MN	HSSC	95.02 %	1	20075	0.82
	Single link	90.70 %	1	19660	0.20
	Complete link	99.17 %	1	33431	0.01
	UPGMA	98.81 %	1	28681	0.02
	Ward's	87.69 %	1	14183	0.63

in Figure 3.7, where there is a large region covering the majority of the landscape in each state. In contrast, our HSSC algorithm has the highest Cbalance, consistently producing regions that are compact and approximately similar in sizes.

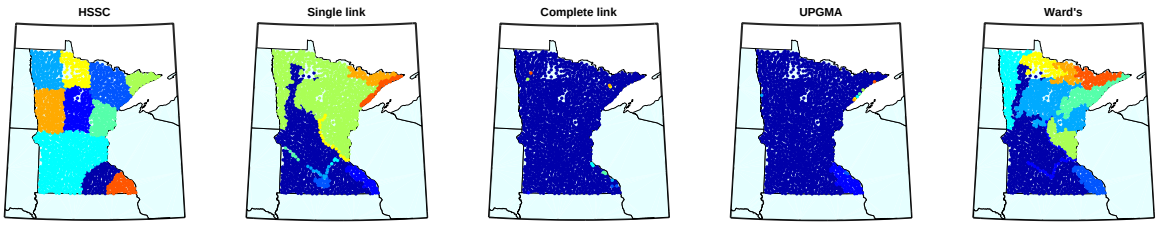
The spatially-constrained single link method has comparable PctML but slightly lower SSW compared to HSSC. It also suffers from the same imbalance region problem as the complete link and UPGMA methods. Meanwhile, the spatially-constrained Ward's method achieves the lowest SSW among all the competing methods, which is not surprising since the algorithm is designed to minimize the SSW in each iteration of the algorithm. However, this comes at the expense of its poor PctML values, which is the worst among all the competing methods. In addition, the Ward's method is known to suffer from the cluster inversion problem [83], in which its objective function is not monotonically non-decreasing as the number of clusters increases. In short, our HSSC algorithm outperforms the complete link, UPGMA, and Ward's methods in 2 of the 3 evaluation criteria. Its PctML and SSW is also quite similar to single link, which suffers from the region imbalanced problem.

Figures 3.8a and 3.8b show a comparison between the regions produced by BSSC, HSSC,



(a) Regions in IA developed by 5 hierarchical algorithm with 10 clusters.

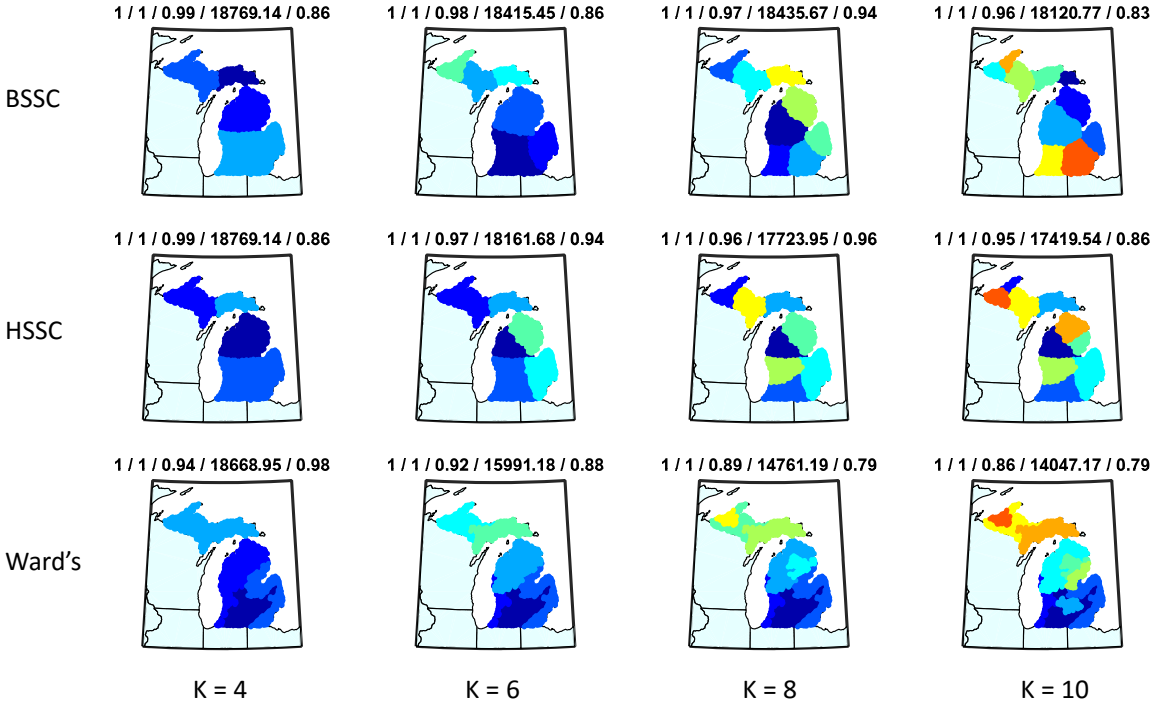
(b) Regions in MI developed by 5 hierarchical algorithm with 10 clusters.



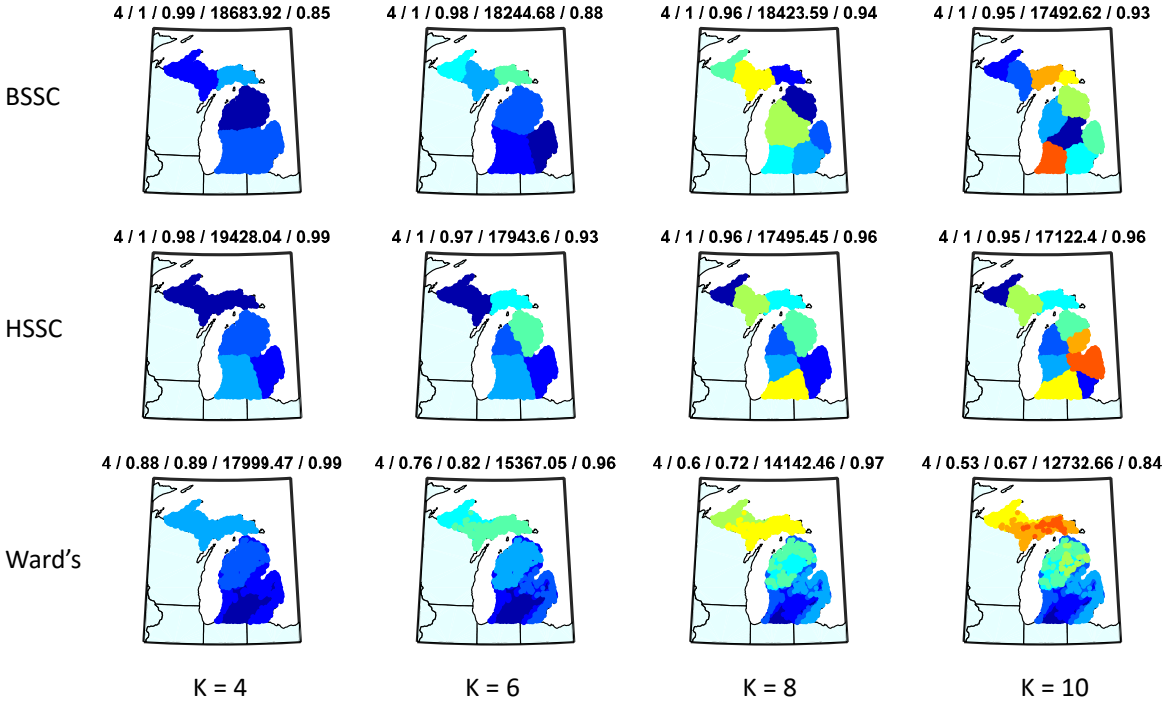
(c) Regions in MN developed by 5 hierarchical algorithm with 10 clusters.

Figure 3.7: Regions developed by 5 hierarchical spatially constrained clustering algorithm for 3 study regions.

and the Ward's method for the state of Michigan as we vary the number of regions from 4 to 10. We show the value of the unnormalized δ along with four metrics—c, PctML, SSW, and Cbalance—at the top of each diagram. Recall that the normalized δ is the ratio between the original δ given in Equation (3.7) and the diameter of the spatial constraint graph. As we increase δ from 1/41 to 4/41, the Ward's method no longer produces regions that are contiguous unlike the BSSC and HSSC methods. The Cbalance for Ward's method is also worse than our algorithms except when the number of clusters is small. The results for BSSC is quite comparable to HSSC, since both of them are based on the same spatially constrained



(a) Results for $\delta = 1/41$



(b) Results for $\delta = 4/41$

Figure 3.8: Comparison between BSSC, HSSC and Ward's method for number of cluster $K = 4, 6, 8, 10$. The five metrics evaluated are listed on top of each figure, namely: unnormalized delta, contiguity metric c, PctML preserved, SSW and cluster balance. Results for $\delta = 1/41$ are shown in (a) and results of the unnormalized $\delta = 4/41$ are shown in (b).

spectral clustering framework. The Cbalance and SSW for HSSC are slightly better than BSSC but its PctML is slightly worse. While this may seem counter-intuitive since BSSC directly optimizes the objective function for spatially-constrained spectral clustering whereas HSSC uses a greedy recursive bisection strategy, it is worth noting that the objective function does not depend solely on the feature similarities within the regions. Instead, it takes into account the spatial constraint matrix C as well. In fact, if we compare the values of the objective functions for BSSC and HSSC at $k = 10$ for the state of Michigan, BSSC has a noticeably lower value (13458.13) compared to HSSC (14284.21).

Finally, we also compare the stability of the regions as we increase ML neighborhood size (δ). For this experiment, we show the results for the state of Michigan (in which the diameter of the constraint graph is 41) and varies the normalized δ from 1/41 to 4/41. We use the adjusted rand index [94] to compare the similarity between two clustering results. A high adjusted rand index would suggest that the regions found are stable, i.e., do not vary significantly with different values of δ. Table 3.5 shows the mean adjusted rand index (averaged over the number of clusters, which varies from 1 to 10) for HSSC, BSSC, and Ward's method. The results suggest that the proposed BSSC and HSSC methods are less sensitive to the change in δ compared to the Ward's method, which is another advantage of using our frameworks.

	BSSC	HSSC	Ward's
Mean Adjusted Rand Index	0.92	0.86	0.66

Table 3.5: Stability of the regions generated by different hierarchical clustering methods for the state of Michigan. The mean Adjusted Rand Index is computed for each method by comparing the similarity between the regions found with $\delta = 1/41$ to the regions found with $\delta = 4/41$.

3.6 Conclusions

This chapter investigated the feasibility of applying constrained spectral clustering to the regionalization task. We compared several constrained spectral clustering methods and

showed the trade-off between landscape homogeneity and spatial contiguity of their resulting regions. We presented two algorithms, SSC and BSSC, that uses a Hadamard product approach to combine the similarity matrix of landscape features with spatial contiguity constraints. The results of our case study showed that the proposed BSSC algorithm is most effective in terms of producing spatially contiguous regions that are homogeneous. The extension of this algorithm to a hierarchical clustering setting also shows its advantages in producing regions that are more balanced in size compared to other hierarchical spatially-constrained algorithms. It also achieves high spatial contiguity and moderate SSW, comparable to the results of its non-hierarchical counterpart (BSSC).

CHAPTER 4

LEARNING HASH-BASED FEATURES FOR INCOMPLETE CONTINUOUS VALUED DATA

4.1 Introduction

Real world data sets are often noisy, making it difficult to develop accurate prediction models from the data. The data are often high-dimensional and may contain redundant or correlated features, as well as missing values, which makes it crucial to derive a good set of features to represent the data. This has led to considerable interest in developing feature learning methods to overcome the limitations of using the original features of the data. Numerous methods are available, from classical methods such as principal component analysis [90] to more recent ones such as hash-based feature learning [64].

Hashing is a popular feature learning technique for transforming high dimensional data into an alternative representation that preserves the similarities between instances in the original data [126]. There are two main advantages of using hashing for feature learning. First, it provides an effective dimensionality reduction technique, especially for applications such as large-scale image and multimedia retrieval problems [59]. Second, the similarity preserving property of the hash functions enables the hash-based features to be used as an approximation to the features defined in the Reproducing Kernel Hilbert Space (RKHS). This allows us to construct linear models on the hash-based features with comparable accuracy as their nonlinear counterpart, but with substantial improvement in computational complexity.

Hash-based feature learning methods can be generally classified into two categories, depending on how the features are created [126]. The first category corresponds to data-independent methods, which create the features by applying randomly generated hash functions to the data. This includes the min-hash [15], random hyperplane-based hashing [20], and shift-invariant kernel hashing [92] methods. One potential limitation of using data-

independent methods is that the number of hash-based features needed to provide a good representation of the data can be very large since the hash functions are generated randomly. Thus, data-driven methods have been developed as an alternative to such methods as they can fit the salient properties of the data using a small set of hash functions. Methods that belong to this second category include spectral hashing [131], semi-supervised hashing [127], and minimal loss hashing [85].

Existing hash-based feature learning methods assume that the input data are complete. Any missing values present in the data must be imputed before the hash functions can be derived. Because the imputation is typically performed during preprocessing, the hash-based features created after the preprocessing may not be optimal for the subsequent modeling task. As an illustration, consider the example shown in Figure 4.1. Suppose P denotes a data point that has a missing value for its predictor variable (the x-axis), but the value of its response variable (the y-axis) is known. After applying mean imputation, the data point P is shifted to its estimated point P', which is much larger than its true value. This process introduces a bias into the data set, resulting in a new regression model (represented by a solid line in the diagram) that deviates from its true model (represented by a dashed line). At first glance, the bias may seem quite small. However, it may become an issue with more missing values in the data. If P is imputed in a way that takes into account the effect of the imputation on the response variable, the regression model would be less affected by biases due to the imputation. To overcome this challenge, we present a novel framework called `H-FLIP` that combines missing value imputation with hash-based feature learning. Specifically, the missing values are imputed in a way that preserves the relationship between the predictor and response variables of the data.

Many existing hash-based feature learning methods are designed for discrete-valued data, such as those encountered in image classification problems [59]. In this chapter, we investigate the application of hash-based feature learning for regression problems. Specifically, the hash-based features created with our approach are continuous rather than binary. In addition,

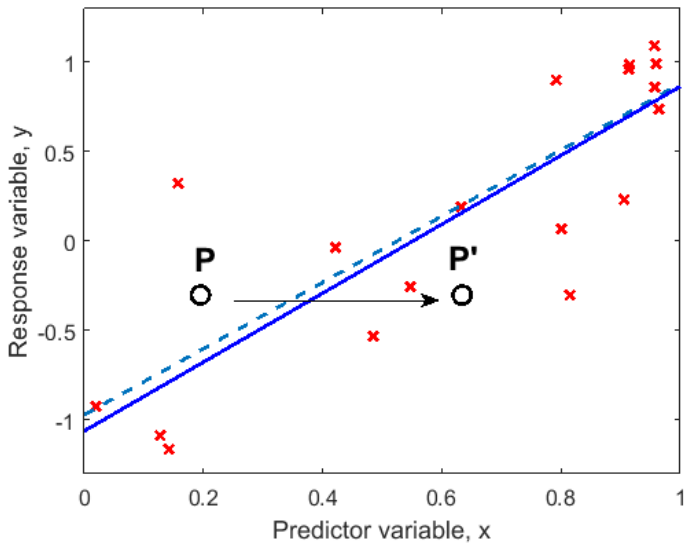


Figure 4.1: Effect of mean value imputation on regression.

many existing methods create their hash-based features using a linear combination of the original features. Since the relationship between the response and predictor variables is often non-linear, these methods make the hash-based features ineffective for more complex prediction tasks. Previous studies have shown that the inner product of random Fourier features provides a good approximation to the shift-invariant kernels that are often used for building nonlinear models [93]. However, since the Fourier features are generated randomly, a relatively large number of such features is needed to provide a succinct representation of the data. We address this problem by training a set of hash functions to fit the response variable using random Fourier features. This enables our framework to model nonlinear relationships in the data using a small set of hash functions. Our framework can thus be regarded as a hybrid method that embeds a data-independent approach, i.e., random Fourier features, into a supervised learning setting.

In summary, the main contributions of this chapter are:

- We developed a supervised feature learning framework for regression problems using nonlinear, random Fourier features as its basis functions.

- We extended this framework to handle incomplete data by enabling the missing value

imputation and hash-based feature learning to be performed jointly.

- We demonstrated the efficacy of our approach through extensive experiments using both synthetic and real-world data sets. Our empirical results showed that the framework is more effective than the standard approach of imputing the missing values before applying feature learning in 7 out of 9 real-world data sets evaluated in this study.

The remainder of this chapter is organized as follows. Section 4.2 reviews the previous works related to this research while Section 4.3 presents the preliminary background of this work. The proposed framework along with its optimization algorithm are described in Section 4.4. Experimental results validating the effectiveness of the framework is presented in Section 4.5. Finally, we present the conclusions of this study in Section 4.6.

4.2 Related Work

High dimensionality is one of the characteristics of real world datasets. To avoid the curse of dimensionality we need a powerful set of features in low dimensional space. Feature learning [10] is the task of learning an better feature representation of a given data set.There are different types of models for feature learning, such as probabilistic models [97, 87], auto-encoders [14, 147], manifold learning [143], and deep networks [61, 149]. Hashing is another feature learning technique that has been well studied. The goal of hash-based feature learning is to transform the data into easily computable features that preserve the underlying properties of the data, such as Min-hash and random hyperplane mapping. However, they were mostly designed to represent complete data with binary hash codes unlike the framework proposed in this chapter.

Missing values can be generally classified into three categories—missing completely at random (MCAR), missing at random (MAR), or missing not at random (MNAR) [70]. A common strategy to deal with missing values is to impute them with the mean or median value of their respective features. Model-based approaches based on maximum likelihood and Bayesian estimation methods have also been developed [69]. These methods assume

that the data follow a certain probability distribution and the missing values are imputed in a way that preserve properties of the distribution. More recently, matrix completion [18] has emerged as a popular method for dealing with missing values. The method assumes that the partially observed data matrix has a low rank, and thus, can be effectively recovered using only a small number of observations. Candès and Recht [18] formulated matrix completion as a trace-norm minimization problem and solved a convex relaxation of the problem using semi-definite programming. The approach was extended by Cai et al. [17] who proposed a singular value thresholding (SVT) algorithm for solving the optimization problem. In this chapter, we integrate the matrix completion formulation into our framework to deal with incomplete data.

4.3 Preliminaries

This section reviews some of the fundamental concepts underlying the framework proposed in this chapter.

4.3.1 Support Vector Regression (SVR)

Support vector regression is a widely-used method to solve large scale prediction problems. The method is grounded in statistical learning theory, based on the structural risk minimization (SRM) principle [118]. The goal of SVR is to learn a linear function f that fits each instance in the training set $\mathbf{D} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$ with an absolute error bounded by ϵ. Formally, this can be expressed by the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_i (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & -\epsilon - \xi_i^* \leq y_i - \mathbf{w}^T \mathbf{x}_i - b \leq \epsilon + \xi_i, \end{aligned}$$

where ξ_i and ξ_i^* are the slack variables that can be used to relax the error bounds on the training instances and C controls the trade-off between minimizing the training error and the magnitude of $\mathbf{w}$.

Table 4.1: Comparison among the various regression methods for modeling lake water quality in terms of their mean square prediction error (MSE).

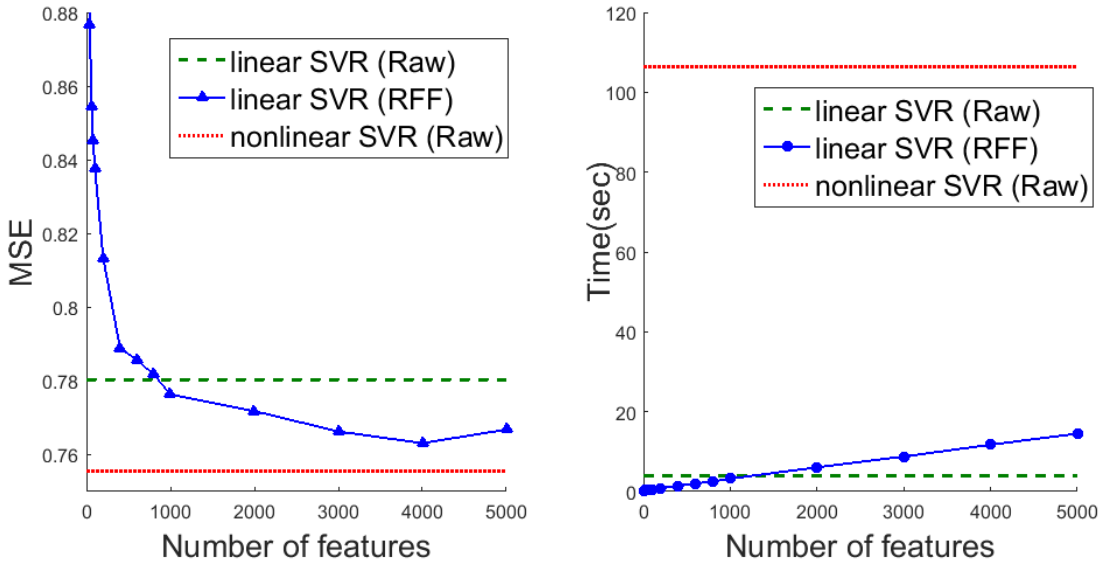
Method\Response variable	TP	TN	Chla	Secchi
Multiple linear regression	1.39	2.39	1.31	0.85
Ridge regression	0.78	0.72	**0.71**	0.59
Lasso regression	0.79	0.74	0.72	0.58
Linear SVR	0.78	0.71	**0.71**	0.58
Nonlinear SVR (RBF)	**0.76**	**0.69**	**0.71**	**0.56**

The linear SVR formulation can be extended to a nonlinear setting by projecting the original features $\mathbf{x}$ to a higher dimensional feature space, $\Phi(\mathbf{x})$, such that the inner product between instances in the new, projected space is equivalent to computing their similarity in the original space using a nonlinear kernel function. For example, the Gaussian radial basis function (RBF), $k(\mathbf{x}_i, \mathbf{x}_j) = \exp[-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}]$ is a popular choice for nonlinear SVR. The implicit mapping to a high-dimensional feature space facilitated by the kernel function enables SVR to capture non-linear dependencies in the data.

To illustrate the advantages of using SVR, we have compared its performance to other regression methods on four lake water quality data sets obtained from the LAGOS-NE database [109]. Details about the data sets can be found in Section 4.5. Table 4.1 shows the mean square error (MSE) obtained using 10-fold cross validation. Due to the noise present in the data, methods such as multiple linear regression may overfit the training data resulting in its high error rate. The performance of linear SVR is either comparable to or better than the results obtained using lasso and ridge regression. However, the best results are obtained using nonlinear SVR with Gaussian RBF kernel.

4.3.2 Random Fourier Features (RFF)

Despite its superior performance in terms of modeling complex relationships in data, the nonlinear SVR method scales poorly with increasing data set size. This is because the method requires considerable computational resources to compute and store the kernel matrices. To overcome this limitation, Rahimi and Recht [93] proposed a mapping function known as



(a) Accuracy (b) Run time

Figure 4.2: Average runtime and accuracy comparison for linear SVR with raw features, nonlinear SVR with raw features, and linear SVR with RFF.

random Fourier feature:

$$\phi(\mathbf{x}) = \sqrt{2/p}\cos(\mathbf{R}\mathbf{x} + \mathbf{t}) \tag{4.1}$$

where $\mathbf{R} \in \Re^{p \times d}$ is a random matrix drawn from a standard normal distribution while $\mathbf{t} \in \Re^{p}$ is a random vector drawn from a uniform distribution between $[0,\ 2\pi]$. Previous research has shown that the RFF provides an unbiased estimate of the RBF kernel in the original feature space [93]. Thus, instead of applying nonlinear SVR with an RBF kernel, comparable performance can be achieved by training a linear SVR on the RFF.

To illustrate this, Figure 4.2 compares the average runtime and accuracy for the following three approaches: (i) linear SVR trained on the original features, (ii) nonlinear SVR trained on the original features, and (iii) linear SVR trained on the random Fourier features. The experiment was performed on the lake water quality data set with total phosphorous (TP) as the response variable. For linear SVR with RFF, we vary the number of random Fourier features from 1 to 5000, each repeated 10 times. The average 10-fold MSE and average runtime for the different methods are shown in Figure 4.2. The results suggest that the

MSE of SVR decreases with increasing number of RFF, approaching the results of nonlinear SVR. Although its runtime increases with larger number of features, it is still significantly lower than the runtime for nonlinear SVR. This result shows the advantage of using RFF as hash-based features for training a linear SVR with comparable accuracy as nonlinear SVR. Nevertheless, the number of hash-based features needed is still large (in the order of several thousands) to produce an error rate that is comparable to nonlinear SVR.

4.3.3 Matrix Completion

Matrix completion is an approach for recovering missing values by assuming that the original data is a low rank matrix. Let $\mathbf{A} \in \Re^{N \times d}$ be the original data matrix with incomplete entries and $\mathcal{P} : \mathcal{R}^{N \times d} \rightarrow \mathcal{R}^p$ be a linear map that identifies indexes of the non-missing entries in the matrix. The matrix completion approach is often cast into the following optimization problem [115, 56]:

$$\min_{\mathbf{X}} \ \frac{1}{2} \parallel \mathcal{P}(\mathbf{X}) - \mathcal{P}(\mathbf{A}) \parallel_2^2 + \mu \parallel \mathbf{X} \parallel_* \tag{4.2}$$

where $\| \cdot \|_*$ denote the trace-norm of a matrix, which is the sum of its singular values. Intuitively, the preceding objective function seeks to learn a "complete" matrix $\mathbf{X}$ of minimal rank that is consistent with the non-missing entries of the original data $\mathbf{A}$. The regularization parameter μ controls the trade-off between maintaining the consistency of the non-missing entries and minimizing the rank of the recovered matrix.

To demonstrate the effectiveness of this method, we performed an experiment on the lake water quality data set using total chlorophyll-a (chla) as the response variable. We introduced missing values randomly into the data set, varying the percentage of missing values from 20% to 80%. We then applied both mean imputation and matrix completion to the altered data followed by linear and nonlinear SVR. The results shown in Figure 4.3 suggested that matrix completion enabled the missing values to be recovered at a higher precision compared to the

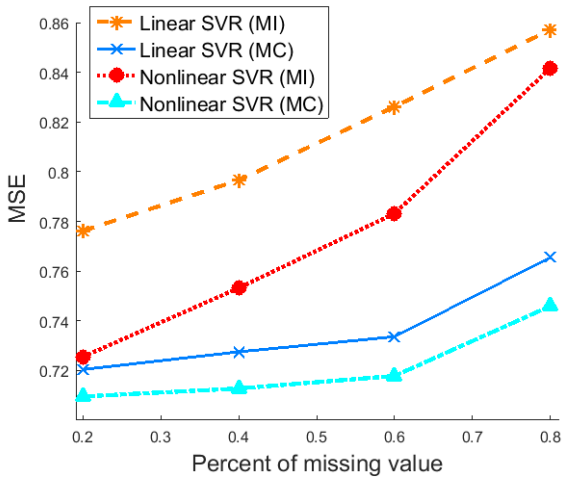


Figure 4.3: Comparing the MSE for linear and nonlinear SVR with mean imputation and matrix completion.

mean imputation method. These results hold true for all percentages of missing values introduced and for both linear and nonlinear SVR.

4.4 Proposed Framework

This section describes the detailed formulation of our proposed hash-based feature learning framework. The unique characteristics of our framework are as follows:

1. It uses a set of sparse random Fourier features as its basis function for creating the hash-based features. The RFF enables the framework to capture nonlinear relationships in the data.

2. It applies supervised learning to identify the best combination of RFF that fits the response variable. Our framework is thus a hybrid method that combines data-independent with data-driven methods.

3. It uses trace-norm regularization to deal with missing values present in the data. Our framework would simultaneously estimate the missing values while learning the hash-based features.

The resulting framework is called **H-FLIP**, which stands for **H**ash-based **F**eature **L**earning for **I**ncom**P**lete data. Although RFF has been used to approximate shift-invariant kernels [93], it has not been used for supervised feature learning. Thus, we first present our supervised hash-based feature learning framework in Section 4.4.1. The framework is extended to incomplete data in Section 4.4.2.

4.4.1 Supervised hash-based feature learning using RFF

Consider a data set $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where each $\mathbf{x}_i \in \Re^d$ denote a set of values for the predictor variables, y_i is the corresponding value for the response variable, and N is the number of observations. Our goal is to learn a set of hash functions $\mathbf{H} = \{h_k\}_{k=1}^K$ where each function $h_k : \Re^d \rightarrow \Re$ transforms the original data in d-dimensional feature space to a 1-dimensional feature space. Instead of using conventional linear hash functions, we employ RFF as our basis function in order to capture nonlinear relationships in the data. Formally, the k^{th} hash function is defined as

$$h_k(\mathbf{x}_i) = \mathbf{w}_k^T \phi_k(\mathbf{x_i}), \tag{4.3}$$

where each basis function $\phi_k(\cdot) : \Re^d \rightarrow \Re^p$ corresponds to a p-dimensional RFF defined in Equation (4.1). The parameters of the hash functions are trained to fit the training data $\mathcal{D}$ using a supervised learning algorithm.

Transforming the original data $\mathcal{D}$ to K RFF requires $\mathcal{O}(NKdp)$ operations, which is expensive when the number of features in the original (d) and projected (p) feature space are large. To reduce the computations, instead of using all d features, each hash function ϕ_k is generated by randomly selecting a subset of the d features and applying the nonlinear transformation given in Equation (4.1) to the selected features. The creation of sparse RFF is illustrated by the following example.

Example 1 *Let $\{x_1, x_2, x_3, x_4\}$ be the set of predictor variables associated with the data*

instance $\mathbf{x}$. *Consider the following RFF,* $\phi(\mathbf{x}) = \cos(\mathbf{R}\mathbf{x} + \mathbf{t})$, *where*

$$\mathbf{R} = \begin{bmatrix} 0.3 & -0.5 & 0 & 0 \\ -0.1 & 0.7 & 0 & 0 \\ 0.2 & 0.6 & 0 & 0 \end{bmatrix}, \ \mathbf{t} = \begin{bmatrix} 1.2 \\ 2.8 \\ 0.66 \end{bmatrix}.$$

This is a sparse RFF as it depends on x_1 *and* x_2 *only.*

The sparse RFF forms the basis function of our supervised hash-based features defined in Equation (4.3). The weights of the hash functions are estimated by optimizing the following objective function:

$$\begin{aligned} J(\mathbf{W}) &= \frac{1}{2}\sum_{k=1}^{K}\sum_{i=1}^{N}(h_k(\mathbf{x}_i) - y_i)^2 + \lambda \sum_{k=1}^{K} \parallel \mathbf{w}_k \parallel_2^2 \\ &= \frac{1}{2}\sum_{k=1}^{K}\left\| \Phi_k(\mathbf{X})\mathbf{w}_k - \mathbf{y} \right\|_2^2 + \lambda \parallel \mathbf{W} \parallel_F^2, \end{aligned} \tag{4.4}$$

where $\mathbf{W} \in \Re^{p \times K} = [\mathbf{w}_1 \ \mathbf{w}_2 \cdots \mathbf{w}_K]$ denote the weight matrix associated with the supervised hash functions and $\Phi_k(\mathbf{X}) \in \Re^{N \times p}$ is the RFF representation of the input data matrix. As the K hash functions can be decoupled from one another in the formulation given in Equation (4.4), the parameter vector $\mathbf{w}_k$ of each hash function can be solved independently as follows.

$$\mathbf{w}_k = \arg\min_{\mathbf{v}} \frac{1}{2} \parallel \Phi_k(\mathbf{X})\mathbf{v} - \mathbf{y} \parallel_2^2 + \lambda \parallel \mathbf{v} \parallel_2^2 \tag{4.5}$$

The closed form solution for $\mathbf{w}_k$ is given by:

$$\mathbf{w}_k = \left[\Phi_k(\mathbf{X})^T\Phi_k(\mathbf{X}) + \lambda\mathbf{I}\right]^{-1}\Phi_k(\mathbf{X})^T\mathbf{y}$$

where $\mathbf{I}$ denotes the identity matrix.

4.4.2 Hash-based feature learning for incomplete data (H-FLIP)

We now extend the previous formulation to data with missing values. Let $\mathbf{A}$ be the incomplete data matrix and $\mathbf{X}$ be the imputed data matrix. Furthermore, let $\mathbf{X} = [\mathbf{X}_l; \mathbf{X}_u]$, where

$\mathbf{X}_l$ is the set of training instances whose response variable values are known and $\mathbf{X}_u$ is the set of test instances whose response variable values are unknown.

`H-FLIP` is designed to simultaneously learn the imputed matrix $\mathbf{X}$ and the weight matrix $\mathbf{W}$ of the sparse RFF by minimizing the following objective function:

$$\min_{\mathbf{W},\mathbf{X}} F(\mathbf{X}, \mathbf{W}) = F_1(\mathbf{X}) + \alpha F_2(\mathbf{X}_l, \mathbf{W}) \tag{4.6}$$

where

$$F_1(\mathbf{X}) = \frac{1}{2} \parallel \mathcal{P}(\mathbf{X}) - \mathcal{P}(\mathbf{A}) \parallel_F^2 + \mu \parallel \mathbf{X} \parallel_*$$
$$F_2(\mathbf{X}, \mathbf{W}) = \frac{1}{2} \sum_{k=1}^{K} \parallel \Phi_k(\mathbf{X}_l)\mathbf{w}_k - \mathbf{y}) \parallel_2^2 + \lambda \parallel \mathbf{W} \parallel_F^2,$$

F_1 measures the error in missing value imputation while F_2 measures the prediction error of using the hash function to fit the response variable y. The regularization parameter α controls the trade-off between minimizing both factors. A trace-norm regularization is applied to ensure that the recovered matrix $\mathbf{X}$ has a low rank. As the missing value imputation must be performed on the entire data set, F_1 involves instances from both the training and test sets while F_2 involves only instances from the training set.

An alternating minimization scheme is employed to solve the objective function given in Equation (4.6). A pseudocode of the `H-FLIP` framework is shown in Algorithm 3. The framework alternates between optimizing for $\mathbf{X}$ and $\mathbf{W}$, until the stopping criteria is satisfied. The optimization steps are outlined below.

4.4.2.1 Updating X

When $\mathbf{W}$ is fixed, the objective function can be simplified as follows:

$$F(\mathbf{X}) \;=\; \frac{1}{2} \parallel \mathcal{P}(\mathbf{X}) - \mathcal{P}(\mathbf{A}) \parallel_F^2 + \frac{\alpha}{2} \sum_{k=1}^{K} \parallel \Phi_k(\mathbf{X}_l)\mathbf{w}_k - \mathbf{y} \parallel_2^2 + \mu \parallel \mathbf{X} \parallel_* \tag{4.7}$$

Algorithm 3 H-FLIP Framework.

Input: $\mathbf{A}, \mathbf{y}$,
Output:$\mathbf{X}, \mathbf{W}$
Initialize $\mathbf{X}^{(0)}$ by solving Equation (4.2).
Generate random matrices $\{\mathbf{R}_k\}$ and $\mathbf{T}$.
Create RFF: $\Phi_k(\mathbf{X}_l) = \sqrt{2/p}\cos(\mathbf{X}_l\mathbf{R}_k^T + \mathbf{T})$
Update $\mathbf{W}$ using Equation (4.11).
while stopping condition is not met **do**
  Initialize $\gamma^{(1)} = \gamma^{(0)} = 1$ and $\mathbf{X}^{(1)} = \mathbf{X}^{(0)}$.
  for $k = 1$ **to** maxIter **do**
    $\mathbf{Y}^{(k)} \leftarrow \mathbf{X}^{(k)} + \frac{\gamma^{(k-1)}-1}{\gamma^{(k)}}\left[\mathbf{X}^{(k)} - \mathbf{X}^{(k-1)}\right]$
    $\mathbf{Z}^{(k)} \leftarrow \mathbf{Y}^{(k)} - \frac{1}{\tau^{(k)}}\nabla f(\mathbf{Y}^{(k)})$
    $[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}^T] \leftarrow \text{SVD } (\mathbf{Z}^{(k)})$
    $\mathbf{X}^{(k+1)} \leftarrow \mathbf{U}\mathbf{D}_{\mu/\tau^{(k)}}(\mathbf{\Sigma})\mathbf{V}^T$
    Compute τ^k using line search.
    $\gamma^{(k+1)} \leftarrow \frac{1+\sqrt{1+4(\gamma^{(k)})^2}}{2}$
  end for
  $\mathbf{X}^{(0)} \leftarrow \mathbf{X}^{(k+1)}$.
  Update $\mathbf{W}$ using Equation (4.11)
end while
return $\mathbf{X}, \mathbf{W}$

Since the trace-norm regularization is a non-smooth function, we separate the objective function into a sum of two functions, $F(\mathbf{X}) = f(\mathbf{X}) + g(\mathbf{X})$, where

$$\begin{aligned} f(\mathbf{X}) &= \frac{1}{2} \parallel \mathcal{P}(\mathbf{X}) - \mathcal{P}(\mathbf{A}) \parallel_F^2 + \frac{\alpha}{2}\sum_{k=1}^{K} \parallel \Phi_k \mathbf{w_k} - \mathbf{y} \parallel_2^2 \\ g(\mathbf{X}) &= \mu \parallel \mathbf{X} \parallel_* \end{aligned}$$

An accelerated proximal gradient descent approach [115] can be used to solve for $\mathbf{X}$ by replacing the smooth part of the objective function with its quadratic approximation, evaluated at some intermediate point $\mathbf{Y}$, i.e.,

$$\begin{aligned} & Q(\mathbf{X}, \mathbf{Y}) \\ = \; & f(\mathbf{Y}) + < \mathbf{X} - \mathbf{Y}, \nabla f(\mathbf{Y}) > + \frac{\tau}{2} \parallel \mathbf{X} - \mathbf{Y} \parallel_F^2 + g(\mathbf{X}) \\ = \; & \frac{\tau}{2} \parallel \mathbf{X} - \left(\mathbf{Y} - \frac{1}{\tau}\nabla f(\mathbf{Y})\right) \parallel_F^2 + g(\mathbf{X}) + K(\mathbf{Y}) \end{aligned}$$

where $K(\mathbf{Y}) = f(\mathbf{Y}) - \frac{1}{2\tau} \parallel \nabla f(\mathbf{Y}) \parallel_F^2$ and

$$\begin{aligned}\nabla f(\mathbf{Y}) &= \mathcal{P}(\mathbf{Y}) - \mathcal{P}(\mathbf{A}) \\ &+ \alpha \sum_{k=1}^{K} \left\{ \left[\sqrt{\frac{2}{p}} \cos(\mathbf{Y}\mathbf{R}_k^T + \mathbf{T})\mathbf{w}_k - \mathbf{y} \right] \mathbf{1}_d^T \right. \\ &\odot \left. \left[-\sqrt{\frac{2}{p}} \sin(\mathbf{Y}\mathbf{R}_k^T + \mathbf{T}) \mathrm{diag}(\mathbf{w}_k)\mathbf{R} \right] \right\} \end{aligned} \tag{4.8}$$

We use $\odot$ in Equation (4.8) to denote the Hadamard product between two matrices and $\mathbf{1_d}$ to denote a d-dimensional vector of all ones. Let $\mathrm{diag}(w_k)$ be a $p \times p$ diagonal matrix, whose k-th diagonal element corresponds to w_k. In addition, $\mathbf{T} = \mathbf{1}_N \mathbf{t}_k^T$ is an $N \times p$ matrix, whose rows correspond to the vector $\mathbf{t}^T$ defined in Equation (4.1). Since the last term $K(\mathbf{Y})$ does not depend on $\mathbf{X}$, $Q(\mathbf{X}, \mathbf{Y})$ can be minimized by solving:

$$\min_{\mathbf{X}} \frac{\tau}{2} \parallel \mathbf{X} - \left(\mathbf{Y} - \frac{1}{\tau} \nabla f(\mathbf{Y}) \right) \parallel_F^2 + \mu \parallel \mathbf{X} \parallel_* \tag{4.9}$$

Let $\mathbf{X}^{(k)}$ denote the most recent estimate after k iterations. The following two steps are performed to update the estimate.

1. Apply accelerated gradient descent method to the smooth part of the objective function.

$$\begin{aligned}\mathbf{Y}^{(k)} &= \mathbf{X}^{(k)} + \frac{\gamma^{(k-1)} - 1}{\gamma^{(k)}} \left[\mathbf{X}^{(k)} - \mathbf{X}^{(k-1)} \right] \\ \mathbf{Z}^{(k)} &= \mathbf{Y}^{(k)} - \frac{1}{\tau^{(k)}} \nabla f(\mathbf{Y}^{(k)}) \end{aligned}$$

   The above equation reduces to the update formula for standard gradient descent when $\forall k : \gamma_k = 1$.

2. Apply singular value shrinkage operator to $\mathbf{Z}^{(k)}$. Let $\mathbf{Z}^{(k)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{\Sigma} = \mathrm{diag}(\sigma_i)$ is a diagonal matrix containing the singular values of $\mathbf{Z}^{(k)}$ while $\mathbf{U}$ and $\mathbf{V}$ are matrices containing its left and right singular vectors. We update $\mathbf{X}$ as follows:

$$\mathbf{X}^{(k+1)} = \mathbf{U}\mathbf{D}_{\mu/\tau^{(k)}}(\mathbf{\Sigma})\mathbf{V}^T \tag{4.10}$$

   where $\mathbf{D}_\nu(\mathbf{\Sigma})$ is a diagonal matrix whose i-th element is $\max(0, \sigma_i - \nu)$. This is equivalent to applying a threshold $\mu/\tau^{(k)}$ to each singular value of $\mathbf{Z}^{(k)}$.

Table 4.2: Summary of the data sets.

Response variable	TP	TN	Chla	Secchi
# instances (lakes)	3694	1961	4834	4684
# features	356	356	356	356
Mean	37.5	821.2	20.9	2.6
Std deviation	68.2	729.6	36.9	1.86

(a) Lake water quality data

Data	Housing	Wine	Parkinson	News	Concrete
# instances	506	4898	5875	5000	1030
# features	14	12	26	61	9

(b) UCI machine learning data

The step size of the gradient descent is determined dynamically using a line search algorithm [115]. The matrix $\mathbf{X}$ is updated until one of the following stopping conditions is satisfied:

(1) the maximum number of iterations is reached,

(2) $\|\mathbf{X}^{(k)} - \mathbf{X}^{(k-1)}\|_F / \|\mathbf{X}^{(k)}\| < \epsilon$,

(3) the objective function given in Equation (4.7) no longer decreases significantly.

4.4.2.2 Updating W

When $\mathbf{X}$ is fixed, we update $\mathbf{W}$ by minimizing the following objective function:

$$\min_{\mathbf{W}} \frac{1}{2} \sum_{k=1}^{K} \| \Phi_k(\mathbf{X}_l)\mathbf{w}_k - \mathbf{y}) \|_2^2 + \lambda \| \mathbf{W} \|_F$$

This is equivalent to solving the objective function for supervised hash-based feature learning as presented in Section 4.4.1. W can be updated using only instances that belong to the training set in the following way:

$$\mathbf{w}_k = (\Phi_k(\mathbf{X}_l)^T \Phi_k(\mathbf{X}_l) + \lambda \mathbf{I})^{-1} \Phi_k(\mathbf{X}_l)^T \mathbf{y} \tag{4.11}$$

4.5 Experimental Evaluation

This section describes the experiments conducted to evaluate the performance of our proposed framework.

4.5.1 Data Sets

We have performed experiments using both synthetic and real-world data sets.

4.5.1.1 Synthetic data

We created a rank-20 data matrix $\mathbf{X}$ containing 5000 instances (rows) and 100 predictor variables (columns) in the following way:

$$\mathbf{X} = \mathbf{PQ} + 0.1\mathbf{E},$$

where $\mathbf{P} \in \Re^{5000\times 20}$, $\mathbf{Q} \in \Re^{20\times 100}$, and $\mathbf{E} \in \Re^{5000\times 100}$. The entries of the matrices $\mathbf{P}$, $\mathbf{Q}$ and $\mathbf{E}$ are generated randomly from a standard normal distribution. Let x_i denote the i-th predictor variable. The value of the response variable y is computed as follows:

$$y = x_1 x_2 + x_{10} x_{11} + x_{12} + \mathcal{N}(0, 0.1).$$

All the columns in $\mathbf{X}$ and the vector $\mathbf{y}$ are standardized to have zero mean and unit variance.

4.5.1.2 Lake water quality data

We used several lake water quality data sets from LAGOS-NE [109], which is a geo-spatial database that contains landscape characterization features and lake water quality data measured at multiple scales covering 17 states in the United States. We used four lake water quality variables—total phosphorus (TP), total nitrogen (TN), total chlorophyll-a (chla) and Secchi depth (Secchi)—as response variables, creating 4 distinct data sets for our experiments. Our goal was to predict the response variables for each lake based on a set of

predictor variables (features) that included land cover, land use, and climate variables. Since there are multiple lake water quality measurements taken at different times for each lake, we computed a single value for each lake by taking the mean of all measurements during the summer months after 2010. The statistics for each data set are shown in Table 4.2.

4.5.1.3 Benchmark data from UCI Machine Learning Repository

We also conducted our experiment on five benchmark data from UCI machine learning repository [68], including `housing` [51], `wine` [27], `Parkinson` [116], `online news` [36], and `concrete strength` [141] (see Table 4.2).

4.5.2 Experimental Setup

We performed our experiments on a Dell PowerEdge R620 server with 2.7GHz Dual Intel Xeon processor. The proposed framework and other baseline methods were written in Matlab.

4.5.2.1 Baseline Methods

First, we compare our proposed supervised hash-based feature learning method for complete data against the following three baseline algorithms: (1) linear SVR trained on the raw features, (2) nonlinear SVR trained on the raw features, (3) linear SVR trained on the random Fourier features.

Second, we compare `H-FLIP`, which is an extension of our supervised hash-based framework to deal with incomplete data, against the following three methods:

- *MC+Raw*: Missing values are imputed during preprocessing using matrix completion. A linear SVR is then trained on the imputed data.
- *MC+PCA*: Missing values are imputed during preprocessing using matrix completion. We then apply principal component analysis (PCA) to extract features from the imputed data.

A linear SVR is then trained to fit the PCA-reduced data.

- *MC+RFF* This is similar to the previous two approaches except we use random Fourier features as feature learning on the imputed data. A linear SVR is then trained on the unsupervised RFF.

For a fair comparison, we extract an equal number of features for all the methods, unless specified otherwise. For example, the number of PCA components, unsupervised RFF, and supervised hash-based features are set to 50. For `H-FLIP`, we first project the data to 200 sparse RFF basis functions before reducing it to the 50-dimensional hash-based features using our supervised learning framework. The regularization parameter λ in `H-FLIP` is determined using cross validation, while the parameter α in Equation (4.6) is set to 0.01. We apply SVR to the features generated by the baseline and proposed methods. As noted in Section 4.3.1, there are two hyper-parameters, ϵ and C, that must be determined when applying SVR. These hyper-parameters are chosen using nested cross validation [119], in which an inner 3-fold cross validation is performed for hyper-parameter tuning and an outer 5-fold cross validation is performed for model assessment.

4.5.2.2 Evaluation Metrics

We consider both the imputation error as well as the prediction accuracy of the induced SVR models. To assess the error in missing value imputation, let $\mathbf{A}_c$ be the true complete data matrix and $\mathbf{X}$ be the estimated (imputed) matrix. The imputation error is computed as follows:

$$\text{Imputation error} = \|\mathcal{P}(\mathbf{X}) - \mathcal{P}(\mathbf{A}_c)\|_2^2 / \|\mathcal{P}(\mathbf{A}_c)\|_2^2.$$

We also evaluate the performance of the SVR models in terms of their mean square prediction error,

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{\mathbf{y}}_i - \mathbf{y}_i)^2,$$

where $\hat{y}_i$ is the predicted response value for the i-th data instance and y_i is its true value.

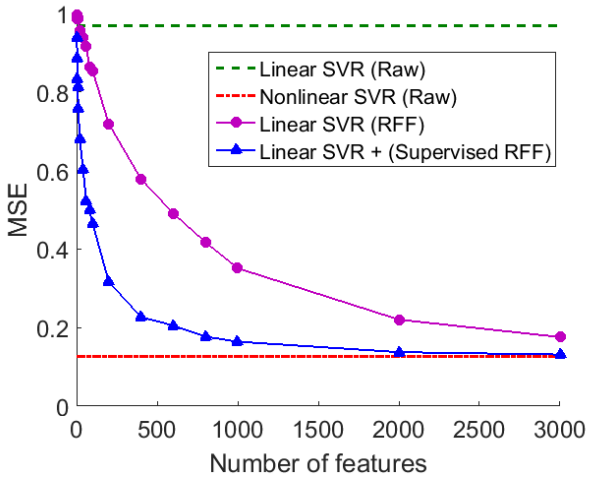


Figure 4.4: Average MSE for linear SVR with raw features, nonlinear SVR with raw features, linear SVR with RFF features, and linear SVR with supervised hash-based features on complete synthetic data.

4.5.3 Experimental Results

This section presents the results of our experiments on both the synthetic and real-world data.

4.5.3.1 Results for Synthetic Data

We begin with the results of our experiments for the complete synthetic data. The proposed framework is compared against linear SVR on raw features, nonlinear SVR (with RBF kernel) on raw features, and linear SVR on unsupervised RFF features. As expected, Figure 4.4 shows that linear SVR on the raw features is worse than other methods as it fails to capture the nonlinear relationships in the data. In addition, the accuracy of linear SVR on both RFF and our supervised hash-based features improves as the number of features increases. More importantly, they are comparable to the accuracy of nonlinear SVR with raw features. This supports the rationale for using RFF to capture nonlinear dependencies in the data. Finally, comparing RFF against the proposed supervised hash-based features, we observe that the supervised approach does not require as many features to achieve high accuracy compared to unsupervised RFF. This justifies the case for using supervised hash-based feature learning

Table 4.3: Imputation error for synthetic data.

Percent of missing value	10%	20%	30%
Mean imputation (MI)	1.0002	1.0002	1.0003
Matrix completion (MC)	0.0271	0.0276	0.0280
H-FLIP	0.0273	0.0280	0.0290

Table 4.4: MSE of linear SVR on synthetic data.

% missing	10%	20%	30%	#features
MC+Raw	0.9680	0.9683	0.9679	100
MC+PCA	0.9681	0.9683	0.9683	50
MC+RFF	0.8960	0.8887	0.8997	50
H-FLIP	**0.4596**	**0.4610**	**0.4626**	50

for nonlinear regression problems.

Next, we add missing values randomly into the synthetic data and compare the imputation error of `H-FLIP` against methods that apply mean imputation and matrix completion during preprocessing. The results in Table 4.3 suggest that mean imputation has the highest imputation error while matrix completion has the lowest error. The imputation error of `H-FLIP` is very close to the imputation error for matrix completion, which is not surprising as `H-FLIP` is designed not only to recover the incomplete data, but also to fit the response variable as accurately as possible. The imputation errors of both matrix completion and `H-FLIP` also do not change significantly as we vary the percent of missing values from 10% to 30%, which shows the robustness of our proposed framework.

In addition to their imputation errors, we also compare the MSE values of their regression models. The results in Table 4.4 show that the raw features and PCA-induced features are worse than unsupervised RFF. `H-FLIP` outperforms all the baseline methods because it learns the appropriate nonlinear features and imputes the missing values without adding significant bias that could degrade the performance of the regression model.

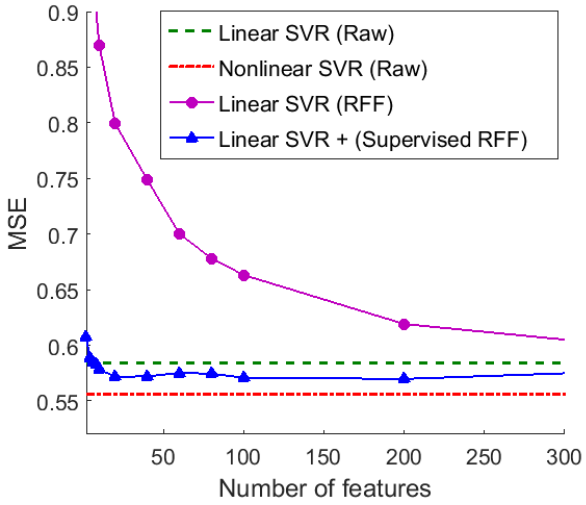


Figure 4.5: Comparing average MSE of linear SVR on the complete Secchi data.

Table 4.5: MSE of linear SVR for lake water quality data.

	TP	TN	Chla	Secchi	#features
MC+Raw	**0.78**	0.73	0.72	0.60	356
MC+PCA	0.81	0.75	0.74	0.63	50
MC+RFF	0.84	0.82	0.81	0.70	50
MC+RFF	0.79	0.74	0.72	0.60	300
H-FLIP	0.79	**0.70**	**0.71**	**0.58**	50

4.5.3.2 Results for Lake Water Quality Data

First, we report the results of applying the various methods to the complete, lake water quality data with no missing values using Secchi as response variable. Figure 4.5 shows that the MSE for linear SVR on the supervised hash-based features is slightly better than linear SVR on the raw features. More importantly, the supervised hash-based features can achieve a low MSE with fewer number of features compared to unsupervised RFF.

We repeat the experiments by adding 20% missing values to the predictor variables and compare the MSE for `H-FLIP` against the baseline methods. The results shown in Table 4.5 suggest that MC+RFF with only 50 features performs the worst, which is consistent with our previous observation that a large number of RFF is needed to effectively represent the data. As we increase the number of RFF from 50 to 300, the MSE improves significantly,

Table 4.6: MSE of linear SVR for UCI data with 20% missing values.

Method	Housing	Wine	Parkinson	News	Concrete
MC+Raw	0.35	0.76	0.80	**0.92**	0.50
MC+PCA	0.36	0.83	0.97	0.94	0.67
MC+RFF	0.38	0.76	0.85	0.94	0.42
H-FLIP	**0.32**	**0.69**	**0.74**	**0.92**	**0.36**

comparable to the results for MC+Raw. Nevertheless, `H-FLIP` with 50 hash-based features outperforms all other methods in 3 of the 4 data sets. In fact, the MSE of linear SVR with `H-FLIP` on the incomplete data is comparable to the results for nonlinear SVR on the complete data (see Table 4.1).

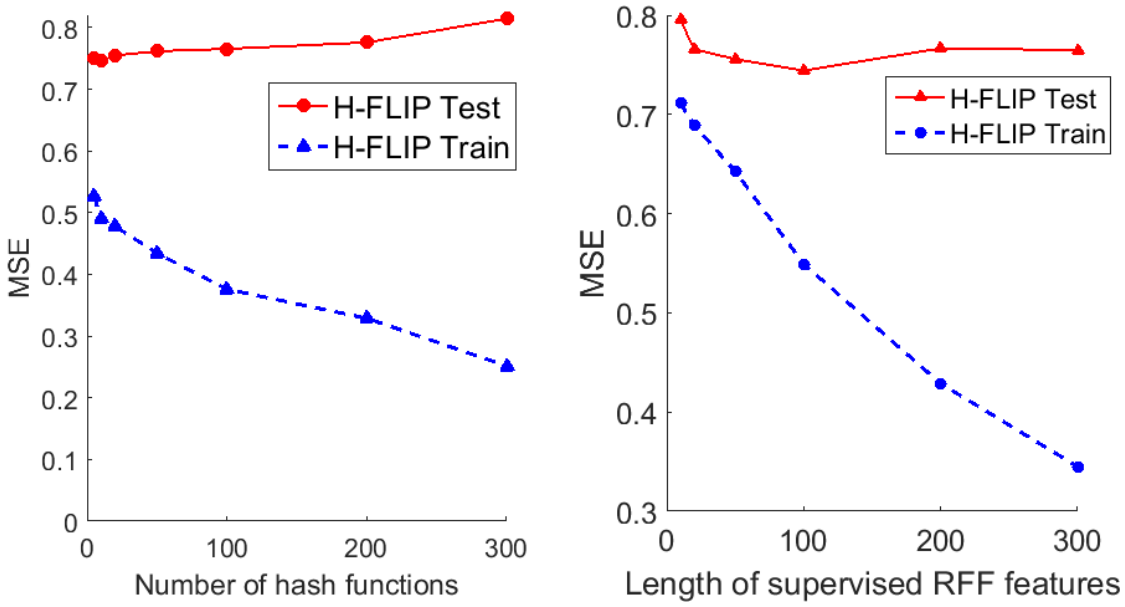
4.5.3.3 Results for UCI Benchmark Data

The results in Table 4.6 show that `H-FLIP` outperforms the baseline methods in 4 of the 5 data sets. The improvement in `H-FLIP` is more significant here compared to the lake data as there are more nonlinear relationships in these data sets. Nonlinear SVR has a lower MSE than linear SVR by more than 0.10 in 3 of the 5 UCI benchmark data but none in the lake data (see Table 4.1).

4.5.4 Sensitivity Analysis

We perform experiments using total nitrogen (TN) as the response variable to evaluate how sensitive the `H-FLIP` results are when varying the number of hash functions (K) and the length of each sparse RFF (p). We first vary the number of hash functions from 5 to 300 while fixing the length of each sparse RFF to be 200. The results shown in Figure 4.6a suggest that the rate of change in the test MSE is quite slow when increasing the number of hash functions compared to the training MSE.

Next, we vary the length of the sparse RFF from 10 to 300 while fixing the number of hash functions to be 50. The results given in Figure 4.6b suggest that the test error of



(a) RFF basis functions (b) Hash-based features

Figure 4.6: MSE of H-FLIP when varying the number of basis functions and supervised hash-based features.

`H-FLIP` is not that sensitive to the increasing length of the RFF compared to its training error.

4.6 Conclusion

This chapter presents `H-FLIP`, a hash-based feature learning framework for incomplete data. Our experimental results show that the framework is particularly effective for data sets that have nonlinear relationships between their predictor and response variables. For future work, we plan to extend the framework to support other hash functions beyond RFF such as spectral hashing [131].

CHAPTER 5

MULTI-TASK LEARNING FOR MULTIPLE RESPONSES AT DIFFERENT LOCATIONS

5.1 Introduction

Predictive modeling of geospatial data has attracted considerable interest due to its wide range of applicability. For example, Brown et al. [16] employed a fusion of two spatial choice models based on logistic regression to predict criminal behaviors. Wimberly et al. [133] used enhanced spatial models to infer the geographical distributions of two tick-borne pathogens. Felicisimo [34] used multiple logistic regression models for the purpose of forested area territorial planning.

One of the challenges in geospatial predictive modeling is dealing with the inherent spatial relationships of the data. Most data mining algorithms implicitly assume that the underlying data are independent and identically distributed. Such an assumption violates the first law of geography [114], which states that: everything is related to everything else but nearby things are more related than distant things. Thus, it would be useful to construct predictive modeling techniques that can explicitly incorporate the spatial relationships of the data as previous studies have suggested that spatial analysis would perform poorly if such relationships are ignored [21].

Another challenge in predictive modeling of geospatial data is that it does not only require performing inference at multiple spatial locations simultaneously, it may also involve predicting multiple, possibly related, response variables. For example, climate scientists are interested in predicting the minimum and maximum temperature as well as precipitation at various locations. These response variables are often correlated with each other. In the field of limnology, researchers are interested in modeling lake nutrients such as total phosphorous and nitrogen in order to monitor the quality of water in freshwater lakes. Since the nutrient

variables are often correlated with one another, the challenge is to build models for the response variables taking into account their joint dependencies.

Sample date	Predictors	Response Variables		
		Chl	TP	Secchi
7/17/2002	Climate data	6.6	27	1.5
6/29/2004		NA	20	NA

Sample date	Predictors	Response Variables		
		Chl	TP	Secchi
6/18/2004	Climate data	NA	8	3.9
6/30/2004		NA	NA	NA
8/10/2008		38.2	NA	3.8

Figure 5.1: Conceptual figure of data with multiple response variable at different locations.

One way to predict the multiple response variables at different locations is to fit a local model for each response variable at each location independently. While this strategy is easy to implement, it has several limitations:

- Some response variables are not always available at certain locations. This makes it difficult to train an accurate local model when there is insufficient training data for a response variable at a given location.
- Such a strategy fails to account for spatial relationships of the data.
- Such a strategy fails to take advantage of the correlation between different response variables.

To overcome these limitations, a multi-task learning framework is proposed for modeling multiple response variables at different locations. As reviewed in Chapter 2.4, multi-task learning (MTL) learns multiple related tasks at the same time so as to improve the model performance. Rather than training the local models independently for each response variable at each location, the proposed formulation simultaneously learns the local models for all response variables by optimizing a joint objective function with trace-norm regularization. The proposed framework also incorporates the spatial relationships between locations as well as the inherent correlation between response variables to improve the model performance.

The main contributions of this chapter are summarized as follows:

- We developed a novel multi-task learning framework for joint prediction of geospatial data that contains multiple response variables at different locations. The proposed framework incorporates spatial autocorrelation among different locations by adding a constraint into the formulation. The proposed framework also takes into account of the inherent correlation between response variables
- With the additional knowledge of the spatial autocorrelation and the correlation among response variables, the proposed framework can be extrapolated to locations with no available training data.
- We demonstrated the effectiveness of the proposed framework on predicting lake water quality data. Our experimental results showed that the proposed framework outperformed other baseline algorithms.

The reminder of the chapter is organized as follows.Section 5.2 defines the multi-response, multi-location prediction problem and Section 5.3 introduces the proposed multi-task learning framework for multiple response variables at different locations for geospatial data. Experimental results are presented in Section 5.4. Finally, Section 5.5 summarizes the findings of this study.

5.2 Multi-Response, Multi-Location Prediction

Consider a geospatial data set $\mathcal{D} = \{\mathbf{X}_i, \mathbf{Y}_i\}_{i=1}^{N}$, where each $\mathbf{X}_i \in \Re^{n_i \times d}$ denote the set of predictors and $\mathbf{Y}_i \in \Re^{n_i \times M}$ denote the corresponding set of response variables for location i. Note that n_i is the number of data instances for location i and d is the number of features. In addition, let M be the number of response variables and N be the number of locations. Our goal is to build a set of models to predict the values of the response variables at each location simultaneously.

Figure 5.2 shows an example application of the multi-response, multi-location prediction problem in the context of ecology data, where the objective is to predict the amount of nutrients in lakes. In this problem, the prediction for each response variable in each lake is considered a separate learning task. The predictor variables correspond to climate information such as monthly temperature and precipitation whereas the response variables include measurements of lake nutrients such as total phosphorous (TP), total nitrogen (TN) and chlorophyll (Chl). Since there are significant spatial autocorrelation in the data and the response variables themselves are highly correlated with each other, this presents an opportunity to develop a multi-task learning framework that can incorporate both types of relationships into the geospatial prediction models.

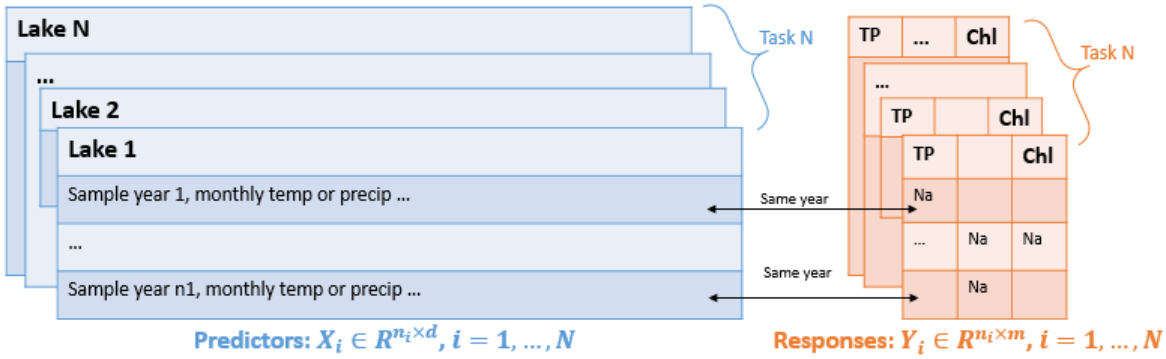


Figure 5.2: An example of predicting lake nutrient data based from climate indexes.

5.3 Proposed Framework

In this section we present the proposed geospatial multi-task learning framework to address the multi-response, multi-location prediction problem.

5.3.1 Objective Function

The objective function for the proposed framework is as follows:

$$F(\mathbf{W}) = \min_{W} \sum_{j=1}^{M} \sum_{i=1}^{N} \| \mathbf{X}_{ij} \mathbf{W}_{\cdot ij} - \mathbf{y}_{ij} \|_F^2 + \Omega(\mathbf{W}) \tag{5.1}$$

where

$$\Omega(W) = \lambda_1 \sum_{j=1}^{M} \text{tr}(\mathbf{W}_{..j}(\mathbf{D} - \mathbf{A}_1)\mathbf{W}_{..j}^T) + \lambda_2 \sum_{i=1}^{N} \text{tr}(\mathbf{W}_{.i.}(\mathbf{D} - \mathbf{A}_2)\mathbf{W}_{.i.}^T) + \lambda_3 \parallel \mathbf{W} \parallel_*$$

In the above formulation, $\mathbf{X}_{ij} \in \Re^{n_{ij} \times d}$ denote the data matrix of predictor variables for i^{th} lake and j^{th} response variable. Similarly, $\mathbf{y}_{ij} \in \Re^{n_{ij} \times 1}$ denote the vector of values for the j^{th} response variable associated with the i^{th} lake. Let $\mathbf{W}$ be a tensor, whose element W_{kij} is the model parameter associated with the k^{th} feature, i^{th} lake, and j^{th} response variable. The matrix $\mathbf{A_1} \in \Re^{N \times N}$ captures the spatial relationship between locations whereas $\mathbf{A_2} \in \Re^{M \times M}$ captures the correlation between response variables.

In this formulation, the prediction for each response variable at each location is a separate learning task. The first term in Equation (5.1) represents the total residual error of the models over all M response variables and N locations. The regularization term, $\Omega(W)$, consists of 3 parts. The first part incorporates the spatial relationship into the model. Specifically, if two locations have a strong, positive spatial relationship, then their model parameters should also be quite similar. The second part of regularization term takes into account the correlation between response variables. If there is a strong, positive correlation between a pair of response variables, then the model parameters associated with the response variables should be highly similar to each other. The last regularization term is used to control the model complexity. The hyperparameters λ_1, λ_2, and λ_3 are used to balance the tradeoff between each regularization term and the rest of the terms in the objective function.

5.3.2 Parameter Estimation

The proposed objective function is convex, and thus, has a global minimum. It can be solved by using the accelerated proximal gradient descent method [56].

Since the nuclear norm regularization is a non-smooth function, we separate the objective

function in Equation (5.1) into a sum of two functions: $F(\mathbf{W}) = f(\mathbf{W}) + g(\mathbf{W})$, where

$$\begin{aligned} f(\mathbf{W}) &= f_1(\mathbf{W}) + f_2(\mathbf{W}) + f_3(\mathbf{W}) \\ g(\mathbf{W}) &= \mu \parallel \mathbf{W} \parallel_* \end{aligned} \tag{5.2}$$

The smooth function $f(\mathbf{W})$ in Equation (5.3) is given by:

$$\begin{aligned} f_1(\mathbf{W}) &= \sum_{j=1}^{M} \sum_{i=1}^{N} \parallel \mathbf{X}_{ij}\mathbf{W}_{\cdot ij} - \mathbf{y}_{ij} \parallel_F^2 \\ f_2(\mathbf{W}) &= \lambda_1 \sum_{j=1}^{M} \mathrm{tr}(\mathbf{W}_{\cdot\cdot j}(\mathbf{D} - \mathbf{A}_1)\mathbf{W}_{\cdot\cdot j}^T) \\ f_3(\mathbf{W}) &= \lambda_2 \sum_{i=1}^{N} \mathrm{tr}(\mathbf{W}_{\cdot i\cdot}(\mathbf{D} - \mathbf{A}_2)\mathbf{W}_{\cdot i\cdot}^T) \end{aligned}$$

Using the proximal gradient decent algorithm, $\mathbf{W}$ is iteratively updated by solving the following problem:

$$\mathbf{W}^{(k)} = prox_{t_k}(\mathbf{W}^{(k-1)} - t_k \nabla f(\mathbf{W}^{(k-1)})), \tag{5.3}$$

where $f(\mathbf{W})$ is the smooth part of the objective function given in Equation (5.3). Here, $prox_t(\mathbf{X})$ is a soft thresholding function on X defined as follows:

$$prox_t(\mathbf{X}) = \mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^T, \tag{5.4}$$

where $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is a singular value decomposition, and $(\mathbf{\Sigma}_t)_{ii} = \max(\mathbf{\Sigma}_{ii} - t, 0)$ The gradient for $f(\mathbf{W})$ is given by the following three expressions:

$$\begin{aligned} \frac{\partial f_1}{\partial \mathbf{W}_{\cdot ij}} &= 2\mathbf{X}_{ij}^T(\mathbf{X}_{ij}\mathbf{W}_{\cdot ij} - \mathbf{y}_{ij}) \\ \frac{\partial f_2}{\partial \mathbf{W}_{\cdot\cdot j}} &= 2\lambda_1 \mathbf{W}_{\cdot\cdot j}(\mathbf{D} - \mathbf{A}_1) \\ \frac{\partial f_3}{\partial \mathbf{W}_{\cdot i\cdot}} &= 2\lambda_2 \mathbf{W}_{\cdot i\cdot}(\mathbf{D} - \mathbf{A}_2) \end{aligned}$$

which can be plugged into Equation (5.3) to compute the new $\mathbf{W}^{(k)}$. The pseudocode of the algorithm is shown in Algorithm 4.

Algorithm 4 Multi-Response Multi-Task learning (MRMT) framework.

Input: $\mathbf{X}, \mathbf{Y}, \mathbf{A_1}, \mathbf{A_2}, \lambda_1, \lambda_2, \lambda_3$
Output: $\mathbf{W}$
Initialization: $k = 0$, $\mathbf{W}^{(0)}$
repeat
  $k = k + 1$
  Update $\mathbf{W}^{(k)} = prox_{t_k}(\mathbf{W}^{(k-1)} - t_k \nabla f(\mathbf{W}^{(k-1)}))$.
until convergence
return W

5.4 Experimental Evaluation

This section presents the experiments performed to evaluate the performance of the proposed multi-task learning framework for predicting multiple response variables at different locations.

5.4.1 Datasets

We use a lake water quality dataset extracted from LAGOS-NE, which is the geospatial database described in Section 1.2. Similar as in the previous chapter, we select four lake water quality features as our response variables, i.e., TP, TN, Chla and Secchi. We use a comprehensive set of climate variables as our predictors, including temperature and precipitation as well as ENSO and NAO climate indices. There are about 12,000 lakes in the dataset. However, each lake contains different number of measurements for each of the four response variables. As a result, the number of data instances may vary for different response variables in the same lake as well as in different lakes. A summary statistics of the dataset is shown in Table 5.1. Furthermore, since Secchi depth was found to be negatively correlated to other response variables, we use the negative value of Secchi depth in our analysis.

5.4.2 Experimental Setup

In this section, we describe the experimental design as well as the baseline methods and evaluation metrics used to compare the performance of the different methods. We first

Table 5.1: Statistics of lake water quality data.

	TP	Chla	TN	Secchi
# instance	33713	35820	70218	8828
# lakes	8245	7393	10247	2398
# features	48	48	48	48
Mean	33.15	17.18	3.06	18.83
Max	1122.50	696.00	18.40	19691.00
Min	0.00	0.00	0.00	0.00
Median	15.50	5.60	2.74	541.00
Avg years per lake	4	5	7	4
# lakes with at least 2 years	4424	5383	7065	1315
# lakes with at least 4 years	2568	2978	4897	687

randomly partition the dataset into 2/3 for training and 1/3 for testing. The training set is then randomly split into two halves, using one half for training and the other half as validation set for tuning the hyperparameters λ_1, λ_2, and λ_3 of our model. We select the hyperparameters that achieve the lowest mean RMSE on the validation set. We then repeat the experiment 10 times and report the results based on the average performance over the 10 repeated trials.

The proposed multi-task, multi-response learning method requires additional side information in the form of the spatial relationship between locations and correlation between the response variables. For the lake water quality dataset, we estimate the spatial relationship matrix $\mathbf{A_1}$ by applying the RBF kernel function on the latitude and longitude of the lakes and the similarity matrix $\mathbf{A_2}$ by computing the correlation between the response variables in the training set.

5.4.2.1 Baseline Methods

To demonstrate its effectiveness, we compare the performance of the proposed multi-task multi-response learning algorithm (denoted as MTMR) against the following methods:

- *Global*: For each response variable, a global model is constructed by combining the data from all the lakes. The global model is then applied to all the lakes to predict their

corresponding value of the response variable.

- *MTL-Lasso*: Unlike the previous global model, this method trains a local model at each location for each response variable. The objective function is as follows:

$$\min_{W} \sum_{i=1}^{t} \parallel W_i^T X_i - Y_i \parallel_F^2 + \rho \parallel W \parallel_1$$

where i is the location index and W_i, X_i, Y_i are the corresponding model weights, predictors and response variables. In this formulation, instead of training the local models independently, the task relationship is achieved by enforcing the same regularization penalty on sparsity of the model. The method is based on an implementation provided by the MALSAR [150] package.

- *MTL-L21*: Similar to the previous method, a local model is trained at each location for each response variable. The difference is in terms of how the task relatedness is defined. The objective function is shown below [2]:

$$\min_{W} \sum_{i=1}^{t} \parallel W_i^T X_i - Y_i \parallel_F^2 + \rho \parallel W \parallel_{2,1}$$

In this learning scheme, the model for different tasks are forced to select the same set of features.

5.4.2.2 Evaluation Metric

We used two metrics to evaluate the performance of the different methods. The first metric is root mean square error (RMSE), which measures the deviation between the observed and predicted values of the response variable. The metric is defined as:

$$RMSE = \sqrt{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2 / N} \tag{5.5}$$

The second metric is the coefficient of determination, R^2, which measures the amount of variations in the response variable explained by the model. The metric is defined as follows:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{5.6}$$

Table 5.2: RMSE on lake water quality data.

	TP	Chla	TN	Secchi
Global	0.928 ± 0.019	0.951 ± 0.016	0.894 ± 0.025	0.841 ± 0.003
MTL-Lasso	0.805 ± 0.022	0.867 ± 0.015	0.609 ± 0.036	0.620 ± 0.003
MTL-L21	0.805 ± 0.022	0.867 ± 0.015	0.572 ± 0.047	0.569 ± 0.002
MTMR	**0.757 ± 0.022**	**0.811 ± 0.016**	**0.557 ± 0.032**	**0.548 ± 0.002**

Table 5.3: Predictive R^2 on lake water quality data.

	TP	Chla	TN	Secchi
Global	0.126 ± 0.004	0.122 ± 0.005	0.155 ± 0.006	0.281 ± 0.002
MTL-Lasso	0.341 ± 0.020	0.270 ± 0.015	0.607 ± 0.041	0.609 ± 0.003
MTL-L21	0.342 ± 0.020	0.270 ± 0.015	0.652 ± 0.051	0.670 ± 0.002
MTMR	**0.417 ± 0.017**	**0.362 ± 0.014**	**0.671 ± 0.031**	**0.695 ± 0.002**

where $\bar{y}$ is the mean of the observed values for the response variable. Unlike RMSE, which goes from 0 to ∞, the R^2 value ranges between 0 to 1. For comparison, we report the average RMSE and predictive R^2 value for the test sets used in our experiments.

5.4.3 Experimental Results

In this section, we report the experimental results obtained for MTMR as well as other baseline methods.

5.4.3.1 Performance Comparison

As mentioned earlier, we repeated our experiment 10 times with different training and test set partitions and report the mean and standard deviation values of RMSE and R^2. The results are shown in Tables 5.2 and 5.3, respectively.

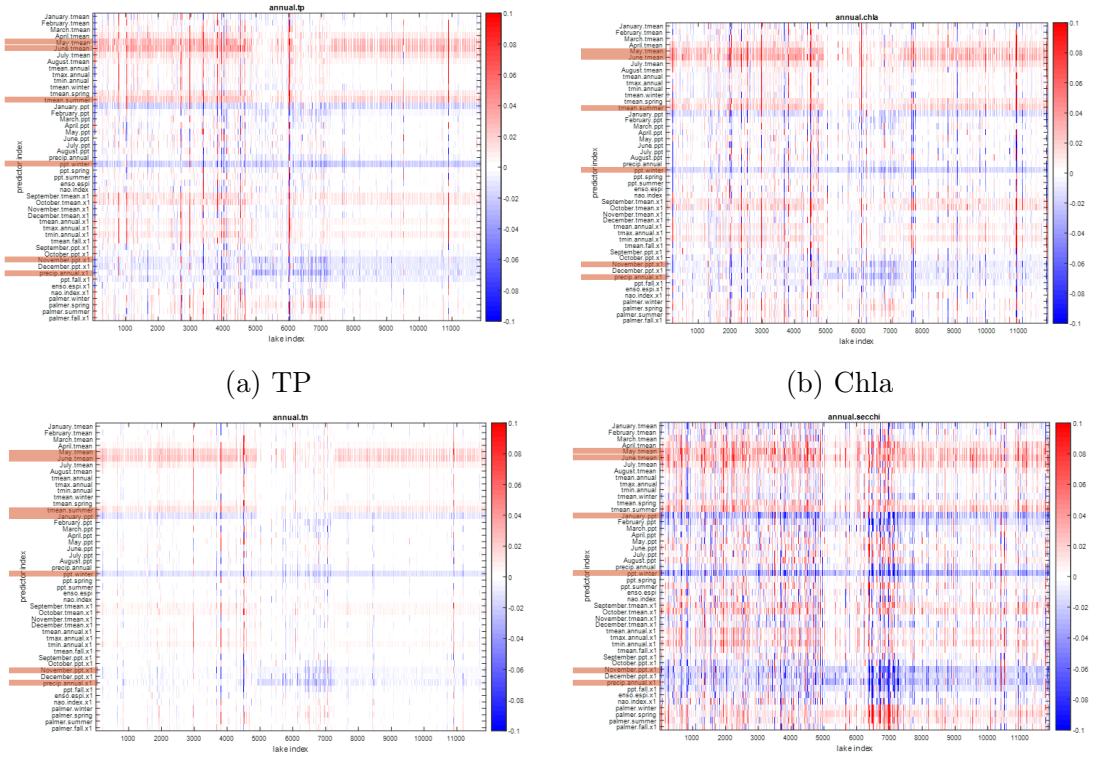
The results suggest that a single, global model for all the lakes performs worst compared to other methods for all four response variables. This is because such a model does not sufficiently capture the lake variations as well as spatial relationships among lakes. MTL-Lasso is better than global model because in this formulation, instead of building one global model, it builds a local model for each lake and uses the lasso regularization constraint to ensure the

models share the same sparsity. In addition, MTL-L21 performs slightly better than MTL-lasso in two of the four response variables. As MTL-L21 used $L_{2,1}$ norm regularization, which assumes that all the tasks share the same set of discriminative features. Compared to Lasso regularization, MTL-L21 results in grouped sparsity. Our proposed MTMR algorithm performs the best for all four responses. This is because it employs a regularization term consisting of three different parts, as shown in Equation (5.1). The first part incorporates spatial relationships into the formulation, the second part takes advantage of the correlation between response variables while the last part controls the model complexity. All three work together in tandem to achieve better predictive performance. The results for R^2 given in Table 5.3 are consistent with the RMSE results, which shows the superiority of our methods compared to other baseline methods.

5.4.3.2 Model Coefficients

We further examine the coefficients of our proposed MTMR model for each response variable in Figure 5.3. The horizontal axis corresponds to each of the lakes we have in the data set while the vertical axis corresponds to the list of predictor variables. Notice that, although we may not have data for certain response variables, we still have a model coefficient associated with the location. A red cell indicates that the model coefficient is highly positive whereas a blue cell indicates the coefficient is highly negative. The cell is almost white in color if the model coefficient is close to zero, meaning the corresponding feature is not important for inferring the value of the response variable at the given location.

Another interesting observation is that there is a consistent pattern when examining all four plots shown in Figure 5.3 even though the proposed framework does not impose an L_{21} regularization. In other words, the set of features with relatively highest positive or negative weights are quite similar for most of the lakes and response variables.



(a) TP (b) Chla

(c) TN (d) Secchi

Figure 5.3: Model weights visualization.

5.5 Conclusion

In this chapter, we proposed a multi-task learning framework for predicting geospatial data with multiple response variables at different locations. The proposed framework incorporates spatial relationship between the locations and takes into account the correlation between different response variables. The framework also allows predictions to be made for locations with no observed data. We demonstrated the effectiveness of our proposed framework on four lake water quality data sets. The results suggest that our proposed framework performs better than other baseline algorithms in all four data sets evaluated in this study.

CHAPTER 6

MULTI-LEVEL MULTI-TASK LEARNING FOR NESTED GEOSPATIAL DATA

6.1 Introduction

The geospatial data available in many applications often contain variables defined at multiple spatial scales. These variables from different scales can interact with each other, a phenomenon also known as cross-scale interactions in the literature. For example, previous studies in lake ecology found strong evidence for cross-scale interactions between geospatial driver variables quantified at local and regional spatial scales for predicting lake nutrients [110]. More formally, cross-scale interactions (CSIs) [91] refer to the coupling between the local and regional variables and their joint effect on the focal response variable. For example, interactions between local wetland cover around a lake and regional agricultural land use have been shown to affect the performance of models predicting lake water total phospho-

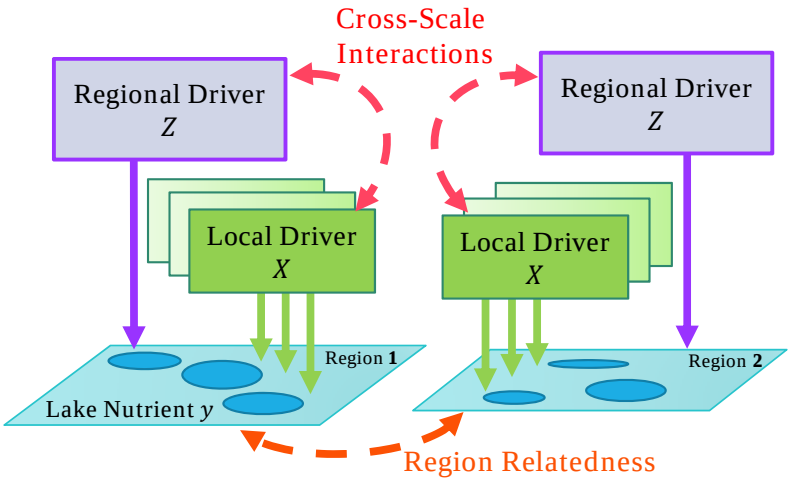


Figure 6.1: Example of nested lake ecology data with cross-scale interactions between the local and regional predictor variables.

rus concentrations [35]. Another example is the cross-scale interaction between broad-scale hurricane-induced disturbance and fine-scale historical land use, which influences the biodiversity of land snails [132]. Nested geospatial data, containing variables measured at multiple spatial scales, are needed to detect such patterns. For example, in the modeling of lake nutrients, the predictor variables may include local drivers such as lake depth, lake type, and amount of wetland areas surrounding the lake as well as regional drivers such as climate and land use/cover (see Fig. 6.1). In this example, the values of the local predictor variables would vary from one lake to another but the values of the regional predictor variables are the same for all lakes within the same region. The nature of such nested data makes it challenging to effectively incorporate both local and regional variables into the model formulation. On one hand, the local and regional variables can be concatenated to form a multi-scale feature vector from which a global regression model can be fitted against the data for all regions. Unfortunately, such a strategy may not be effective since the relationship between the predictor and response variables may vary across regions, making it difficult to construct an accurate, one-size-fits-all model for all the regions. On the other hand, a local regression model can be trained to fit the data in each region separately. However, such a model would ignore the regional variables altogether as their values would be identical for all lakes in the same region. In addition, the model may not be able to capture the cross-scale interactions present in the data.

Another challenge in the predictive modeling of nested geospatial data is the unbalanced sample sizes across different regions. If the underlying relationships in the geospatial domain are complex, the predictive models developed for data-poor regions are likely to be inferior compared to those developed for data-rich regions. Thus, a key research question is how to effectively leverage data from other regions to improve the prediction models for the data-poor regions.

Finally, the original set of regions from which the nested data were obtained may not be ideal for predictive modeling as they were often defined for other purposes (e.g., based on

political boundaries, management policies, etc.). Indeed, it is possible that the lakes from the same region may not share the same relationship between their predictor and response variables. It would be useful to develop a modeling framework that can infer a set of regions that better capture the relationship between the predictor and response variables of the data.

To address the above challenges, this chapter presents a novel multi-level multi-task learning framework for the predictive modeling of nested geospatial data from the ecology domain. The framework enables a distinct prediction model to be trained for each region using both its local and regional predictor variables. The framework assumes that the nested geospatial data are characterized by a set of low-rank latent factors, which relates the dependencies between the local and regional predictor variables to the response variable. Instead of building the models for each region independently, the models are jointly trained by inferring their local and regional latent factors. The shared latent factors provide several advantages for our framework. First, they enable the data-poor regions to leverage information from other regions in order to construct more robust models. Second, the latent factors can be used to identify cross-scale interactions in each region. Finally, they provide a new feature representation for each lake, which allows us to cluster the lakes into a new set of regions based on the similarity of their local latent factors. Empirical results using four lake water quality datasets from the LAGOS-NE_{NE} database [109] showed significant performance improvement in the local prediction models when trained on the new set of regions instead of the original, pre-defined regions of the data.

In summary, the main contributions of this chapter are as follows:

1. We introduced a multi-level multi-task learning framework for nested geospatial data. The framework can incorporate both the local and regional predictor variables into a unified formulation

2. We showed that the framework can be used to identify cross-scale interactions among the local and regional predictor variables. The framework also allows more robust models to

be constructed especially for regions with limited or no training data.

3. We derived a new set of regions from the latent factors and showed that they are more aligned with the response variables of interest compared to the original regions of the lake ecology data.

4. We demonstrated the effectiveness of the proposed framework on four lake water quality datasets. Our experimental results showed that the proposed framework outperformed four other baseline methods in more than 64% of the regions in 3 out of 4 lake water quality datasets evaluated in this study.

The reminder of the chapter is organized as follows: Section 6.2 reviews previous work on multi-level modeling and multi-task learning. Section 6.3 formulates the nested geospatial predictive modeling problem while Section 6.4 introduces the proposed multi-level multi-task learning framework. Experimental results are presented in Section 6.5. Section 6.6 summarizes the findings of this study.

6.2 Related Work

Geospatial data of many types, including ecological, climatalogical, epidemiological, and social data [123, 65] are inherently nested, containing grouped variables defined at multiple, hierarchically-ordered spatial scales. For predictive modeling applications, the response (target) variable of interest is usually defined at the finest spatial scale, while the predictor variables are available at both finer and coarser scales. The complexity introduced by the nested geospatial data has led to growing interests in applying multi-level modeling techniques [107, 35, 124, 110] that can capture the influence of the fine-level and coarse-level predictors on the response variable along with their cross-scale interactions. Multi-level statistical models have been proposed as a powerful analytical approach that accounts for dependencies among observations within grouping levels [38], and thus, can help accommodate the assumption of independence that, if violated, can reduce the effective sample size

and lead to exclusion of relevant predictors in the model. In addition, this modeling approach can address contradictory relationships, where relationships in one area may differ in magnitude or direction from relationships in another area [35]. Such variation in slope estimates often emerge as a result of spatial heterogeneity in drivers of ecological response variables [43]. Multi-level models are well-suited to macroscale studies where it is expected that both fine- and broad-scale variables influence system structures and functions[9] and cross-scale interactions may be present but are challenging to quantify[122]. For example, Filstrup et al. [37] used a 4-parameter multi-level logistic model to describe regional relationships between lake nutrients and algal productivity and found that regional land use and land cover mediated the nutrient-productivity relationships across space.

Multi-task learning (MTL) [19] provides an alternative approach that can be used to model nested geospatial data by considering the prediction problem for each region as a separate learning task. Instead of training the model for each task (region) independently, MTL learns the models for all tasks (regions) simultaneously, taking into account the tasks' relationships. Over the past two decades, numerous MTL algorithms have been proposed in the literature. These algorithms vary in terms of how the task relatedness are defined and incorporated into the learning formulation. For example, a standard assumption is that the model parameters for closely related tasks are similar to each other, which led to the development of the mean regularized MTL approach by Evgeniou and Pontil [33]. Another common assumption is that the model parameters for different tasks share a low-rank representation. MTL algorithms that employ such a strategy include the works in [23, 2, 63].

More recently, there has been considerable interest in applying MTL to spatial, temporal, and spatio-temporal prediction problems as many of these problems can be naturally cast into a multi-task learning formulation [41, 72, 137, 138, 139]. For example, Xu et al. [137] presented an MTL framework for ensemble time series forecasting problems. The application of the MTL framework to spatio-temporal data has also been studied in [138, 139, 72] for var-

ious applications including climate modeling and urban water quality prediction. However, none of these approaches consider the nested structure of the data, and thus, are unable to handle cross-scale interactions present in the data. Another related work is the multi-level lasso approach proposed by Lozano et al.[74]. However, the approach is not designed for geospatial data. It also assumes that the second-level variables are unobserved, unlike the formulation presented in our study, which assumes that the coarser-level (regional) variables are observed. Despite the difference, we use a variant of this multi-level lasso formulation, assuming the second-level variables are observed, as one of the baseline methods for comparing our proposed framework.

6.3 Preliminaries

We begin by considering a two-level nested geospatial data, $\mathcal{D} = \{\mathbf{X}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^{r}$, where r is the number of regions. Let $\mathbf{X}_i \in \mathbb{R}^{n_i \times d}$ be the design matrix containing the local predictor variables for all the geospatial objects in region i, $\mathbf{y}_i \in \mathbb{R}^{n_i}$ be their corresponding values of response variables, and $\mathbf{z}_i \in \mathbb{R}^k$ be their corresponding values of the regional predictor variables. Here, d is the number of local predictors, k is the number of regional predictors, and n_i is the number of geospatial objects (e.g., lakes) in region i. Note that our proposed framework can be extended to more than 2 levels (see Section 6.4.5).

The goal of geospatial predictive modeling is to learn a target function $f(\mathbf{x}, \mathbf{z})$ that maps the local and regional predictor variables of a geospatial object ($\mathbf{x} \in \mathbb{R}^d, \mathbf{z} \in \mathbb{R}^k$) to its response value, y, with minimal prediction error. In this chapter, we consider only linear models, though the approach can be extended to nonlinear models using the well-known kernel trick [118] or random Fourier features [93] approaches. We assume that the local and regional variables are augmented with dummy variables, whose values are set to 1. This allows us to simplify the notation for a linear model of the form $f(\mathbf{x}) = \sum_{j=1}^{d-1} w_j x_j + w_0$ to $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$, where $\mathbf{w} \in \mathbb{R}^d$.

A trivial way to model nested geospatial data is by fitting a single, global model f_{global}

to the entire data set $\mathcal{D}$. Unfortunately, the global model may not provide a good fit to the data especially if the relationship between the predictor and response variables may vary by region. Alternatively, one could train an independent, local model f_{local} for each region, but this approach is also not as effective especially for regions that have very few training examples available. Furthermore, the local models will not be able to utilize the regional variables since their values are the same for all the training examples in the same region. Alternative techniques are therefore needed for modeling nested geospatial data.

Multi-level modeling [107] is a widely-used statistical technique for assessing the influence of multi-scale variables on the response variable of interest. For a two-level model, the relationship between the response and predictor variables for a geospatial object $(\mathbf{x}_i, \mathbf{z}_i)$ in region i is given as follows:

$$\begin{aligned} y_i &= \mathbf{w}_i^T \mathbf{x}_i + \epsilon_1, \qquad \epsilon_1 \sim \mathbb{N}(0, \sigma_1^2) \\ \mathbf{w}_i &= \mathbf{G}^T \mathbf{z}_i + \boldsymbol{\epsilon}_2, \qquad \boldsymbol{\epsilon}_2 \sim \mathbb{N}(\mathbf{0}, \boldsymbol{\Sigma}_\mathbf{2}), \end{aligned} \tag{6.1}$$

where $\mathbf{G} \in \mathbb{R}^{k \times d}$ is a matrix that captures the cross-scale interactions between the local and regional predictors. Specifically, the (i, j)-th element of $\mathbf{G}$ corresponds to the cross-scale interaction term between the i-th regional predictor and the j-th local predictor. It can be shown that the maximum likelihood estimation (MLE) of $\mathbf{G}$ can be found by minimizing the following loss function: $L(\mathbf{G}) = \sum_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i \mathbf{G}^T \mathbf{z}_i \parallel_2^2$. Several variants of the formulation have also been proposed in the literature. For example, Zhao et al. [149] presented a multi-level modeling approach for hierarchical multi-source event forecasting. Lozano et al. [74] presented the following multi-level lasso formulation for multi-task regression:

$$\min_{\mathbf{G}} \ \frac{1}{2} \sum_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i \mathbf{G}^T \mathbf{z}_i \parallel_2^2 + \rho_1 \parallel \mathbf{G} \parallel_1 \tag{6.2}$$

Since the $\{\mathbf{z}_i\}$ is given, the optimization problem can be solved by using the proximal gradient descent method. During the prediction step, a test instance $(\mathbf{x}^*, \mathbf{z}^*)$ can be predicted as follows:

$$\hat{y} = \mathbf{z}^{*T} \mathbf{G} \mathbf{x}^* = G_{11} + \sum_{p=2}^{d} G_{1p} x_p^* + \sum_{q=2}^{k} G_{q1} z_q^* + \sum_{p,q>1} z_q^* G_{qp} x_p^* \tag{6.3}$$

In the preceding equation, $x_1^* = z_1^* = 1$ and G_{11} is the intercept term of the model. The second term measures the effect of the local predictors on the response variable whereas the third term measures the effect of the regional predictors. The last term of the equation quantifies the influence due to joint coupling of the local and regional predictors on the response variable y. A non-zero value in G_{qp}, where $p > 1$ and $q > 1$, can thus be regarded as evidence for cross-scale interaction between the p-th local and q-th regional predictors.

6.4 Multi-Level Multi-Task Learning (MLMT) Framework

This section presents our proposed multi-level multi-task learning framework for modeling cross-scale interactions in nested geospatial data. We first present the objective function of our framework in Section 6.4.1. Section 6.4.2 describes the optimization algorithm used to estimate the model parameters while Section 6.4.3 provides a proof of convergence of the algorithm. The description on how to identify cross-scale interactions from the proposed framework is given in Section 6.4.4. Finally, Section 6.4.5 generalizes the framework to an N-level setting, where $N > 2$.

6.4.1 Objective Function

The traditional multi-level model formulation shown in Equation (6.1) restricts the regression coefficients for all the regions to lie in the column space of $\mathbf{G}^T$, i.e., each w_i is a linear combination of the column vectors in $\mathbf{G}^T$ with the weights of the linear combination given by the regional variables $\mathbf{z}_i$'s. In contrast, our proposed formulation assumes that the regression coefficients for all the regions share a common set of latent factors. Specifically, each $\mathbf{w}_i$ is decomposed into a product of two terms: a latent factor matrix $\mathbf{U} \in \mathbb{R}^{d \times m}$ that is shared by all the regions and a vector $\mathbf{v}_i \in \mathbb{R}^m$ that is shared by all the geospatial objects in region i, where d is the number of local predictors and m is the number of latent factors. Instead of regressing $\mathbf{w}_i$ directly against the regional variables $\mathbf{z}_i$, we regress the latent factor $\mathbf{v}_i$

against $\mathbf{z}_i$, which leads to the following optimization problem:

$$\begin{aligned}\min_{\mathbf{U},\mathbf{V},\mathbf{R}} \quad & \frac{1}{2}\sum_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i\mathbf{U}\mathbf{v}_i \parallel_2^2 + \frac{\rho_1}{2}\sum_{i=1}^{r} \parallel \mathbf{z}_i - \mathbf{R}\mathbf{v}_i \parallel_2^2 \\ & + \rho_2 \parallel \mathbf{U} \parallel_1 + \rho_3 \parallel \mathbf{V} \parallel_1 + \rho_4 \parallel \mathbf{R} \parallel_1, \end{aligned} \tag{6.4}$$

where $\mathbf{V} = [\mathbf{v}_1\mathbf{v}_2 \cdots \mathbf{v}_r]$ and r is the number of regions. The first term in Equation (6.4) corresponds to the squared loss prediction error of the model while the second term corresponds to the error in fitting the regional predictors $\mathbf{Z}$ to $\mathbf{V}$. The last 3 terms of the objective function controls the sparsity of the model by enforcing L1-regularization to the latent factors $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{R}$. Finally, ρ_1, ρ_2, ρ_3, and ρ_4 are the user-specified parameters. In this formulation, $\mathbf{U}$ represents the latent factors for the local predictors while $\mathbf{R}$ represents the latent factors for the regional predictors. Furthermore, the feature representation for each region i in the m-dimensional latent space is given by $\mathbf{v}_i$ whereas the feature representation for each lake $\mathbf{x}$ in the latent space is given by $\mathbf{U}^T\mathbf{x}$.

6.4.2 Parameter Estimation

We employ the block coordinate descent approach to minimize the objective function given in Equation (6.4). Since there are three latent factors ($\mathbf{U}$, $\mathbf{V}$, and $\mathbf{R}$) to be estimated, the algorithm iteratively estimates one of the three latent factors while keeping the other two latent factors fixed. The update formula for each latent factor is given below.

Update formula for V Assuming $\mathbf{U}$ and $\mathbf{R}$ are given, the optimization for $\mathbf{V}$ is obtained by minimizing the following objective function:

$$\mathcal{L}(\mathbf{V}) \;=\; \tfrac{1}{2}\textstyle\sum_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i\mathbf{U}\mathbf{V}_i \parallel_2^2 + \frac{\rho_1}{2} \parallel \mathbf{Z}^T - \mathbf{R}\mathbf{V} \parallel_F^2 + \rho_3 \parallel \mathbf{V} \parallel_1 \tag{6.5}$$

Since $\mathcal{L}(\mathbf{V})$ is not a smooth function, we solve the optimization problem using the proximal gradient descent algorithm. Specifically, $\mathbf{V}$ is iteratively updated by solving the following problem:

$$\mathbf{V}^{(s)} = prox_{\lambda}(\mathbf{V}^{(s-1)} - \lambda\nabla g(\mathbf{V}^{(s-1)})), \tag{6.6}$$

where $g(\mathbf{V})$ is the smooth part of the objective function given in Equation (6.5) and $prox_\lambda(x)$ is a soft thresholding function on x defined as follows:

$$prox_\lambda(x) = \text{sign}(x)\ \max(x - \lambda, 0) \tag{6.7}$$

The gradient for $g(\mathbf{V})$ is given by:

$$\nabla g(\mathbf{v_i}) = -(\mathbf{X}_i\mathbf{U})^T(\mathbf{y}_i - \mathbf{X}_i\mathbf{U}\mathbf{v}_i) - \rho_1\mathbf{R}^T(\mathbf{z}_i - \mathbf{R}\mathbf{v}_i)$$

which can be plugged into Equation (6.6) to obtain the new $\mathbf{V}^{(k)}$.

Update formula for U Assuming $\mathbf{V}$ and $\mathbf{R}$ are given, the latent factors $\mathbf{U}$ are estimated by minimizing the following objective function:

$$\mathcal{L}(\mathbf{U}) = \frac{1}{2}\sum\nolimits_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i\mathbf{U}\mathbf{v}_i \parallel_2^2 + \rho_2 \parallel \mathbf{U} \parallel_1 \tag{6.8}$$

Once again, since $\mathcal{L}(\mathbf{U})$ is not a smooth function, we apply proximal gradient descent to update $\mathbf{U}$ as follows:

$$\mathbf{U}^{(s)} = prox_\lambda(\mathbf{U}^{(s-1)} - \lambda\nabla g(\mathbf{U}^{(s-1)})) \tag{6.9}$$

where the proximal mapping is the same as the soft thresholding function given in Equation (6.7). The gradient of the smooth part of the objective function given in Equation (6.8) is

$$\nabla g(\mathbf{U}) = \sum\nolimits_{i=1}^{r} \mathbf{1}_{m\times 1}(\mathbf{y}_i - \mathbf{X}_i\mathbf{U}\mathbf{v}_i)^T)\mathbf{X}_i \odot \mathbf{v}_i\mathbf{1}_{1\times d}$$

Update formula for R Finally, assuming $\mathbf{U}$ and $\mathbf{V}$ are fixed, the latent factor $\mathbf{R}$ is updated by minimizing the following terms in the objective function that depend on $\mathbf{R}$:

$$\mathcal{L}(\mathbf{R}) = \frac{\rho_1}{2} \parallel \mathbf{Z}^T - \mathbf{R}\mathbf{V} \parallel_F^2 + \rho_4 \parallel \mathbf{R} \parallel_1 \tag{6.10}$$

The update formula for $\mathbf{R}$ is derived using the proximal gradient descent approach as follows:

$$\mathbf{R}^{(s)} = prox_\lambda(\mathbf{R}^{(s-1)} - \lambda \bigtriangledown\ g(\mathbf{R}^{(s-1)})) \tag{6.11}$$

Algorithm 5 Multi-Level Multi-Task learning (MLMT) framework.

Input: $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{m}, \rho_1, \rho_2, \rho_3, \rho_4$
Output: $\mathbf{U}, \mathbf{V}, \mathbf{R}$
Initialization: $k = 0$, $\mathbf{U}^{(0)}$, $\mathbf{V}^{(0)}$, and $\mathbf{R}^{(0)}$
repeat
  $k = k + 1$
  Update $\mathbf{V}^{(k)}$ by solving Equation (6.6)
  Update $\mathbf{U}^{(k)}$ by solving Equation (6.9)
  Update $\mathbf{R}^{(k)}$ by solving Equation (6.11)
until convergence
return $\mathbf{U}, \mathbf{V}, \mathbf{R}$

where the gradient of the smooth function $\nabla g(R)$ is given by:

$$\nabla g(\mathbf{R}) = -\rho_1(\mathbf{Z}^T - \mathbf{R}\mathbf{V})\mathbf{V}^T$$

The pseudocode of the proposed framework called **MLMT** is summarized in Algorithm 5. The latent factors are initialized as follows. We first compute an initial model $\mathbf{W}^{(0)}$ by applying existing methods such as lasso regression or multi-task learning [71] on the local predictors only. We then factorize $\mathbf{W}^{(0)}$ into a product of $\mathbf{U}^{(0)}$ and $\mathbf{V}^{(0)}$. The initial value for $\mathbf{R}^{(0)}$ is then obtained by solving Equation (6.10). After the initialization, the latent factors are iteratively updated using the formula given in Equations (6.6), (6.9), and (6.11) until one of the the following two stopping conditions are met: (1) if the maximum number of iterations is reached, or (2) the value of the objective function does not change significantly.

6.4.3 Proof of Convergence

The following proposition can be used to prove the convergence of our block coordinate descent algorithm.

Proposition 1 *Let* $\mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{R})$ *be the objective function given in Equation* (6.4). *The sequence* $\mathcal{L}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k)})$ *computed by the MLMT framework is non-increasing for* $k = 1, 2, \cdots$.

Proof 1 *It is easy to see that*

$$\mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{R}) = f(\mathbf{U}, \mathbf{V}) + g(\mathbf{R}, \mathbf{V})$$

. The latent factors are iteratively updated using the block coordinate descent approach. First, the latent factor V is updated by using the proximal gradient descent algorithm to solve Equation (6.5)*. The following inequality holds for $\mathbf{V}^{(k)}$ after applying proximal gradient descent:*

$$\mathcal{L}(\mathbf{V}^{(k)}) - \mathcal{L}(\mathbf{V}^{*(k)}) \leq \frac{\| \mathbf{V}^{(k-1)} - \mathbf{V}^{*(k)} \|_2^2}{2tn_k},$$

where n_k is number of interations in proximal gradient descent, t is the step size, and $\mathbf{V}^{*(k)}$ is the optimal solution for Equation (6.6)*. The inequality states that the proximal gradient descent solution converges at the rate of $O(\frac{1}{n_k})$. The proof for this inequality is given in [8]. Similar inequalities also hold when applying proximal gradient descent to update $\mathbf{U}$ and $\mathbf{R}$.*

Since $\mathbf{V}$ is updated by minimizing the terms in the objective function that depend on $\mathbf{V}$, we have: $\mathcal{L}(\mathbf{U}^{(k-1)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k-1)}) \leq \mathcal{L}(\mathbf{U}^{(k-1)}, \mathbf{V}^{(k-1)}, \mathbf{R}^{(k-1)})$ Similarly, the objective function after updating $\mathbf{U}$ is:

$$\begin{aligned} & \mathcal{L}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k-1)}) \\ = \; & f(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}) + g(\mathbf{R}^{(k-1)}, \mathbf{V}^{(k)}) \\ \leq \; & f(\mathbf{U}^{(k-1)}, \mathbf{V}^{(k)}) + g(\mathbf{R}^{(k-1)}, \mathbf{V}^{(k)}) \\ = \; & \mathcal{L}(\mathbf{U}^{(k-1)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k-1)}) \end{aligned}$$

Finally, upon updating $\mathbf{R}$, we have:

$$\mathcal{L}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k)}) \leq \mathcal{L}(\mathbf{U}^{(k)}, \mathbf{V}^{(k)}, \mathbf{R}^{(k-1)}),$$

which completes the proof.

Since the objective function is bounded from below by 0, the convergence of the algorithm is guaranteed by the monotonicity theorem.

6.4.4 Cross-scale Interactions (CSIs)

This section presents an approach for deriving the cross-scale interactions of the proposed MLMT framework. Specifically, the cross-scale interactions can be inferred by examining the regression coefficient that relates the local and regional predictors of the data, analogous to Equation (6.3). To illustrate this, we consider a variation of the traditional multi-level modeling method given in Equation (6.1) by casting its formulation into the following optimization problem:

$$\min_{\mathbf{G},\mathbf{W}} \frac{1}{2}\sum_{i=1}^{R} \parallel \mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i \parallel_2^2 + \frac{\rho_1}{2}\sum_{i=1}^{R} \parallel \mathbf{w}_i - \mathbf{G}^T\mathbf{z}_i \parallel_2^2 \tag{6.12}$$

In this relaxed multi-level modeling approach, the first term of the objective function penalizes the regression error for each region while the second term fits the regression coefficient to the regional predictors. Taking the partial derivative of the objective function with respect to W and setting it to zero yields the following solution:

$$\mathbf{w}_i = (\mathbf{X}_i^T\mathbf{X}_i + \rho_1\mathbf{I})^{-1}\mathbf{X}_i^T\mathbf{y}_i + \rho_1(\mathbf{X}_i^T\mathbf{X}_i + \rho_1\mathbf{I})^{-1}\mathbf{G}^T\mathbf{z}_i \tag{6.13}$$

Observe that the first term of the regression coefficient is equivalent to the solution for ridge regression using only the local predictor variables. The second term, on the other hand, is a correction factor due to the regional variables. Given a test example $(\mathbf{x}^*, \mathbf{z}_i)$ from region i, we can predict its response value as follows:

$$\hat{y} = \mathbf{x}^*\mathbf{w}_i = \mathbf{x}^*(\mathbf{X}_i^T\mathbf{X}_i + \rho_1\mathbf{I})^{-1}\mathbf{X}_i^T\mathbf{y}_i + \mathbf{x}^*\hat{\mathbf{G}}_i^T\mathbf{z}_i$$

where $\hat{\mathbf{G}}_i = \rho_1\mathbf{G}(\mathbf{X}_i^T\mathbf{X}_i + \rho_1\mathbf{I})^{-1}$. In other words, the predicted value can be decomposed into two parts: a local prediction term and a cross-scale interactions term involving both $\mathbf{x}^*$ and $\mathbf{z}_i$. Therefore, $\hat{\mathbf{G}}_i$ is a modified cross-scale interactions term for the relaxed version of the multi-level modeling formulation given in Equation (6.12). It is also worth noting that $\hat{\mathbf{G}}_i$ is no longer a constant. Instead, it may vary from one region to another, depending on the covariance matrix of its local predictors, $\mathbf{X}_i^T\mathbf{X}_i$.

Using the same strategy, the cross-scale interactions term for our proposed formulation is given by the following theorem.

Theorem 1 *Let* $\mathbf{U}$ *be the latent factors associated with the local predictors and* $\mathbf{R}$ *be the latent factors associated with the regional predictors for the multi-level multi-task learning framework given in Equation* (6.4)*. The cross-scale interactions term for the formulation is*

$$\bar{\mathbf{G}}_i = \rho_1 \mathbf{R}(\mathbf{U}^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{U} + \rho_1 \mathbf{R}^T \mathbf{R})^{-1} \mathbf{U}^T. \tag{6.14}$$

Proof 2 *Ignoring the L*1*-regularization terms, the objective function can be re-written as follows:*

$$\frac{1}{2} \sum_{i=1}^{r} \parallel \mathbf{y}_i - \mathbf{X}_i \mathbf{U} \mathbf{v}_i \parallel_2^2 + \frac{\rho_1}{2} \parallel \mathbf{Z}^T - \mathbf{R}\mathbf{V} \parallel_F^2 \tag{6.15}$$

Taking the partial derivative of the objective function with respect to $\mathbf{V}$ *and setting it to zero yields the following:*

$$\frac{\partial L}{\partial \mathbf{v}_i} = -(\mathbf{X}_i \mathbf{U})^T (\mathbf{y}_i - \mathbf{X}_i \mathbf{U} \mathbf{v}_i) - \rho_1 \mathbf{R}^T (\mathbf{z}_i - \mathbf{R}\mathbf{v}_i) = 0$$

$$\Rightarrow \mathbf{v}_i = \Big[(\mathbf{X}_i \mathbf{U})^T (\mathbf{X}_i \mathbf{U}) + \rho_1 \mathbf{R}^T \mathbf{R} \Big]^{-1} \Big[\mathbf{U}^T \mathbf{X}_i^T \mathbf{y}_i + \rho_1 \mathbf{R}^T \mathbf{z}_i \Big]$$

Thus, the predicted value for a test example $(\mathbf{x}^*, \mathbf{z}_i)$ *can be computed as follows:*

$$\begin{aligned} \hat{y} &= \mathbf{x}^* \mathbf{U} \mathbf{v}_i \\ &= \mathbf{x}^* \mathbf{U} \Big[\mathbf{U}^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{U} + \rho_1 \mathbf{R}^T \mathbf{R} \Big]^{-1} \mathbf{U}^T \mathbf{X}_i^T \mathbf{y}_i + \mathbf{x}^* \bar{\mathbf{G}}_i^T \mathbf{z}_i \end{aligned} \tag{6.16}$$

where $\bar{\mathbf{G}}_i = \rho_1 \mathbf{R}(\mathbf{U}^T \mathbf{X}_i^T \mathbf{X}_i \mathbf{U} + \rho_1 \mathbf{R}^T \mathbf{R})^{-1} \mathbf{U}^T$*. The first term corresponds to the value predicted using the local predictors only whereas the second term* $\mathbf{x}^* \bar{\mathbf{G}}_i^T \mathbf{z}_i$ *involves the local and regional variables. Thus,* $\bar{\mathbf{G}}_i^T$ *corresponds to the cross-scale interactions term between the two variables for our proposed framework.*

The original multi-level modeling formulation imposes a strict constraint where the expected value $E[\mathbf{w}_i] = \mathbf{G}^T \mathbf{z}_i$. Thus, its cross-scale interactions ($\mathbf{G}$) is identical for all the regions. In the relaxed version of the multi-level modeling formulation given in Equation

(6.12), $\mathbf{w}_i$ is a combination of the regression coefficient for the local predictors as well as a term proportional to $\hat{\mathbf{G}}_i^T \mathbf{z}_i$. Thus, its cross-scale interactions term $\hat{\mathbf{G}}_i$ may vary from one region to another, based on the covariance matrix $\mathbf{X}_i^T \mathbf{X}_i$ of its local predictors. For our proposed framework, the regression coefficient $\mathbf{w}_i$ also comprises of two terms, one depending on the local predictors while the other is proportional to $\bar{\mathbf{G}}_i^T \mathbf{z}_i$, where $\bar{\mathbf{G}}_i$ depends on the covariance matrix of the local predictors as well as the local and regional latent factors, i.e., $\mathbf{U}$ and $\mathbf{R}$.

6.4.5 Generalization to N-level Modeling

For N-level modeling where $N > 2$, let $\mathbf{X}^{(1)}$ be the design matrix for the local predictors and $\{\mathbf{X}^{(2)}, \mathbf{X}^{(3)}, \cdots, \mathbf{X}^{(N)}\}$ be the set of matrices corresponding to coarser-level predictors from level 2 to N. Similar to the 2-level model, the response variable can be factorized into a set of latent factors ($\mathbf{U}$) associated with the local predictors as well as the feature representation $\mathbf{V}^{(1)}$ of coarser "regions" defined at level 2:

$$\min_{\mathbf{U}, \mathbf{V}^{(1)}} \sum_i \parallel \mathbf{y}_i - \mathbf{X}_i^{(1)} \mathbf{U} \mathbf{v}_i^{(1)} \parallel_2^2 .$$

Let $\mathbf{X}_i^{(l)} \in \mathbb{R}^{r_i^{(l)} \times d_l}$ denote the matrix of "regional" predictors associated with the i^{th} region in level l, where d_l is the number of regional predictors and $r_i^{(l)}$ is the number of finer subregions at level $l-1$ contained within the i^{th} region at level l. Figure 6.2 shows an example of a multi-level nested data, where each geographical object corresponds to a city. In this example, $\mathbf{X}_1^{(1)}$ corresponds to the matrix of local predictors for all cities located in county1, $\mathbf{X}_1^{(2)}$ corresponds to the matrix of county-level predictors for all counties located in state1, and $\mathbf{X}_1^{(3)}$ corresponds to the matrix of state-level predictors for all the states located in the given country.

Assume the regional predictors at level l are factorized into m_l latent factors. Furthermore, let $\mathbf{X}_{(i,j)}^{(l-1)} \in \mathbb{R}^{r_{(i,j)}^{(l-1)} \times d_{l-1}}$ be the sub-matrix of regional predictors for the j^{th}

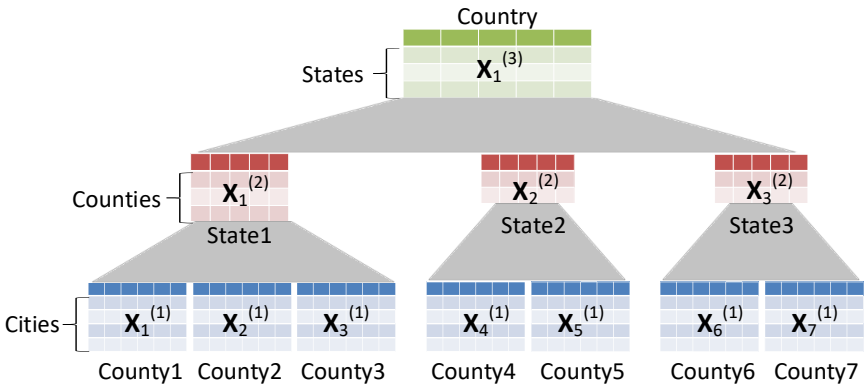


Figure 6.2: Example of a multi-level nested data. The finest level represents the cities. Each city belongs to a county, which in turn, is located within a state in a given country.

sub-region of the i-th region at level l. The second subscript j is introduced to allow the region at a given level to be related to its subregion at a lower level. For example, in Figure 6.2, $\mathbf{X}_{(2,1)}^{(2-1)}$ refers to the first subregion of the second region of level 2, i.e., the matrix $X_4^{(1)}$. Note that $\mathbf{X}_i^{(l)}$ and $\mathbf{X}_{(i,j)}^{(l-1)}$ can be jointly factorized as $\min_{\mathbf{R},\mathbf{P},\mathbf{V}} \parallel (\mathbf{X}_i^{(l)})^T - \mathbf{R}^{(l)}\mathbf{P}_i^{(l)}\mathbf{V}_i^{(l)} \parallel_2^2$ and $\min_{\mathbf{R},\mathbf{P},\mathbf{V}} \parallel (\mathbf{X}_{(i,j)}^{(l-1)})^T - \mathbf{R}^{(l-1)}\mathbf{P}_{(i,j)}^{(l-1)}\mathbf{V}_{(i,j)}^{(l-1)} \parallel_2^2$, respectively, where $\mathbf{R}^{(l)} \in \mathbb{R}^{d_l \times m_l}$ is the latent factor shared by all the regions at level l. The matrix $\mathbf{P}_i^{(l)} \in \mathbb{R}^{m_l \times m_{l+1}}$ captures the relationship between the latent factors at different levels. Furthermore, $\mathbf{V}_i^{(l)} \in \mathbb{R}^{m_{l+1} \times r_i^{(l)}}$ and $\mathbf{V}_{(i,j)}^{(l-1)} \in \mathbb{R}^{m_l \times r_i^{(l-1)}}$. Each column of $\mathbf{V}_{(i,j)}^{(l-1)}$ is defined as the j^{th} column of $\mathbf{P}_i^{(l)}\mathbf{V}_i^{(l)}$.

Putting them together, the objective function for our N-level multi-task learning framework can be expressed as follows:

$$\mathcal{L}(\mathbf{U},\mathbf{V},\mathbf{P},\mathbf{R}) = \sum_{i=1}^{r_1} \parallel \mathbf{y}_i - \mathbf{X}_i^{(1)}\mathbf{U}\mathbf{v}_i^{(1)} \parallel_2^2 \tag{6.17}$$
$$+ \sum_{l=2}^{N} \sum_{i=1}^{r_l} \parallel (\mathbf{X}_i^{(l)})^T - \mathbf{R}^{(l)}\mathbf{P}_i^{(l)}\mathbf{V}_i^{(l)} \parallel_F^2 + \Omega(\mathbf{U},\mathbf{V},\mathbf{P},\mathbf{R})$$

where $\mathbf{v}_i^{(1)}$ is a column in $\mathbf{V}_i^{(1)}$. The first term in (6.17) corresponds to the prediction error at level 1 while the second term learns the latent factors $\mathbf{V}$ of the nested data. $\Omega(\mathbf{V},\mathbf{P},\mathbf{R})$ denote the regularization term on the parameters. The objective function can be solved using a block coordinate descent approach, similar to the algorithm used for our previous 2-level data.

6.5 Experimental Evaluation

We have performed extensive experiments to evaluate the performance of the proposed multi-level multi-task learning (MLMT) framework using nested datasets from the lake ecology domain.

6.5.1 Datasets

The lake water quality datasets were obtained from LAGOS-NE [109], which is a geospatial database containing land cover/use features and lake chemistry data measured at multiple scales covering the Northeastern part of the United States. We selected four water quality metrics as response variables, including two lake nutrient variables (total phosphorus [TP] and total nitrogen [TN] concentrations), a measure of algal biomass (chlorophyll-a [chla]), and Secchi depth (Secchi), a measure of water clarity. The sampling years for the response variables span from 2000 to 2013. For each lake, we extracted the sample data from the summer months of June, July, and August, and took their average values over all the sampling years to represent the ground truth value for each response variable. We also selected 13 variables, including lake hydrogeomorphic variables and land cover/use data from 2001, as the local predictors. Ecological Drainage Units (EDUs) [48] were used to define the spatial regions of the study. The regionalization scheme has been used in a previous study on multi-scale modeling of lake water quality [124]. We extracted 8 regional predictors, including the hydrogeomorphic and land cover/use variables measured at the coarser EDU-level. All the local and regional predictors are standardized to have zero mean and unit standard deviation while the response variables are log-transformed similar to the approach used in [124]. As shown in Table 6.1, the number of instances (lakes) in each region (EDU) with ground truth data available varies from one response variable to another.

Table 6.1: Summary statistics for 4 lake water quality data.

Response variable	TP	TN	Chla	Secchi
# regions (EDUs)	86	83	87	88
# instances (lakes)	4352	1946	5592	5796
#instances/region	1 - 369	1 - 236	1 - 575	1 - 583
mean value	37.58	739.25	17.19	2.78
standard deviation	66.75	1015.99	29.56	1.87

6.5.2 Experimental Setup

The proposed multi-level multi-task (MLMT) learning framework along with the baseline algorithms were implemented in Matlab. Our source code for the proposed framework and other baselines are available at [144]

6.5.2.1 Baseline Methods

We compared the performance of our framework against the following four baseline methods:

- *Global-L*: This method makes the following two assumptions: (1) the relationship between the predictor and response variables are the same across all the regions and (2) the regional predictors do not influence the response variable. With these assumptions, a single, lasso regression model is trained to fit the local predictors of the training data from all regions.

- *Global-LR*: This method is similar to the previous baseline except it assumes that the regional predictors also influence the response variable. A single, lasso regression model is trained to fit both the local and regional predictors of the training data from all regions.

- *STL*: This method applies lasso regression independently to each region using only the local predictors of the regions. Regional predictors are not used since their values are identical for all the training instances in the same region.

- *MLM*: This method applies L1-regularization to the multi-level modeling formulation (see Equation (6.1)) to build a separate model for each region [74]. This multi-level lasso

method assumes that the regression coefficients for each region are directly tied to the regional variables via their shared cross-scale interactions term, $\mathbf{G}$.

6.5.2.2 Evaluation Metric

We employed two metrics to evaluate the performance of the different methods. The first metric is root mean square error (RMSE), which measures the deviation between the observed and predicted values of the response variable. The metric can be calculated as follows:

$$RMSE = \sqrt{\sum_{i=1}^{N} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2 / N} \tag{6.18}$$

where $\hat{\mathbf{y}}_i$ is the predicted value and N is the number of predicted instances.

The second metric is the coefficient of determination, R^2, which measures the variance in the response variable explained by the model. The metric is calculated as follows:

$$R^2 = 1 - \frac{\sum_i (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2}{\sum_i (\mathbf{y}_i - \bar{\mathbf{y}})^2} \tag{6.19}$$

where $\bar{y}$ is the mean of the observed values for the response variable.

6.5.3 Experimental Results

6.5.3.1 Performance Comparison for All Regions

For evaluation purposes, we partitioned each dataset into separate training and test sets, using 2/3 of the data for training and the remaining 1/3 for testing. We further divide the training set into two halves, one for training and the other for validation (hyperparameter tuning). We repeated this 10 times with different training and test partitions and reported the average and standard deviation of RMSE and R^2 values in Table 6.2. The results in this table suggest that single task learning (STL) performs the worst among all five competing methods on 3 of the 4 datasets, which is not surprising for several reasons. First, the sample sizes are highly imbalanced among the regions. In fact, for all the response variables, more than 20% of the EDUs (regions) have 10 or less lakes with observation data available. The

Table 6.2: Results for 4 lake water quality data.

Response	TP	TN	Chla	Secchi
Global-L	0.330±0.003	0.231±0.007	0.426±0.013	0.274±0.019
Global-LR	0.310±0.004	0.214±0.006	0.413±0.018	0.258±0.022
STL	0.564±0.475	0.546±0.033	0.529±0.195	0.260±0.015
MLM	0.302±0.005	0.210±0.006	0.423±0.044	0.242±0.011
MLMT	**0.286±0.004**	**0.203±0.006**	**0.381±0.014**	**0.231±0.010**

(a) RMSE results

Response	TP	TN	Chla	Secchi
Global-L	0.414±0.009	0.515±0.018	0.359±0.031	0.351±0.093
Global-LR	0.485±0.014	0.584±0.023	0.399±0.044	0.421±0.101
STL	0.095±0.075	0.011±0.035	0.056±0.033	0.275±0.021
MLM	0.511±0.016	0.599±0.020	0.364±0.140	0.494±0.046
MLMT	**0.560±0.010**	**0.624±0.024**	**0.489±0.030**	**0.540±0.044**

(b) R^2 results

models for many of these data-poor regions are likely to be inferior. Second, the independent models were not able to fully utilize the regional predictors as their values are identical for all the lakes in the same region. Third, the existing regions (EDUs) for which the lasso models were trained may not be ideal for the predictive modeling task. This last point will be further illustrated in Section 6.5.3.4. For Secchi, STL performs relatively better than the global models because there are more instances available in each region to construct an effective model.

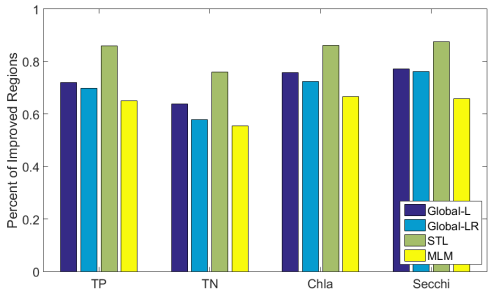


Figure 6.3: Percentage of regions in which MLMT performs better than baseline methods.

We also observe that global-L is worse than global-LR, which suggests the value of incorporating regional predictors into the predictive modeling framework. Nevertheless, the

performances of the global models are inferior compared to the multi-level modeling (MLM) approach since they both apply the same model to all the regions. Finally, the proposed MLMT framework consistently outperforms all the baseline methods on all four datasets.

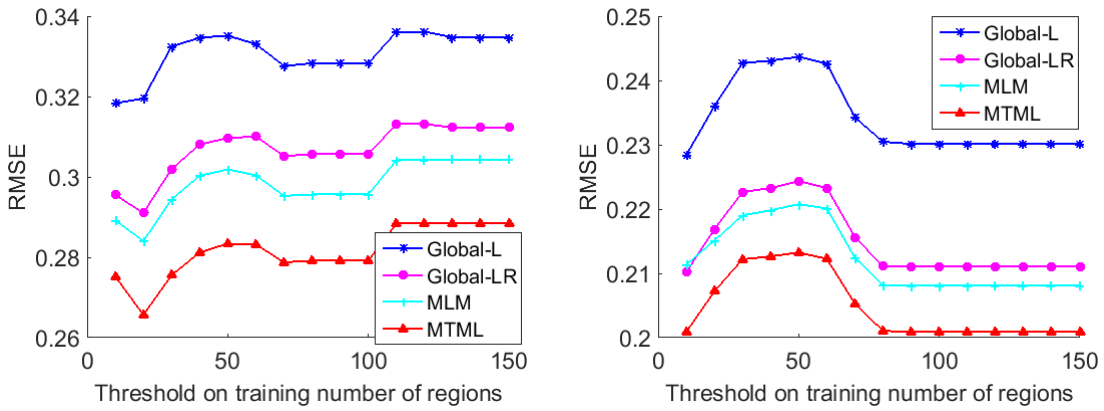
In addition to comparing the magnitude of their RMSE and R^2 values, we also compare the number of regions in which MLMT outperforms the baseline methods. Specifically, for each dataset, we calculate the percentage of regions in which our method outperforms each baseline and take the average percentages over the 10 training and test set partitions. As can be seen from the results shown in Figure 6.3, MLMT outperforms all the baseline methods in more than 64% of the regions in 3 of the 4 datasets. The percentage increases to over 70% of the regions when compared against STL. For the TN dataset, which has fewer instances available, MLMT still performs better than MLM in more than 55% of the regions.

6.5.3.2 Performance Comparison for Data-Poor Regions

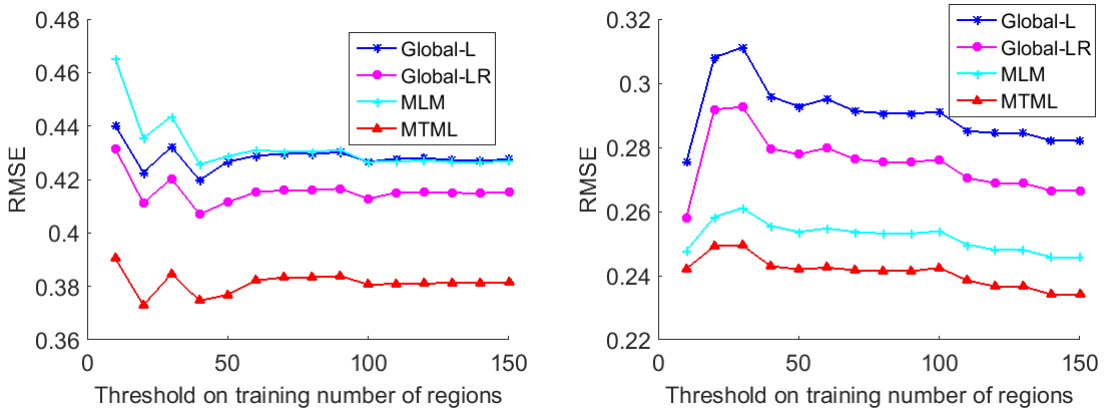
In this experiment, we evaluate the performance of all the methods for regions with small training set sizes. To identify such regions, we define a maximum sample size threshold τ and calculate the RMSE values for the test examples located in regions that have less than τ training examples. We vary τ from 10 to 150 and plot the results in Figure 6.4. The results in this figure suggest that the RMSE value of MLMT for the data-poor regions is consistently lower than all the baseline methods. This validates our assertion that the shared latent factors enable the data-poor regions to leverage information from other regions in order to construct more effective models.

6.5.3.3 Cross-scale Interactions

In this section, we first examine the cross-scale interactions (CSIs) found by MLMT that contribute to the prediction of total phosphorous (TP). As mentioned in Section 6.4.1, our framework allows the CSIs to vary by region. The CSIs can be visualized by plotting the $\bar{\mathbf{G}}_i$'s given in Equation (6.14). For TP, we found all the 86 regions follow a similar CSI



(a) TP (b) TN



(c) Chla (d) Secchi

Figure 6.4: Performance comparison for regions with limited number of training data.

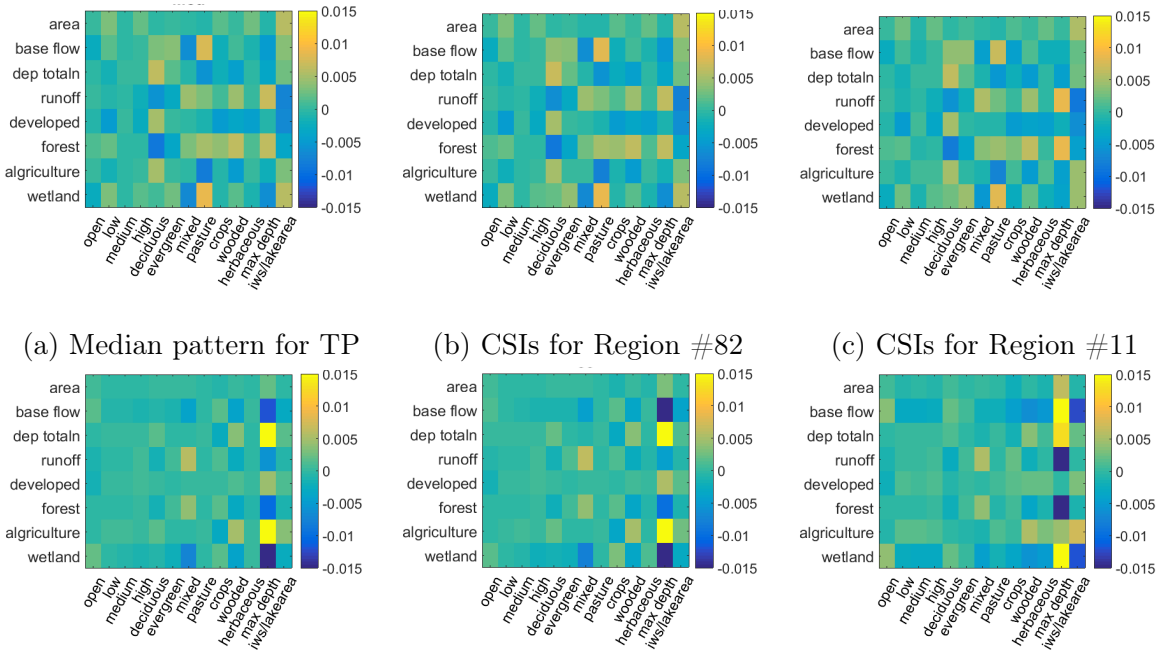
pattern, as evidenced by the high average correlation (0.991) between the $\bar{\mathbf{G}}$ matrices of all the regions. The median pattern, computed from the median value of $\hat{\mathbf{G}}$, is shown in Figure 6.5(a). We also show the regions whose CSI patterns are most and least correlated with the median pattern in Figures 6.5(b) and (c), respectively. These figures suggest there is very little difference between the CSI patterns of TP for all the regions.

To validate the CSI patterns found by MLMT, we examine its median CSI pattern (see Figure 6.5(a)) and compare it against previous results reported in the literature. For example, our analysis showed that the coefficient in $\bar{\mathbf{G}}$ for regional agriculture and local wetland (wooded) is negative (-0.005). This coefficient denotes the relationship between regional agriculture and the slope of the wetland-TP relationship. The CSI term suggests that when the proportion of agricultural land use in a region is low, the wetland-TP relationship is

positive. In contrast, when the proportion of agricultural land use in a region is high, the wetland-TP relationship is negative. This result matches the previous finding given in [35, 110]. An explanation to this is that in regions with little agriculture, wetlands may be the source of phosphorus to lakes (positive slope), but when agriculture increases, wetland effects on lake phosphorus becomes negative since the wetlands may be retaining phosphorus from getting into lakes.

Among the CSIs identified by the median pattern include regional wetland and local pasture(0.009), regional forest and local deciduous(-0.009), regional base flow and local pasture (0.008) and regional wetland and max depth (-0.006). These CSIs represent the complex interactions that exist between local lake properties and the broader regional context in which a lake is located. For example, the negative CSI between the proportion of regional wetland cover and max depth suggests that the lake morphological controls on TP, as a result of increasing lake depth, are stronger when a lake is embedded in a landscape with abundant wetlands. This may be due to the additional reduction of TP entering a lake due to retention of nutrients within wetland complexes [35].

Although the CSI patterns for total phosphorous are very similar for all the regions, the patterns do vary by region for Secchi depth. The median CSI pattern for Secchi depth is shown in Figure 6.5(d). Many regions have CSI patterns that are very similar to the median pattern. This includes the CSI pattern for region #54, which is shown in Figure 6.5(e). However, there are several regions whose CSI patterns are considerably different than the median pattern. For example, Figure 6.5(f) shows the CSI pattern for region #88. For this region, some relationships such as those between regional base flow and local max depth and between regional wetland and local max depth have completely opposite sign compared to the relationships shown by the median CSI pattern.



(a) Median pattern for TP (b) CSIs for Region #82 (c) CSIs for Region #11

(d) Median pattern for Secchi (e) CSIs for Region #54 (f) CSIs for Region #88

Figure 6.5: Cross-scale interactions between local and regional predictors for the prediction of total phosphorous (a)-(c) and Secchi depth (d)-(f). For each plot, the horizontal axis denotes the local predictors while the vertical axis denotes the regional predictors.

6.5.3.4 Comparison Between the New and Original Regions

Since the original regions (EDUs) are created for other purposes, we hypothesized that a better set of regions can be derived for predictive modeling using the latent factors associated with the lakes. To do this, we first compute the latent feature representation of each lake, which is given by $\mathbf{XU}$. We then apply k-means clustering to generate the new set of regions. For a fair comparison, we set the number of clusters to be the same as the number of regions (EDUs) in the original data. A lasso regression model is independently trained for each new region using only their local predictor variables. Similarly, we also train lasso regression models for each of the original regions. We then compare the performances of the models for the new regions against those for the original regions. Table 6.3 summarizes their RMSE values. The results in this table suggest that the local models trained on the new regions have a lower RMSE compared to the local models trained on the original (EDU) regions in 3

Table 6.3: RMSE comparison for original and new regions.

Response variable	TP	TN	Chla	Secchi
Original regions (EDUs)	0.564	0.546	0.529	**0.260**
New regions	**0.403**	**0.519**	**0.487**	0.267

out of the 4 datasets. This supports our hypothesis that the new regions created by MLMT can be used to build more accurate local prediction models compared to the original regions.

6.5.3.5 Sensitivity Analysis

The proposed framework requires tuning the following 4 hyper-parameters: ρ_1, ρ_2, ρ_3 and ρ_4. For the experiments described in the previous subsections, the hyper-parameters are tuned using the validation set. This section examines the sensitivity of MLMT to changes in the hyper-parameter values. To do this, we vary the value of each hyper-parameter from 0.001 to 100 and plot the changes in their RMSE values in Figure 6.6. Note that the hyper-parameter values shown on the horizontal axis are plotted on the log scale. The results suggest that the RMSE values are quite stable for a relatively wide range of hyper-parameter values. In fact, the RMSE values do not change significantly when varying ρ_4. To achieve a low RMSE, ρ_1 prefers smaller values while both ρ_2 and ρ_3 appear to favor larger values between 1 to 10.

6.6 Conclusions

This chapter presents a novel framework called MLMT for modeling nested geospatial data. The framework jointly trains a set of models that can incorporate both the local and regional predictors into a unified formulation. We also show how cross-scale interactions can be derived for each region using the proposed framework. Experimental results suggest that MLMT outperforms four other baseline methods on the lake water quality datasets evaluated in this study. Finally, we show that MLMT can be used to derive new regions for building more accurate local prediction models compared to using the original regions of the data.

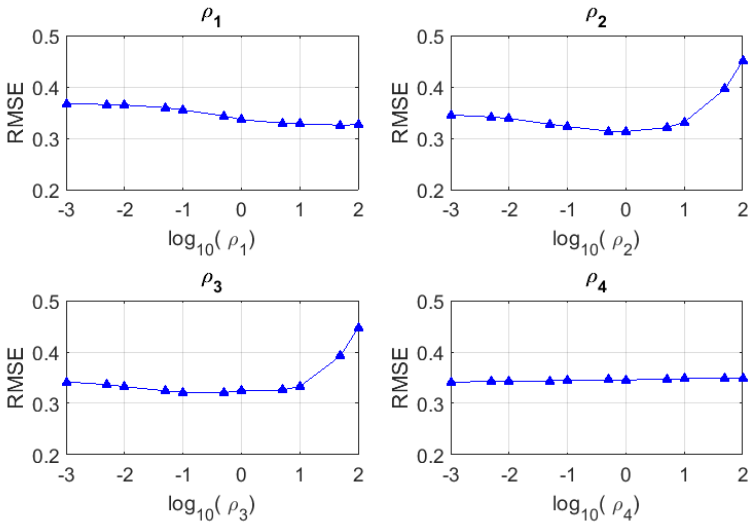


Figure 6.6: Sensitivity analysis for ρ_1, ρ_2, ρ_3 and ρ_4.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

This thesis presents novel learning algorithms to address the practical challenges of mining geospatial data. Specifically, the contributions of the thesis are summarized below.

- I proposed a framework for spatially constrained spectral clustering with application to regionalization. The framework introduces a way to represent the adjacent information into a graph and incorporate the relationship between spatial units as a constraint to ensure the spatial contiguity. The proposed framework can balance the trade off between spatial contiguity and landscape homogeneity. This framework is further extended to a hierarchical setting, thus enabling each cluster to be nested wholly within coarser level clusters.

- I developed a supervised hash-based feature learning approach for incomplete geospatial data with application to predict lake nutrients. The proposed algorithm simultaneously infers the missing feature values while constructing a set of nonlinear hash-based features from the incomplete data. The new feature representation has the following properties: (i) complete without missing values, (ii) reduce the dimensionality, (iii) derived via a supervised learning strategy, (iv) can be subsequently trained to capture the nonlinear relationship between predictor and response variables using efficient linear models.

- I proposed a multi-task learning framework for jointly modeling geospatial data with multiple response variables at different locations. Instead of learning many independent local models for each response variable, this method enhances prediction ability

by incorporating spatial autocorrelation between different locations and correlations among multiple response variables.

- I proposed a novel multi-level multi-task learning framework for nested geospatial data that can be applied to regions that have no training data. The proposed framework can effectively incorporate both the local and regional predictor variables into its formulation. In addition, it can automatically identify potential cross-scale interactions among variables.

7.2 Future Work

In Chapter 3 we proposed a spatially constrained spectral clustering framework. The framework based on two different kernel representation of the geographical graph. In the future, we can try different variation of the representation and compare with the existing representation. In this work, we applied our framework to form regionalization system. However, the application of this framework can go beyond region delineation. We can apply our framework to any dataset that contains both profile of the nodes and adjacent information of the nodes. For example, it can be used in social network data for community detection. The features will be profiles of the user, the data extracted from a user's text posting, etc. The spatial constraint can be the friend status of two users – whether they are linked or not. The task is to cluster users into groups with similar interest.

In Chapter 4 we presented the supervised hash-based feature learning for data contains missing values. This algorithm learn features and impute missing values simultaneously. However, it is a batch learning algorithms and one of the limitation is that when a new incomplete test data comes in, we need to re-train the model based on all the available data. The framework can be extended to incremental learning or online learning fashion, in a way to deal with new unseen data without re-train the whole model. In addition, we apply the Random Fourier feature(RFF) to capture the non-linear relationship between predictors and response variables. It will be interesting to try other non-linear mapping as well.

In Chapter 5 we presented a multi-task learning framework for data with multiple response variables at multiple locations. In this current model, we use latitude and longitude as the constraint information of relationship between different locations. We can try other spatial autocorrelation metrics in the future. In chapter 5 we applied our framework in lake water quality database. We can also apply the algorithm on lots of other data sets as well. For example the framework can be used to predict school's test scores where each school is a different task and there's relationship between different schools. The response variables is different course scores and scores for different courses are not independent from each other.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Robin Abell, Michele L. Thieme, Carmen Revenga, Mark Bryer, Maurice Kottelat, Nina Bogutskaya, Brian Coad, Nick Mandrak, Salvador Contreras Balderas, William Bussing, Melanie L. J. Stiassny, Paul Skelton, Gerald R. Allen, Peter Unmack, Alexander Naseka, Rebecca Ng, Nikolai Sindorf, James Robertson, Eric Armijo, Jonathan V. Higgins, Thomas J. Heibel, Eric Wikramanayake, David Olson, Hugo L. López, Roberto E. Reis, John G. Lundberg, Mark H. Sabaj Pérez, and Paulo Petry. Freshwater ecoregions of the world: A new map of biogeographic units for freshwater biodiversity conservation. BioScience, 58, 2008.

[2] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. Mach. Learn., 73(3):243–272, 2008.

[3] Fernando Bação, Victor Lobo, and Marco Painho. Geo-self-organizing map (geo-som) for building and exploring homogeneous regions. Geographic Information Science, pages 22–37, 2004.

[4] Robert G. Bailey. Ecoregions map of north america: Explanatory note. U.S. Dept. of Agriculture, Forest Service, 1998.

[5] Robert G. Bailey. Ecosystem Geography: From Ecoregions to Sites. Springer-Verlag, 2009.

[6] Arindam Banerjee and Joydeep Ghosh. Scalable clustering algorithms with balancing constraints. Data Mining and Knowledge Discovovery, 13(3):365–395, 2006.

[7] Sugato Basu, Ian Davidson, and Kiri Wagstaff. Constrained Clustering: Advances in Algorithms, Theory, and Applications. Taylor and Francis, 2008.

[8] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. Imaging sciences, 2(1):183–202, 2009.

[9] Justin M Becknell, Ankur R Desai, Michael C Dietze, Courtney A Schultz, Gregory Starr, Paul A Duffy, Jerry F Franklin, Afshin Pourmokhtarian, Jaclyn Hall, Paul C Stoy, et al. Assessing interactions among changing climate, management, and disturbance in forests: a macrosystems approach. BioScience, 65(3):263–274, 2015.

[10] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. TPAMI, 35:1798–1828, 2013.

[11] Stefania Bertazzon, Markey Johnson, Kristin Eccles, and Gilaad G Kaplan. Accounting for spatial effects in land use regression for urban air pollution modeling. Spatial and spatio-temporal epidemiology, 14:9–21, 2015.

[12] Tijl De Bie, Johan A. K. Suykens, and Bart De Moor. Learning from general label constraints. In SSPR/SPR, volume 3138, pages 671–679. Springer, 2004.

[13] Daniel Boley and Jaya Kawale. Constrained spectral clustering using l1 regularization. In SIAM Int'l Conference on Data Mining, pages 103–111. SIAM, 2013.

[14] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. Biological cybernetics, 59(4):291–294, 1988.

[15] Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. Min-wise independent permutations. J. of Comp. Sys. Sci., 60:327–336, 1998.

[16] Donald Brown, Jason Dalton, and Heidi Hoyle. Spatial forecast methods for terrorist events in urban environments. In International Conference on Intelligence and Security Informatics, pages 426–435. Springer, 2004.

[17] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. SIAM Journal on Optimization, 20:1956–1982, 2010.

[18] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. Found. Comput. Math., 9:717–772, 2009.

[19] Rich Caruana. Multitask learning. In Machine Learning, pages 41–75, 1997.

[20] Moses Charikar. Similarity estimation techniques from rounding algorithms. In STOC, pages 380–388, 2002.

[21] Sanjay Chawla, Shashi Shekhar, Weili Wu, and Uygar Ozesmi. Modeling spatial dependencies for mining geospatial data. In Proceedings of the 2001 SIAM International Conference on Data Mining, pages 1–17. SIAM, 2001.

[22] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A convex formulation for learning shared structures from multiple tasks. In Proceedings of the 26th Annual International Conference on Machine Learning, pages 137–144, 2009.

[23] Jianhui Chen, Jiayu Zhou, and Jieping Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 42–50, 2011.

[24] Kendra S. Cheruvelil, Patricia A. Soranno, Katherine E. Webster, and M.T. The multi-scaled drivers of ecosystem state: Quantifying the Ecological Applications, 23:1603–1618, 2013.

[25] Casey Cleve, Maggi Kelly, Faith R. Kearns, and Max Moritz. Classification of the wildland- urban interface: A comparison of pixel- and object-based classifications using high-resolution aerial photography. Transportation Research Record, 32(4):317–326, 2008.

[26] Tom Coleman, James Saunderson, and Anthony Wirth. Spectral clustering with inconsistent advice. In ICML, pages 152–159, 2008.

[27] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. Decis. Support Syst., 47:547–553, 2009.

[28] R Cameron Craddock, G Andrew James, Paul E Holtzheimer, Xiaoping P Hu, and Helen S Mayberg. A whole brain fMRI atlas generated via spatially constrained spectral clustering. Human brain mapping, 33:1914–1928, 2012.

[29] Ian Davidson and S. S. Ravi. Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. In Lecture notes in computer science, pages 59–70. Springer, 2005.

[30] Chris Ding and Xiaofeng He. Cluster merging and splitting in hierarchical clustering algorithms. In Proc. IEEE Int'l Conf. Data Mining, pages 139–146, 2002.

[31] Juan Carlos Duque, Raúl Ramos, and Jordi Suriñach. Supervised regionalization methods: A survey. International Regional Science Review, 30:195–220, 2007.

[32] Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. J. Mach. Learn. Res., 6:615–637, 2005.

[33] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 109–117. ACM, 2004.

[34] Angel M. Felicísimo. Uses of spatial predictive models in forested areas territorial planning. In CIOT-IV international conference on spatial planning, pages 2–4, 2003.

[35] Emi Fergus, Patricia A. Soranno, Kendra S. Cheruvelil, and Mary T Bremigan. Multiscale landscape and wetland drivers of lake total phosphorus and water color. Limnology and Oceanography, 56(6):2127–2146, 2011.

[36] Kelwin Fernandes, Pedro Vinagre, and Paulo Cortez. A proactive intelligent decision support system for predicting the popularity of online news. In EPIA, pages 535–546, 2015.

[37] Christopher T Filstrup, Tyler Wagner, Patricia A Soranno, Emily H Stanley, Craig A Stow, Katherine E Webster, and John A Downing. Regional variability among nonlinear chlorophyll—phosphorus relationships in lakes. Limnology and Oceanography, 59(5):1691–1703, 2014.

[38] Andrew Gelman and Jennifer Hill. Data analysis using regression and multilevel hierarchical models, volume 1. Cambridge University Press New York, 2007.

[39] John A George, Bruce W Lamar, and Chris A Wallace. Political district determination using large-scale network optimization. Socio-Economic Planning Sciences, 31(1):11–28, 1997.

[40] Bardan Ghimire, John Rogan, and Jennifer Miller. Contextual land-cover classification: incorporating spatial dependence in land-cover classification models using random forests and the getis statistic. Remote Sensing Letters, 1(1):45–54, 2010.

[41] André Ricardo Gonçalves, Fernando J. Von Zuben, and Arindam Banerjee. A multi-task learning view on the earth system model ensemble. Computing in Science and Engineering, 17(6):35–42, 2015.

[42] Oleksandr Grygorash, Yan Zhou, and Zach Jorgensen. Minimum spanning tree based clustering algorithms. In 18th IEEE International Conference on Tools with Artificial Intelligence, pages 73–81, Arlington, VA, 2006.

[43] Kelly-Ann Dixon Hamil, Basil V Iannone III, Whitney K Huang, Songlin Fei, and Hao Zhang. Cross-scale contradictions in ecological relationships. Landscape ecology, 31(1):7–18, 2016.

[44] Jiawei Han, M. Kamber, and A.K.H. Tung. Spatial clustering methods in data mining: A survey. In H.J. Miller and J. Han, editors, Geographic data mining and knowledge discovery, pages 188–217. Taylor and Francis, 2001.

[45] Jiawei Han and Krzysztof Koperski. Discovery of spatial association rules in geographic information databases. In 4th Int'l Symp. on Large Spatial Databases, 1995.

[46] Fatma Haouas, Zouhour Ben Dhiaf, and Basel Solaiman. Fusion of spatial auto-correlation and spectral data for remote sensing image classification. In Advanced Technologies for Signal and Image Processing (ATSIP), pages 537–542. IEEE, 2016.

[47] William Hargrove and Forrest Hoffman. Potential of multivariate quantitative methods for delineation and visualization of ecoregions. Environmental Management, 34:S39–S60, 2004.

[48] Jonathan V Higgins, Mark T Bryer, Mary L Khoury, and Thomas W Fitzhugh. A freshwater classification approach for biodiversity conservation planning. Conservation Biology, 19(2):432–445, 2005.

[49] Jeffrey W Hollister, W Bryan Milstead, and Betty J Kreakie. Modeling lake trophic state: a random forest approach. Ecosphere, 7(3), 2016.

[50] George E. Host, Philip L. Polzer, David J. Mladenoff, Mark A. White, and Thomas R. Crow. A quantitative approach to developing regional ecosystem classifications. Ecological Applications, 6:608–618, 1996.

[51] Housing Data Set. `http://archive.ics.uci.edu/ml/datasets/Housing`, 1993.

[52] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proc of the Annual ACM Symposium on Theory of Computing, pages 604–613, 1998.

[53] Cem Iyigun, Murat Türkeş, İnci Batmaz, Ceylan Yozgatligil, Vilda Purutçuoğlu, Elçin Kartal Koç, and Muhammed Z Öztürk. Clustering current climate regions of turkey by using a multivariate statistical method. Theoretical and applied climatology, 114(1-2):95–106, 2013.

[54] Anil K. Jain and Richard C. Dubes. Algorithms for Clustering Data. Prentice-Hall, Inc., 1988.

[55] Anil K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. ACM Computing Surveys, 31(3):264–323, 1999.

[56] Shuiwang Ji and Jieping Ye. An accelerated gradient method for trace norm minimization. In ICML, pages 457–464, 2009.

[57] Sepandar D. Kamvar, Dan Klein, and Christopher D. Manning. Spectral learning. In In IJCAI, pages 561–566, 2003.

[58] Risi Imre Kondor and John D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In 19th Int'l Conference on Machine Learning, pages 315–322. Morgan Kaufmann Publishers Inc., 2002.

[59] Weihao Kong, Wu-Jun Li, and Minyi Guo. Manhattan hashing for large-scale image retrieval. In Proc of SIGIR, pages 45–54, 2012.

[60] Holger Kreft and Walter Jetz. A framework for delineating biogeographical regions based on species distributions. Journal of Biogeography, 37:2029—-2053, 2010.

[61] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Proc of NIPS, 2012.

[62] Brian Kulis and Trevor Darrell. Learning to hash with binary reconstructive embeddings. In NIPS, pages 1042–1050, 2009.

[63] Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. In ICML. Omnipress, 2012.

[64] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan. Simultaneous feature learning and hash coding with deep neural networks. In CVPR, 2015.

[65] Andrew B Lawson. Hierarchical modeling in spatial epidemiology. Wiley Interdisciplinary Reviews: Computational Statistics, 6(6):405–417, 2014.

[66] Pierre Legendre and Louis Legendre. Numerical ecology. Amsterdam, 2012.

[67] Yan Li, Zhou Shi, Feng Li, and Hong-Yi Li. Delineation of site-specific management zones using fuzzy clustering analysis in a coastal saline land. Computers and Electronics in Agriculture, 56:174–186, 2007.

[68] Moshe Lichman. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2013.

[69] Roderick J A Little. Regression with missing x's: A review. Journal of the American Statistical Association, 87:1227–1237, 1992.

[70] Roderick J A Little and Donald B Rubin. Statistical Analysis with Missing Data. John Wiley & Sons, Inc., 1986.

[71] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient l2, 1-norm minimization. In Proc of Conf. on Uncertainty in Artificial Intelligence, pages 339–348, 2009.

[72] Ye Liu, Yu Zheng, Yuxuan Liang, Shuming Liu, and David S. Rosenblum. Urban water quality prediction based on multi-task multi-view learning. In Proc. of the 25th Int'l Joint Conference on Artificial Intelligence, pages 2576–2582, 2016.

[73] Jed A. Long, Trisalyn A. Nelson, and Michael A. Wulder. Regionalization of landscape pattern indices using multivariate cluster analysis. Journal of Environmental Management, pages 134–142, 2010.

[74] Aurelie C. Lozano and Grzegorz Swirszcz. Multi-level lasso for sparse multi-task regression. In Proc of Int'l Conf on Machine Learning, 2012.

[75] Ulrike Luxburg. A tutorial on spectral clustering. Statistics and Computing, 17(4):395–416, 2007.

[76] Christopher R. Margules, Daniel P. Faith, and Lee Belbin. An adjacency constraint in agglomerative hierarchical classifications of geographic data. Environment and Planning A, 17(3):397–412, 1985.

[77] Edward McCauley and Susan Downing, John A.and Watson. Sigmoid relationships between nutrients and chlorophyll among lakes. Can J Fish Aquat Sci, 46:1171–1175, 1989.

[78] Gerard McMahon, S.M. Gregonis, S.W. Waltman, J.M. Omernik, T.D. Thorson, J.A. Freeouf, A.H. Rorick, and J.E. Keys. Developing a spatial framework of common ecological regions for the conterminous united states. Environmental Management, 28(3):293–316, 2001.

[79] Kristin Meseck and et.al. Is missing geographic positioning system data in accelerometry studies a problem, and is imputation the solution? Geospatial Health, 11(034), 2016.

[80] Vincent Miele, Franck Picard, and Staphane Dray. Spatially constrained clustering on ecological networks. Methods in Ecology Evolution, 5(8):771–779, 2014.

[81] Jennifer Miller and Janet Franklin. Modeling the distribution of four vegetation alliances using generalized linear models and classification trees with spatial dependence. Ecological Modelling, 157(2):227–247, 2002.

[82] Richard Tran Mills, Forrest M. Hoffman, Jitendra Kumar, and William W. Hargrove. Cluster analysis-based approaches for geospatiotemporal data mining of massive data sets for identification of forest threats. In Proceedings of the International Conference on Computational Science, pages 1612–1621, 2011.

[83] Fionn Murtagh. A survey of algorithms for contiguity-constrained clustering and related problems. The computer journal, 28(1):82–88, 1985.

[84] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In Advances in Neural Information Processing Systems, pages 849–856, 2001.

[85] Mohammad Norouzi and David J. Fleet. Minimal loss hashing for compact binary codes. In ICML, pages 353–360, 2011.

[86] Vladimir Novotny, Alena Bartosova, and Nealand Ehlinger Timothy O'Reilly. Unlocking the relationship of biotic integrity of impaired waters to anthropogenic stresses. Water Research, 39:184–198, 2005.

[87] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature, 381(6583):607, 1996.

[88] James M Omernik. Ecoregions: A spatial framework for environmental management. In W. S. Davis and T. P. Simon, editors, Biological assessment and criteria: tools for water resource planning and decision making, pages 49–62. Lewis Publishers, Boca Raton, Florida, 1995.

[89] Stan Openshaw. A regionalisation program for large data sets. Computer Applications, 136:47, 1973.

[90] Karl Pearson. On lines and planes of closest fit to systems of points in space. Phil. Mag., 6:559–572, 1901.

[91] Debra P. C. Peters, Roger A. Pielke Sr., Brandon T. Bestelmeyer, Craig D. Allen, Stuart Munson-McGee, and Kris M. Havstad. Cross-scale interactions, nonlinearities, and forecasting catastrophic events. Proc National Academy of Science, 101(42):15130––15135, 2004.

[92] Maxim Raginsky and Svetlana Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In NIPS, pages 1509–1517, 2009.

[93] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In NIPS, pages 1177–1184, 2008.

[94] William M Rand. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association, 66(336):846–850, 1971.

[95] A. Ramachandra Rao and V.V. Srivnivas. Some problems in regionalization of watersheds. Water Resources Systems -Water Availability and Global Change, 280:301–308, 2003.

[96] Anthony Recchia et al. Contiguity-constrained hierarchical agglomerative clustering using sas. J. Stat. Softw, 33, 2010.

[97] Sam Roweis. Em algorithms for pca and spca. In NIPS, pages 626–632, 1998.

[98] Leonard Sandin and Richard K. Johnson. Ecoregions and benthic macroinvertebrate assemblages of swedish streams. Journal of the North American Benthological Society, 19:462–474, 2000.

[99] Sergio M. Savaresi and Daniel L. Boley. A comparative analysis on the bisecting k-means and the PDDP clustering algorithms. Intelligent Data Analysis, 8(4):345–362, 2004.

[100] William Robson Schwartz and Hélio Pedrini. Texture classification based on spatial dependence features using co-occurrence matrices and markov random fields. In 2004 International Conference on Image Processing, volume 1, pages 239–242. IEEE, 2004.

[101] Paul R. Seaber, F.Paul Kapinos, and George L. Knapp. Hydrologic unit maps. U.S. Geological survey water-supply papers, 1987.

[102] Richard R. Shaker and Timothy J. Ehlinger. Exploring non-linear relationships between landscape and aquatic ecological condition in southern wisconsin: A GWR and ANN approach. International Journal of Applied Geospatial Research, 5(4):1–20, 2014.

[103] Shashi Shekhar, Pusheng Zhang, and Yan Huang. Spatial Data Mining, pages 833–851. Springer US, 2005.

[104] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):888–905, 1997.

[105] Xiaoxiao Shi, Wei Fan, and Philip S. Yu. Efficient semi-supervised spectral co-clustering with constraints. In Proc of IEEE Int'l Conf on Data Mining, pages 1043–1048. IEEE Computer Society, 2010.

[106] PHA Sneath and RR Sokal. Numerical taxonomy: the principles and practice of numerical classification, 1973.

[107] Tom A.B. Snijders and Roel J. Bosker. Multilevel Analysis: An Introduction to Basic and Advanced Multilevel Modeling. SAGE Publications Ltd, 2012.

[108] Robert R Sokal. A statistical method for evaluating systematic relationships. Univ Kans Sci Bull, 38:1409–1438, 1958.

[109] Patricia A Soranno, Edward G Bissell, Kendra S Cheruvelil, Samuel T Christel, Sarah M Collins, C Emi Fergus, Christopher T Filstrup, Jean-Francois Lapierre, Noah R Lottig, Samantha K Oliver, et al. Building a multi-scaled geospatial temporal ecology database from disparate data sources: fostering open science and data reuse. GigaScience, 4(1):28, 2015.

[110] Patricia A Soranno et al. Cross-scale interactions: quantifying multi-scaled cause–effect relationships in macrosystems. Frontiers in Ecology and the Environment, 12(1):65–73, 2014.

[111] Thorvald Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. Biol. Skr., 5:1–34, 1948.

[112] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. Introduction to data mining, First Edition. Addison-Wesley Longman Publishing Co., Inc., 2005.

[113] Jean-Claude Thil and Aaron Wheeler. Tree induction of spatial choice behavior. Transportation Research Record, 1719:250–258, 2000.

[114] Waldo R Tobler. Cellular geography. In Philosophy in geography, pages 379–386. Springer, 1979.

[115] Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. Pacific J. of Optimization, 2009.

[116] Athanasios Tsanas, Max A. Little, Patrick E. McSharry, and Lorraine O. Ramig. Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests. IEEE Trans. Biomed. Engr., 57:884–893, 2010.

[117] U.S. EPA's National Geospatial Data Policy. `http://www.epa.gov/geospatial`, 2005.

[118] Vladimir N. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag New York, Inc., 1995.

[119] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. BMC Bioinformatics, 7:91, 2006.

[120] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. Annals of Statistics, 36(2):555–586, 2008.

[121] Ulrike Von Luxburg, Olivier Bousquet, and Mikhail Belkin. Limits of spectral clustering. In Advances in Neural Information Processing Systems, pages 857–864, 2004.

[122] Tyler Wagner, C Emi Fergus, Craig A Stow, Kendra S Cheruvelil, and Patricia A Soranno. The statistical power to detect cross-scale interactions at macroscales. Ecosphere, 7(7), 2016.

[123] Tyler Wagner, Daniel B Hayes, and Mary T Bremigan. Accounting for multilevel data structures in fisheries data using mixed models. Fisheries, 31(4):180–187, 2006.

[124] Tyler Wagner, Patricia A Soranno, Katherine E Webster, and Kendra Spence Cheruvelil. Landscape drivers of regional variation in the relationship between total phosphorus and chlorophyll in lakes. Freshwater Biology, 56(9):1811–1824, 2011.

[125] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In Proc of IEEE Int'l Conf on Machine Learning, pages 577–584. Morgan Kaufmann, 2001.

[126] Jingdong Wang, Heng Tao Shen, Jingkuan Song, and Jianqiu Ji. Hashing for similarity search: A survey. CoRR, abs/1408.2927, 2014.

[127] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for large scale search. TPAMI, 34:2393–2406, 2012.

[128] Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 563–572. ACM, 2010.

[129] Xiaogang Wang, Cha Zhang, and Zhengyou Zhang. Boosted multi-task learning for face verification with applications to web image and video search. In CVPR, pages 142–149, 2009.

[130] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. Journal of the American statistical association, 58:236–244, 1963.

[131] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In Advances in neural information processing systems, pages 1753–1760, 2009.

[132] Michael R. Willig, Christopher P. Bloch, Nicholas Brokaw, Christopher Higgins, Jill Thompson, and Craig R. Zimmermann. Cross-scale responses of biodiversity to hurricane and anthropogenic disturbance in a tropical forest. Ecosystems, 10:824–838, 2007.

[133] Michael C Wimberly, Adam D Baer, and Michael J Yabsley. Enhanced spatial models for predicting the geographic distributions of tick-borne pathogens. International Journal of Health Geographics, 7(1):15, 2008.

[134] David M. Wolock, Thomas C. Winter, and Gerard McMahon. Delineation and evaluation of hydrologic-landscape regions in the united states using geographic information system tools and multivariate statistical analyses. Environmental Management, 34:S71–S88, 2004.

[135] Jinguo Wu. Hierarchy theory: An overview. Linking Ecology and Ethics for a Changing World, 1:281–301, 2013.

[136] Xiaolan Wu and Alan T Murray. A new approach to quantifying spatial contiguity using graph theory and spatial interaction. International Journal of Geographical Information Science, 22(4):387–407, 2008.

[137] Jianpeng Xu, Pang-Ning Tan, and Lifeng Luo. ORION: online regularized multi-task regression and its application to ensemble forecasting. In 2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014, pages 1061–1066, 2014.

[138] Jianpeng Xu, Pang-Ning Tan, Lifeng Luo, and Jiayu Zhou. Gspartan: a geospatio-temporal multi-task learning framework for multi-location prediction. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016, pages 657–665, 2016.

[139] Jianpeng Xu, Jiayu Zhou, Pang-Ning Tan, Xi Liu, and Lifeng Luo. WISDOM: weighted incremental spatio-temporal multi-task learning via tensor decomposition. In Proc of the IEEE International Conference on Big Data, 2016.

[140] Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. J. Mach. Learn. Res., 8:35–63, 2007.

[141] I-Cheng Yeh. Modeling of strength of high performance concrete using artificial neural networks. Cement and Concrete Research, 28:1797–1808, 1998.

[142] Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In Proceedings of the 22Nd International Conference on Machine Learning, pages 1012–1019, 2005.

[143] Kai Yu and Tong Zhang. Improved local coordinate coding using local tangents. In ICML, pages 1215–1222, 2010.

[144] Shuai Yuan. Code available at. `https://github.com/shuaiyuan-msu/csi-mlmt`, 2017.

[145] Shuai Yuan, Pang-Ning Tan, Kendra S Cheruvelil, C Emi Fergus, Nicholas K Skaff, and Patricia A Soranno. Hash-based feature learning for incomplete continuous-valued data. In Proceedings of the 2017 SIAM International Conference on Data Mining, pages 678–686. SIAM, 2017.

[146] Shuai Yuan, Pang Ning Tan, Kendra Spence Cheruvelil, Sarah M. Collins, and Patricia A. Soranno. Constrained spectral clustering for regionalization: Exploring the trade-off between spatial contiguity and landscape homogeneity. In Proceedings of the IEEE International Conference on Data Science and Advanced Analytics, 12 2015.

[147] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Rob Fergus. Deconvolutional networks. In CVPR, 2010.

[148] Liang Zhao, Qian Sun, Jieping Ye, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. Multi-task learning for spatio-temporal event forecasting. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1503–1512, 2015.

[149] Liang Zhao, Jieping Ye, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. Hierarchical incomplete multi-source feature learning for spatiotemporal event forecasting. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 2085–2094, 2016.

[150] J. Zhou, J. Chen, and J. Ye. MALSAR: Multi-tAsk Learning via StructurAl Regularization. Arizona State University, 2011.

[151] Jiayu Zhou, Lei Yuan, Jun Liu, and Jieping Ye. A multi-task learning formulation for predicting disease progression. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 814–822, 2011.