Abnormal Pattern Recognition in Spatial Data

by

Yufeng Kou

Dissertation submitted to the Faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Applications

Advisory Committee:
Dr. Chang-Tien Lu, Chair
Dr. Roger W. Ehrich
Dr. Mohamed Eltoweissy
Dr. Thomas Grizzard
Dr. Yao Liang
Dr. Ing-Ray Chen

Date: November 29th, 2006
Falls Church, Virginia

Keywords: spatial data mining, spatial outlier detection, pattern recognition, change detection, image segmentation, similarity search

Abnormal Pattern Recognition in Spatial Data

Yufeng Kou

Abstract

In the recent years, abnormal spatial pattern recognition has received a great deal of attention from both industry and academia, and has become an important branch of data mining. Abnormal spatial patterns, or spatial outliers, are those observations whose characteristics are markedly different from their spatial neighbors. The identification of spatial outliers can be used to reveal hidden but valuable knowledge in many applications. For example, it can help locate extreme meteorological events such as tornadoes and hurricanes, identify aberrant genes or tumor cells, discover highway traffic congestion points, pinpoint military targets in satellite images, determine possible locations of oil reservoirs, and detect water pollution incidents.

Numerous traditional outlier detection methods have been developed, but they cannot be directly applied to spatial data in order to extract abnormal patterns. Traditional outlier detection mainly focuses on "global comparison" and identifies deviations from the remainder of the entire data set. In contrast, spatial outlier detection concentrates on discovering neighborhood instabilities that break the spatial continuity. In recent years, a number of techniques have been proposed for spatial outlier detection. However, they have the following limitations. First, most of them focus primarily on single-attribute outlier detection. Second, they may not accurately locate outliers when multiple outliers exist in a cluster and correlate with each other. Third, the existing algorithms tend to abstract spatial objects as isolated points and do not consider their geometrical and topological properties, which may lead to inexact results.

This dissertation reports a study of the problem of abnormal spatial pattern recognition, and proposes a suite of novel algorithms. Contributions include: (1) formal definitions of various spatial outliers, including single-attribute outliers, multi-attribute outliers, and region outliers; (2) a set of algorithms for the accurate detection of single-attribute spatial outliers; (3) a systematic approach to identifying and tracking region outliers in continuous meteorological data sequences; (4) a novel Mahalanobis-distance-based algorithm to detect outliers with multiple attributes; (5) a set of graph-based algorithms to identify point outliers and region outliers; and (6) extensive analysis of experiments on several spatial data sets (e.g., West Nile virus data and NOAA meteorological data) to evaluate the effectiveness and efficiency of the proposed algorithms.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Dr. Chang-Tien Lu, for his support and guidance during my Ph.D. study. It is my great pleasure to be his first Ph.D. student. From Dr. Lu, I have learned advanced methodology to conduct rigorous research, strong sense of responsibility to be a serious scientist, and firm persistence to seek the truth.

I am very thankful to all the committee members. Without their guidance, I would never have been able to finish my dissertation. Thanks Dr. Grizzard for offering financial support for more than two years. Thanks Dr. Ehrich, Dr. Liang, and Dr. Eltoweissy for providing insightful comments and valuable suggestions from my proposal to final defense. Special thanks goes to Dr. Chen, who was willing to participate in my final defense committee at the last moment.

I would also like to thank the kind help from my collaborators, Dr. Li Chen, Dr. Dechang Chen, Dr. Adil Godrej, and Dr. Lily Liang. Dr. Li Chen and Dr. Dechang Chen rendered tremendous help in building mathematic models and provided significant suggestions on experiment design. Dr. Godrej guided my OWML project on both software functionalities and user interface design. Dr. Liang offered summer support and introduced me to the interesting project of biological data analysis.

I would like to express appreciation to my friends in the Spatial Data Management Laboratory, Jing Dai, Feng Chen, Ying Jin, Bing Liu, Arnold Boedijardjo, Edward Devilliers, Ray Dos Santos, Wendell Jordan-Brangman, Manu Shukla, and Chad Steel. Many thanks for their precious comments on my dissertation. Each discussion with them sparked new thoughts in my research. They made my Ph.D. study an enjoyable journey with a lot of happy memory.

Finally, I would dedicate this dissertation to my parents and my elder brother. Without their support, I could not have completed fundamental education in China, not mention to going abroad and pursuing advanced study in Virginia Tech.

Contents

List of Tables vii

List of Figures ix

1 Introduction 1

1.1 Traditional Outlier Detection Methods . . . 2

1.2 Abnormal Pattern Recognition in Spatial Data . . . 4

1.2.1 Spatial Data . . . 4

1.2.2 Spatial Outlier Detection . . . 5

1.3 Image Segmentation and Wavelet Analysis . . . 10

1.3.1 Image Segmentation . . . 10

1.3.2 Wavelet Analysis . . . 12

1.4 Contributions . . . 14

1.5 Organization of the Dissertation . . . 16

2 Single Attribute Outlier Detection 17

2.1 Spatial Outlier Detection . . . 17

2.1.1 Problem Formulation . . . 17

2.1.2 Deficiency of the Existing Approaches . . . 19

2.1.3 Detection Algorithms . . . 21

2.1.4 Computational Complexity . . . 25

2.1.5 Experiments . . . 26

2.2 Weighted Spatial Outlier Detection . . . 34

2.2.1 Motivation . . . 34

2.2.2 Problem Formulation . . . 35

2.2.3 Algorithms . . . 36
2.2.4 Experiments . . . 40
2.3 Summary . . . 45

3 Region Outlier and Change Detection 47
3.1 Region Outlier Detection . . . 47
3.1.1 Problem and Approach . . . 48
3.1.2 Algorithm Design . . . 52
3.1.3 Experiments . . . 60
3.1.4 Discussion . . . 71
3.2 Change Detection . . . 71
3.2.1 Problem and Approach . . . 72
3.2.2 Experimental Analysis . . . 75
3.2.3 Summary . . . 76

4 Multiple Attribute Outlier Detection 82
4.1 Motivation and Problem Formulation . . . 82
4.2 Detection Algorithms . . . 83
4.3 Computational Complexity . . . 86
4.4 Experiments . . . 87
4.4.1 Analysis of Census Data . . . 87
4.4.2 Analysis of West Nile Virus Data . . . 89

5 Graph-based Outlier Detection 94
5.1 Motivation and Problem Formulation . . . 94
5.2 Detection Algorithms . . . 97
5.2.1 Algorithm 1: Point Outlier Detection . . . 100
5.2.2 Algorithm 2: Region Outlier Detection . . . 101
5.2.3 Time Complexity . . . 103
5.3 Experiments . . . 104
5.3.1 Experiment Design . . . 104
5.3.2 Result Analysis . . . 104
5.4 Summary . . . 107

6 Conclusion and Future Work **109**

6.1 Contributions **109**

6.1.1 Single Attribute Outlier Detection 109

6.1.2 Multi-attribute Outlier Detection 111

6.1.3 Region Outlier Detection 112

6.1.4 Outlier Detection in Continuous Data Sequences 113

6.1.5 Related Surveys and Demo Systems 113

6.2 Future Work **113**

Bibliography **114**

List of Tables

2.1 Top three spatial outliers detected by z, iterative z, iterative r, median, scatterplot, and Moran scatterplot algorithms. 25
2.2 The top five spatial outlier candidates detected by z, iterative z, iterative r, and median algorithms in Indiana. 27
2.3 The top ten spatial outlier candidates detected by z, iterative z, iterative r, and Median algorithms. 30
2.4 Top six spatial outlier candidates detected by z, iterative z, and median algorithms. . . 32
2.5 The top 30 spatial outlier candidates detected by z, weighted z, and $AvgDiff$ algorithms. 41
2.6 The common border length, center distance, and weight of the 8 neighbors of York Co. (PA) 43
2.7 The common border length, center distance, and weight of the 4 neighbors of Camden Co. (NJ) 44
2.8 The time complexity comparison between Median algorithm, Iterative-z, iterative-r, weighted z, and $AvgDiff$ algorithms. 46

3.1 Scale Table for Mexican Hat Wavelet 53
3.2 Scale Table for Morlet Wavelet 54
3.3 The tracking data of hurricane center. 64
3.4 The execution time of image segmentation. 65
3.5 Running time of K-Means. 69
3.6 Running time of Maximum Entropy. 70
3.7 Running time of $\lambda - connectedness$. 70

4.1 The top 10 spatial outlier candidates detected by the Mean algorithm and their associated standardized attribute values 88

4.2 The top 10 spatial outlier candidates detected by the Median algorithm and their associated standardized attribute values 88
4.3 Top 18 spatial outlier candidates, detected by the Median single-attribute algorithm, and their original/standardized attribute values. 91
4.4 Top 162 spatial outlier candidates, detected by the multiple attribute algorithm, and their associated standardized attribute values. The rank is based on the magnitude of the Mahalanobis distance. 92

5.1 Top three spatial outliers detected by scatterplot, Moran scatterplot, z-value, and graph-based method. 95
5.2 Top 10 spatial outliers detected by z-value, scatterplot, Moran scatterplot, and graph-partition algorithms based on 1-bedroom rent. 105
5.3 2 region outliers detected by ROD Alg. based on two-bedroom rent data in 2005. . . . 106

6.1 The time complexity comparison between Median algorithm, Iterative-based algorithms, spatial weighted algorithms, and graph-based algorithms. 111

List of Figures

1.1 Tiling of time-frequency space by (a) Short time Fourier transformation, (b) Wavelet transformation 13

2.1 A spatial data set. Objects are located in the $X-Y$ plane. The height of each vertical line segment represents the attribute value of the corresponding object. 19

2.2 Graphical illustration of the z algorithm. Data are shown in Figure 2.1. The height of each vertical line segment indicates the absolute value of the z-score. $S1$, $E1$, and $E2$ would be identified as possible spatial outliers since their absolute z-score values are the largest. 20

2.3 Moran scatterplot used to detect the spatial outliers of the data in Figure 2.1. Here z scores represent the normalized attribute values. The three best candidates of outliers are $S1$, $E1$, and $E2$, located in the upper left and lower right quadrants. 21

2.4 Scatterplot used to analyze the data in Figure 1. $E1$, $E2$, and $S2$ are the three best spatial outlier candidates since their distances to the regression line are the largest. . . . 22

2.5 Population Density of Marion Co., Shelby Co., Hancock Co., and their Neighboring Counties. 27

2.6 Population Density of Allen Co., St. Joseph Co, and their Neighboring Counties. 28

2.7 Population Density of Tippecanoe Co., Vanderburgh Co., and their Neighboring Counties. 29

2.8 US Population at the County Level in year 2001 29

2.9 Cases of dead birds infected by WNV across all counties in the United States in 2002 . 31

2.10 Distribution of the values of attribute Bird-2002 in the vicinity of Allegheny, Bucks, and Montgomery. The numbers represent the values for Bird-2002. 33

2.11 Distribution of the values of attribute Bird-2002 in the vicinity of Berks, Lancaster, and Westmoreland. The numbers represent the values for Bird-2002. 33

2.12 Distribution of the values of attribute Bird-2002 in the vicinity of Dauphin and Armstrong. The numbers represent the values for Bird-2002. 34
2.13 The 2003-Vet WNV case density for Hot Spring Co.(WY) and its neighbors 42
2.14 The 2003-Vet WNV case density for York Co.(PA) and its neighbors 43
2.15 The 2003-Vet WNV case density for Camden Co.(NJ) and its neighbors 44
2.16 The 2003-Vet WNV case density for Anne Arundel Co. (MD) and its neighbors 45

3.1 A regional outlier (hurricane) in meteorological data. 49
3.2 A sample output of the Mexican hat wavelet (a: original water vapor data, b: wavelet power at scale 2, c: wavelet power at scale 3). 54
3.3 A sample output of Morlet wavelet (a: original water vapor data, b: wavelet power at scale 6, c: wavelet power at scale 7). 55
3.4 Global distribution of water vapor. 62
3.5 Mexican hat wavelet power with locations and scales (a: original and filtered data, b: wavelet for original data at all scales, c: wavelet for filtered data at all scales). 63
3.6 Wavelet power distribution at scale index 3. 64
3.7 Reconstruction on both X Y dimension. 65
3.8 Water vapor data at 0AM Sept. 18th, 2003 with Hurricane Isabel identified. 66
3.9 Water vapor data at 6AM Sept. 18th, 2003 with Hurricane Isabel identified. 67
3.10 Trajectory of moving region with "noise" data. 68
3.11 Trajectory of moving region without "noise" data. 69
3.12 Two source images for image segmentation comparison. 70
3.13 The effectiveness of K-$Means$ image segmentation 77
3.14 Maximum Entropy image segmentation: single threshold. 78
3.15 Maximum Entropy image segmentation: single threshold. 78
3.16 Maximum Entropy image segmentation: two-threshold. 78
3.17 λ-connectedness image segmentation. 79
3.18 Spatial wavelet power distribution at scale index 3 on day 1. 79
3.19 Spatial wavelet power distribution at scale index 3 on day 2. 80
3.20 Temporal wavelet power distribution across latitude and date. 80
3.21 Distribution of high temporal wavelet power (threshold=35) across longitude, latitude, and date. 81

4.1 Experimental design for multiple attribute outlier detection. 90
4.2 Distribution of the values of attribute vet-2001 in the vicinity of Marion County. 93

5.1 An example data set, where X and Y coordinates denote the spatial locations and Z coordinate represents the value of nonspatial attribute. 95
5.2 The Scatterplot and Moran-scatterplot of the data set shown in Figure 5.1. 96
5.3 The kNN based graph representation of a small data set, $k = 3$. 98
5.4 The result of graph-based outlier detection: 2 point outliers and 1 region outlier are identified. 98
5.5 The data structure for graph representation. 99
5.6 Dorchester Co.(MD) is detected by Moran-scatterplot algorithm based on one-bedroom rent data in 2005. 105
5.7 Dukes Co.(MA) identified by POD algorithm based on one-bedroom rent data in 2005. 106
5.8 A region outlier detected by ROD algorithm based on two-bedroom rent data in 2005. 107
5.9 A region outlier detected by ROD algorithm based on two-bedroom rent data in 2005. 108

Chapter 1

Introduction

In recent years, our society has experienced a data explosion. Automated data collection tools and mature database technology are now used to collect tremendous amounts of data, far outstripping our ability to analyze and interpret them. These data are in various formats, including unstructured texts on the web, digital images for medical diagnosis, video streams from surveillance cameras, and gene sequences from biological laboratories. However, these raw data cannot provide useful information unless they are turned into significant insights or knowledge to help people make important decisions.

Data mining is a process used to dig out useful "nuggets of information" from large amounts of data stored either in databases, data warehouses, or other information repositories [36]. These "nuggets" can be used to identify the patterns that occur frequently, illustrate the interesting associations among different patterns, and classify data items based on their characteristics. Besides extracting general rules and patterns, an equally important task of data mining is to identify abnormal patterns (outliers), which were often ignored or discarded as noise.

"One person's noise is another person's signal." Rare events often reveal more important information than common ones, especially in case of the computer security, emergency responses, and credit card fraud detection. For example, a credit card company is expected to handle millions of transactions from users, but in order to identify fraud, it is necessary to be able to identify the exceptional credit card usage, which are exceptional in shopping address, expenditure amount and purchase frequency. Therefore, abnormal pattern recognition, or outlier detection, has attracted a great deal of attention from researchers and engineers and has become an indispensable branch of data mining.

Although there is no consensus on how to describe outliers, Barnet's definition is accepted by

many statisticians and computer scientists, describing an outlier as "one observation that appears to deviate markedly from other members of the sample in which it occurs" [7]. In the recent years, outlier detection has begun to be widely used in numerous applications. In different applications, outliers may have different names such as anomalies, deviations, exceptions, faults and irregularities. Outlier detection can help identify intrusions in computer networks [109], locate malfunctioning parts in a manufacture streamline [19], pinpoint suspicious usage of credit cards [11], and monitor unusual changes of stock prices [13].

In the following sections, we classify the outlier detection problem into two types: traditional outlier detection, which deals with transaction data, and spatial outlier detection, which focuses on spatial data.

1.1 Traditional Outlier Detection Methods

According to Breunig's classification [9], the existing traditional outlier detection algorithms can be categorized into four categories: clustering-based, distribution-based, depth-based, and distance-based. Outliers emerge as a byproduct of clustering, thus a few clustering algorithms can be used to detect outliers. These algorithms identify objects as outliers if they are far away from any cluster [25, 73, 112]. Since these algorithms are aimed to extract clusters, their efficiency and effectiveness may not be optimized for outlier detection.

Distribution-based methods use a standard distribution such as Normal or Gaussian to measure the data set and the data points deviating from this distribution are defined as outliers [106]. If the distribution of a data set is known these methods can be efficient and accurate, but in most cases the exact distribution of a data set is unknown beforehand and it is both costly and difficult to derive the distribution. Furthermore, many data sets may not exactly follow a standard distribution. Depth-based methods organize the data in different layers of k-d convex hulls where data in the outer layers tend to be outliers [79, 85]. However, these methods are not widely used due to their high computation cost for multi-dimensional data.

Distance-based methods may be the most widely used techniques which define an outlier as a data point having an exceptionally far distance to the other data points [45, 80]. Formally, for a k dimensional data set T with N objects, an object O in T is a $DB(p, D)$-outlier if at least a fraction p of the objects in T lies at greater than a distance D from O. Ramaswamy *et al.* proposed a formulation for distance-based outliers by calculating the distance of a point from its

k^{th} nearest neighbor [80]. After ranking points based on the distance to their k^{th} nearest neighbors, the top n points are declared as outliers. Distance-based methods are easy to understand, simple to implement, and do not need to know the distribution of a data set. They work well under certain conditions, but are not satisfactory when there are multiple clusters with different densities. The reason is that distance-based methods are a "global" method, which define outliers by considering distances in the whole data set.

Breunig *et al.* proposed a "local density" based method to address the defects in the distance-based methods [9]. The local density of an object is defined based on its k-distance neighborhood. For a given point, the lower its local density is, the higher its probability of being an outlier. Another advantage of this approach is that it provides a degree of outlierness that is different from the binary results of other methods, which simply classify objects as "normal" or "outlying." Jin *et al.* extends Breunig's algorithm to detect the top n local outliers without the need to first compute local densities for all the data elements. Since the number of outliers n is much smaller than the size of the data set N, this variant algorithm can significantly reduce the computation complexity [42]. Note, the term "local" here denotes the neighborhood relationship determined by both spatial and non-spatial attributes, whereas in spatial outlier detection, the term "local" means that the neighborhood is defined only by spatial attributes, i.e., the spatial neighborhood.

The above methods for detecting outliers focus on the case of a single attribute. For detecting outliers with multiple attributes, classic outlier detection approaches are ineffective due to the "curse of high dimensionality," *i.e.*, the sparsity of the data objects in high dimensional space [5]. It has been shown that the distance between any pair of data points in high dimensional space is so similar that either every data point or none of the data points can be viewed as outliers if the concept of proximity is used to define outliers [2]. As a result, the traditional Euclidean distance cannot be used to effectively detect outliers in high dimensional data sets. Two categories of research work have been conducted in order to address this issue. The first is to project high dimensional data to low dimensional data [4, 5, 8, 39] and the second is to re-design distance functions to accurately define the proximity relationship between data points [2].

In addition to processing static data sets, outlier detection has been extended to identify outliers in dynamic data sequences. For example, Yamanishi *et al.* presented a unifying framework for processing both outlier detection and change detection [107]. In this framework, a probabilistic model of the data source evolves over time based on an on-line discounting learning algorithm, which can track the changing data by gradually "forgetting" the effect of past data. However, since

this method does not consider spatial relationships among data objects, it cannot be appropriately used for detecting spatial outliers.

1.2 Abnormal Pattern Recognition in Spatial Data

Many algorithms have been developed to detect abnormal patterns in large data sets. However, most of them are designed to process traditional transaction data and are not effective for handling spatial data. Thus, it is an important and challenging task to study the identification of abnormal patterns in spatial data.

1.2.1 Spatial Data

Spatial data refer to data objects pertaining to geometry and topology [82]. Geometry describes the shape, location, and size of an spatial object, while topology studies the spatial relationships among several objects, such as overlapping, adjacency, and inclusion. In contrast to nonspatial data, which simply deal with numbers and characters, spatial data are more complex and require special operations. First, spatial data are composed of complex structures such as points, lines, regions, and even 3-D objects. Moreover, spatial objects can be treated together to form even more complex objects. Second, due to their complex structures, the size of spatial data sets is much larger than that of nonspatial data sets. For example, thousands of (latitude,longitude) pairs may be required to represent the boundaries of a county. Third, spatial dependance, or spatial correlation, is common in many geospatial applications where the characteristics between neighboring objects tend to be similar. For instance, adjacent areas are likely to have similar temperature and humidity. Fourth, spatial data call for specific mechanisms for storage, indexing, and querying. Traditional database management techniques cannot be directly used for spatial data, since they do not support topological and geometric operations.

Spatial data can be categorized into two groups, macro-spatial data and micro-spatial data. Macro-spatial data are also referred to as geospatial data, as they contain spatial information for geological features. Geospatial data often exist in Geographic Information Systems (GIS), related to the earth environment or man-made architectures such as meteorological images, terrain maps, and mansion design figures. Micro-spatial data refer to those smaller objects whose characteristics are closely tied to their locations and shapes, for example, genetic sequences, Very Large-Scale Integration (VLSI) designs, and medical scan images.

In many applications, spatial data not only relate to geometry and topology, but also contain temporal information. These data are called spatio-temporal data or spatial data sequences. Spatio-temporal data can be meteorological images continuously sent from satellites, real-time water quality data collected by monitoring stations, or traffic flow changes gathered by road detectors. These data contain complex structures, arrive continuously, evolve over time, and need to be processed in real time.

To sift the implicit knowledge out of large amounts of spatial data, efficient and effective data mining methods are in great demand. Over the last twenty years, extensive research in this area has formed a new branch of data mining, known as spatial data mining.

1.2.2 Spatial Outlier Detection

From satellite observation systems to urban planning, geography related spatial data are widely used. Other types of spatial data, such as medical images and gene maps, have received a significant amount of attention from medical professionals and researchers. Due to the ever-increasing volume of spatial data, spatial data mining has become an important research area over the past few years [35, 89]. As defined in [46], spatial data mining is the process of discovering hidden but valuable patterns from large spatial data sets. Spatial data mining techniques can be classified into four categories: classification, clustering, association analysis, and outlier detection.

This dissertation focuses on spatial outlier detection, where the objective is to discover local instabilities which break the spatial correlation and continuity. "A spatial outlier is a spatially referenced object whose non-spatial attribute values are significantly different from those of other spatially referenced objects in its spatial neighborhood" [89]. In contrast to traditional outliers, spatial outliers are local anomalies that are extreme compared to their neighbors [10], but do not necessarily deviate from the remainder of the whole data set. Informally, Spatial outliers can be called "local outliers" because they pay more attention to local differences, while traditional outliers can be called "global outliers" since they focus on global comparisons.

In certain situations, an outlier may not appear as a single spatial object but in the form of a group of adjacent objects, i.e., a region. The spatial objects within the region have similar characteristics, and their characteristics are significantly different from the objects outside the region. We define this type of outlier as a "region outlier". Region outliers can be tornadoes in meteorology, traffic congestions in highway systems, and crude oil reservoirs in geology.

Although numerous traditional outlier detection methods have been developed, they cannot

be directly applied to spatial data to extract abnormal patterns. This is because classical outlier detection is designed to process numerical and categorical data, whereas spatial outlier detection has to handle spatial objects such as points, lines, polygons, and 3D objects. Also, traditional outlier detection does not handle spatial relationships among input variables, while spatial patterns often exhibit continuity and high autocorrelations with nearby samples. As stated by the geographical rule of thumb, "Everything is related to everything else, but nearby things are more related than distant things [97]." For spatial outlier detection, spatial and non-spatial dimensions should be considered separately. The spatial dimension is used to define the neighborhood relationship, while the non-spatial dimension is used to define the distance function.

The identification of spatial outlier detection plays an important role in knowledge discovery and decision-support. It can help locate extreme meteorological events such as tornadoes and hurricanes [62, 113], identify disease outbreaks [103] and tumor cells [78], and discover abnormal highway traffic patterns [92]. In addition, spatial outlier detection can potentially be applied to pinpoint significant military targets in satellite images, determine the locations of possible gas/oil reservoirs, and detect water pollution incidents. In the identification of spatial outliers, attribute space is generally divided into two parts, non-spatial attributes and spatial attributes. Spatial attributes record information related to locations, boundaries, directions, sizes, and volumes, which determine the spatial relationships among neighbors. Based on the neighborhood relationship, non-spatial attributes are then used to identify abnormal observations.

In this section, spatial outlier detection algorithms will be classified into two groups and discussed in turn. The first group identifies outliers in static spatial data. The second group is designed to detect spatio-temporal outliers in continuous data sequences.

Outlier Detection in Static Spatial Data

Numerous spatial outlier detection algorithms have been developed. Several algorithms are based on visualization, which illustrates the distribution of neighborhood differences in a figure and identifies the points in particular portions of the figure as spatial outliers. These methods include variogram clouds, pocket plots, scatterplots and Moran-scatterplots [35, 37, 66, 76]. A Scatterplot [35, 65] visualizes the data in a two dimensional space, where the X-axis denotes non-spatial attribute values and Y-axis represents the average of the non-spatial attribute values in spatial neighborhood. Spatial outliers are those points which are far away from the regression line. A Moran scatterplot [66] presents normalized attribute values against the neighborhood average of normalized attribute val-

ues. A Moran scatterplot consists of four quadrants. The upper left quadrant indicates objects with low values surrounded by neighbors with high values. The lower right quadrant reveals that objects with high values are surrounded by objects with low values. Objects located in these two quadrants are flagged as spatial outliers.

Other algorithms perform statistical tests to discover local inconsistencies. The z-value approach is a typical example [91]. The Z-value is the normalized difference between a spatial object and the average of its spatial neighbors. The absolute z-value is used for the measurement of outlierness. The higher the z-value is, the greater deviation of the object from its neighbors and the more likely the object is to be a spatial outlier. As a variant of the z-value approach, a graph-based spatial outlier detection method was proposed by Shekhar *et al.* [92]. Their methods define the spatial neighborhood based on graph connectivity and define the temporal neighborhood based on time series. In [91], Shekhar *et al.* provided a unified definition of spatial outlier detection and proved that this definition generalizes the existing algorithms such as z-value, Scatterplot, and Moran Scatterplot. In addition, they built cost models for outlier detection procedures and evaluated the I/O efficiency of their approach based on highway traffic data.

Spatial data have various formats and semantics. Thus, many outlier detection algorithms are designed to accommodate the special properties of the given spatial data. Adam *et al.* proposed an algorithm which considers both the spatial relationship and the semantic relationship among neighbors [1]. They refined the definition of neighborhood by grouping similarly behaving objects into one neighborhood, measuring the similarity in terms of both spatial relationships and semantic relationships based on spatial features in the vicinity. Liu and Jezek proposed a method for detecting outliers in an irregularly distributed spatial data set [56], assuming that each object o has 8 neighbors, each of which is located in one octant of o and has the smallest distance to o. The outlierness of o can then be measured by both its spatial interpolation residual and the surface gradient of its neighborhood.

Graph theory plays an important role in data mining, e.g., clustering and outlier detection. Karypis *et al.* proposed a hierarchical clustering method, Chameleon, based on graph partition and merge [44]. One of the key features of Chameleon is that it considers both the relative interconnectivity and relative closeness to model cluster similarity. Since outlier detection is derived from clustering, most of the clustering methods can be used to detect outliers. However, clustering methods focus on locating large clusters and view the single points or small clusters as noise. For example, many clustering methods try to generate balanced partition, which avoids small clusters.

Thus, their input/output and internal similarity evaluation are not optimized for outlier detection and may not guarantee best efficiency and effectiveness. On the contrary, the objective of outlier detection is to find single points or small clusters. Complex comparisons among large clusters are not necessary. Yu *et al.* proposed a graph-based method for numerical and categorical outlier detection [108]. This method calculates the ratio between the open k-walk distance from a point and the close k-walk distance for this point. The large ratio value indicates the large degree of outlierness. In their experiments, the authors claim that this method is more accurate than the density-based method LOF [9]. One limitation of this method is that the similarity graph is a complete graph and may introduce expensive computation cost. In addition, both Karypis' method and Yu's method do not separate the spatial attributes from non-spatial attributes and use the entire attribute space to compute the similarity. Therefore, they can not be directly used for spatial outlier detection.

Outlier detection techniques have been applied to medical fields such as the idenfication of disease outbreaks and tumors. Wong *et al.* proposed a rule-based disease outbreak detection method which can identify specific case groups whose profiles are anomalous relative to their typical profile [103]. In [104], a Bayesian network was employed to produce a baseline distribution based on both the disease case distribution and condition attributes (season, age, etc.). Cases deviating from the baseline distribution are then classified as outliers (disease outbreaks). Prastawa proposed a brain tumor segmentation framework based on outlier detection, which considers samples as tumors if they are more than three standard deviations away from the mean value [78]. In this dissertation, spatial outlier detection methods are applied on the West Nile virus data set collected in the United States, in order to discover unusual disease distribution patterns.

Pattern Recognition in Spatio-temporal Data Sequence

Substantial attention has been devoted to identifying useful patterns and tracking their changes from continuous data sequences. These patterns include clusters, evolutions, deviations, and anomalies.

(1) Clusters: Guha *et al.* proposed a divide-and-conquer approach for continuous data clustering, which divides a data sequence into blocks, clusters each block, and then clusters the centers of all the blocks [33]. Li and Han also explored a clustering technique on moving objects to catch moving patterns of a set of similar data points [54].

(2) Evolutions: By extending an existing spatio-temporal data model, Tripod [32], Djafri *et al.* developed a general approach to characterize the evolution queries in a spatio-temporal database [21]. Aggarwal presented a framework to detect changes and identify useful trends in evolving data sequences [3], while Giannella *et al.* designed an algorithm to maintain frequent patterns under a tilted-time window framework in order to answer time-sensitive queries [29].

(3) Deviations: Palpanas *et al.* utilized kernel density estimators for online deviation detection in continuous data sequences [75]. Their research consisted of two steps. First, a model was constructed based on kernel density estimators to estimate the distribution of values generated by the sensors. Second, the low and high capacity sensors were combined and managed in a hierarchical way to offer a multi-resolution view for identifying deviation.

(4) Anomalies: A neighborhood-based anomaly detection approach was proposed by Adam *et al.* for high dimensional spatio-temporal sensor data streams [1]. They addressed the problem of outlier detection by refining the concept of neighborhood, which groups similarly behaving objects into one neighborhood. A sensor whose reading is inconsistent with other sensors in neighboring regions will be identified as an outlier. At the same time, the period when the anomaly occurs is recorded for change detection. Cheng *et al.* developed a multi-scale approach to detect spatial-temporal outliers by evaluating the change among consecutive spatial and temporal scales [17]. If an object is significantly different from its spatial neighbors or temporal neighbors, it is tagged as a spatial-temporal outlier.

With the explosion in the amount of meteorological data, extensive research has been conducted to assist meteorologists in accurately identifying severe weather events. Several approaches, including fuzzy clustering [6], neural networks [20], genetic algorithms [52], and support vector machine [81, 95], have been proposed to classify storm cells. Peters *et al.* presented a rough-set-based method capable of classifying four types of storm events: hail, heavy rain, tornadoes and wind [77]. However, the existing approaches suffer from several limitations. They require training, mainly support off-line processing instead of continuous data sequence handling, and focus on weather pattern classification but are inadequate for tracking pattern changes. In this dissertation, a systematic approach is proposed to detect abnormal weather events (e.g., hurricanes) and track their movements in the continuous meteorological data sequences.

A number of methods have been proposed for spatial outlier detection [35, 65, 66, 92], whereas they suffer from the following limitations. First, most focus on identifying single (nonspatial) attribute outliers and tend to falsely classify normal objects as spatial outliers if their neighborhoods

contain true spatial outliers. A strong outlier will significantly impact the overall characteristics of its neighborhood and may prevent (or "mask") other nearby outliers from being identified. Second, the existing algorithms tend to abstract spatial objects as isolated points and do not consider spatial relationships over a neighborhood. Neighboring objects are closely correlated based on their shape, area, and common border. In certain situations, adjacent objects should be considered together as one single "big point" if they have similar characteristics. Third, little research has been conducted on outlier detection and tracking in continuous data sequences. Although several studies have been performed to discover useful patterns in data streams [22, 29, 41], they focused on online pattern recognition rather than on processing spatial data.

To address these issues, a suite of algorithms are introduced here for single and multiple attribute spatial outlier detection. The proposed algorithms not only identify "true" outliers that are ignored by existing algorithms but also avoid detecting "false" outliers. In addition, a region outlier detection algorithm is proposed to identify abnormal weather events in continuous meteorological data sequences. A wavelet based approach has also been designed to detect changes in spatio-temporal data sequences. Extensive experiments have been conducted on multiple data sets in order to validate the effectiveness of our proposed algorithms.

1.3 Image Segmentation and Wavelet Analysis

In certain cases, spatial data can be viewed as digital images, where the spatial locations can be represented by the rows and columns and the nonspatial attribute of an object is denoted as the color intensity of the corresponding pixel. Therefore, image segmentation techniques can be applied to group similar data objects and delineate prominent data changes in spatial data. Wavelet transformation, a technique widely used in digital signal processing, can be employed to sharpen the data variations and represent the data with multiple resolutions, offering an indispensable preprocessing technique for image segmentation. This section introduces the state-of-art in the fields of image segmentation and wavelet analysis.

1.3.1 Image Segmentation

Image segmentation is a significant area in digital image processing. Segmentation partitions an image into connected subsets (segments). Each segment is uniform, and no union of adjacent segments is uniform. A region (or segment) in an image may be viewed as a connected group of pixels, all with similar gray scales or color vectors.

Image segmentation partitions an image into different components or objects. This is a key procedure for image preprocessing, object detection, and movement tracking. The existing image segmentation approaches can be categorized into five groups. The **first** and most popular, threshold segmentation, uses a threshold or clip-level to transform a grey-scale image into a binary image. Cheriet *et al.* proposed an approach that explores the use of an optimal threshold for minimizing the ratio of between-segments variance and the total variance [18]. Another approach, called the maximum entropy approach, is to define a threshold based on comparing the entropies of the segmented image [74]. The **second** method, proposed by Rosenfeld, treats an image as a 2D fuzzy set and uses α-cut to develop a fuzzy connectivity [83]. A variation of this fuzzy connectedness is to measure two pixels to evaluate if they are "fuzzy connected." A pixel set is referred to as λ-connected if for any two points there is a path that is λ-connected where λ is a fuzzy value between 0 and 1 [14]. Both threshold segmentation and λ-connected segmentation can be executed in linear time. The **third** method is called split-and-merge segmentation [31], or quad-tree segmentation. This method splits an image into four blocks or parts and checks if each part is homogenous. If not, the splitting process will be repeated; otherwise a merging process will be performed. This method is accurate for complex image segmentation, but is complicated to implement and costs more computation time ($O(nlogn)$). The **fourth** category is related to the K-means or fuzzy c-means. This is a standard classification method that is often applied in image segmentation [16]. Here, the pixels are classified into different clusters to reach a minimum total "error," where the "error" refers to the distance from a pixel to the center of its own cluster. This method may produce very convincing results. Nevertheless, it employs an iterative process to reach convergence. The **fifth** method, the Mumford-Shah method, uses the variational principal [72]. This method considers three factors in segmentation: the length of the edges of all segments, the unevenness of the image without edges, and the error between the original image and the segmented images. When the three weighted factors reach a minimum, this iterative segmentation process stops. Chan and Vese employed level-sets to confine the search of segment edges based on contour boundaries [12]. Their approach is more efficient than the Mumford-Shah method, although level-sets may limit its reflexibility compared to the original method.

The theoretical analysis of segmentation algorithms in which the iteration processes are involved has had significant development in recent years. Researchers started to view the algorithms from different angles. For instance, the performance focuses not only on the segmentation details but also on the time cost. The state-of-the-art results for these segmentation algorithms are listed below: (1)

For the K-means and the fuzzy c-means, researchers already have some good algorithms. Kanungo *et al.* improved Lloyd's K-means clustering algorithm. Their algorithm has the time complexity of at least $O(nlogn)$ since a kd-tree is required [43]. For example, to process a 128x128 image, it needs 14 times as much time as a linear algorithm that only requires one single scan. Runkler, Bezdek, and Hall [84] concluded the algorithm for fuzzy c-means is $O(c^2n)$ where c is the number of classes. When there are 10 segments in the image, the execution time of this algorithm equals the time it takes to scan the image 100 times. The time complexity of the maximum entropy method used to select thresholds was studied by several researchers [15, 30, 55]. The time complexity is $O(n^2)$ for one or two thresholds. Even though this is a very good result, it is far from a log linear algorithm. A nearest neighbor based algorithm for finding the thresholds was proposed in [100], with a time complexity of $O(nlogn)$.

Recently, a new approach called Knowledge-based segmentation has been developed for medical image interpolation [24] and aerial image understanding [111]. Zhang and Hall proposed a knowledge-guided image segmentation method which can automatically segment and analyze previously unseen multi-feature images after training [110]. This approach uses a fuzzy clustering algorithm to extract clusters from easily detectable portions of objects. Meanwhile, cluster labelling rules are generated to form a knowledge base which can guide the segmentation of less detectable objects.

1.3.2 Wavelet Analysis

Wavelet transform, an extension of the conventional Fourier transform, is an ideal tool to divide data, functions, or operators into different frequency components and then study each component with a resolution matched to its scale. Wavelet Analysis has been widely used in various fields, including mathematics, signal processing, and image processing.

Wavelet Transformation versus Short Time Fourier Transformation

Like Fourier transformation, Wavelet transformation can be used to obtain further information which cannot be directly extracted from raw signals. One of the strengths of wavelet transformation is that it supports a time-frequency representation of the signal. Thus, using wavelet transformation, we can know not only what spectral components exist in the signal, but also the locations of these components (time intervals).

Short Time Fourier Transformation (STFT) offer an improvement on Fourier transformation to overcome its limitation of supporting only static data. In STFT, the signal is divided into small segments, and the signal can be assumed to be stationary within these segments (windows) [53]. STFT can provide a time-frequency representation of the signal, but its window size is fixed and may not be suitable for the signals in the entire time range. Another problem of STFT is that it is a nonreversible transformation. Wavelet transformation is proposed to deal with the "fixed window (resolution)" problem of STFT. It uses an approach called multi-resolution analysis (MRA) to present the signal at different frequencies with different resolutions and selects dynamic window sizes for various spectral components. As shown in Figure 1.1 [53], representations based on Fourier analysis offer only a single-scale resolution, since the transformed signal is completely localized in frequency and global in time. However, wavelet analysis is localized in both time and frequency, and is multi-scale in nature, because the resolution changes at different locations in the frequency-time domain.

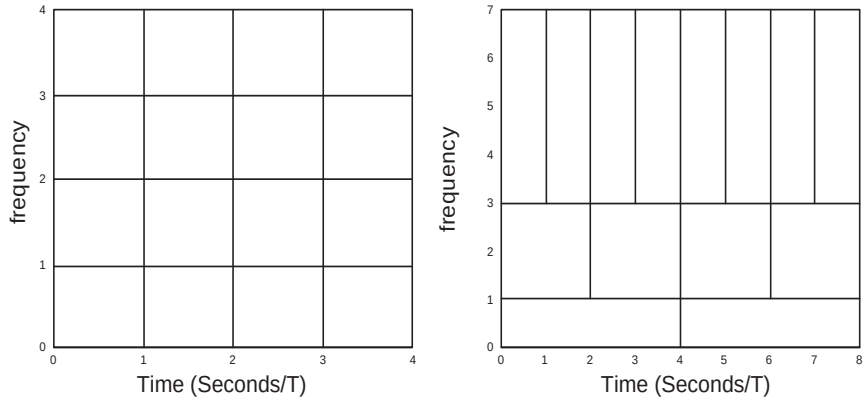


Figure 1.1. Tiling of time-frequency space by (a) Short time Fourier transformation, (b) Wavelet transformation

Wavelet Transformation in Data Analysis

Wavelet transformation provides a natural hierarchical structure to represent data with multiple resolutions, and it therefore can be used as an efficient tool for both data representation and

analysis. TSA-tree is a typical example, where a binary tree structure is built based on the wavelet decomposition [86,87]. A TSA(Trend and Surprise Abstraction) tree is designed to facilitate queries of various time granularities. Like a binary tree, TSA tree is constructed from root to leaf by iteratively decomposing the original signal into two parts: a high frequency part and a low frequency part. The leaf nodes contain all the wavelet coefficients and a subset of the coefficients(leaf nodes) can provide an approximation of the original time sequences. In recent years, the application of wavelet transformation has been extended to many data analysis areas, including clustering [88], similarity search [94], and visualization [70].

Signals contain regular and irregular changes and the irregular changes often carry important information. For example, intensity jumps in digital images usually indicate the edges of an object, while peaks and jumps in stock price curves are often associated with important financial events. Wavelet analysis is widely used in change detection because of its special advantages such as localization in both the time and frequency domains. When time series are analyzed to discover variations, the result depends on the time shift and scale. The time shift represents the location of a change, and the scale tells the period of that change. Mallat *et al.* applied wavelet analysis to detect the isolated and non-isolated singularities in signals [69]. Wang proposed an algorithm to identify jumps and sharp cusps in a signal [101]. The method checks the absolute value of the wavelet transformation at the fine scale levels. If extremely large values exist at higher resolution levels, jumps can be located; if extremely large values exist at lower resolution levels, cusps can be detected. Whitcher *et al.* proposed a wavelet based method to detect and locate multivariance changes in the presence of long range dependencies. Their method can help identify the non-stationary features or change points in a time series [102]. Dong *et al.* proposed an efficient approach to mining emerging patterns, which are item sets whose support significantly increases from one data set to another data set [23]. A comprehensive framework for measuring changes in data characteristics was provided by Ganti *et al.* [28].

1.4 Contributions

This dissertation gives a comprehensive study of detecting various outliers in spatial data. This paper has two objectives: (1) extend the current algorithms to improve their accuracy; (2) Define new types of spatial outliers (e.g., multi-attribute outliers and region outliers) and design effective methods to identify them. The major contributions of this dissertation are listed as follows.

- We formally defined various spatial outlier detection problems, including single attribute spatial outliers, multi-attribute spatial outliers, and region outliers.

- A set of single attribute point outlier detection algorithms are developed to reduce the "Masking" and "Swamping" problems of the existing methods. "Masking" denotes that true outliers fail to be identified and "Masking" means that normal objects are erroneously identified as outliers. Two approaches are developed to address these problems. One is based on robust representation of neighborhood average, median. Another method utilizes multiple iterations to detect multiple outliers in spatial data.

- To consider the impact of spatial relationships on neighborhood comparison, two spatial weighted spatial outlier detection methods are proposed. They assign a weight for each neighbor based on its spatial relationship with the center object, therefore providing a more accurate representation of neighborhood average.

- To detect multi-attribute spatial outliers, a Mahalanobis-distance-based method is proposed. This method has several advantages. First, it considers the correlations among non-spatial attributes. It provides a weighted distance to compare the neighborhood difference where highly correlated attributes have lower weight. Second, it considers the distribution of each attribute. If each attribute follows normal distribution, the overall behavior of all the attributes can be approximated by Chi-square distribution. Thus, the density function of Chi-square distribution can be used to set the cutting-off threshold for outlier detection.

- A framework is proposed to provide a systematic way for region outlier detection in continuous meteorological data sequences. This framework models geospatial data as digital images, employs efficient image preprocessing and segmentation techniques to detect region outliers, and utilizes a tracking mechanism to monitor their movements. In addition, wavelet-based algorithms are proposed to detect spatial changes and temporal changes in spatio-temporal data.

- A graph-based method is developed for identifying region outliers in static spatial data. This method constructs a k-nearest neighbor dissimilarity graph to represent a spatial data set, then iteratively partitions the graph to detect isolated small regions as outliers. One advantage of this method is that it can detect both region outliers and point outliers.

The median-based and iterative-based algorithms are extension and improvement of the exiting methods for point outlier detection. As far as I know, the proposed spatial weighted algorithms are the first spatial outlier detection method which explicitly incorporates the impact of spatial relationships. Mahalanobis distance was invented long time ago and has been used for many fields. We first applied Mahalanobis distance to the field of multi-attribute spatial outlier detection. The proposed graph-based approach is designed to detect a new type of spatial outlier, region outlier. This is the first method which can effectively identify both point outliers and region outliers. All above methods are designed for outlier detection in static spatial data. The image-based framework is especially designed for region outlier detection in continuous spatial data sequences. This framework was not proposed by others before.

1.5 Organization of the Dissertation

The remainder of this dissertation is organized as follows: Chapter 2 focuses on single (nonspatial) attribute outlier detection. The problem of spatial outlier detection is formally defined and 3 algorithms are presented to reduce the "masking effect" of an outlier on its neighboring objects. In addition, two spatial weighted algorithms are proposed, which take into account the impact of spatial relationships on outlier detection; Chapter 3 discusses region outlier detection and tracking in continuous meteorological data sequences; Chapter 4 extends the use of single attribute algorithms to identify spatial outliers with multiple attributes and proposes a set of Mahalanobis-distance-based approaches; Chapter 5 introduces a graph-based method which can detect both region outliers and point outliers; and finally, Chapter 6 summarizes the dissertation and discusses several future research directions for abnormal spatial pattern recognition.

Chapter 2

Single Attribute Outlier Detection

In this chapter, we propose two sets of algorithms to detect single-attribute spatial outliers. The first set of algorithms aim to reduce the "masking effect" (true outliers are ignored) and "swamping" (normal objects are identified as outliers) problems. These methods are the extension of the existing spatial outlier detection algorithms to improve the detection accuracy. The second set of algorithms consider the impact of spatial relationships on neighborhood comparison. They assign a weight for each neighbor to represent their importance to impact the center object. This weight can support more appropriate neighborhood comparison to meet the specific requirements of different applications.

2.1 Spatial Outlier Detection

In this section, we first define the problem of detecting spatial outliers with a single attribute and discuss the deficiency of the existing detection methods. Then we propose a set of new algorithms and analyze their computational complexities. Finally, experiments are conducted on two data sets to evaluate the effectiveness of our proposed algorithms.

2.1.1 Problem Formulation

Spatial Outlier Definition

Suppose there exists a set of spatial points $X = \{x_1, x_2, \ldots, x_n\}$ in a space with dimension $p \geq 1$. An attribute function f is defined as a mapping from X to R (the set of all real numbers) such that

$f(x_i)$ represents the attribute value of spatial point x_i. For a given point x_i, let $NN_k(x_i)$ denote the k nearest neighbors of point x_i, where $k = k(x_i)$ depends on the value of x_i. A neighborhood function g is defined as a map from X to R such that for each x_i, $g(x_i)$ returns a summary statistic of attribute values of all the spatial points inside $NN_k(x_i)$. For example, $g(x_i)$ can be the average attribute value of the k nearest neighbors of x_i. To detect spatial outliers, we compare the attribute value of each point x_i with those attribute values of its neighbors in $NN_k(x_i)$. Such comparison is done through a comparison function h, which is a function of f and g. There are many choices for the form of h. For example, h can be the difference $f - g$ or the ratio f/g. Let $h_i = h(x_i)$ for $i = 1, 2, \ldots, n$. Given the attribute function f, function k, neighborhood function g, and comparison function h, a point x_i is a **spatial outlier** or simply **S-outlier** if h_i is an extreme value of the set $\{h_1, h_2, \ldots, h_n\}$. We note that the definition depends on the choices of functions k, g and h.

The definition given above is quite general. As a matter of fact, outliers involved in various existing spatial outlier detection techniques are special cases of S-outliers [91]. These include outliers detected by scatterplot [35], Moran scatterplot [66], and pocket plots [37, 76].

The following formulation characterizes the task of designing algorithms for detecting spatial outliers:

Spatial Outlier Detection Problem:
Given:

- a set of spatial points $X = \{x_1, x_2, \ldots, x_n\}$
- a sequence of neighborhoods $NN_k(x_1), NN_k(x_2), \ldots, NN_k(x_n)$
- an attribute function $f : X \rightarrow R$
- a neighborhood function $g : X \rightarrow R$
- a comparison function $h : X \rightarrow R$

Objective:

- design algorithms to detect spatial outliers

A straightforward method to detect S-outliers can be stated as follows. Assume all $k(x_i)$ are equal to a fixed number, denoted as k. The neighborhood function g evaluated at a spatial point

x is taken to be the average attribute value of all the k nearest neighbors of x. The comparison function $h(x)$ is chosen to be the difference $f(x) - g(x)$. Applying such an h to the n spatial points leads to the sequence $\{h_1, h_2, \ldots, h_n\}$. A spatial point x_i is treated as a candidate of S-outlier if its corresponding value h_i is extreme among the data set $\{h_1, h_2, \ldots, h_n\}$. To identify the extreme values of this data set, we begin with standardizing the data set $\{h_1, h_2, \ldots, h_n\}$. Let μ and σ denote the mean and standard deviation of $\{h_1, h_2, \ldots, h_n\}$. The standardized value for each h_i is $z_i = \frac{h_i - \mu}{\sigma}$. Clearly, h_i is extreme in the data set $\{h_1, h_2, \ldots, h_n\}$ iff z_i is extreme in the standardized data set. Correspondingly, x_i is a possible S-outlier if $y_i = |z_i|$ is large enough. This algorithm is described in [92]. We call this algorithm z algorithm, since it is based on the z-score $\frac{h_i - \mu}{\sigma}$.

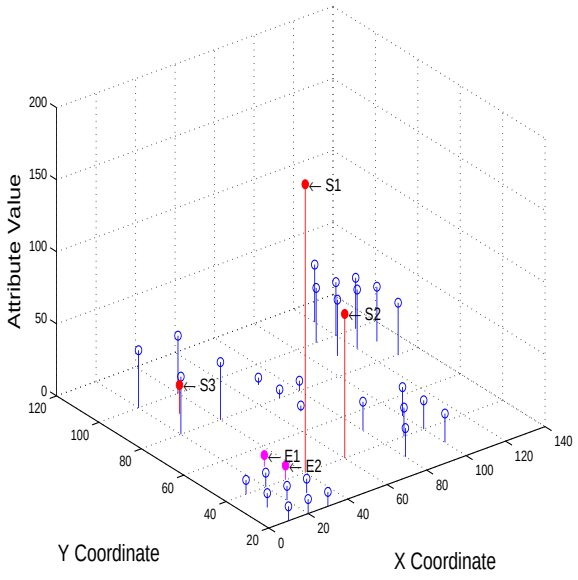


Figure 2.1. A spatial data set. Objects are located in the $X - Y$ plane. The height of each vertical line segment represents the attribute value of the corresponding object.

2.1.2 Deficiency of the Existing Approaches

One drawback of the above z algorithm is that regular spatial points may be falsely detected as spatial outliers due to the presence of neighboring points with very high or low attribute values. A consequence of this disadvantage is that true spatial outliers could be ignored due to the falsely detected spatial outliers if the expected number of spatial outliers is limited. We show these two

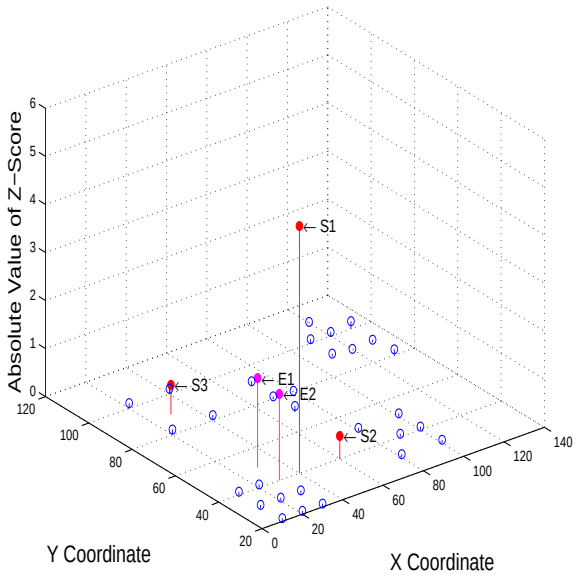


Figure 2.2. Graphical illustration of the z algorithm. Data are shown in Figure 2.1. The height of each vertical line segment indicates the absolute value of the z-score. $S1$, $E1$, and $E2$ would be identified as possible spatial outliers since their absolute z-score values are the largest.

problems using an illustrative example. In figure 2.1, each object is located in the X-Y plane with its associate attribute value in the Z-coordinate. Assuming the expected number of spatial outliers is 3 and k is taken to be 3, then we can easily observe that points $S1$, $S2$ and $S3$ are spatial outliers, since their attribute values are significantly different from those of their neighbors. However, the obtained result from running the above z algorithm indicates that $S1$, $E1$ and $E2$ are spatial outliers, as shown in Figure 2.2. This detection error is mainly due to the large attribute value difference between point $S1$ and its neighboring points. For example, since $S1$ is inside the neighborhood of $E1$, the neighborhood function at $E1$ obtains a value much larger than the attribute value of $E1$, so that $E1$ is taken as an outlier. In general, the z algorithm will lead to some true spatial outliers being ignored and some false spatial outliers being identified. This disadvantage is also shared by other existing detection approaches. For example, $S1$, $E1$, and $E2$ are detected as the top three spatial outliers by the Moran scatterplot method, since these three points are located in the upper-left and lower-right quadrants and are far away from the origin (0,0), as shown in Figure 2.3. $E1$, $E2$, and $S2$ are identified as the top three spatial outliers by the scatterplot approach, since the distances of the three points to the regression line are the largest, as shown in Figure 2.4.

To eliminate the above mentioned defect of the existing algorithms, iterative algorithms can be used, where spatial outliers are detected by multi-iterations. Each iteration identifies only one

outlier and modifies the attribute value of this outlier so that this outlier will not significantly impact the subsequent iterations. We also propose a non-iterative algorithm, where the median is used as the neighborhood function. The use of median potentially reduces the impact caused by the extreme neighboring points.

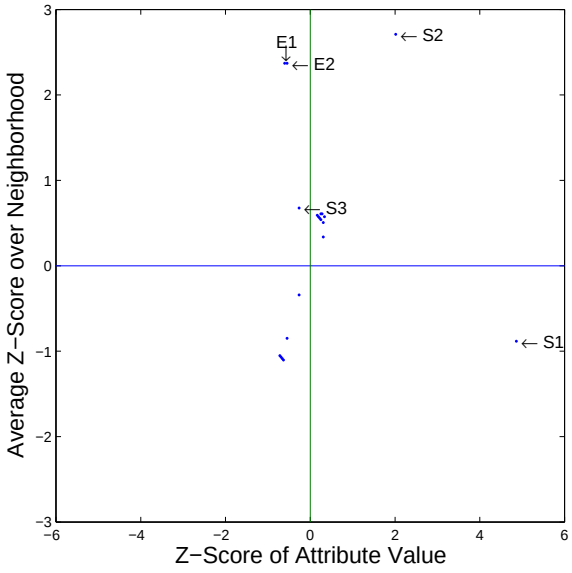


Figure 2.3. Moran scatterplot used to detect the spatial outliers of the data in Figure 2.1. Here z scores represent the normalized attribute values. The three best candidates of outliers are $S1$, $E1$, and $E2$, located in the upper left and lower right quadrants.

2.1.3 Detection Algorithms

In this section, we review three detection algorithms reported in [59]. Additional insights into the algorithms will be made along the line of description of the algorithms. Assume that all $k(x_i)$ are equal to a fixed number k. (The algorithms can be easily generalized by replacing the fixed k by a dynamic $k(x_i)$.) Under the framework of Section 2.1.1, outlier detection algorithms depend on the choices of the neighborhood function g and comparison function h. Selection of g and h determines the performance of the algorithm. In Algorithm 1 below, $g(x)$ is taken to be the average attribute value of all the k nearest neighbors of x and $h(x)$ is chosen to be the ratio of $f(x)$ to $g(x)$. x is an S-outlier if $h(x)$ is very large or very small. Algorithm 1 tries to identify possible S-outliers one by one. Due to the introduction of a positive integer m, Algorithm 1 will locate at most m possible

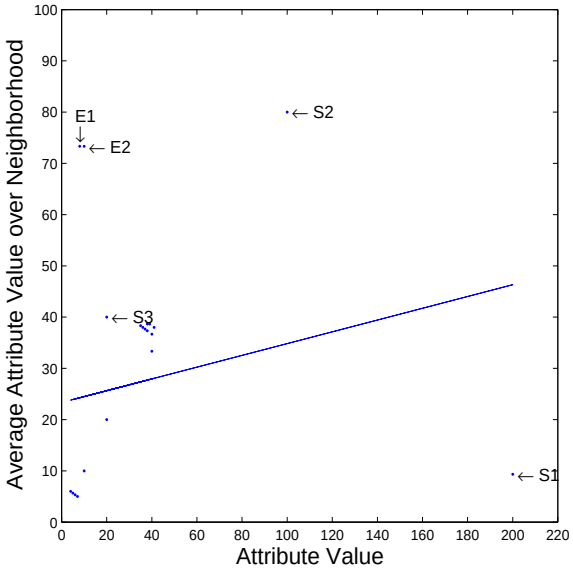


Figure 2.4. Scatterplot used to analyze the data in Figure 1. $E1$, $E2$, and $S2$ are the three best spatial outlier candidates since their distances to the regression line are the largest.

S-outliers. Note that in the algorithm, after x is identified as a candidate, its attribute value $f(x)$ will be replaced by the average value of its neighboring points., i.e., $g(x)$. Correspondingly, some ratios h will be updated during the next run. This will prevent the points close to x from becoming outlier candidates. An example illustrating this "correction" is shown as follows. Suppose x is an S-outlier ($h(x)$ or $h(x)^{-1}$ is large) and x^* is a regular spatial point close to x such that $NN_k(x^*)$ contains x. If $h(x)$ is large, $g(x^*)$ may be large and so $h(x^*) = f(x^*)/g(x^*)$ may be small enough for us to claim that x^* is an outlier candidate. Similarly, if $h(x)$ is small, $h(x^*)$ may be big and hence x^* may be detected as an outlier. With the replacement of $f(x)$ by $g(x)$ and some necessary updating stated in the algorithm, the risk of regular point x^* becoming an outlier is reduced. Algorithm 1 is also termed as an iterative r(ratio) algorithm, since iterations are coupled with the ratios.

Algorithm 1 (Iterative r Algorithm): Given a spatial data set $X = \{x_1, x_2, \ldots, x_n\}$, an attribute function f, and two positive integer numbers k and m,

1. Compute, for each spatial point x_i, the k nearest neighbor set $NN_k(x_i)$, the neighborhood function $g(x_i) = \frac{1}{k} \sum_{x \in NN_k(x_i)} f(x)$ and the comparison function $h_i = h(x_i) = \frac{f(x_i)}{g(x_i)}$.

2. If h_t or h_t^{-1} equals the maximum of h_1, h_2,..., h_n, h_1^{-1}, h_2^{-1}, ..., h_n^{-1}, treat x_t as a candidate of S-outlier.

3. Update $f(x_t)$ to be $g(x_t)$. For each spatial point x_i whose $NN_k(x_i)$ contains x_t, update $g(x_i)$

and h_i.

4. Repeat steps 2 and 3 until the total number of S-outlier candidates equals m.

In Algorithm 2 below, g is the same as in Algorithm 1, but $h(x)$ is chosen to be the difference $f(x) - g(x)$. Applying such an h to the n spatial points leads to the sequence $\{h_1, h_2, \ldots, h_n\}$. A spatial point x_i is treated as a candidate of S-outlier if its corresponding value h_i is extreme among the data set $\{h_1, h_2, \ldots, h_n\}$. Let μ and σ denote the sample mean and sample standard deviation of $\{h_1, h_2, \ldots, h_n\}$. The standardized value for each h_i is $z_i = \frac{h_i - \mu}{\sigma}$, and the standardized data set is $\{z_1, z_2, \ldots, z_n\}$. Now it is clear that x_i is extreme in the original data set iff z_i is extreme in the standardized data set. Correspondingly, x_i is a possible S-outlier if $|z_i|$ is large enough. Similar to Algorithm 1, Algorithm 2 tries to identify possible S-outliers one by one. After x is identified as a candidate outlier, its attribute value $f(x)$ will be replaced by $g(x)$. In addition, the neighborhood function g and comparison function h must be recomputed for points whose k nearest neighbors include point x, and the parameters μ and σ need to be updated as well. Algorithm 2 will locate at most m possible S-outliers. Since Algorithm 2 is based on iterations and the z scores $\frac{h_i - \mu}{\sigma}$, we call it an iterative z algorithm.

Algorithm 2 (Iterative z Algorithm): Given a spatial data set $X = \{x_1, x_2, \ldots, x_n\}$, an attribute function f, and two positive integer numbers k and m,

1. Compute, for each spatial point x_i, the k nearest neighbor set $NN_k(x_i)$, the neighborhood function $g(x_i) = \frac{1}{k} \sum_{x \in NN_k(x_i)} f(x)$, and the comparison function $h_i = h(x_i) = f(x_i) - g(x_i)$.

2. Let μ and σ denote the sample mean and sample standard deviation of the data set $\{h_1, h_2, \ldots, h_n\}$. Standardize the data set and compute the absolute value $y_i = |\frac{h_i - \mu}{\sigma}|$ for $i = 1, 2, \ldots, n$. If y_t equals the maximum of $y_1, y_2, \ldots, y_n$, treat x_t as a S-outlier candidate.

3. Update $f(x_t)$ to be $g(x_t)$. For each spatial point x_i whose $NN_k(x_i)$ contains x_t, update $g(x_i)$ and h_i.

4. Recalculate μ and σ of the data set $\{h_1, h_2, \ldots, h_n\}$. For $i = 1, 2, \ldots, n$, update $y_i = |\frac{h_i - \mu}{\sigma}|$.

5. Repeat steps 2, 3, and 4 until the total number of S-outlier candidates equals m.

There is no clear guideline on which of the two algorithms (iterative r and z algorithms) should be selected in applications. If the attribute function f can take negative values, then the iterative

r algorithm should not be used due to the current description of the algorithm. If the attribute function f is nonnegative, the selection depends on the properties of the practical applications. In general, we recommend that both algorithms be used. We note that these two algorithms do not always yield the same set of S-outliers, as shown in our experiments on the U.S. Census data and the West Nile virus data described later. There exist cases in which the two methods may lead to close detection results. For example, if k is large enough so that we can assume that the variation of the neighborhood function g is small, then it follows from $\frac{f}{g} - 1 = \frac{f-g}{g}$ that the effect of $\frac{f}{g}$ in the iterative r algorithm and the effect of $f - g$ in the iterative z algorithm are about the same. This suggests that the two algorithms may produce the same set of S-outliers.

In both Algorithms 1 and 2, once an S-outlier is detected, one immediately makes some corrections such as replacing the attribute value of the outlier by the average attribute value of its neighbors and updating the subsequent computation. The effect of these corrections is to avoid normal points close to the true outliers to be claimed as possible outliers. In the following, Algorithm 3 presents a direct method to reduce the risk of overstating the number of outliers without replacing the attribute value of the detected outlier. In Algorithm 3, $g(x)$ is chosen to be the median of the attribute values of the points in $NN_k(x_i)$ and $h(x)$ is selected to be the difference $f(x) - g(x)$. The motivation of using median is the fact that median is a robust estimator of the "center" of a sample. Therefore, if $g(x_i)$ represents the median of the attribute values of the points in $NN_k(x_i)$, then when x is detected as an outlier, $g(x^*)$ for any x^* close to x will not be affected by $f(x)$ if the number of neighbors of x^* is large.

Algorithm 3 (Median Algorithm): Given a spatial data set $X = \{x_1, x_2, \ldots, x_n\}$, an attribute function f, and two positive integer numbers k and m,

1. Compute, for each spatial point x_i, the k nearest neighbor set $NN_k(x_i)$, the neighborhood function $g(x_i) =$ median of the data set $\{f(x) : x \in NN_k(x_i)\}$, and the comparison function $h_i = h(x_i) = f(x_i) - g(x_i)$.

2. Let μ and σ denote the sample mean and sample standard deviation of the data set $\{h_1, h_2, \ldots, h_n\}$. Standardize the data set and compute the absolute values $y_i = |\frac{h_i - \mu}{\sigma}|$ for $i = 1, 2, \ldots, n$.

3. Let $i_1, i_2, \ldots, i_m$ be the m indices such that their y values in $\{y_1, y_2, \ldots, y_n\}$ represent the m largest. Then the m S-outliers are $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$.

A quick illustration of Algorithms 1, 2, and 3 is to apply them to the data in Figure 1. Table 2.1 shows the results using the three algorithms with parameters $k = m = 3$, compared with the existing approaches. As can be seen, all the three proposed algorithms accurately detect $S1$, $S2$, and $S3$ as spatial outliers, but the z algorithm, scatterplot, and Moran scatterplot, falsely identify $E1$ and $E2$ as spatial outliers. In this table, the rank of the outliers is defined in an obvious way. For example, in iterative r and z algorithms, the rank is the order of iterations, while in both z and median algorithms, the rank is determined by the the size of y value, i.e., objects with larger y values receive smaller ranks.

Rank	Methods					
	Scatter-plot	Moran Scatterplot	z Alg.	Iterative z Alg.	Iterative r Alg.	Median Alg.
1	E1	S1	S1	S1	S1	S1
2	E2	E1	E1	S2	S2	S2
3	S2	E2	E2	S3	S3	S3

Table 2.1. Top three spatial outliers detected by z, iterative z, iterative r, median, scatterplot, and Moran scatterplot algorithms.

2.1.4 Computational Complexity

In the first step of the iterative r algorithm, the neighborhood and comparison functions are computed for each spatial point, in which a k nearest neighbor (KNN) query is issued. The complexity of the algorithm is then based on the complexity of KNN query with two choices. We can use a grid-based approach, which processes KNN query in constant time if the grid directory resides in memory, leading to the complexity of $O(n)$ for the first step. If an index structure (e.g. R-tree) exists for the spatial data set, spatial index can be used to process KNN query, which needs an $O(logn)$ calculation, leading to the complexity of $O(nlogn)$ for the first step. For each iteration in 2-3 steps, we need to retrieve the maximum ratio value, modify the attribute value of the spatial outlier, and update the values of g and h functions for the impacted objects. The complexity of the above operations is $O(n)$. Therefore, m iterations of steps 2-3 have a time complexity of $O(mn)$. The total complexity of the iterative r algorithm is then $O(n) + O(mn) = O(n)$ for the grid-based structure, or $O(nlogn) + O(mn) = O(nlogn)$ for the spatial index structure (if $n \gg m$).

For the first step in the iterative z algorithm, the computational structure and complexity are the same as those of the iterative r algorithm. The complexity for the steps 2-4 of the algorithm is $O(mn)$. Then the total complexity of the iterative z algorithm is exactly the same as that of the

iterative r algorithm, i.e., $O(n)$ for the grid-based structure, or $O(nlogn)$ for the spatial index-based structure (if $n \gg m$).

In the first step of the median algorithm, one needs to compute, for each object, the k nearest neighbors and the median attribute value. Its computation cost is $O(kn)$ for the grid-based structure, or $O(n(logn + k))$ for the spatial index structure. For the second step, computation of mean, standard deviation and standardization costs $O(n)$. In the third step, $O(nlogm)$ is required for selecting the top m objects with the largest y values. The total complexity for the algorithm is then $O(kn) + O(n) + O(nlogm) = O(n)$ for the grid-based structure (if $n \gg k$ and $n \gg m$), or $O(n(logn + k)) + O(n) + O(nlogm) = O(nlogn)$ for the spatial index-based structure (if $n \gg k$ and $n \gg m$).

In summary, all the three algorithms have the complexity $O(n)$ for the grid-based structure, or $O(nlogn)$ for the spatial index-based structure, if the number of objects n is much greater than the number of neighbors k and the number of spatial outliers m. In addition, we can see that the computation costs of these three algorithms are primarily determined by the KNN query.

2.1.5 Experiments

In this section, experiments are performed on two data sets, U.S. Census Data and West Nile Virus Data. The experimental results from both data sets validate the effectiveness of our proposed algorithms.

1. Experiments on Census Data

We empirically compared the detection performance of our proposed methods with the z algorithm through mining a real-life census data set. We first tested the four algorithms based on the counties in the State of Indiana and then extended to all counties in the U.S.A. The experiment results indicate that our algorithms can successfully identify spatial outliers ignored by the z algorithm and can avoid detecting false spatial outliers.

The census data is the most detailed tabulation of American demographic data compiled by the U.S. Census Bureau [99]. It contains population, race and ethnicity, age, gender, education, employment, income, poverty, housing, and many other attributes for each of the following different levels of geography: 1) the United State and major regions of the country; 2) each state and metropolitan area; 3) all 3000+ counties in the United States; 4) municipalities, census tracts, and block groups.

Rank	Methods			
	z alg.	Iterative z alg.	Iterative r alg.	Median
1	Marion	Marion	Marion	Marion
2	Lake	Lake	Vanderburgh	Lake
3	Vanderburgh	Vanderburgh	Lake	Vanderburgh
4	Shelby	Allen	Tippecanoe	Posey
5	Hancock	St. Joseph	Allen	St. Joseph

Table 2.2. The top five spatial outlier candidates detected by z, iterative z, iterative r, and median algorithms in Indiana.

County Population Density in the State of Indiana

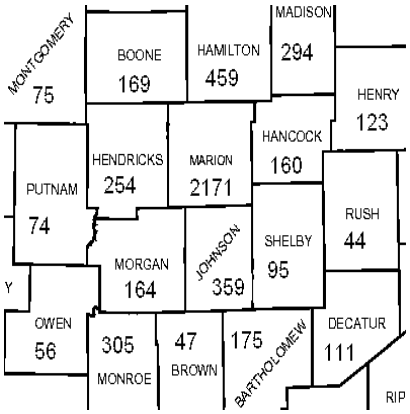


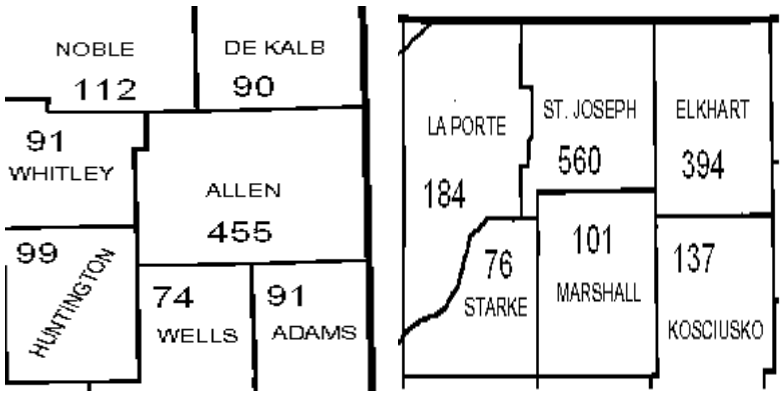
Figure 2.5. Population Density of Marion Co., Shelby Co., Hancock Co., and their Neighboring Counties.

In this experiment, we used population density (population per square mile) as the attribute for each county in the state of Indiana. We ran the four algorithms (z, iterative r, iterative z, and Median algorithms) to detect the counties with abnormal population densities. The state of Indiana has 91 counties, whose locations are denoted by their "central" longitude and latitude. We expected 5% of these counties have abnormal population densities, and 5 counties (91∗5%) are therefore identified as spatial outliers, namely, $m = 5$. Correspondingly, for the parameter setting we chose $m = 5$ and $\theta = 1$ for all four algorithms. The number of neighbors is dynamic, i.e., the neighborhood of a county is chosen to be the set of adjacent counties.

Table 2.2 shows the execution results of the four algorithms. As can be seen, the top three spatial outliers, Marion County, Lake County, and Vanderburgh County, are identified by all the four algorithms but in different order. There is an intuitive justification for the selection of these counties: the state capital of Indiana, Indianapolis, is located in Marion county; Lake county has higher population density than its surrounding counties because it is immediately next to the city of Chicago; and in Vanderburgh County, there are the city of Evansville, the University of Evansville,

and the University of Southern Indiana.

For the fourth and fifth outliers, z algorithm identified Shelby County and Hancock County. As shown in Figure 2.5, these two counties were falsely selected, because a true spatial outlier, Marion County, exists in their neighborhood. Marion county has such a high population density (2171 per square mile) that it dominates the neighborhood averages of Hancock county and Shelby county, which leads to the large differences between these two counties and their neighbors. In contrast, iterative z algorithm correctly selects Allen County and St. Joseph County as the fourth and fifth spatial outliers, and Figure 2.6 shows the population density of these two counties. The iterative r algorithm selects Tippecanoe County as the fourth spatial outlier and Allen County as the fifth spatial outlier. Figure 2.7(a) shows that Tippecanoe County is a true spatial outlier, because its population density is considerably higher than its neighboring counties. Note that Posey county is falsely labelled as the fifth spatial outliers by the median algorithm, because one of its neighbors, Vanderburgh County, has a extremely high population density (please refer to Figure 2.7(b)). Since Posy county has only two neighbors, Vanderburgh County and Gibson county, the median of attribute values of these two neighbors is equal to the mean. Here, we can see that if the number of neighbors is too small, even the median cannot be a robust estimator for the neighborhood average.

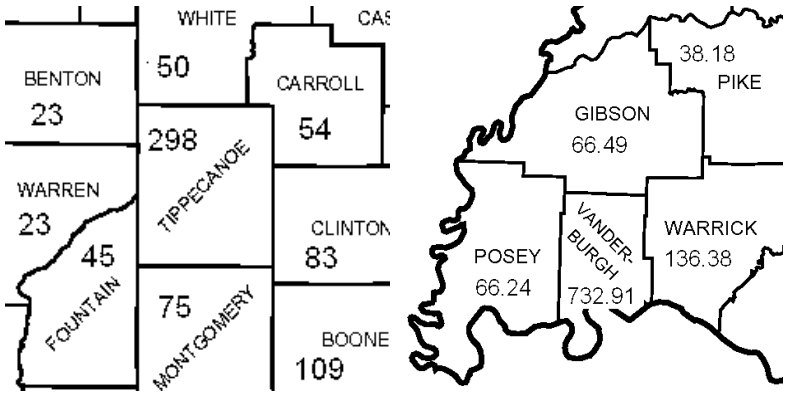


(a) Allen Co. (b) St. Joseph Co.

Figure 2.6. Population Density of Allen Co., St. Joseph Co, and their Neighboring Counties.

Population of the USA

In this experiment, we extended the data set to all the counties in the United States. We tested various attributes, including population, population density, percentage of white people, percentage of African Americans, percentage of American Indians, percentage of Asian Americans,



(a) Tippecanoe Co. (b) Vanderburgh Co.

Figure 2.7. Population Density of Tippecanoe Co., Vanderburgh Co., and their Neighboring Counties.

and percentage of females. Figure 2.8 shows the population distribution of the USA at county level in 2001. Four algorithms (z, iterative-r, iterative-z, and Median) were applied on the 3192 counties in the USA, and 10 counties were identified as spatial outliers.

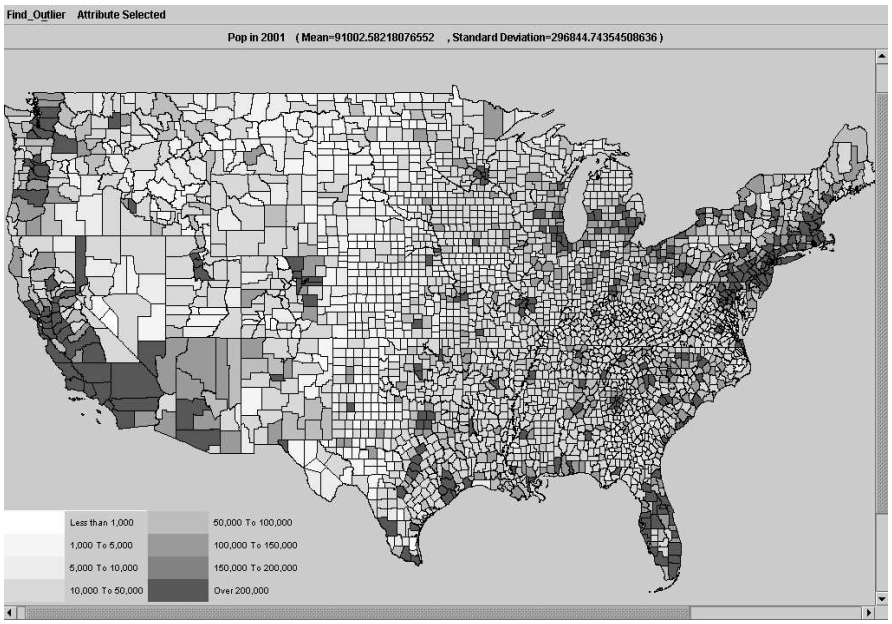


Figure 2.8. US Population at the County Level in year 2001

Table 2.3 shows that the z, iterative-z, and Median algorithms have similar results, identifying eight identical counties in the top 10 outliers. Further examination demonstrates that the top 10 counties selected by the iterative-z and Median algorithms are true outliers. However, Ventura Co. in California was falsely detected by the z algorithm, but avoided by the iterative z and Median algorithms. The last column of Table 2.3 shows the top ten candidate outliers identified

Rank	Methods			
	z Alg.	Iterative z Alg.	Median Alg.	Iterative r Alg.
1	Los Angeles,CA,9637494.0	Los Angeles,CA,9637494.0	Los Angeles,CA,9637494.0	Kenedy,TX,413.0
2	Cook,IL,5350269.0	Cook,IL,5350269.0	Cook,IL,5350269.0	Loving,TX,70.0
3	Harris,TX,3460589.0	Harris,TX,3460589.0	Harris,TX,3460589.0	Treasure,MT,802.0
4	Maricopa,AZ,3194798.0	Maricopa,AZ,3194798.0	Maricopa,AZ,3194798.0	Lincoln,NV,4198.0
5	Ventura,CA,770630.0	Dallas,TX,2245398.0	Miami-Dade,FL,2289683.0	Falls Church city,VA,10612.0
6	Dallas,TX,2245398.0	Miami-Dade,FL,2289683.0	Dallas,TX,2245398.0	La Paz,AZ,19759.0
7	Miami-Dade,FL,2289683.0	Wayne,MI,2045473.0	Wayne,MI,2045473.0	Alpine,CA,1192.0
8	Wayne,MI,2045473.0	Bexar,TX,1417501.0	Clark,NV,1464653.0	Hudspeth,TX, 3318.0
9	Bexar,TX,1417501.0	King,WA,1741785.0	Bexar,TX,1417501.0	Fairfax city,VA,21674.0
10	King,WA,1741785.0	San Diego,CA,2862819.0	Tarrant,TX,1486392.0	Gilpin, CO,4823.0

Table 2.3. The top ten spatial outlier candidates detected by z, iterative z, iterative r, and Median algorithms.

by the iterative r algorithm. Although these 10 candidates have been validated as true outliers by a practical examination, they are much different from those outliers obtained from the other three methods. This difference is due to the fact that the iterative-r algorithm focuses on the ratio between the attribute values.

The experimental results validate that the two iterative algorithms and the Median algorithm are more accurate than the original z algorithm in terms of the false alarm rate. For the detailed information about our experiments, we refer readers to [63], where a software package is available on the Internet, which implements all the discussed algorithms.

2. Experiment on West Nile Virus Data

In this section, we perform experiments on the West Nile virus(WNV) data provided by the US Centers for Disease Control and Prevention(CDC), to further validate the effectiveness of our single attribute outlier detection algorithms.

WNV is commonly found in Africa, West Asia, and the Middle East. The virus can infect humans, birds, mosquitoes, horses and some other mammals. Since its first appearance in 1999, WNV has been detected in wildlife, primarily birds, in 45 states and the District of Columbia in the United States. WNV is maintained in the environment of a bird-mosquito life cycle. Birds are the natural host and reservoir of WNV. For reasons still unknown, American crows (Corvus brachyrhynchos) and other corvids (e.g., blue jays) seem more susceptible to fatal infection. This has allowed many local health departments to utilize dead birds as an indicator of the virus's emergence in their jurisdictions. Mosquitoes become infected when they feed on infected birds, which may circulate the virus in their blood for a few days. During blood feeding, the virus may be injected into the animal or human, where it may multiply, possibly causing illness.

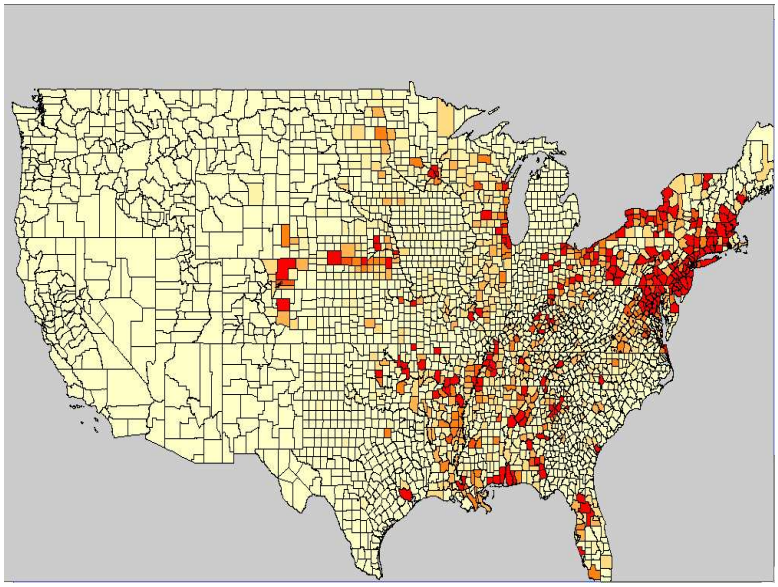


Figure 2.9. Cases of dead birds infected by WNV across all counties in the United States in 2002

The WNV data set is based on all 3000+ counties in the United States. It reports the case numbers of wild birds, mosquitoes, and veterinaries found within each county between January 1, 2001 and December 31, 2003. The location of each county is determined by the "central" longitude and latitude provided by the data. A case number in each year can be treated as an attribute. Since 3 years are considered, there are 9 attributes available for each county: Bird-2001, Mosquito-2001, Veterinary-2001, Bird-2002, Mosquito-2002, Veterinary-2002, Bird-2003, Mosquito-2003, and Veterinary-2003. Understanding of the definitions of these attributes is straightforward. For example, Bird-2001 denotes the number of cases of WNV infected birds in the year 2001, Mosquito-2001 denotes the number of cases of WNV infected mosquitoes in the year 2001, and Veterinary-2001 denotes the number of cases of WNV infected veterinaries identified in the year 2001. Figure 2.9 shows the visualization of the cases of birds infected by West Nile Virus for all counties in the United States in 2002. Darker colors denote higher number of cases.

The purpose of the experiments is to identify which counties are abnormal in terms of the WNV cases. In our analysis, each county was treated as a spatial object, and the number of neighbors for each county was chosen to be dynamic, i.e., the neighborhood of a county was chosen to be the set of adjacent counties. The proposed algorithms in this paper were used to facilitate the observation and discovery of abnormal WNV cases by comparing disease reports for a county with those from its adjacent counties.

We empirically compared the performance of our proposed single attribute algorithms with the z algorithm. Since there are many counties having 0 for some attribute variables (which means no WNV case was detected), iterative ratio method can not be used. Thus, in general, only the z, iterative z, and median algorithms can be used to show the effectiveness of the proposed single attribute algorithms. As an illustration, in this section we report the results obtained by examining the State of Pennsylvania.

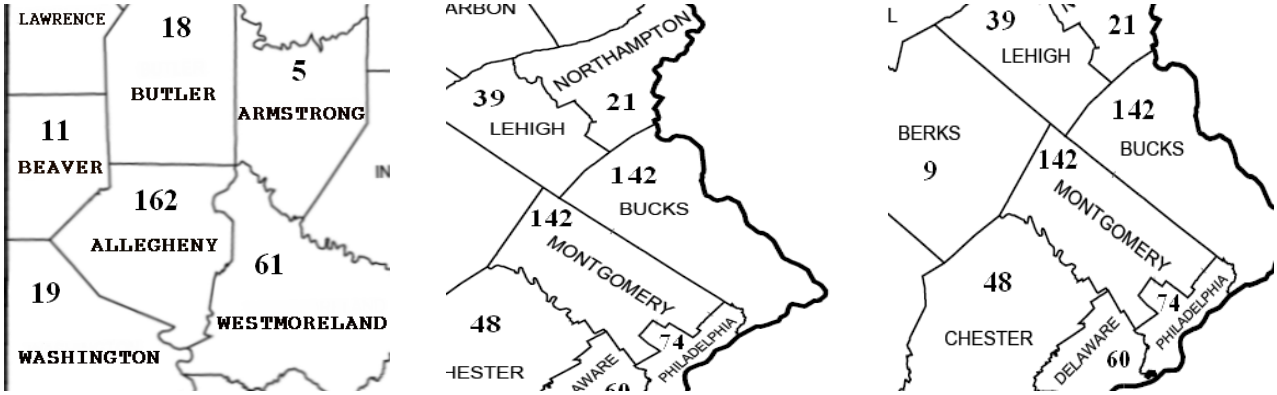
In this experiment, we used Bird-2002 as the attribute for each county in the State of Pennsylvania. Three algorithms (z, iterative z, and median) were executed to discover which counties have abnormal Bird WNV cases within the state. The State of Pennsylvania has 67 counties. We selected top 6 counties (about 9% of 67) as outlier candidates, which have abnormal attribute values compared with their neighbors. Correspondingly, for the parameter setting we chose $m = 6$ for all the three algorithms.

Table 2.4 shows the top six spatial outlier candidates detected by the three algorithms. As can be seen, the top three candidates, Allegheny County, Bucks County, and Montgomery County, are the same for all these algorithms. They should be treated as true outliers since they have much higher attribute values (162, 142, 142) than their neighbors (far less than 100). See Figure 2.10 for more details.

Rank	Methods		
	z alg.	Iterative z alg.	Median alg.
1	Allegheny	Allegheny	Allegheny
2	Bucks	Bucks	Bucks
3	Montgomery	Montgomery	Montgomery
4	Berks	Westmoreland	Westmoreland
5	Lancaster	Lancaster	Berks
6	Armstrong	Dauphin	Dauplin

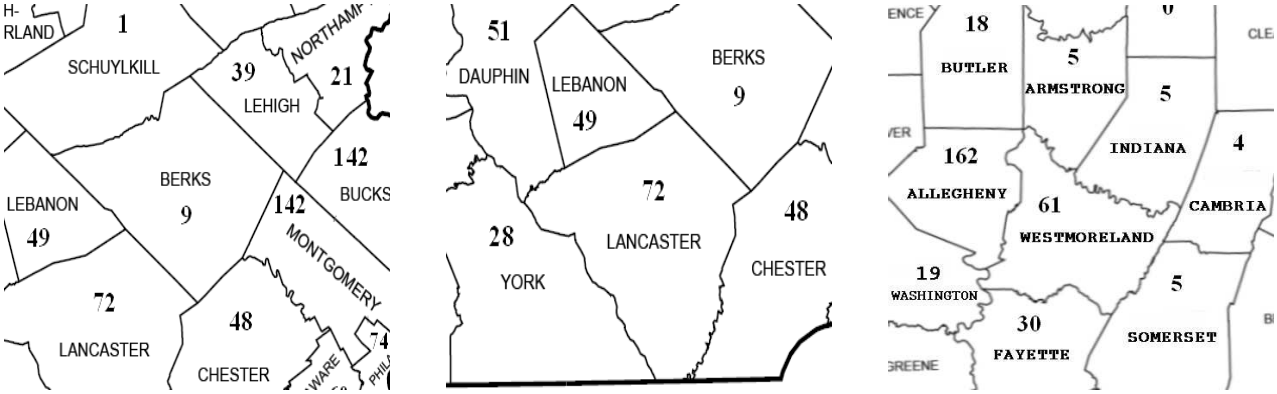
Table 2.4. Top six spatial outlier candidates detected by z, iterative z, and median algorithms.

The fourth and fifth spatial outlier candidates were selected from 3 counties: Berks, Lancaster, and Westmoreland. As shown in Figure 2.11(a), Berks should be treated as a real outlier, since its attribute value is quite different from those of its neighbors whose average is 58.6. Lancaster (Figure 2.11(b)) should also be treated as a real outlier, since it has an attribute value of 72, and all the values from its neighbors are less than 50. Note that Lancaster has other two neighboring counties that do not show up in this figure. They are Harvard County and Cecil County, with attribute values of 45 and 5, respectively. Westmoreland County is identified by both iterative z



(a) Allegheny Co. (b) Bucks Co. (c) Montgomery Co.

Figure 2.10. Distribution of the values of attribute Bird-2002 in the vicinity of Allegheny, Bucks, and Montgomery. The numbers represent the values for Bird-2002.



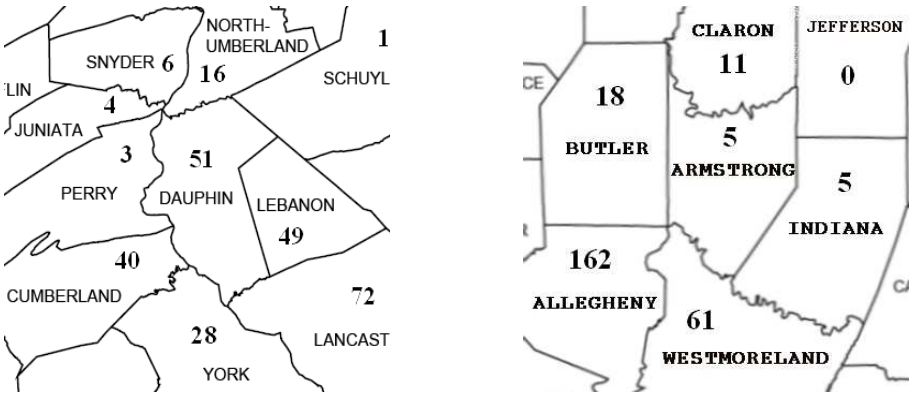
(a) Berks Co. (b) Lancaster Co. (c) Westmoreland Co.

Figure 2.11. Distribution of the values of attribute Bird-2002 in the vicinity of Berks, Lancaster, and Westmoreland. The numbers represent the values for Bird-2002.

and median algorithms, while it does not appear in the result of the z algorithm. Westmoreland county is a real outlier, since the attribute value of this county is much higher than most of its neighbors (see Figure 2.11(c)). The z algorithm fails to detect it since its neighbor, Allegheny County, has such a high attribute value (162) that the averaged value of the neighborhood of Westmoreland is close to the attribute value of Westmoreland. The iterative z algorithm does not have this disadvantage because once Allegheny County is identified as an outlier, its attribute value is substituted with the average of its neighbors, 22.8. Thus the average of Westmoreland's neighbors (13.6) is actually much less than its own attribute value (61). The median algorithm can also detect Westmoreland County because it uses the median value (11.5) to represent the "center" of the values from Westmoreland's neighbors (4,5,5,5,18,19,30,162). For the sixth outlier, the iterative z and median algorithms identify the same county, Dauplin. As shown in Figure

2.12(a), Dauplin is selected because it has four neighbors which have very small attribute values (3,4,6,16). The z approach detects Armstrong County. However, this county is not a real outlier. Figure 2.12(b) shows that Armstrong has 6 neighbors, 4 of which have values similar to that of Armstrong. But the other two neighbors, Allegheny and Westmoreland, have much higher values, which make the Armstrong County falsely detected as an outlier.

The above example shows that the iterative z and median algorithms tend to identify spatial outliers ignored by the z algorithm and avoid detecting false spatial outliers. This conclusion is valid for all the experiments we have conducted so far.



(a) Dauphin Co. (b) Armstrong Co.

Figure 2.12. Distribution of the values of attribute Bird-2002 in the vicinity of Dauphin and Armstrong. The numbers represent the values for Bird-2002.

2.2 Weighted Spatial Outlier Detection

In this section, we propose two spatial outlier detection methods which integrate the impact of spatial properties to the outlierness measurement. In this way, a more accurate representation of the difference between an object and its neighbors can be provided. Experimental results on a real data set demonstrate the effectiveness of the proposed algorithms.

2.2.1 Motivation

In most of the existing spatial outlier detection algorithms, spatial attributes are only used for determining the neighborhood relationship. The computation of the outlierness of a spatial object is solely based on the non-spatial attributes of this object and its neighbors. If two neighbors of an object have the same nonspatial attribute values, they are deemed to have equal impact on this central object. However, in many real applications, spatial objects can not be simply abstracted

as isolated points. They have different location, area, contour, and volume, which play important roles in determining the impact of a spatial object on its neighbors and should not be ignored. For example, suppose we would investigate the expansion of a chemical pollution across a number of adjacent counties. The impact of a county to its neighbor county is closely related to the distance and common border length between them. The smaller the distance and the larger the common border, the higher possibility of pollution expansion between these two counties.

Based on this observation, we propose a spatial outlier detection method, which assigns different weights for different neighbors in computing the outlierness of the central object. The weight is determined by spatial relationships such as the distance and common border length.

2.2.2 Problem Formulation

We formalize the spatial outlier detection as follows.

Given:

- **X** is a set of spatial objects $\{x_1, x_2, \ldots, x_n\}$ with single or multiple attributes, where $x_i \in \Re^d$.
- **k** is an integer denoting the number of adjacent data objects which form the neighborhood relationship. Every object x_i has k neighbor objects based on its spatial location, denoted as $NN_k(x_i)$.
- **Y** is a set of attribute values $\{y_1, y_2, \ldots, y_n\}$, where y_i is the attribute value of x_i.
- **m** is the number of outliers to be identified, and generally $m \ll n$.

Objective:

- Design a mapping function $f : (X, Y, k) \longrightarrow O_f$. $O_f = \{OF_1, OF_2, \ldots, OF_n\}$ where OF_i is the outlier factor describing the degree of outlierness for object x_i, $OF_i \in \Re^d$.
- Find a set Z of m data objects where $Z \subset X$ and for $\forall x_i \in Z$ and $\forall x_j \in (X - Z)$, $OF_i > OF_j$.

The major task of spatial outlier detection is to design an appropriate function f, which can effectively represent the outlierness of an object. The outlierness can be viewed as the difference between an object and its neighbors. Most of the existing outlier methods use the average attribute value to represent the behavior of a set of neighbor points. Then this average value is compared with the attribute value of the center object to determine its outlierness. The definition given

above is quite general. As a matter of fact, outliers involved in various existing spatial outlier detection techniques are special cases of above definition, including Scatterplot [35, 65], Moran scatterplot [66], and pocket plots [37, 76].

2.2.3 Algorithms

In this section, we define the problem of detecting spatial outliers with single nonspatial attribute, discuss the impact of spatial properties on the outlier detection, and propose two algorithms which consider the impact of spatial properties. The computational complexities of these algorithms are examined as well.

Algorithm 1: Weighted z value approach

In this section, we propose an algorithm to detect spatial outliers by using spatial relationship as the weight of nonspatial attribute values. Additional insights into the algorithm will be provided through the description below the pseudo-code. We assume that all $k(x_i)$ are equal to a fixed number k. (The algorithms can be easily generalized by replacing the fixed k by a dynamic $k(x_i)$.)

The proposed algorithm has four input parameters. X is a set of n objects containing spatial attributes such as location, boundary, and area. The non-spatial attributes are contained in another set Y. In many applications, these attribute values can be the results of preprocessing like dimension reduction or data standardization. k is the number of neighbors. For description simplicity, the value of k is fixed for every object. m is the number of requested outliers. Generally, m should not be greater than 5% of n.

For each data object, the first step is to identify its k nearest neighbors. Calculating the Euclidean distance between the centers of two objects is the most frequently used method. Next, for each object x_i, compute the weighted average of the non-spatial attribute values for all x_i's neighbors. Different neighbors have different impact on x_i, which is represented by a *weight*. The weight is determined by the spatial relationships between x_i and its neighbor x_j. There may be more than one spatial relationship which contributes to the weight, for example, the inverse of distance between x_i and x_j and the common border length between x_i and x_j. The value of *weight* for a neighbor x_j is between 0 and 1, and the sum of weights for all x_i's neighbors is 1. Assuming x_j is the rth neighbor of x_i, the *weight* of x_j can be obtained by the following equation:

$$weight = \sum_{p=1}^{q} \alpha_p \bullet \frac{S_{pr}}{\sum_{l=1}^{k} S_{pl}}$$

q denotes the maximum number of spatial properties which determine the weight. S_{pl} represents

Algorithm 1: Weighted z Value Algorithm

Input:
X is a set of n spatial objects;
Y is the set of attribute values for X;
k is the number of neighbors;
m is the number of requested spatial outliers;

Output:
O_s is a set of m outliers

for(i=1; i $\leq n$; i++) {
  /* *calculate the neighbor hood relationship* */
  $NN_k(x_i)$ = GetNeighbors(X, x_i);
  /* calculate the weighted average of all x_i's neighbors */
  $NbrAvg(x_i) = 0$;
  for each $x_j \in NN_k(x_i)$ {
    $weight$ = getWeight($NN_k(x_i)$, x_j)
    $NbrAvg(x_i) = NbrAvg(x_i) + y_j * weight$
  }
  $Diff(x_i) = y_i - NbrAvg(x_i)$
}
/* calculate the standardized $Diff(x_i)$ as the outlierness factor */
$\mu = getMean(Diff)$
$\sigma = getStd(Diff)$
for (each $x_i \in X$) {
  $OF(x_i) = |\frac{Diff(x_i)-\mu}{\sigma}|$
}
O_s = getTopMOutliers(OF, m)

the value of a particular spatial property S_p for the lth neighbor of x_i. α_p is the factor which determines the importance of spatial property S_p, and $\sum_{p=1}^{q} \alpha_p = 1$. For example, if two spatial properties, the inverse of distance and the length of common border are used for weight calculation, the equation can be represented as:

$$weight = \alpha_1 \bullet \frac{invDist_r}{\sum_{l=1}^{k} invDist_l} + \alpha_2 \bullet \frac{Border_r}{\sum_{l=1}^{k} Border_l}.$$

Weighted average $NbrAvg$ is obtained by summing the product of the weight and the non-spatial attribute value y_j for each neighbor x_j. Next, the difference between x_i's non-spatial attribute value and $NbrAvg$ is computed, denoted as $Diff(x_i)$. Based on the mean and standard deviation of $Diff$, the standardized $Diff(x_i)$ is then computed as the outlierness factor for x_i. Finally, the top m objects with largest outierness factors will be identified as outliers and be output to the result set O_s.

If the impact of spatial properties on the nonspatial attribute is ignored, that is, $q = 0$, we designate the weight as $\frac{1}{k}$. In this case, $NbrAvg$ is the arithmetic average of neighbors, which

makes this algorithm the same as the statistical z value approach in [91]. Thus, weighted z value approach can be viewed as the generalization of z value approach.

Algorithm 2: Averaged Difference Algorithm

In this section, we present a variant of Algorithm 1, Averaged Difference Algorithm. For simplicity, we call it $AvgDiff$ algorithm. Unlike Algorithm 1, $AvgDiff$ is based on the weighted average of the absolute difference between x_i and each of its neighbors. The main idea is to compare an object with each of its neighbors one by one, instead of obtaining the average of all its neighbors before comparison. The reason lies in that the simple average of neighbors may conceal their variances. For example, suppose we have an object O_1 with an attribute value of 50. It has two neighbors O_2 and O_3, with attribute values of 0 and 100 respectively. The average of O_2 and O_3 is 50, which is identical to the value of O_1. However, both O_2 and O_3 are quite different from O_1. By computing the absolute difference first and then computing the average, we can retain the variances among neighbors. Since the difference is absolute, it will not follow normal distribution. Therefore, it is not necessary to be normalized. Similar to Algorithm 1, spatial properties are employed as the weight of the difference.

The $AvgDiff$ algorithm has the same input and output parameters as the weighted z value approach. For each data object x_i, the first step is to identify its k nearest neighbors. Unlike algorithm 1, $AvgDiff$ algorithm does not compute the weighted average of x_i's neighbors or calculate the difference between the attribute value of x_i and this average. Instead, it first calculates the absolute difference $diff$ between x_i and each of its neighbors x_j, and then obtain the weighted average $AvgDiff(x_i)$ of these difference values. Here, the computation of x_j's weight is the same as Algorithm 1. The weighted average difference can be directly used as outlierness factor OF. Finally, the top m objects with largest OF values will be identified as outliers and are output to the result set O_s.

Time Complexity

For the weighted z value approach, the neighborhood is computed first for each spatial point, in which a k nearest neighbor (KNN) query is issued. For the KNN query, there are two choices. We can use a grid-based approach, which processes KNN query in constant time if the grid directory resides in memory, leading to a complexity of $O(n)$ for determining k neighbors for all objects in the data set. If an index structure (e.g. R-tree) exists for the spatial data set, spatial index can be

Algorithm 2: AvgDiff Algorithm

Input:
X is a set of n spatial objects;
Y is the set of attribute values for X;
k is the number of neighbors;
m is a number of requested spatial outliers;

Output:
O_s is the set of m outliers

```
for(i=1; i ≤ n ;i++) {
   /* calculate the neighbor hood relationship */
   NN_k(x_i) = GetNeighbors(X, x_i);
   /* calculate the weighted average difference between x_i and */
   /* its neighbors */
   AvgDiff(x_i) = 0;
   for each x_j ∈ NN_k(x_i) {
      diff = |y_i − y_j|
      weight = getWeight(NN_k(x_i), x_j)
      AvgDiff(x_i) = AvgDiff(x_i) + diff * weight
   }
}
for ( each x_i ∈ X ) {
   OF(x_i) = AvgDiff(x_i)
}
O_s = getTopMOutliers(OF, m)
```

used to process KNN query, whose cost is $O(logn)$, leading to a complexity of $O(nlogn)$. The cost of computing the weighted average for all the neighbors of object x_i is $O(k)$, which is very small compared with the cost of KNN query and can be ignored. The time complexity of computing mean, standard deviation and standardized value of the difference between x_i and the average of its neighbors is $O(n)$. The cost of picking the top m outliers from n objects is $O(nlogm)$. Since $m \ll n$, this cost can be viewed as $O(n)$. In summary, if the number of point n is much greater than the number of neighbors k and the number of spatial outliers m, the time complexity is $O(n)$ for grid-base structure, or $O(nlogn)$ for spatial index structure. The computation cost is primarily determined by the KNN query.

For the $AvgDiff$ algorithm, the time complexity of computing k nearest neighbors is the same as that of weighted z value approach. The computation of averaged difference has the cost of $O(k)$, which can be ignored compared with the time complexity of KNN query. Therefore, the total time complexity is mainly determined by the KNN query, which is $O(n)$ for grid-base structure, or $O(nlogn)$ for spatial index structure.

2.2.4 Experiments

In this section, we conduct experiments on a real data set, West Nile Virus (WNV) Data provided by the US Centers for Disease Control and Prevention (CDC), to validate the effectiveness of our proposed outlier detection algorithms. Experimental design is discussed first, followed by the experiment result analysis. For details for the WNV data set, please refer to Section 2.1.5.

The proposed algorithms can facilitate the discovery of abnormal WNV cases by comparing disease reports from neighboring counties. The number of neighbors was chosen to be dynamic, i.e., the neighborhood of a county consists of the set of adjacent counties. When calculating the weighted difference between a county and its neighbors, two spatial properties are employed, inverse center distance and common border length. The shorter the center distance and the longer the common border length, the higher probability of the spread of West Nile Virus between two neighboring counties. The distance between an object and its neighbor is obtained by calculating the Euclidean distance between their centers. The common border length is determined by the overlap of boundary polygons of the two counties. We assume that the inverse center distance and the common border length have equal impact on the outlierness of a county, so the weight can be expressed as:

$weight = \frac{1}{2} \bullet \frac{invDist_r}{\sum_{l=1}^{k} invDist_l} + \frac{1}{2} \bullet \frac{Border_r}{\sum_{l=1}^{k} Border_l}$.

Here, r denotes the r-th neighbor of the central county and k refers to the number of neighbors. The nonspatial attribute is the number of WNV cases detected in birds, veterinaries, and mosquitos. Since different counties have different size of area, we use the density of WNV cases to make them comparable. The density is expressed by the number of cases per square kilometer. As area has been used for calculating the WNV case density, its impact will not be considered in the weight computation.

Our experiment was based on the cases of veterinaries infected by West Nile Virus in 2003. In addition to the weighted z algorithm and $AvgDiff$ algorithm, we also conducted experiment on the existing z value approach for comparison. Table 2.5 provides the experimental result for all these three spatial outlier detection algorithms. The top 30 spatial outliers are presented, which account for about 1% of all the 3109 counties.

Weighted z algorithm vs. z algorithm

Due to the application of spatial weight, the weighted z algorithm has much different result compared with z approach, where 15 counties are different in the top 30 outliers. For example, Hot

Rank	Methods		
	z Alg.	Weighted z Alg.	$AvgDiff$ Alg.
1	Harford County,MD,0.0158	**York County,PA,0.0175**	**Lancaster County,PA,0.0683**
2	**Hot Springs County,WY,0.0008**	**Berks County,PA,0.0121**	Chester County,PA,0.0501
3	Delaware County,PA,0.0126	Lebanon County,PA,0.0245	Lebanon County,PA,0.0245
4	Adams County,PA,0.0178	Delaware County,PA,0.0126	New Castle County,DE,0.0000
5	Sandoval County,NM,0.0014	Cecil County,MD,0.0078	Carroll County,MD,0.0378
6	Torrance County,NM,0.0018	New Castle County,DE,0.0000	**Berks County,PA,0.0121**
7	Los Alamos County,NM,0.0035	Lancaster County,PA,0.0683	Gloucester County,NJ,0.0321
8	Berks County,PA,0.0121	Chester County,PA,0.0501	Cecil County,MD,0.0078
9	Lancaster County,PA,0.0683	Cumberland County,NJ,0.0063	Salem County,NJ,0.0309
10	Carroll County,MD,0.0378	Montgomery County,PA,0.0184	Delaware County,PA,0.0126
11	**York County,PA,0.0175**	Harford County,MD,0.0158	York County,PA,0.0175
12	Baltimore city,MD,0.0000	Adams County,PA,0.0178	Baltimore city,MD,0.0000
13	Howard County,MD,0.0199	Carroll County,MD,0.0378	Rockwall County,TX,0.0210
14	McKinley County,NM,0.0002	Frederick County,MD,0.0175	Cumberland County,NJ,0.0063
15	Philadelphia County,PA,0.0029	Howard County,MD,0.0199	Philadelphia County,PA,0.0029
16	Weld County,CO,0.0050	Dauphin County,PA,0.0044	Dauphin County,PA,0.0044
17	Cumberland County,NJ,0.0063	Philadelphia County,PA,0.0029	Bucks County,PA,0.0216
18	Cecil County,MD,0.0078	Baltimore County,MD,0.0129	Montgomery County,PA,0.0184
19	Denton County,TX,0.0056	Camden County,NJ,0.0087	Monmouth County,NJ,0.0147
20	Baltimore County,MD,0.0129	Baltimore city,MD,0.0000	Union County,PA,0.0134
21	Johnson County,WY,0.0012	Salem County,NJ,0.0309	Ramsey County,MN,0.0149
22	Boulder County,CO,0.0094	Gloucester County,NJ,0.0321	**Camden County,NJ,0.0087**
23	Montgomery County,PA,0.0184	Mercer County,NJ,0.0051	Baltimore County,MD,0.0129
24	Lebanon County,PA,0.0245	Atlantic County,NJ,0.0062	**Anne Arundel County,MD,0.0158**
25	Santa Cruz County,AZ,0.0000	Cumberland County,PA,0.0133	Howard County,MD,0.0199
26	Guadalupe County,NM,0.0004	Ocean County,NJ,0.0049	Frederick County,MD,0.0175
27	Hood County,TX,0.0018	Dallas County,TX,0.0070	Harford County,MD,0.0158
28	Arapahoe County,CO,0.0072	Bucks County,PA,0.0216	Bernalillo County,NM,0.0139
29	Santa Fe County,NM,0.0075	Burlington County,NJ,0.0062	Montgomery County,MD,0.0125
30	Tarrant County,TX,0.0085	Queen Anne's County,MD,0.0000	Hancock County,WV,0.0093

Table 2.5. The top 30 spatial outlier candidates detected by z, weighted z, and $AvgDiff$ algorithms.

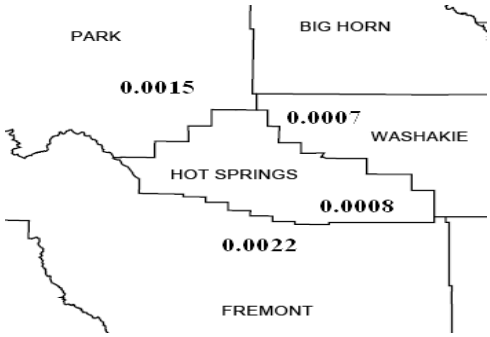


Figure 2.13. The 2003-Vet WNV case density for Hot Spring Co.(WY) and its neighbors

Spring Co.(WY) is marked as the 2nd outlier by z algorithm. However, it does not appear in the top 30 outliers for both the weighted z and $AvgDiff$ algorithms. As shown in Figure 2.13, the attribute value of Hot Spring Co. (0.0008) is much different from the average of its three neighbors, Park Co., Washake Co., and Fremont Co., whose attribute values are 0.0015, 0.0007, and 0.0022 respectively. That is why it is selected by z algorithm. However, if we consider the impact of center distance and common border length, Washake Co. has a large weight since its center distances to HotSpring Co. is short and their common border is long. The attribute of Washake Co. is similar to that of Hot Spring Co.. Therefore, its large spatial weight help reduce the difference between Spring Co. and its weighted neighborhood average. Consequently, Hot Spring Co. can not be recognized by weighted z algorithm and $AvgDiff$ algorithm.

York Co.(PA) is identified as the top outlier by weighted z approach. However, it is only ranked the 11th by z algorithm. As shown in Figure 2.14, with attribute value of 0.0175, York Co. has 7 neighbors whose attribute values are small (0.0044, 0.0133, 0.0178, 0.0378, 0.0129, 0.0158, and 0.0078) and one neighbor, Lancaster Co., whose attribute value is large (0.0683). If we do not consider spatial weight, the big difference between York Co. and Lancaster Co. will be significantly counteracted by the other 7 neighbors. Nevertheless, when the common border length and center distance are taken into account, Lancaster Co. has a very large weight (0.21, see Table 2.6), thus dominating the average of York Co.'s neighbors. Consequently, the difference between York Co. and the weighted average of its neighbors will be large, which leads to a high ranking for York Co. by weighted z approach.

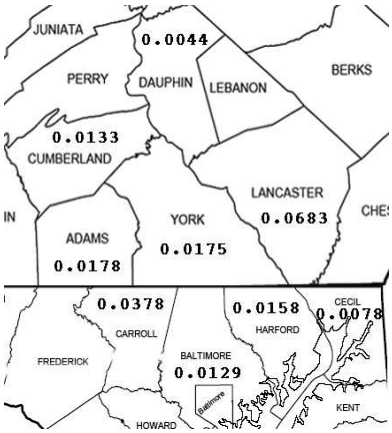


Figure 2.14. The 2003-Vet WNV case density for York Co.(PA) and its neighbors

	Adams	**Lancaster**	Dauphin	Cumberland	Carroll	Harford	Baltimore	Cecil
Common Border	44.04 (km)	**66.18** (km)	23.45 (km)	31.52 (km)	21.11 (km)	31.67 (km)	24.98 (km)	4.46 (km)
Center Distance	38.43 (km)	**42.25** (km)	44.48 (km)	45.14 (km)	47.53 (km)	58.31 (km)	62.71 (km)	77.16 (km)
Weight	0.16	**0.21**	0.12	0.13	0.11	0.12	0.10	0.05

Table 2.6. The common border length, center distance, and weight of the 8 neighbors of York Co. (PA)

$AvgDiff$ algorithm vs. z algorithm

Another weighted algorithm, $AvgDiff$, also has significantly different result compared with z algorithm. As shown in Table 2.5, there are 16 different counties in the top 30 outliers detected by these two algorithms. For example, Camden Co. (NJ) is identified by $AvgDiff$ algorithm with a ranking of 22th. Nevertheless, it is not marked as an outlier by z algorithm. In Figure 2.15, we can see Camden Co., with attribute value of 0.0087, has four neighbors, Gloucester Co. (0.0321), Philadelphia Co. (0.0029), Burlington Co. (0.0062) and Atlantic Co.(0.0062). If spatial weight is not taken into consideration, the high value of Gloucester Co. will be offset by the other three neighbors when calculating the neighborhood average. In this case, the difference between Camden Co. and the average of its four neighbors is 0.0032. When the spatial weight is considered, Gloucester Co. possesses a bigger weight (0.37) than other neighbors since it has a long common border and short center distance with Camden Co.. Thus, the large difference between Gloucester Co. and Camden Co is strengthened so that it dominates the difference between Camden Co. and the overall characteristics of its neighborhood. The average absolute difference between Camden Co. and its neighbors is 0.0106, which makes Camden Co. the 22th outlier by $AvgDiff$ algorithm. Note that Camden Co. is also selected as the 19th outlier by the weighted z algorithm.

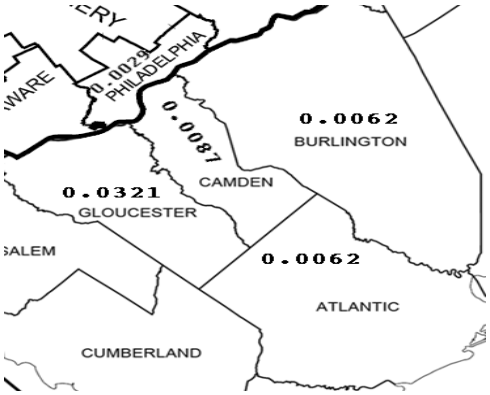


Figure 2.15. The 2003-Vet WNV case density for Camden Co.(NJ) and its neighbors

	Gloucester Co.	Philadelphia Co.	Burlington Co.	Atlantic Co.
Common Border Length	**42.27km**	16.74km	41.50km	18.69km
Center Distance	**15.84km**	17.38km	23.62km	58.43km
Weight	**0.37**	0.22	0.29	0.12

Table 2.7. The common border length, center distance, and weight of the 4 neighbors of Camden Co. (NJ)

Weighted z algorithm vs. $AvgDiff$ algorithm

Weighted z value algorithm and $AvgDiff$ algorithm have similar results, identifying the same 22 counties in the top 30 outliers but in different order. For example, Berks Co.(PA) is selected as the 2nd outlier by weighted z algorithm, but ranked as the 6th outlier by $AvgDiff$ algorithm. Lancaster Co., the top outlier for $AvgDiff$ algorithm, is marked as the 7th outlier by weighted z algorithm. The ranking variation is caused by the different mechanisms used by these two algorithms to calculate the average neighborhood difference.

There are 8 outlier counties identified by $AvgDiff$ algorithm but not identified by weighted z algorithm. Anne Arundel Co.(MD) is an example, which is ranked as the 24th outlier by $AvgDiff$ algorithm. As shown in Figure 2.16, Anne Arundel Co. has 4 neighbors, Baltimore city, Howard Co., Prince George's Co., and Baltimore Co. The attribute value (0.0199) of Howard Co. is larger than that of Anne Arundel Co. (0.0158), while the other 3 neighboring counties have smaller attribute values (0.0000, 0.0000, 0.0129) than Anne Arundel Co.. Thus, the difference between Howard Co. and Anne Arundel Co. is "neutralized" by the difference between other 3 neighbors and Anne Arundel Co., which makes the weighted z algorithm unable to identify Anne Arundel Co.. $AvgDiff$ does not have this "neutralization" issue, because it uses the absolute difference

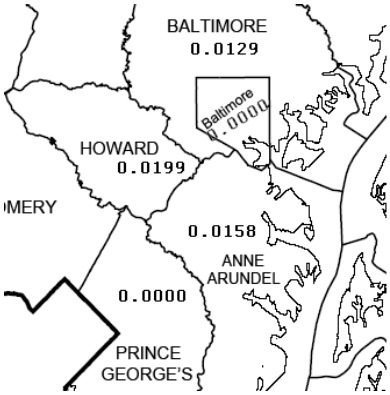


Figure 2.16. The 2003-Vet WNV case density for Anne Arundel Co. (MD) and its neighbors

between a county and its neighbors.

2.3 Summary

In this chapter, a suite of algorithms are discussed to detect single attribute outliers. The median-based and iterative-based methods are designed to reduce the "masking" and "swamping" problems. When multiple outliers exist in a spatial data set, some outliers may have negative impact on its neighbors so that true outliers are ignored and normal objects are erroneously identified. Median-based method can alleviate this problem because it is a robust estimator of the center of a sample and its value will not be impacted by the neighbor with the largest or smallest attribute values. Iterative-based methods are effective because they detect multiple outliers in multiple iterations. After one outlier has been identified, its attribute value is replaced with the average of its neighborhood. Therefore, it will not have negative impact on the following iterations. Median-based method does not require multiple iterations, which makes it more efficient than the iterative methods. However, if the values in a sample set are very irregular and not evenly distributed, median may not represent the average of a sample appropriately. Iterative methods can be used in this situation. In addition, if the number of outliers are not very large, iterative methods can be selected as well. As for the iterative-r and iterative-z, which one to choose is dependent on the requirements of different applications. If the application focuses more on the difference among neighbors, the iterative-z should be used. If the application pays more attention to the ratio between an object and its neighbors, iterative-r should be selected.

To consider the impact of spatial relationship in neighborhood comparison, we proposed two

algorithms: weighted z and $AvgDiff$. They share the same weight computation scheme where multiple related spatial properties have their own impacting factors. The only difference between these two algorithms is that weighted z uses z-value to measure the neighborhood difference and $AvgDiff$ employs the aggregated absolute difference between the center object and each of its neighbor. $AvgDiff$ can be selected if the users focus on measuring the absolute difference among neighborhood. The performance comparison between the two weighted algorithms is highly dependent on the data set and domain experts. From the view of data mining, these two proposed algorithms focus on rendering a filtering mechanism to present a small set of outlier candidates for further investigation by domain experts.

Table 2.8 shows the time complexity comparison between all above methods, where m denotes the number of iterations (or the number of detected outliers). The computation cost is mainly determined by the kNN search. If we choose grid-based indexing, iterative-based method has higher computation cost than other methods. If R-tree is used for indexing spatial objects, all these methods have similar time cost.

	Methods				
Time complexity	Median alg.	Iterative-z	Iterative-r	Weighted z	$AvgDiff$
grid indexing	$O(n)$	$O(mn)$	$O(mn)$	$O(n))$	$O(n))$
R-tree indexing	$O(nlogn)$	$O(nlogn)$	$O(nlogn)$	$O(nlogn)$	$O(nlogn)$

Table 2.8. The time complexity comparison between Median algorithm, Iterative-z, iterative-r, weighted z, and $AvgDiff$ algorithms.

Chapter 3

Region Outlier and Change Detection

Detecting spatial outliers can help identify significant events in spatial data sequences. In the field of meteorological data processing, spatial outliers are frequently associated with natural disasters such as tornadoes and hurricanes. Previous studies on spatial outliers mainly focused on identifying single location points in the static data. In this chapter, we define a new type of spatial outlier, region outlier, and propose a set of algorithms to identify region outliers (abnormal weather events, e.g., hurricanes) in continuous meteorological data sequences and track their movements. In addition, two wavelet based change detection algorithms are presented for capturing both temporal changes and spatial movements of weather patterns in meteorological data.

3.1 Region Outlier Detection

In this section, we propose and implement a systematic approach to detecting and tracking region-outliers in a sequence of meteorological data frames. First, a wavelet transformation such as the Mexican Hat or Morlet is used to sharpen and enhance the data variation. Second, an image segmentation method, λ-connected segmentation, is employed to identify the outlier regions. Finally, a regression technique is applied to track the center movement of the outlier regions for consecutive frames. In addition, we conducted experimental evaluations using real-world meteorological data and events, in this case, Hurricane Isabel, to demonstrate the effectiveness of our proposed approach.

3.1.1 Problem and Approach

As the most widely-used spatial data, geographic data not only deals with three dimensional volumes, but also contains temporal information. Together, these form spatial data sequences. In recent years, spatio-temporal data has attracted a great deal of attention from computer scientists, geographers, environmental researchers, resource managers, and biologists. This data contains complex structures, arrives continuously, evolves over time, and needs to be processed in real time. However, unlike a video stream, the frame sampling period can be as long as minutes, and there are no strict restrictions on processing speed. Several recent studies have been conducted to develop specific data mining techniques to detect useful patterns from continuous data streams [22, 29, 41]. Because these techniques are not specifically designed for processing spatial data, they may not be effectively utilized by geospatial applications. Intensive research is therefore needed to extract patterns from spatio-temporal data and to accurately predict their trends [21, 54].

Outlier detection is a process that is often used to identify objects which differ from the other members in the same data set [7, 38]. In the research on atmospheric sciences, huge amounts of spatial data are continuously collected from both observation and simulation modelling. Discovering useful patterns from these data, especially spatial outliers and their movements, will have great practical value for weather forecasting, environmental monitoring, and climate analysis. In meteorological data, spatial outliers are those observations that are inconsistent with their surrounding neighbors. Spatial outliers or anomalies are often associated with severe weather events such as tornadoes and hurricanes. These events do not usually happen at a single location but cover an extended region, so spatial outliers are usually two dimensional regions. Furthermore, the spatio-temporal changes in these regions are frequently associated with variations of weather phenomena and climate patterns.

The ability to automatically extract these outlier regions is therefore a crucial issue. Typically, the methods used to address this problem rely on image segmentation and pattern recognition techniques [31, 51]. Image segmentation divides an image into constituent regions. This technique has been widely used in several practical applications, such as military satellite image analysis. Wavelet transformation is an important tool for digital signal processing, image processing, and data mining. Wavelet transformation can represent data in a hierarchical structure, with multiple resolutions ranging from gross to fine. In addition, it can provide the time and frequency information simultaneously, thus rendering a time-frequency representation of the signal. Another advantageous property of wavelet transformation is that it can distinctly capture the differences between a data

item and its neighboring items [53].

In the Earth's atmosphere, anomalies emerge at different spatial scales and may appear in different shapes, which presents a daunting challenge to those seeking to detect outliers from continuous meteorological data sequences. Figure 3.1 shows an image of the water vapor distribution over the east coast of the U.S., the Atlantic Ocean, and the Gulf of Mexico. The color intensity of each region reflects its water vapor content, and the "hot spot" located in the left portion of the image (28^oN, 90^oW) indicates a hurricane in the Gulf of Mexico. This outlier spot is not a single point but a group of points, a region. This region has a much higher water vapor content than its surrounding neighbors. Thus, *a regional outlier is a group of adjacent points whose features are inconsistent with those of their surrounding neighbors.* The red-colored hot spot, a hurricane, in Figure 3.1 is a regional outlier. Regional outliers are determined by domain experts based on a predefined threshold. The challenge is to design an efficient and practical approach to automatically detect regional outliers, which could be in irregular shapes, from spatial data sequences. In real applications, such approaches can help identify spatial anomalies such as hurricanes, tornadoes, thunder storms, and other severe weather events in the observation data.

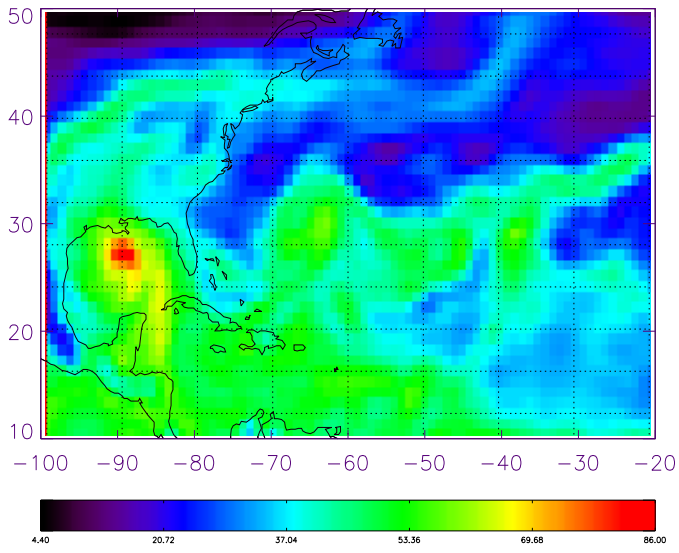


Figure 3.1. A regional outlier (hurricane) in meteorological data.

In order to accurately extract regional outliers, it is preferable to decompose the original observations into different spatial scales in order to reduce the complexity and centralize the target object. Wavelet transformation provides such a capability with its multi-resolution characteristics.

First, wavelet tranformation can be used to decompose the original spatial variation of the data into different scales, allowing users to focus on the scale of interest and identify the potential outliers at that scale. Second, the localization of variations in the frequency domain is useful in determining the spatial location of outliers.

In this application, we will apply wavelet transformation in the real spatial domain and then analyze the transformed data for a particular set of scales. As spatial outliers are usually small in size compared with the environment, relatively small scales will be selected for hurricanes and tornadoes. The wavelet power indicates the strength of the variation and the localization of high values reveals the places where anomalies exist. In the next section, we will discuss how the wavelet transformation is used in our application.

Image segmentation can be employed to extract spatial regions within which the meteorological characteristics are similar. The segmentation algorithm needs to be fast in order to process sequential frames and even high-speed image streams. For example, the selected algorithms should not scan the whole frame multiple times. Ideally, the original frame or part of the original frame should be scanned only once. With $O(nlogn)$ time complexity, split-and-merge methods would not be practical for this purpose. K-means and fuzzy c-means, as well as the Mumford-Shah method, need extensive computation time because they require numerous iterations. Thus, to achieve a satisfactory speed, the threshold method and λ-connected method are the only two options since they both have linear time complexity. Threshold segmentation seems to offer the simplest solution, but when an image involves multiple thresholds, the determination of these thresholds values will be both difficult and time-consuming. The advantage of the λ-connectedness approach is that it can determine segments at different intensity levels without the need to calculate different thresholds or clip-level values. Based on the above reasoning, we chose the λ-connectedness approach to segment the meteorological data.

Our goal here is to identify the largest outlying region in which the value of each pixel is above a reasonable, predefined threshold. If we select the threshold method, the image is translated into a binary image based on a specified threshold, then a breadth-first search algorithm is used to label each connected component and select the largest one. The major advantage of this approach is that the process is easy to perform. Its disadvantage, however, is that it does not tolerate any noise. Using a λ-connected search algorithm [14], we can start with any pixel above a threshold, and find all the neighbors that have similar values by comparing them with the starting pixel. This method is therefore a generalized version of the threshold method. The details of a λ-connected search are

described as follows.

An image is a mapping from a two dimensional space to the real space R. Without loss of generality, let Σ_2 be the two-dimensional grid space, the 2D digital space. A digital image can be represented by a function: $f : \Sigma_2 \to [0, 1]$. Let $p = (x, y), q = (u, v) \in \Sigma_2$, and p, q are said to be adjacent if $\max\{\|x-u\|, \|y-v\|\} \leq 1$. (A pixel, i.e. picture element, is a pair of $(p, f(p))$.) So, if p, q are adjacent and $f(p), f(q)$ have only a "little" difference, then pixels $(p, f(p))$ and $(q, f(q))$ are said to be λ-adjacent. If there is a point r that is adjacent to q and $(q, f(q))$, $(r, f(r))$ are λ-adjacent, then $(p, f(p)), (r, f(r))$ are said to be λ-connected. Similarly, we can define the λ-connectness along a path of pixels.

Mathematically, let (Σ_2, f) be a digital image. If p and q are adjacent, we can define a measure called "neighbor-connectivity" as given below:

$$\alpha_f(p, q) = \begin{cases} 1 - \|f(p) - f(q)\|/H & \text{if } p,\ q \text{ adjacent} \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where $H = \max\{f(x) | x \in \Sigma_2\}$.

Let $x_1, x_2, ..., x_{n-1}, x_n$ be a simple path. The path-connectivity β of a path $\pi = \pi(x_1, x_n) = \{x_1, x_2, ..., x_n\}$ is defined as

$$\beta_f(\pi(x_1, x_n)) = \min\{\alpha_f(x_i, x_{i+1}) | i = 1, ..., n-1\} \tag{3.2}$$

or

$$\beta_f(\pi(x_1, x_n)) = \prod\{\alpha_f(x_i, x_{i+1}) | i = 1, ..., n-1\} \tag{3.3}$$

Finally, the degree of connectivity between two vertices x, y with respect to ρ is defined as:

$$C_f(x, y) = \max\{\beta_f(\pi(x, y)) | \pi \text{ is a (simple) path.}\} \tag{3.4}$$

For a given $\lambda \in [0, 1]$, point $p = (x, f(x))$ and $q = (y, f(y))$ are deemed λ-connected if $C_f(x, y) \geq \lambda$.

If equation 3.2 applies, λ-connectedness is reflexive, symmetrical, and transitive. Thus, it is an equivalence relation. If equation 3.3 applies, λ-connectedness is reflexive and symmetrical. Therefore, it is a similarity relation.

3.1.2 Algorithm Design

In this section, we first describe a wavelet transformation on image data. Second, we design a segmentation algorithm to obtain the largest connected region whose wavelet power is above background. Third, after the center point and boundary of the region are stored, a linear regression will be employed to construct the approximate trajectory of the moving region in consecutive frames. The existence of some disturbances may introduce incorrect outlier regions. Regression can help remove these "noise" center points to obtain an accurate trajectory.

Wavelet Transformation

Wavelet transformation is a practical technique that is widely used in signal analysis and image processing. Wavelet transformation possesses several attractive features: (1)**Multi-resolution**: Wavelet transformation examines the signal at different frequencies with different resolutions, using a wider window for low frequencies and a narrower window for high frequencies. This feature works especially well for signals whose high frequency components have short durations and low frequency components have long durations. Thus, wavelet transformation is an effective tool with which to filter a signal and focus on specific scales. (2)**Localization of the frequency**: In Fourier transformation, the frequency domain has no localization information. Thus, if the frequency changes with time in the signal, it is hard to distinguish which frequency occurs within which time range, although all the frequencies may be detected. In the real world, signals are usually complicated and are non-stationary. If we want to know exact information for a variation, such as the frequency and the location of a certain variation or the strength of the variation at a certain location, wavelet transformation has an advantage over Fourier transforms.

In this paper, we use continuous wavelet transformation. For a wavelet function $\Psi(t)$, the continuous wavelet transformation of a discrete signal $X_i (i = 0, \ldots, N-1)$ is defined as the convolution of X with scaled and translated Ψ:

$$W(n,s) = \sum_{i=0}^{N-1} x(i)\Psi^*[\tfrac{(i-n)\delta t}{s}]$$

where (*) indicates the complex conjugate, n is the localization of the wavelet transformation and s is the scale. The wavelet transformation can also be inversely transformed to (or used to reconstruct) the original data set :

$$x_i = \tfrac{\delta j \delta t^{1/2}}{C_\delta \Psi_0(0)} \sum_{j=0}^{J} \tfrac{RealW(n,s_j)}{s_j^{1/2}}$$

index	0	1	2	3	4
scale	2	2.83	4	5.65	8
period	7.95	11.23	15.9	22.47	31.79

Table 3.1. Scale Table for Mexican Hat Wavelet

Where C_δ is a constant for each wavelet function; Ψ_0 is the normalized wavelet function; and J is the maximum scale index, which will be explained later. For more details of the wavelet transformation method, please refer to [98].

Here, not all scales of the wavelet transformation are included in the reconstruction. In order to filter out the non-related information, the data will be reconstructed based on the scales that are of interest. For example, if the low frequency range of the variations in the data set is to be studied, a low pass data set may be reconstructed in order to filter out the high frequency variations and make low frequency variations more visible. Many functions can be used as the base or mother function for wavelet transformations. Here, we use two of the most widely used bases: the Morlet base and the Mexican hat base. The Morlet function is:

$$\Psi_0(\eta) = \pi^{-1/4} e^{\omega_0 \eta} e^{-\eta^2/2}$$

The Mexican hat function is:

$$\Psi_0(\eta) = \frac{(-1)}{\sqrt{\Gamma(5/2)}} \frac{d^2}{d\eta^2}(e^{-\eta^2/2})$$

When performing the wavelet transformation, the scales are selected by $S_0 * 2^{j/2} (j = 0, 1, \ldots, J)$, where J is the maximum scale index, which satisfies $J \leq 2\log_2(\frac{N}{2})$, and N is the length of the signal, in this case $S_0 = 2\delta x$, $N = 360$. We use j as the scale index; Scale 2 means the real scale is $S_0 2^{0.5*2} = 4$. Table 3.1 and Table 3.2 provide the relationship between the scale index, real scale, and the corresponding period of the Fourier transform (here, since we are performing wavelet transformation on the spatial domain, it is in fact the wavelength of the spatial variation) for the Mexican hat and Morlet wavelets. From the tables, it can be seen that as the scale grows, the period (or wavelength) of the real object the wavelet focuses on also grows. However, the growth rates are different for the two wavelets. For the Morlet wavelet, the period grows more slowly than it does for the Mexican hat wavelet. Thus, the Morlet wavelet has a better frequency resolution than the Mexican hat wavelet. This also implies that Morlet has a poorer localization resolution.

index	1	2	3	...	6	7	8
scale	2.83	4	5.65	...	16	22.6	32
period	2.92	4.13	5.84	...	16.52	23.4	33.05

Table 3.2. Scale Table for Morlet Wavelet

The Morlet wavelet is a complex wavelet and the Mexican hat wavelet is a real wavelet. The Mexican hat model captures both the positive and negative variations as separate peaks in wavelet power, while the Morlet wavelet power combines both positive and negative peaks into a single broad peak [98]. Figures 3.2 and 3.3 show examples of the two wavelet transformations. Figure 3.2(a) is the original water vapor distribution along a particular latitude. Figures 3.2(b) and (c) show the wavelet transformation power at two different scales for a Mexican hat wavelet. Figure 3.3 uses the Morlet wavelet and higher scale indices. From Figures 3.2 and 3.3, we can see that the power of the wavelet transformation can be used to depict the distribution or localization of the variation at certain scales. The Mexican hat wavelet provides a better localization (spatial resolution), therefore we will use the Mexican hat wavelet to perform the analysis.

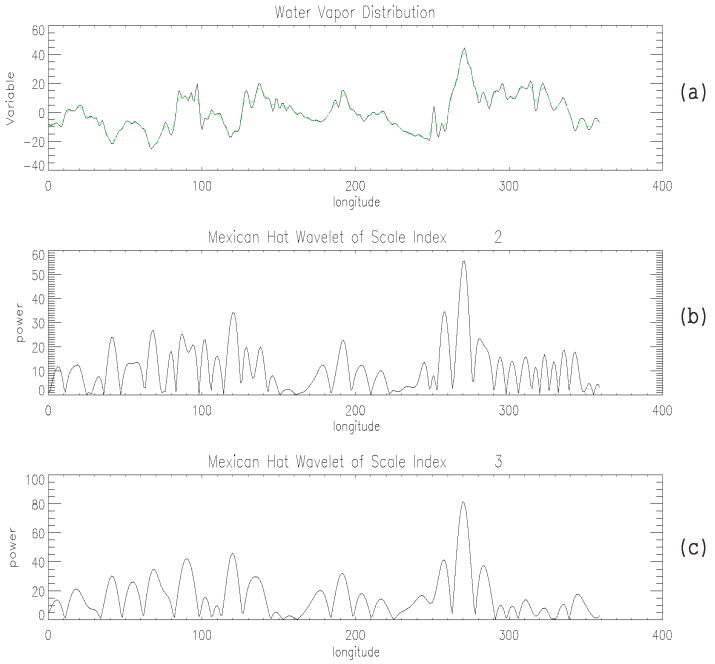


Figure 3.2. A sample output of the Mexican hat wavelet (a: original water vapor data, b: wavelet power at scale 2, c: wavelet power at scale 3).

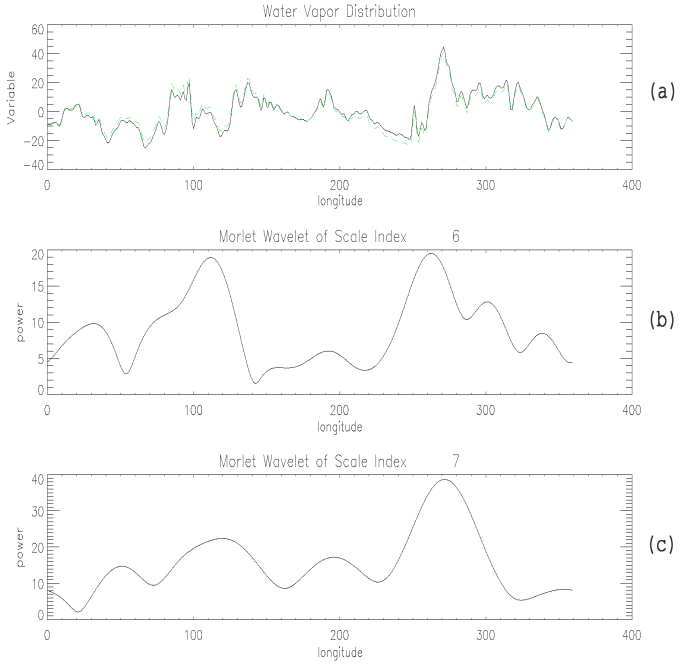


Figure 3.3. A sample output of Morlet wavelet (a: original water vapor data, b: wavelet power at scale 6, c: wavelet power at scale 7).

Detection Algorithms

The proposed algorithm has two major purposes: detecting a sequence of region outliers in consecutive frames and tracking their movements. First, a wavelet transformation is performed on the image data to identify the regions with prominent spatial variations at certain scales. Then, segmentation is employed to extract the largest outlier region and trace its trajectory. The algorithm is designed based on the following assumptions. First, the CPU speed is capable of processing at least a number k of data windows ($k \geq 1$). This means that the algorithm can process the continuous data window by window. The size of the window can be adjusted according to the arrival speed of the data sequence. Second, the data arrive in a specific sequence, for example, in the order of latitude or longitude. Each arriving data element is thus spatially adjacent to the previous data element.

The primary algorithm is *Main*, which invokes other sub-algorithms, including *WaveletAnalysis*, *Segmentation*, and *Trajectory*. The input of the algorithm *Main* includes a sequence of continuously arriving data DS, a set of selected scales S for wavelet transformation, a threshold θ, a similarity level λ for segmentation, and a trajectory T of the outlier region in previous frames. The output is the largest outlier region O_r for each image frame and its updated trajectory T.

In the algorithm *Main*, a set of scales of interest must first be determined by domain experts.

The continuous and unbounded data sequence DS will be processed in each window, and the window size will be determined by the size of each data item and the memory capacity. We designate each window to represent an integral view of the global meteorological data (180 degrees by 360 degrees) as one time frame. From the I/O buffer, a sequence of data elements are fetched and stored in window W. Then, the algorithm *WaveletAnalysis* is performed on W, and the filtered data $wDomain$ is generated, which focuses on particular scales. Next, based on $wDomain$, the algorithm *Segmentation* is employed to extract the outlier regions, which are connected components with attribute values above a predefined threshold θ. In particular, we focus on the largest connected region whose attribute values exceed the threshold, that is to say, only one region outlier will be detected. Finally, the boundary and center point of the outlier region can be calculated in order to trace the region's movement. Trajectory T will be recalculated and updated once a new region is added.

In fact, identifying the moving outlier region does not require processing the entire frame, as the locations of the outlier region in adjacent frames are not likely to change dramatically. Thus, based on the region location in the previous frame, the function $getPredictedArea()$ can define the predicted area Σ_p, an approximate rectangle which contains all the possible positions of the moving region but is much smaller than the whole image $wDomain$. Instead of processing $wDomain$, we can obtain the outlier region by applying image segmentation only to Σ_p. This way, the costs of region detection can be significantly reduced. The center of Σ_p can be obtained by considering both the region center in the previous frame and its moving speed. Note that for the first several frames, Σ_p is set to be $wDomain$ and the whole image will be processed for segmentation. This utilization of the predicted area will make the segmentation process four times faster if its size is a quarter of the original frame. However, the area cannot be too small in order to maintain the quality of the search.

The three sub-algorithms are discussed in detail as follows. The algorithm *WaveletAnalysis* is designed to filter out unimportant information from the source image data. The input of this algorithm is a sequence of data points W and a set of selected scales S. The output is the filtered image. Performing the wavelet transformation and reconstructing data based on a particular subset of scales can help filter noise and identify patterns with particular sizes. The algorithm first extracts the boundary of W. α_1 denotes the beginning latitude or longitude, and α_n denotes the ending latitude or longitude of the current window. Note that for meteorological data, the wavelet transformation will be performed along latitude lines. We will discuss the justification for this in

Algorithm : Main

Input:
DS is a data sequence
S is a set of selected scales;
θ is the threshold used for segmentation;
λ is the similarity level for segmentation;
T is the trajectory of the outlier region in the previous frames;

Output:
O_r is the set of points in the outlier region
T is the trajectory after appending the outlier region in the current frame

```
T = φ;
/* continuously process the sequence window by window */
while (true) {
   /* get a window of data from the sequence */
   W = getWinFromBuf(DS);
   /*Call algorithm WaveletAnalysis to process current window*/
   wDomain = WaveletAnalysis(W,S);
   /* Define the predicted area to speed the image segmentation */
   Σ_p = getPredictedArea(wDomain, T);
   /* Call algorithm Segmentation to obtain the largest region*/
   O_r = Segmentation(Σ_p, θ, λ);
   /* Call algorithm Trajectory to track movement*/
   T = Trajectory(T, O_r);
   /* output the detected region and its moving trajectory*/
   Output(O_r, T); }
```

the experimental section of this paper.

The algorithm *Segmentation* aims to extract the largest connected region above a threshold θ. It contains three input parameters: Σ, θ, and λ. The parameter Σ denotes the set of data points to be segmented; θ is a threshold to filter-out unwanted points (points whose values are less than θ will not be processed); and λ is the similarity level. The value of θ is determined by domain experts. Ordinarily, we will designate it as 75 percent of the difference between the maximum value and the minimum value of the data set. The output is the largest connected component in the data set, consisting of points with values greater than θ and similarity levels greater than λ. First, the algorithm picks a point p_0 from Σ whose value is greater than θ and is not labelled as '*', which means "not processed." Then, p_0 is added into $QUEUE$. For each point in this $QUEUE$, its "unprocessed" neighboring points will be examined to see if they have a similarity level greater than λ. If the condition holds, the corresponding neighboring point will be stored into $QUEUE$ and marked as "processed." Repeating the "marking" process for all the points in the $QUEUE$, we can obtain a result set S', containing the connected part of Σ. Next, the number of points in S and

Algorithm : WaveletAnalysis

Input:
W is a data window from the sequence;
S is a set of selected scales;

Output:
wDomain is the wavelet power of the data window

```
/* get the minimum latitude(or longitude) of current window */
α1 = getMinBound(W);
/* get the maximum latitude(or longitude) of current window */
αn = getMaxBound(W);
/*wavelet transformation and inverse transformation along all latitudes(or longitudes)*/
for(i=α1;i ≤ αn ;i++) {
   wDomain = WaveletTransform(W,S,i); }
/* output the filtered data window*/
Output(wDomain);
```

S' will be compared. If S is smaller than S', S will be replaced by S', ensuring that S maintains the largest component discovered so far. The loop repeats until there is no "unprocessed" point with a value greater than θ. Finally, S is returned as the largest component discovered by the algorithm.

The objective of the algorithm *Trajectory* is to track the moving direction and speed of a given region and to determine the validity of the current detected region. The input parameters are the previously recorded trajectory T, the newly detected region R, and the number K of the latest points in T. The data structure of T includes the time, center, moving speed, and boundary of previous K regions. The detected region R is from the output of the *Segmentation* algorithm. It is possible for a region to be erroneously detected by the algorithm *Segmentation* due to errors in the raw data or an inappropriate segmentation threshold. Therefore, a verification function is needed in order to determine the validity of R based on the trajectory of the previous K regions. In the algorithm, the boundary point B is first extracted and the center C of the region R is computed. Then, a verification procedure is performed to compare C with the statistics of the past K center points along the trajectory. The mean μ and standard deviation σ of the past K center points are calculated. If C is located within 2σ from μ, R is considered as a valid region and C is appended to the trajectory T. Otherwise, R is flagged as a "noise" point that will be discarded. The speed and moving direction of the region center can thus be obtained from two valid consecutive center points. Finally, the new trajectory T will be updated and stored in permanent storage for a specified period of time.

Algorithm: Segmentation

Input:
Σ : Set of data points
θ : Threshold for the clip level
λ: Similarity level

Output:
S: the largest connected component with value above θ

```
Σ = ∅; {
while (Σ contains unlabelled points)
   p0 = pickOneUnLabeledPoint(Σ, θ);
   L(p0) = '*'; /*labeling p0 as processed*/
   /*insert p0 into a Queue*/
   QUEUE = InsertQueue(QUEUE, p0);
   while (not Empty(QUEUE)){
      /*get an element from the head of QUEUE*/
      p0 = RemoveQueue(QUEUE);
      For each p that is adjacent to p0 {
         if (L(p) ≠ '*' and C(p, p0) ≥ λ)
            QUEUE = InsertQueue(QUEUE,p);
            L(p) = '*'; }}}
   S' = {p : L(p) = 0}; /*S' is a λ-connected component*/
   if (S' has more points than S)
      S = S'; /* save the largest component to S*/
}
return(S);
```

Time Complexity and Memory Usage

The water vapor attribute value of each point is represented by a 4-byte double. If one window contains all the global water vapor data for a specific time (360*180 locations), it will take 260K byte of memory. The computation of the wavelet transformation is efficient. A fast wavelet transformation needs $O(N)$ operations, where N is the number of objects (locations). Its memory usage is also linear [53]. For each data window, the time complexity of the *WaveletAnalysis* algorithm is $O(m)$, where m refers to the window size (or number of pixels in the image). The time complexity of identifying the largest λ-connected part is $O(m)$, because in the search algorithm, each pixel will be visited twice. This also validates that the breadth-first based search technique is an efficient searching algorithm. For trajectory tracking, the time complexity is $O(p + K)$ where $O(p)$ is used for extracting the boundary and center point of the outlier region (with an average of p points) and $O(K)$ is the cost of "noise" point elimination and speed calculation. Since p and K are very small compared to m, the running time will be dominated by the wavelet transformation and image segmentation operations. The total time complexity will correspond to the total number of objects

Algorithm : Trajectory

Input:
T: Previous trajectory;
R: Current detected region;
K: Number of latest center points along T;
Output:
T: Updated trajectory with R appended

/* extract boundary of in R */
B = getBoundary(R);
/* Calculate the central point of R */
C = getCenter(R);
/*Eliminate "noise" points*/
T = verification(T,C,K);
/*Compute the moving speed of center point*/
T = calculateSpeed(T);
/*Output the new trajectory*/
Output(T);

N(the aggregation of m for all windows), that is, $O(N)$.

3.1.3 Experiments

We used the NOAA/NCEP (National Oceanic and Atmospheric Administration/National Centers of Environmental Prediction) global reanalysis data set, which provides multiple parameters with a resolution of 1 degree $\times$ 1 degree. This data set covers the entire globe and is updated 4 times a day, at 0AM, 6AM, 12PM, and 6PM. Our main objective was to trace hurricanes or tropical storms by studying water vapor data from satellites. Even though a hurricane is not defined by a high concentration of water vapor, it is always accompanied by a high concentration of water vapor. Usually, the stronger the circulation wind, the lower the surface pressure, the stronger the convection, and the higher the concentration of water vapor. Although surface wind and surface air pressure are generally considered better indicators of a hurricane, these parameters are very difficult to retrieve from satellite observation under cloud cover, especially for hurricanes, which have deep convections and thick clouds. In contrast, the total water vapor (integrated from the surface to the top of the atmosphere) is a well-validated satellite product which provides a good estimation of the situation even under heavy cloud. Figure 3.4 shows an image of global water vapor distribution on October 3, 2002. In most cases, the tropical region is covered by high values for the water vapor. Our methodology can be used to identify high concentrations of water vapor that could be caused by hurricanes. The results need to be sent to domain experts for further

validation.

The human eye can visually identify and track this type of extreme weather phenomena. Actually, the current weather forecast system is semi-automatic, mainly based on human recognition. However, when huge amounts of meteorological data continuously arrive, even the most experienced expert will be overwhelmed. Human eyes alone cannot handle a large number of data frames in a short period of time. In addition, humans are prone to making mistakes due to many factors such as distractions, lack of experiences, operator errors, and fatigue.

In order to automate accurate outlier identification and tracking, we proposed our systematic approach, which can handle continuous data sequences. The wavelet transform can focus on particular spatial scales and filter out the unnecessary and distracting information. Also, after the wavelet transformation, the difference between adjacent locations will be highlighted. Fast image segmentation algorithms can automatically extract regions with extreme weather patterns within a short period of time. Tracking functionality quickly reveals the movement of a weather pattern, with accurate information on its location, direction and speed, which is impossible even for human experts to find. In summary, the ultimate objective is to provide an efficient machine-based anomaly detection method to relieve weather forecasters and climate analyzers from intensive and error-prone work.

In this section, we will demonstrate the experimental results of wavelet transformation, image segmentation, and trajectory tracking.

Wavelet Transformation

We first performed a Mexican hat wavelet transformation on the data over all latitudes. Figure 3.5 shows the water vapor data for 26^o North and its wavelet power. In Figure 3.5(a), the solid line is the original data and the dashed line is the filtered data (reconstructed with scales 2 and 3). Figure 3.5(b) is the plot of the wavelet power of the original data. Figure 3.5(c) is the plot of the wavelet power of the filtered data. Figure 3.5 shows that the variation exists on all scales and the power of the variation changes at different locations. This figure also shows that the Mexican hat wavelet has a satisfactory localization resolution. We mainly focused on the anomalies with sub-weather scales, that is with variations of 1000km or 10 degrees in longitude at the mid-latitude region. Figure 3.6 shows the global map of wavelet transformation power with the scale index 3. Clearly, there are some areas where the power is especially high. In these areas, the spatial variations with the scale index 3 are prominent, and therefore these areas are viewed as suspected

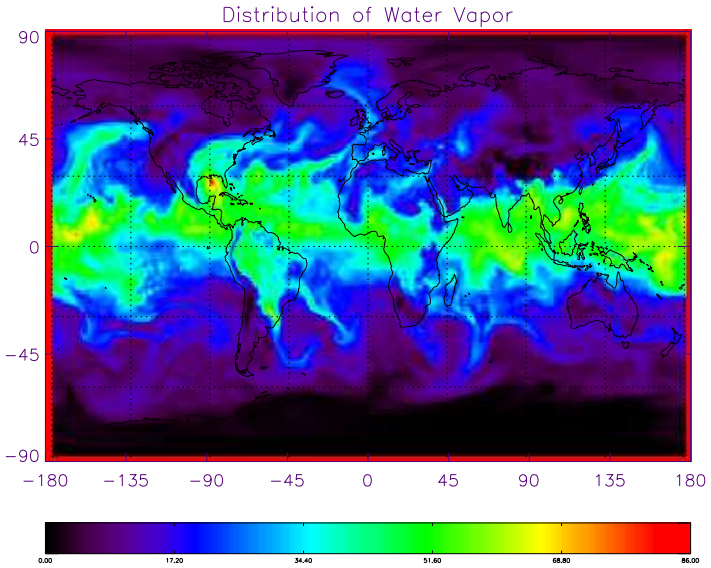


Figure 3.4. Global distribution of water vapor.

region outliers.

Comparing Figure 3.6 with Figure 3.4, in Figure 3.4, the area over the Gulf of Mexico with a high value also has a high wavelet power. However, the high vapor value areas near 160^oW in the tropic region do not show strong wavelet powers in Figure 3.6, and the low value areas in South America show high wavelet powers in Figure 3.6. Therefore, a high value does not necessarily guarantee a high wavelet power. Here, we focus on the spatial variation, not the value of the variable. Wavelet power mainly represents the variation of the signal in the spatial domain. Another advantage of using a wavelet transformation is its multi-scale capability, as mentioned earlier: we are able to focus only on the scales in which we are interested. This makes it easier to study the complicated variations in multi-scale meteorological data.

We performed the wavelet transformation in the X dimension along lines of latitude because for weather systems, the scale is usually represented based on the latitude. For the basic atmospheric parameter distribution, there is a strong variation between different latitudes such as between the tropics and high latitude areas. This variation is the normal pattern of the general atmosphere and is not an anomalous feature. Thus, when detecting spatial variations, it is useful to focus on the variation along the latitude line (X-axis). Technically, however, we can also perform a wavelet transformation along the longitude line (Y-axis). Figure 3.7 shows the reconstructed water vapor distribution using an inverse wavelet transformation along both the latitude line and longitude line (X and Y). Figure 3.7 reveals many more patterns than Figure 3.6. However, these patterns are

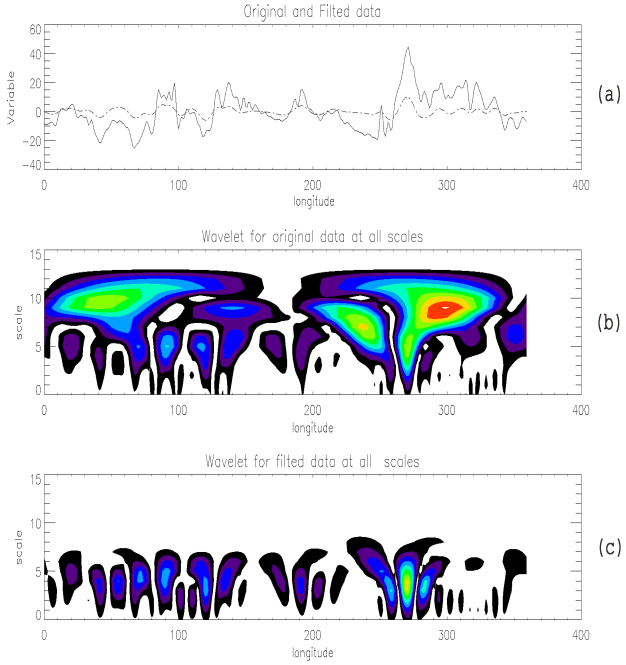


Figure 3.5. Mexican hat wavelet power with locations and scales (a: original and filtered data, b: wavelet for original data at all scales, c: wavelet for filtered data at all scales).

caused by the normal variations along the longitude Y and are merely noise in most cases.

Image Segmentation and Tracking

In this experiment, we examined the the water vapor data over the period of 9/17/2003-9/19/2003, during which Hurricane Isabel landed on the east coast of the United States. Hurricane Isabel formed in the central Atlantic Ocean on September 6th, 2003. It moved in a generally west-northwestward direction and strengthened to a category five hurricane by September 11th. Weakening began on September 16th as the hurricane turned north-northwestward. On September 18th, Isabel made landfall on the outer banks of North Carolina as a category two hurricane, while portions of eastern North Carolina and southeastern Virginia experienced hurricane-force winds. The experimental results for Hurricane Isabel demonstrate the effectiveness of our algorithms in detecting abnormal meteorological patterns. Figure 3.8 shows the wavelet image at 0AM on September 18th, 2003. When the boundary of Hurricane Isabel was extracted by the algorithm *Segmentation*, it showed that the center is located at (32.54°N, 71.80°W). Figure 3.9 shows another experimental result on September 18th, 2003, at 6:00AM. The boundary of Hurricane Isabel was clearly identified, showing the center was located at (33.05°N, 72.28°W). During these six hours, the trend of Hurricane Isabel can be observed as it moves northwestward overland.

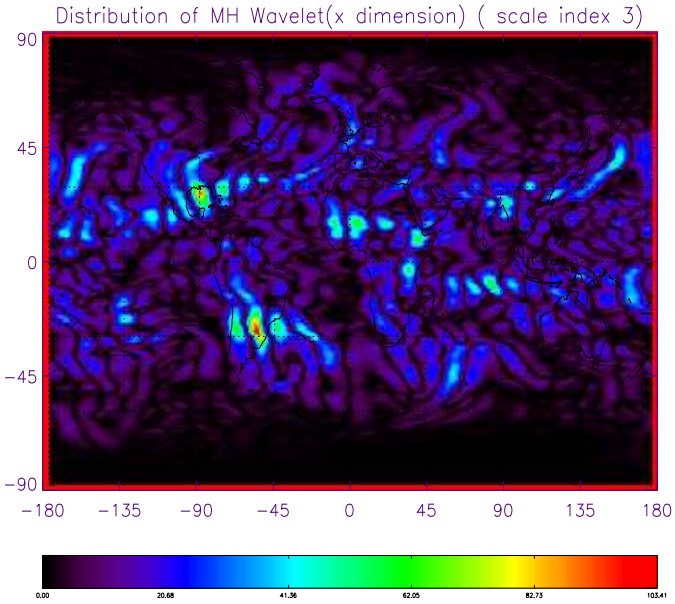


Figure 3.6. Wavelet power distribution at scale index 3.

Figure 3.10 shows the 3D trajectory of Hurricane Isabel from September 17th, 2003 to September 19th, 2003. Since the location of hurricane was measured every six hours, 12 regions are illustrated in this figure covering these three days. The boundary of each outlier region is depicted by a dotted line and the center points are connected so that its trajectory can be observed. As can be seen from the figure, region 4 is not consistent with the locations of the other regions. It is a "noise" outlier caused by other weather patterns or inappropriate segmentation parameters. Region 4 is flagged by the verification procedure in the algorithm *Trajectory*. Figure 3.11 shows the new trajectory after eliminating the "noise" region. The northwestward movement of Hurricane Isabel can be clearly

SN	Latitude	Longitude	Time	Flag
1	35.27	-70.07	09/17/2003/0Z	1
2	34.41	-70.42	09/17/2003/6Z	1
3	33.31	-71.28	09/17/2003/12Z	1
4	-29.20	-167.82	09/17/2003/18Z	0
5	32.54	-71.80	09/18/2003/0Z	1
6	33.05	-72.28	09/18/2003/6Z	1
7	33.91	-72.34	09/18/2003/12Z	1
8	34.53	-72.70	09/18/2003/18Z	1
9	38.05	-74.86	09/19/2003/0Z	1
10	41.41	-76.52	09/19/2003/6Z	1
11	43.61	-78.68	09/19/2003/12Z	1
12	45.46	-78.97	09/19/2003/18Z	1

Table 3.3. The tracking data of hurricane center.

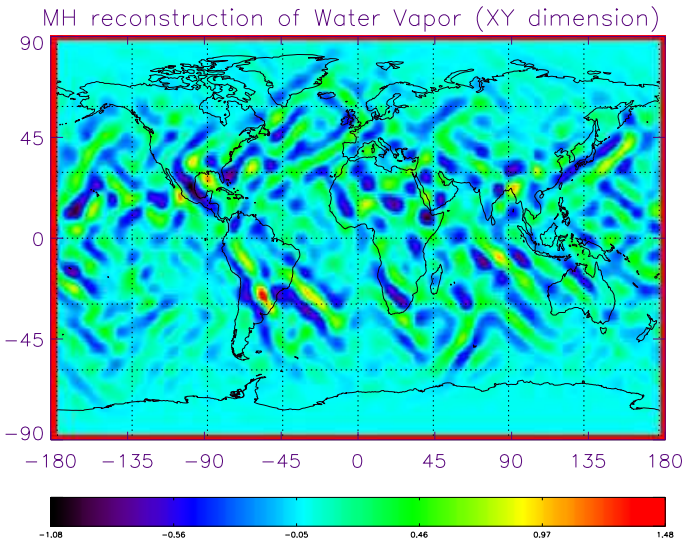


Figure 3.7. Reconstruction on both X Y dimension.

Data Size (180×360)	1	4	9	16	64
Time (Sec)	0.003	0.017	0.030	0.048	0.218

Table 3.4. The execution time of image segmentation.

observed. The latitude and longitude of the hurricane center are listed in Table 3.3. "Flag=1" denotes that the region is correctly detected and "Flag=0" denotes that the region is "noise" data and is not recorded.

Table 3.4 shows the processing time of the proposed λ-connectedness based image segmentation algorithm. The size denotes the number of data frames, where each frame is made up of 180×360 data points, and the time is measured in seconds. In this experiment, we used a Pentium4 (2.8GHz) PC with 512MB memory. The experimental results show that our image segmentation algorithm can efficiently process a high speed meteorological data sequence, taking only 0.218 seconds to process 64 image windows, each window containing 180×360 points.

Comparison with other image segmentation methods

Image segmentation directly impacts the performance of region outlier detection in continuous data sequences. To validate the efficiency and effectiveness of the proposed λ-connectedness algorithm, two widely used methods, K-Means and Maximum Entropy, were compared. Three key elements are considered in the comparison: segmentation quality, stability, and running time. Stability

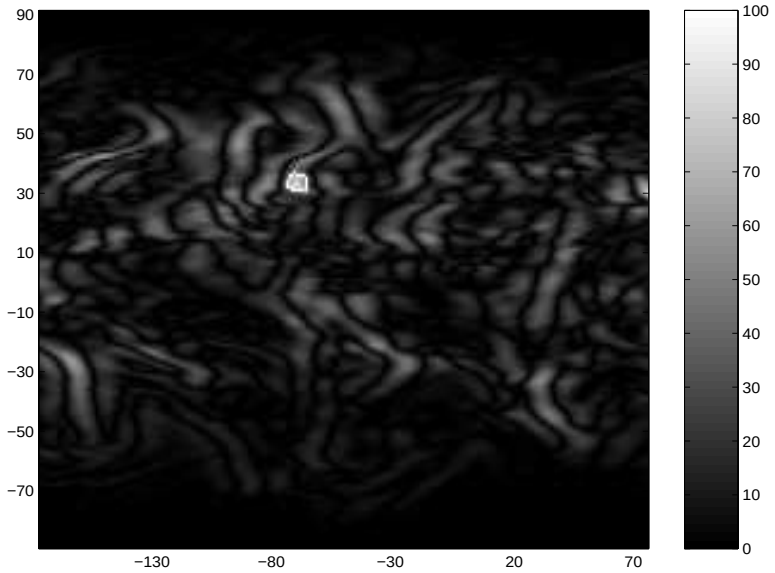


Figure 3.8. Water vapor data at 0AM Sept. 18th, 2003 with Hurricane Isabel identified.

measures the parameter variance between consecutive images. For continuous data sequences, a stable segmentation method is preferred, which has smaller parameter variance.

The three methods are conducted on two source images in the water vapor data which have been preprocessed by wavelet. Figure 3.12(a) shows Image 1, the data collected at 0AM Sept 17, 2003. Figure 3.12(b) illustrates Image 2, the data collected at 6AM Sept 18, 2003. The boundary of Hurricane Isabel is highlighted in white in both images and the center is at (35.27°N, 70.7°W,) and (33.05°N, 72.28°W), respectively. For K-Means image segmentation, we experimented on various K values and demonstrated the results for K=2, 4, and 8, respectively. For the Maximum Entropy method, the results are recorded for both single threshold and two thresholds. The λ-connectedness approach is tested based on $\lambda = 0.95$ and $\theta = 45\%$.

(1) Quality and stability comparison:

***K*-Means**

The results of the K-Means segmentation are illustrated in Figure 3.13. Figure 3.13(a) and Figure 3.13(b) show the results for $K = 2$, where Image 1 and Image 2 are segmented into 2 distinct gray levels, dark gray and black. The largest connected component is marked within a white rectangle. For both figures, the results are not satisfactory since the largest connected regions are not Hurricane Isabel. The results for $K = 4$ are illustrated in Figure 3.13(c) and Figure 3.13(d),

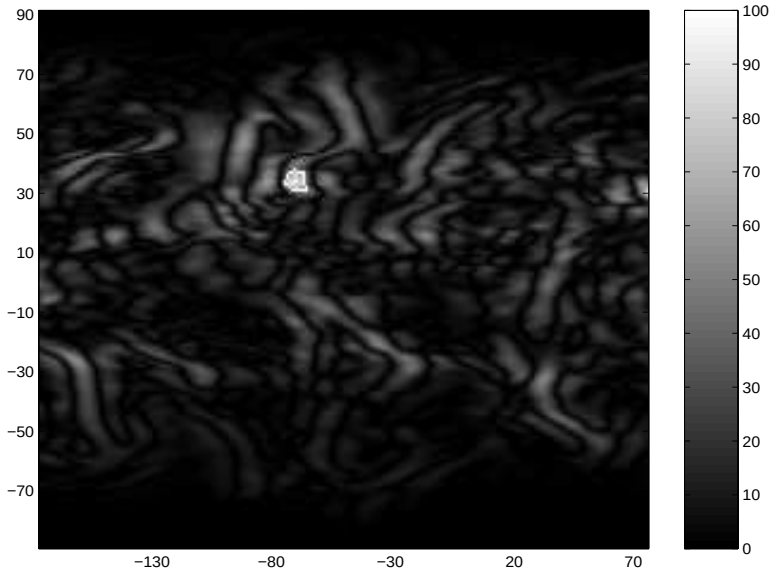


Figure 3.9. Water vapor data at 6AM Sept. 18th, 2003 with Hurricane Isabel identified.

where the images were segmented into 4 gray levels. In Figure 3.13(c), Hurricane Isabel is correctly identified as the largest connected component as shown by a white arrow. In Figure 3.13(d), the segmentation is not acceptable for Image 2. The region designated by a white arrow represents the location of Hurricane Isabel. However, it is far from the largest connected component. For example, the region marked by a white rectangle is apparently larger. When $K = 8$, the segments of the images are represented by 8 different gray levels as shown in Figure 3.13(e) and Figure 3.13(f). In both figures, the boundary of Hurricane Isabel is identified accurately with a white arrow. In summary, the K-Means method is not effective in segmenting the water vapor data and locating the hurricane when $K = 2$ or $K = 4$. When K is increased to 8, the hurricane can be identified accurately. However, the running time will increase significantly. Please refer to Section 3.1.3 for details.

Maximum Entropy:

The Maximum Entropy method was tested on Image 1 and Image 2 as well. We experimented with the segmentation using a single threshold and 2 thresholds. The results of single threshold 67 are demonstrated in Figure 3.14(a) and Figure 3.14(b). The threshold 67 is automatically calculated based on Image 1. Based on this threshold, Figure 3.14(a) shows that the segmentation of Image 1 is effective, with Hurricane Isabel identified using a white arrow. However, as shown in Figure 3.14(b), the threshold 67 is not suitable for Image 2: the largest connected region centered at (44°N, 90°W) is marked with a white rectangle and it is apparently not Hurricane Isabel, which

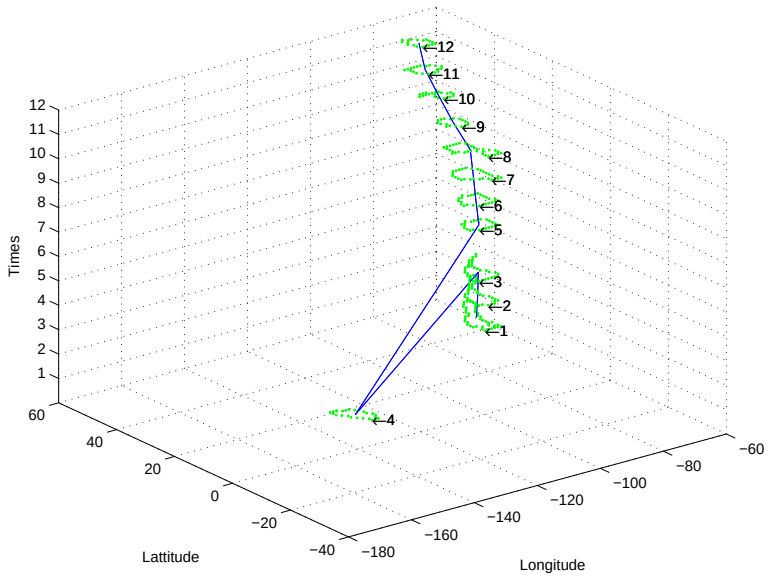


Figure 3.10. Trajectory of moving region with "noise" data.

is marked by a white arrow. Based on Image 2, another threshold value 140 can be generated automatically. Figure 3.15(a) shows that although this threshold can help determine the center of Hurricane Isabel in Image 1, the boundary of the detected region is not accurate and the size of the region is much smaller than the actual hurricane. As shown in Figure 3.14(b), the threshold 140 is effective for Image 2, with the location of Hurricane Isabel accurately pinpointed by a white arrow. Nevertheless, one limitation of the single threshold Maximum Entropy is that different images need different thresholds and these thresholds vary significantly. This means that the threshold is not stable for continuous data sequences. The 2-threshold Maximum Entropy image segmentation achieves satisfactory results (please refer to Figure 3.16(a) and Figure 3.16(b)), with the boundary of Hurricane Isabel identified accurately. The threshold values are (56, 129) for Image 1 and (58, 140) for Image 2. However, the running time of the two-threshold Maximum Entropy exceeds 5 seconds, which may not be suitable for high-speed stream data processing. Please see Section 3.1.3 for details.

λ-connectedness

Figure 3.17 demonstrates the results of the λ-connectedness image segmentation. λ was selected as 0.95 and the clip parameter θ was set to 45%. For both images, the location of Hurricane Isabel is accurately identified using a white arrow. The segmentation quality of the λ-connectedness is comparable to that of Maximum Entropy and the λ-connectedness method has excellent stability: only one set of parameters are required for all correlated images.

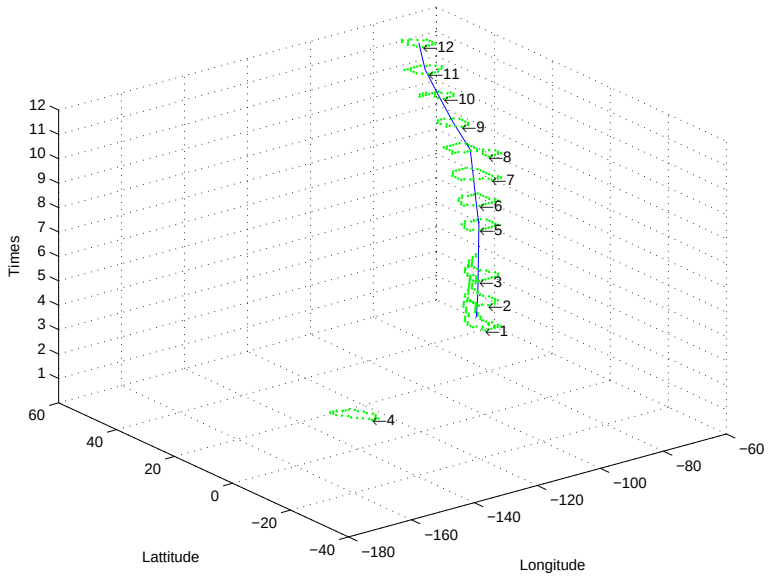


Figure 3.11. Trajectory of moving region without "noise" data.

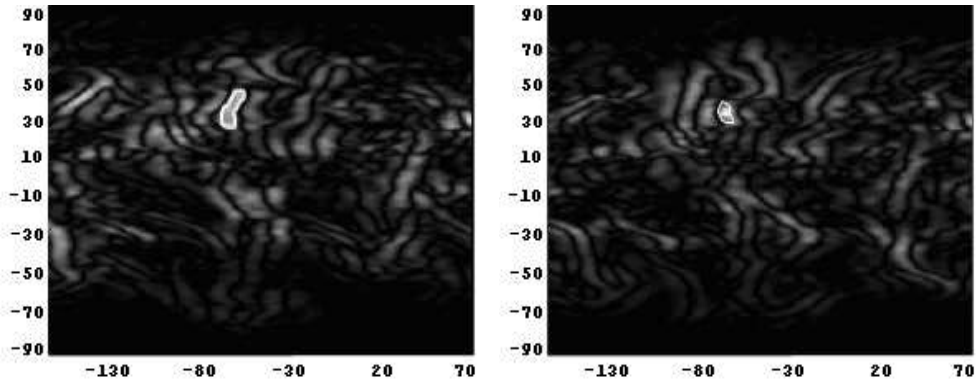
Image	$K = 2$	$K = 4$	$K = 8$
Image 1: data at 0AM Sept 17, 2003	62ms	219ms	1094ms
Image 2: data at 6AM Sept 18, 2003	47ms	188ms	485ms

Table 3.5. Running time of K-Means.

(2) Running time comparison

The running time of the three methods were recorded for both testing images. Table 3.5 lists the running time of the K-Means for $K = 2$, $K = 4$, and $K = 8$. When $K = 2$, the average image segmentation time is 62ms and 47ms for the two images respectively. The efficiency may be acceptable for low-speed data streams but not appropriate for high-speed data streams. When $K = 4$ and $K = 8$, the running time increases dramatically (around 500ms), which is apparently not suitable for fast image segmentation.

Table 3.6 shows the running time of the Maximum Entropy. The single threshold Maximum Entropy segmented the two images in 63ms and 16ms respectively, similar to the running time of K-Means when $K = 2$. However, when two thresholds are employed, the running time jumps to 5 $\sim$ 7 seconds, which is not suitable for online processing of data sequences. Table 3.7 demonstrates the efficiency of the λ-connectedness approach, where $\lambda = 0.95$ and $\theta = 45\%$. Compared with the other two methods, it has the shortest processing time of only 3ms. In addition, the running time variation between different images is very small.



(a) Image 1: 0AM, Sept 17, 2003 (b) Image 2: 6AM, Sept 18, 2003

Figure 3.12. Two source images for image segmentation comparison.

Image	Single threshold	2 thresholds
Image 1: data at 0AM Sept 17, 2003	63ms	7155ms
Image 2: data at 6AM Sept 18, 2003	16ms	5359ms

Table 3.6. Running time of Maximum Entropy.

Based on the above comparisons, we can see that the K-Means is not appropriate for data sequence processing. When $K = 2$ and $K = 4$, the image segmentation quality can not be guaranteed. When $K = 8$, the quality is acceptable, but the efficiency is not satisfactory. The Maximum Entropy achieves satisfactory segmentation quality. If the data arriving rate is not very high, the single threshold entropy method can be used. However, it has a limitation in that both the threshold value and the running time vary significantly with different images. The two-threshold Maximum Entropy provides excellent quality. Nevertheless, the long running time makes it not suitable for stream data processing. The λ-connectedness method can support high processing speed and render acceptable segmentation quality. Moreover, it provides excellent stability: for different images, one single set of parameters can be used and the processing time is steady. Based on the above considerations, the λ-connectedness method was selected to perform the image segmentation.

Image	$\lambda - connectedness$
Image 1: data at 0AM Sept 17, 2003	3ms
Image 2: data at 6AM Sept 18, 2003	3ms

Table 3.7. Running time of $\lambda - connectedness$.

3.1.4 Discussion

Our major contribution is to provide a framework to detect region outlier based on image processing techniques for spatio-temporal data. Several techniques are carefully selected and combined together to fulfil this task. Wavelet is not a new idea and it has been widely used in many fields. We choose wavelet as the preprocessing tool because it is highly efficient and supports multiple resolutions. λ-connected method was proposed by one of our collaborators years ago, however its efficiency was not fully examined in real applications. We extended the λ-connectedness method for region outlier detection. For example, a threshold for background noise filtering is used to further improve its processing speed. In addition, extensive experiments in real NOAA water vapor data were conducted to evaluate its performance against other image segmentation methods. Finally, we proposed a regression-based method to verify the correctness of the detected region outlier and track its movement.

Wavelet analysis and λ-connectedness are not the only choice. Any image preprocessing method can be used to substitute wavelet as long as it can provide high efficiency and multi-resolutions. Any new image segmentation method can be used to replace λ-connectedness as long as it can support linear computation time and achieve acceptable segmentation result.

The parameter selection in this framework is dependent on the requirements of different applications. For NOAA water vapor data, the domain experts select scale 2 and 3 for wavelet analysis, since their resolution is appropriate for hurricane detection. For image segmentation, we designate the background clipping threshold as 75 percent of the difference between the maximum value and minimum value in the data set since hurricanes have high water vapor values. As for the similarity level λ, 0.95 is selected based on the experiment results on some sampled data. One interesting and challenging future direction is to support automatic parameter selection, which means the parameter can be adjusted automatically according to the requirement change and data evolution.

3.2 Change Detection

In research on atmospheric sciences, huge amounts of spatial data have been collected from both observation and modelling. Discovering useful patterns from these data sets would have significant practical value and would greatly assist weather forecasting and environmental monitoring efforts. Temporal and spatial changes of the parameters (especially the prominent changes) are frequently associated with variations in weather phenomena and climate patterns. Consequently, detecting

these temporal/spatial changes in the atmospheric parameters is crucial in weather forecasting and climate analysis.

Wavelet analysis has been widely used in detecting temporal changes and trends at various ranges from turbulence (minutes) [26, 40] to climate (years) [50, 68, 98]. As meteorological data contain both spatial and temporal information, it is beneficial to take into account the spatial information while examining the temporal changes, since these changes could be different from place to place. In this section, new wavelet based change detection procedures are proposed in order to reveal changes in meteorological observation data.

3.2.1 Problem and Approach

As is well known, the weather always changes and so does the climate. These changes occur in both the temporal and spatial domains, and the combined effects of spatial movement and temporal variation make the change detection task difficult. For example, temporal changes may vary from place to place, and spatial changes may evolve as time elapses. What makes it even more challenging is that the temporal changes happen at different time scales, ranging from turbulence (minutes) to climate change (years or decades), and the spatial changes also occur at different spatial scales, varying from tornados (kilometers or less) to El Ninos (global). This is a complicated multi-scale problem. Different tasks require a focus on particular scales: tornado watches concentrate on imminent changes in the atmosphere, whereas global warming research is concerned with changes in the inter-annual or decades range. It is therefore necessary to find an effective procedure to detect the prominent changes that occurred at given scales of interest.

Wavelet analysis is a useful tool with which to study the subjects from signal analysis to image processing [27, 98]. Wavelet analysis offers some specially attractive features as it can analyze the signal at different frequencies with different resolutions. The changes of the signal at different scales can be studied with different focuses and the signal can be separated and recomposed at will. This feature makes wavelet transformation an effective tool for filtering out the noise and focusing on certain scales. In the real world, signals are usually complicated and non-stationary. Wavelet transformation provides the strength, frequency and the location of a particular variation. This study utilizes continuous wavelet analysis. For a wavelet function $\Psi(t)$, the continuous wavelet transform of a discrete signal $X_i(i = 0, \ldots, N-1)$ is defined as the convolution of X with scaled and translated Ψ: $W(n,s) = \sum_{i=0}^{N-1} x(i)\Psi^*[\frac{(i-n)\delta t}{s}]$, where (*) indicates the complex conjugate, n is the localization of the wavelet transform and s is the scale. Many functions can be used as the

base function for wavelet analysis. Here, one of the most widely used bases, the *Mexican Hat*, is used as it provides good localization (spatial/temporal resolution). The *Mexico Hat* base function is: $\Psi_0(\eta) = \frac{(-1)}{\sqrt{\Gamma(21/2)}} \frac{d^2}{d\eta^2}(e^{-\eta^2/2})$. The scales in the wavelet analysis are selected by $S_0 * 2^{j/2}(j = 0, 1, \ldots, J)$. Here J is the maximum scale index which satisfies : $J \leq 2\log_2(\frac{N}{2})$, where N is the length of the signal.

Algorithm 1 :Spatial Change Detection

Input:
D_1 is the meteorology data for time t_1;
D_2 is the meteorology data for time t_2;
S is a set of selected scales;
θ_w is the pre-defined threshold of wavelet power;
α_1 is the beginning latitude/longitude;
α_n is the ending latitude/longitude;
$idxSet_1$ is a set of location indices in D_1;
$idxSet_2$ is a set of location indices in D_2;
Output:
M is a set of (region, direction) pair;

```
/* wavelet transform along all latitudes or longitudes */
for(i=α1;i ≤ αn ;i++)
    wDomain1 = WaveletTransform(D1,S,i);
    wDomain2 = WaveletTransform(D2,S,i);
/*record points with prominent wavelet power*/
for every point p in wDomain1
    if ( p > θw )
        AddToLocationSet(p,idxSet1);
/*record points with prominent wavelet power*/
for every point p in wDomain2
    if ( p > θw )
        AddToLocationSet(p,idxSet2);
/* group points to regions */
R1 = GroupIndices(idxSet1)
R2 = GroupIndices(idxSet2)
/*output the time and location with prominent changes*/
M=CompareLocation(R1,R2)
```

By applying wavelet transformation in the spatial domain and repeating the process at different time instances, it is possible to monitor the movements of the spatial anomalies. Also, wavelet transformation can be applied to time series in order to detect temporal changes. Here, two algorithms are proposed: Algorithm 1 tracks the spatial movements of meteorological objects, while Algorithm 2 discovers the temporal changes of meteorological objects.

In Algorithm 1, a set S of scales of particular interest must first be provided by domain experts. Then the wavelet transformation is performed on two data sets, D_1 and D_2, along all the latitudes or longitudes. Here D_1 and D_2 are meteorological data for time t_1 and time t_2, α_1 denotes the

beginning latitude(or longitude), and α_n denotes the ending latitude(or longitude). $wDomain_1$ *and* $wDomain_2$ are the domain of wavelet power values transformed from the original data D_1 and D_2. The algorithm selects the points with wavelet power greater than the threshold θ_w and records their location indices in two sets named $idxSet_1$ and $idxSet_2$. $idxSet_1$ stores the indices of interesting points in D_1 and $idxSet_2$ stores the indices of interesting points in D_2. Then the algorithm groups points with adjacent location indices in $idxSet_1$ and $idxSet_2$ to regions, and stores these regions in set R_1 and set R_2 respectively. Finally, the locations of the same region in R_1 and R_2 are compared to find the direction of movement of this region between time t_1 and time t_2. The comparison is based on the coordinates of the center points of the regions. The output is a set M of (region, direction) pairs, which denotes the direction of movement of a specific region.

Algorithm 2 :Temporal Change Detection

Input:
D is the given dataset;
θ_t is the pre-defined threshold of wavelet power on time series;
β_1 is the beginning time series;
β_2 is the ending time series;
$idxSet$ is a set of location indices;
Output:
O_s is the set of change regions;

```
/* wavelet transform along all time series */
for(i=β1;i ≤ β2 ;i++)
    wDomain = WaveletTransform(D,i);
/*record points with prominent wavelet power*/
for every point p in wDomain
    if ( p > θt )
        AddToLocationSet(p,idxSet);
/* group points to regions */
Os = GroupIndices(idxSet)
Output(Os) /* output the time and location with prominent changes
```

Algorithm 2 is similar to Algorithm 1 with two major distinctions. First, Algorithm 2 applies wavelet analysis in the time domain instead of the spatial domain. Second, in Algorithm 2, the wavelet analysis picks only one particular scale rather than a set of scales S. The output of Algorithm 2 is a set of regions where prominent changes occur in the time domain. Note that the result of the wavelet transformation is a 3-dimension array of (time,latitude,longitude). It is possible to observe the temporal changes from both latitude and longitude by visualizing this array.

3.2.2 Experimental Analysis

The experiment used NOAA/NCEP global water vapor data set, which has been introduced in Section 3.1.3. A 40-day water vapor data was used to study changes in the weather system.

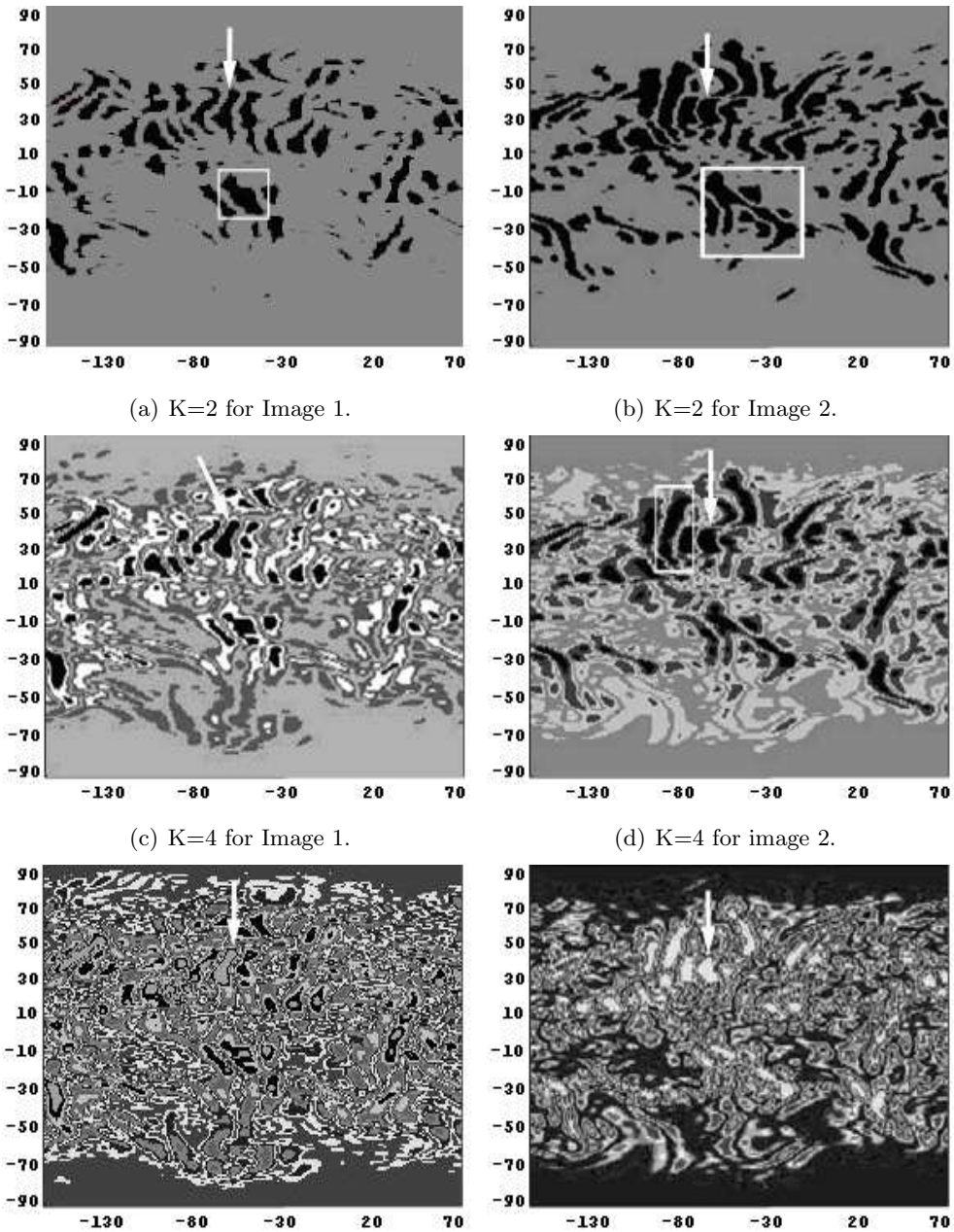
First, wavelet analysis was performed in the spatial domain over all latitudes, focusing on the anomalies with sub-weather scales, i.e, around around 1000 km. The left panel of Figure 3.18 shows the global map of wavelet transformation power with scale index 3. As can be seen, there are several areas where the power is extremely high. In these areas the spatial variation is prominent and these areas are possible spatial anomalies (weather events). The corresponding locations of the strong wavelet powers beyond a threshold are plotted in the right panel: one region (a hurricane) over the Gulf of Mexico (-90^oW,26^oN) and another region (a tropical storm) over South America (-70^oW,-25^oS). Figure 3.19 shows the spatial wavelet power for another date. The Comparison between Figure 3.18 and Figure 3.19 reveals the northward movement of the anomaly region (a hurricane) near the Gulf of Mexico, while the tropical storm over south America is still there but gradually diminishing in magnitude. This procedure can be used for consecutive date/time instances to track the changes in weather systems.

Algorithm 2 applied wavelet transformation in the time domain to detect the temporal changes of the water vapor. As the preliminary tests show that the strongest variations are often short-term changes for this data set, the analysis focused on the short temporal scales in our analysis, performing wavelet analysis on a time series over all 1×1 grids. Figure 3.20 shows the cross section (at 90^oW) of the wavelet power (with temporal scale 0) along date(0-39) and the latitudes. The changes are mainly located in the mid-latitudes, especially around 25^oN, and the prominent changes happen during days 20-35. The figure also shows that some change signals are moving northward with elapsed time, which is consistent with the pattern observed from Figures 3.18 and 3.19. Figure 3.21 provides the 3 dimension distribution of the high wavelet transformation power across latitudes, longitudes, and dates (with threshold 35), revealing several change signals and their spatial-temporal locations. Of particular interest are the continuous changes revealing the development and movement of the weather systems. One of the significant changes occurs in South America (-70^oW,-25^oS), corresponding to a tropical storm. The signals are persistent and move northward and eastward at times. Another one is in the Gulf of Mexico region (-90^oW,26^oN), where a strong change signal is formed around day 20 and then moves northward as time progresses. This is a developing hurricane over the Gulf of Mexico.

3.2.3 Summary

In this section, a wavelet based approach is proposed to detect temoral/spatial changes in meteorological data. This approach can help identify distinct change patterns. Some of these patterns may be visible in the original data set, whereas other patterns may be hidden in the original data and might be overlooked without the use of wavelet analysis. By applying wavelet transformation, the location and the time of a change's occurrence can be identified, which assists in monitoring changes in the weather and climate. This study focuses on short range changes, but in the future, other scales of changes will be explored. In addition, the proposed algorithms will be extended to other meteorology parameters, so that the detected changes will be more comprehensive and representative.

Currently, the proposed change detection algorithms can only process static data sets or low speed data sequences. To identify prominent changes in high speed stream data, several key issues must be addressed. First, the "unbounded size" and "online processing" properties of stream data require efficient data structures and indexing schemes for representing changes in a data stream. These data structures should provide flexible containers for hierarchical data summarization, support multiple granularities of different time frames, and render efficient indexing schemes. Second, high performance mining methods for change detection are in great demand for stream data applications, including satellite-based land cover monitoring, video surveillance , traffic volume control, and fault diagnosis in industrial production systems. Most of these applications are based on images or videos, which contain both spatial and non-spatial information. Image processing skills, such as edge detection, image segmentation, and image similarity comparison, will need to be combined with traditional data mining techniques, for example, Neural Networks and Bayesian Networks, in order to identify abnormal patterns and support trend analysis.



(a) K=2 for Image 1.

(b) K=2 for Image 2.

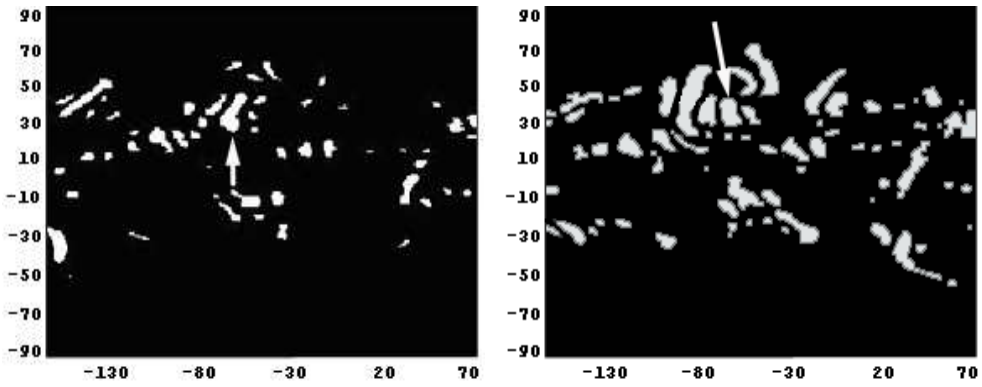
(c) K=4 for Image 1.

(d) K=4 for image 2.

(e) K=8 for Image 1.

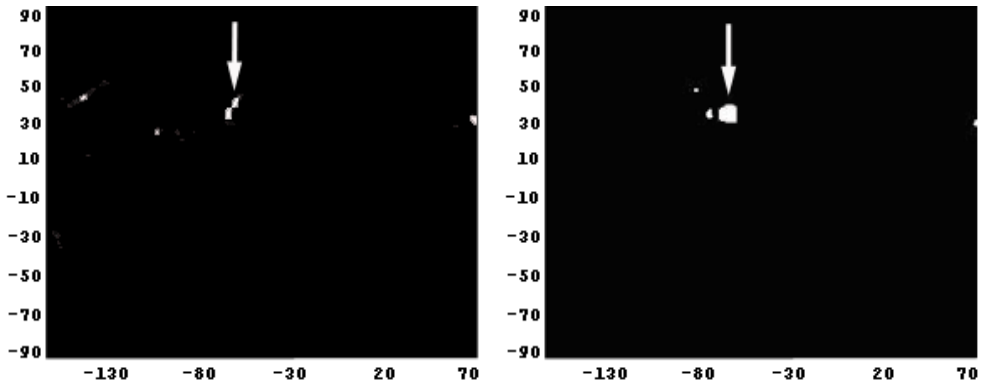
(f) K=8 for Image 2.

Figure 3.13. The effectiveness of K-$Means$ image segmentation



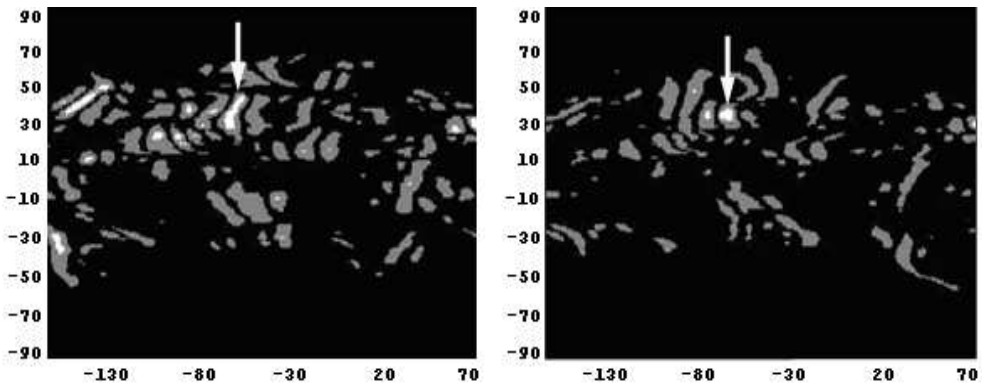
(a) Threshold=67 for Image 1. (b) Threshold=67 for Image 2.

Figure 3.14. Maximum Entropy image segmentation: single threshold.



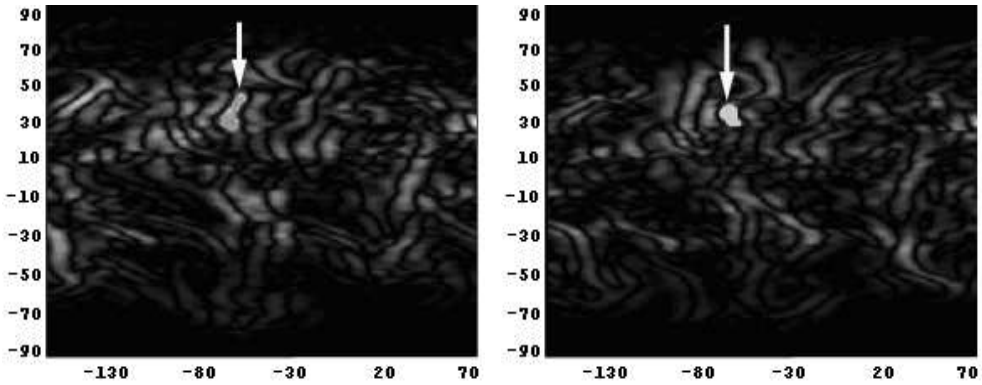
(a) Threshold=140 for Image 1. (b) Threshold=140 for Image 2.

Figure 3.15. Maximum Entropy image segmentation: single threshold.



(a) Thresholds=(56,129) for Image 1. (b) Thresholds=(58,140) for Image 2.

Figure 3.16. Maximum Entropy image segmentation: two-threshold.



(a) based on Image 1. (b) based on Image 2.

Figure 3.17. λ-connectedness image segmentation.

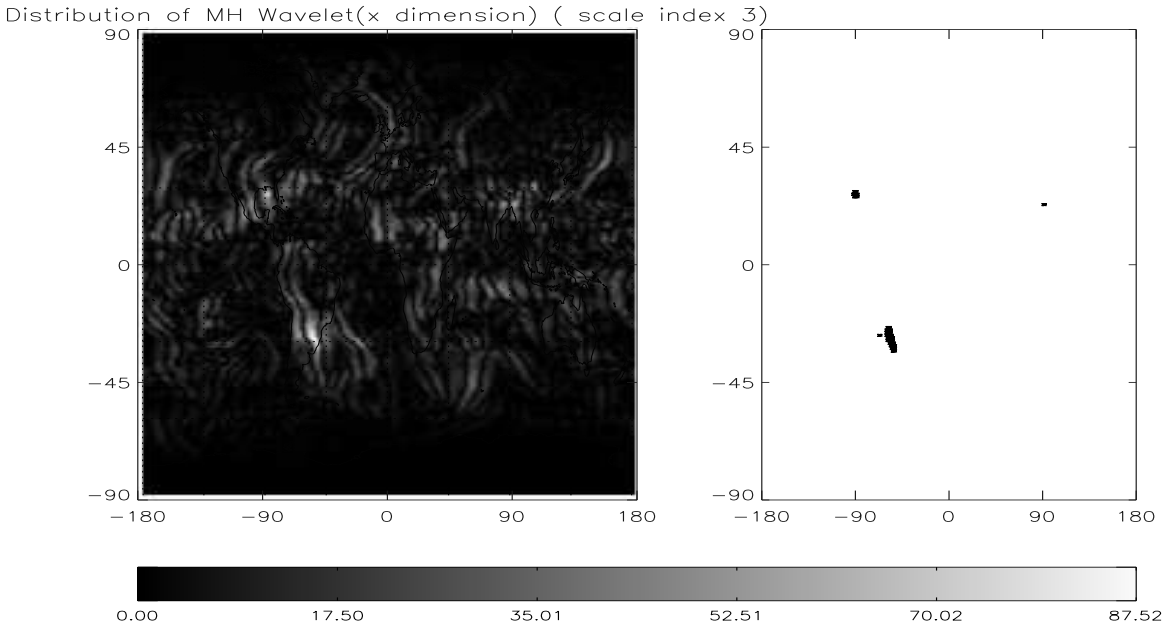


Figure 3.18. Spatial wavelet power distribution at scale index 3 on day 1.

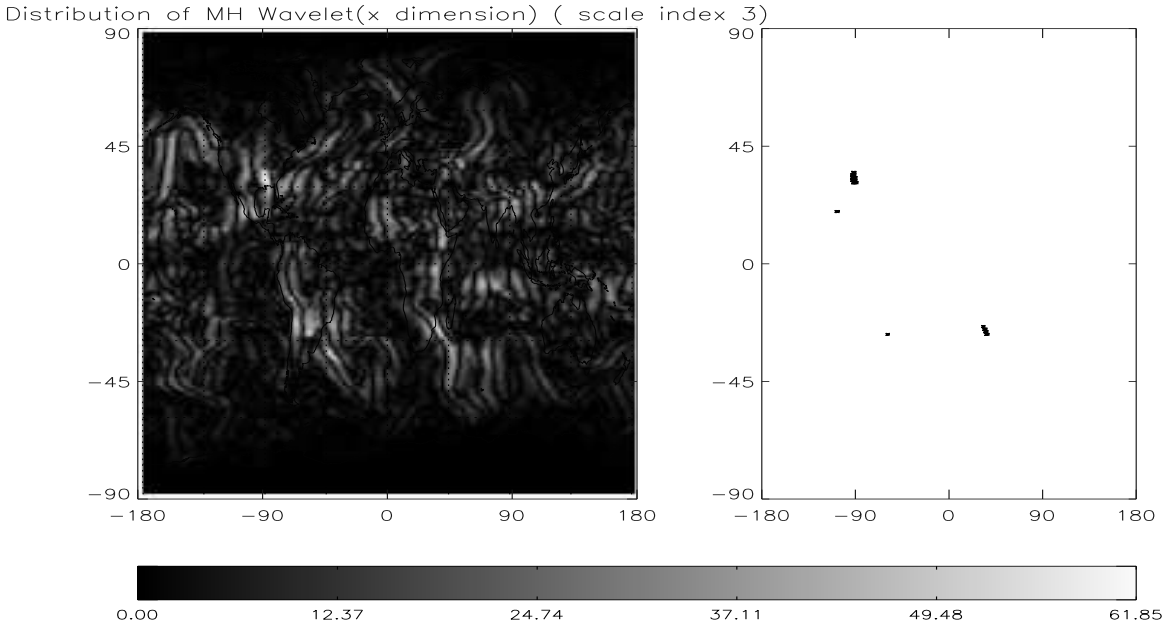


Figure 3.19. Spatial wavelet power distribution at scale index 3 on day 2.

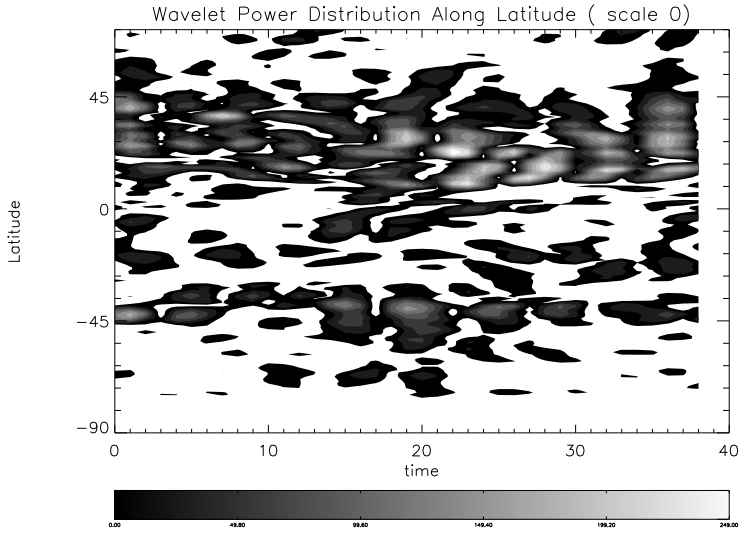


Figure 3.20. Temporal wavelet power distribution across latitude and date.

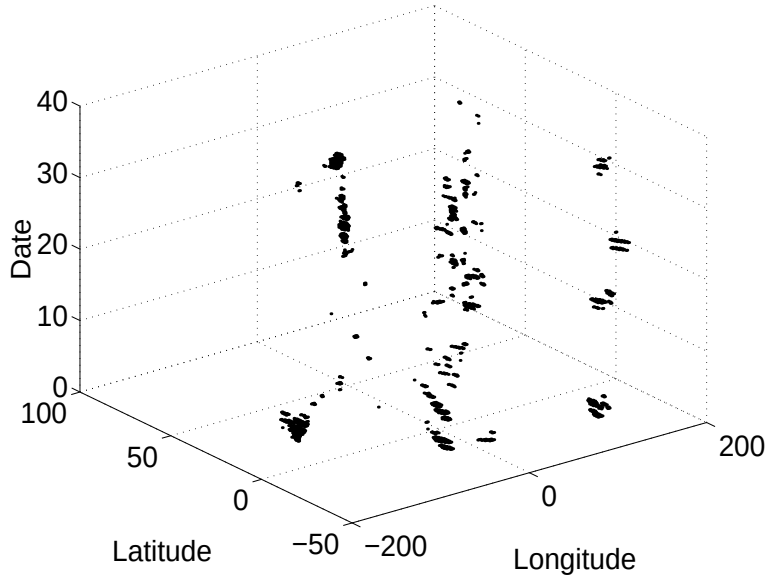


Figure 3.21. Distribution of high temporal wavelet power (threshold=35) across longitude, latitude, and date.

Chapter 4

Multiple Attribute Outlier Detection

Geospatial data sets often contain multiple nonspatial attributes. For example, the monitoring stations on the highway system collect not only volume information, but also speed, occupancy, and road condition. In certain situations, we need consider multiple attributes together to discover abnormal patterns, or outliers. This chapter defines the multi-attribute spatial outlier detection problem, proposes several algorithms to detect spatial outliers with multiple attributes, discusses their computational complexities, and analyzes their effectiveness based on two real data sets.

4.1 Motivation and Problem Formulation

Detecting spatial outliers is useful in many applications of geographic information systems and spatial databases [89, 90, 93]. These application domains include transportation, ecology, public safety, public health, climatology, and location based services. In these applications, there may be more than one non-spatial attributes associated with each spatial object. For example, in census data set, each census tract contains several non-spatial attributes, including population, population density, income, poverty, housing, education, and race [99]. Detecting outliers from these spatial data with multiple attributes will help demographers and social workers to identify local anomalies for further analysis.

Spatial Outliers with Multiple Attributes

Suppose $q(\geq 1)$ measurements (attribute values) are made on the spatial object x. We use a

to denote the vector of these q values at x. Given a set of spatial points $X = \{x_1, x_2, \ldots, x_n\}$ in a space with dimension $p \geq 1$, an attribute function f is defined as a mapping from X to R^q (the q dimensional Euclidean space) such that for each spatial point x, $f(x)$ equals the attribute vector a. For convenience, we write $a_i = f(x_i) = (f_1(x_i), f_2(x_i), \ldots, f_q(x_i))^T$ for $i = 1, 2, \ldots, n$. Denote the set $\{a_1, a_2, \ldots, a_n\}$ by A.

Let $NN_k(x_i)$ denote the k nearest neighbors of point x_i with $k = k(x_i)$ for $i = 1, 2, \ldots, n$. A neighborhood function g is defined as a mapping from X to R^q such that the jth component of $g(x)$, denoted $g_j(x)$, returns a summary statistic of the jth attribute values from all the spatial points inside $NN_k(x)$. For the purpose of detecting spatial outliers, we compare all of the components of a at x with the corresponding quantities from the neighbors of x. A comparison function h is a function of f and g, whose domain is X and range is in R^r with $r \leq q$. Examples of h include $h = f - g$, a mapping from X to R^q with $r = q$, and $h = f_1/g_1$, a mapping from X to R with $r = 1$. Denote $h(x_i)$ by h_i. A point x_i is an **S-outlier** if h_i is an extreme point of the set $\{h_1, h_2, \ldots, h_n\}$. The following problem characterizes the task of designing algorithms for detecting spatial outliers with multiple attributes:

Detection Problem: Given a set of spatial points $X = \{x_1, x_2, \ldots, x_n\}$, a sequence of neighborhoods $NN_k(x_1), NN_k(x_2), \ldots, NN_k(x_n)$, an attribute function $f : X \to R^q$, a neighborhood function $g : X \to R^q$, and a comparison function $h : X \to R^r$, design algorithms to detect spatial outliers with multiple attributes.

4.2 Detection Algorithms

Different choices of g and h may lead to different algorithms and thus potentially different outliers. The criterion on the selection of g and h is that most of the resulting outliers should possess practical meanings. For example, examining outliers should often lead to causation investigations.

When attribute vectors are of high dimension, a natural way to detect S-outliers is to check the marginal attribute value one by one. Specificallly, let us fix $j (1 \leq j \leq q)$ and consider the jth component data set $A_j = \{a_{1j}, a_{2j}, \ldots, a_{nj}\}$. View A_j as the one-dimensional attribute value data set of the points $x_1, x_2, \ldots, x_n$. If using any method from Section 2.2.3, one finds that x_i is an S-outlier with respect to A_j, then clearly x_i is an S-outlier with respect to A. This corresponds to taking h to be a single-valued function of f_i and g_i (so $r = 1$).

Detecting unusual attribute vectors by using all the attribute values simultaneously is available.

We do this by computing the difference between f and g, i.e., $h = f - g$ and then checking the Mahalanobis distance between $h(x)$ and the average h value from the neighbors of x. To describe this method, let us first note the following. Under certain conditions, we may show that $h(x)$ follows a multivariate normal distribution. And if $h(x)$ is distributed as $N_q(\mu, \Sigma)$, i.e., q-dimensional vector $h(x)$ follows a multivariate normal distribution with mean vector μ and variance-covariance matrix Σ, then $(h(x)-\mu)^T\Sigma^{-1}(h(x)-\mu)$ is distributed as χ^2_q, where χ^2_q is the chi-square distribution with q degrees of freedom. Therefore the probability that $h(x)$ satisfies $(h(x)-\mu)^T\Sigma^{-1}(h(x)-\mu) > \chi^2_q(\alpha)$ is α, where $\chi^2_q(\alpha)$ is the upper (100α)th percentile of a chi-square distribution with q degrees of freedom. Thus intuitively, for a point x, if $(h(x) - \mu)^T\Sigma^{-1}(h(x) - \mu)$ is sufficiently large, x should be treated as a S-outlier candidate.

Now suppose there are n spatial referenced objects $x_1, \cdots, x_n$. For the sample $h(x_1), \cdots, h(x_n)$, calculate the sample mean

$$\mu_s = \frac{1}{n}\sum_{i=1}^{n} h(x_i)$$

and sample variance-covariance matrix

$$\Sigma_s = \frac{1}{n-1}\sum_{i=1}^{n}[h(x_i) - \mu_s][h(x_i) - \mu_s]^T.$$

Then we should expect that the probability of $h(x)$ satisfying $(h(x) - \mu_s)^T\Sigma_s^{-1}(h(x) - \mu_s) > \chi^2_q(\alpha)$ is roughly α. Set $d(x) = [(h(x) - \mu_s)^T\Sigma_s^{-1}(h(x) - \mu_s)]^{1/2}$. Therefore, if $d(x)$ or $d^2(x)$ is unusually large, x will be teated as a spatial outlier.

The quantity $d(x)$ defined above is actually the Mahalanobis distance from $h(x)$ to μ_s. The Mahalanobis distance, introduced in 1936, provides a way of finding points which are far from all of the others in a multidimensional space [67]. It has been widely used in discriminant analysis, clustering, and principle analysis [71,96,105]. It has many advantages over Euclidian distance when dealing with multivariate data. For example, the basic Euclidian distance treats each variable as equally important in calculating the distance, while Mahalanobis distance automatically accounts for the scaling of the coordinate axes.

In practice, it is possible that the inverse Σ_s^{-1} does not exist. If this happens, the Moore-Penrose generalized inverse Σ_s^+ can be used to replace Σ_s^{-1}. The Moore-Penrose generalized inverse Σ_s^+ satisfies the following properties: $\Sigma_s\Sigma_s^+\Sigma_s = \Sigma_s$, $\Sigma_s^+\Sigma_s\Sigma_s^+ = \Sigma_s^+$, $(\Sigma_s^+\Sigma_s)' = \Sigma_s^+\Sigma_s$, $(\Sigma_s\Sigma_s^+)' = \Sigma_s\Sigma_s^+$. The Moore-Penrose generalized inverse Σ_s^+ is unique. Σ_s^+ equals the inverse Σ_s^{-1} if the latter

exists. Σ_s^+ provides an approximation of Σ_s^{-1} when Σ_s is not a square matrix or does not have a full rank. In addition, it guarantees that the approximation error is small in a least square sense. Replacing Σ_s^{-1} with Σ_s^+ can greatly extend the range of solvable problems and the applicability of the proposed algorithms. For more information on Moore-Penrose matrix inverse, we refer the readers to [34].

We now present the following algorithms that handle all the attributes simultaneously.

Given a spatial data set $X = \{x_1, x_2, \ldots, x_n\}$, an attribute function f, and two positive integer numbers k and m,

Algorithm 1: (Mean Algorithm).

1. For each fixed j $(1 \leq j \leq q)$, standardize the attribute function f_j, i.e., $f_j(x_i) \leftarrow \frac{f_j(x_i) - \mu_{f_j}}{\sigma_{f_j}}$ for $i = 1, 2, \ldots, n$.

2. For each spatial point x_i, compute the k nearest neighbor set $NN_k(x_i)$.

3. For each spatial point x_i, compute the neighborhood function g such that $g_j(x_i) =$ average of the data set $\{f_j(x) : x \in NN_k(x_i)\}$, and the comparison function $h(x_i) = f(x_i) - g(x_i)$.

4. Compute $d^2(x_i) = (h(x_i) - \mu_s)^T \Sigma_s^+ (h(x_i) - \mu_s)$. Those m x_i's with the largest d^2 values will be treated as candidates of S-outliers with respect to A.

Standardization is used in Step 1 of the above algorithm. Therefore, different units of multiple dimensions will not impact the Mahalanobis distance. Motivated by the fact that median is a robust estimator of the "center" of a sample, we obtain the following median detection algorithm by choosing g to be the median of the attribute vectors from the neighborhood.

Algorithm 2: (Median Algorithm).

1. For each fixed j $(1 \leq j \leq q)$, standardize the attribute function f_j, i.e., $f_j(x_i) \leftarrow \frac{f_j(x_i) - \mu_{f_j}}{\sigma_{f_j}}$ for $i = 1, 2, \ldots, n$.

2. For each spatial point x_i, compute the k nearest neighbor set $NN_k(x_i)$.

3. For each spatial point x_i, compute the neighborhood function g such that $g_j(x_i) =$ median of the data set $\{f_j(x) : x \in NN_k(x_i)\}$, and the comparison function $h(x_i) = f(x_i) - g(x_i)$.

4. Compute $d^2(x_i) = (h(x_i) - \mu_s)^T \Sigma_s^+ (h(x_i) - \mu_s)$. Those m x_i's with the largest d^2 values will be treated as candidates of S-outliers with respect to A.

The only difference between these two algorithms lies in the third step, where the Algorithm 1 uses average attribute values to compute the neighborhood function while the Algorithm 2 uses the median attribute values. We note that in the above two algorithms, if the expected number m of S-outliers is given, instead of θ, then those m outliers could be picked up according to the m largest values of d^2.

It should be noted that a point x which is an S-outlier with respect to A may not be an S-outlier with respect to any A_j. This means that there can be S-outliers detected by Algorithm 1 (or Algorithm 2) that can not be detected from the marginal attribute values using any algorithm in Section 2.1.3. This fact is not surprising, since S-outliers with respect to A take into account all the attribute values at the same time whereas methods based on the marginal attribute values consider the attributes separately. On the other hand, S-outliers based on individual marginal attributes may not be detected by algorithms based on all the attributes. Therefore, to effectively detect S-outliers with multiple attributes, a good practice is to run algorithms designed for individual attributes and run algorithms handling all the attributes simultaneously (see the experiment section).

4.3 Computational Complexity

The complexity of Algorithm 1 is analyzed as follows. Step 1 is to standardize the attribute function, which costs $O(qn)$. In Step 2, the neighborhood is computed for each spatial point, in which a k nearest neighbor (KNN) query is issued. Again, the corresponding complexity is $O(n)$ for the grid-based approach in the query or $O(logn)$ for the spatial indexed-based approach. For Step 3, the computation of neighborhood function g and comparison function h takes $O(qkn)$. In Step 4, the computation costs $O(q^2 * n)$. In summary, the total computational cost for Algorithm 1 is $O(qn) + O(n) + O(qkn) + O(q^2 * n)$ for the grid-based structure, or $O(qn) + O(nlogn) + O(qkn) + O(q^2 * n)$ for the index-based structure. If $n \gg k$ and $n \gg q$, the total time complexity is $O(n)$ for the grid-based structure, or $O(nlogn)$ for the index-based structure. Therefore, when n is large, algorithms 1 discussed above have the same complexity as the single attribute algorithms discussed in Section 2.1.3. Algorithm 2 has the same time complexity as Algorithm 1, since it takes the same computation cost for calculating the mean and median of the $k-$nearest neighbors.

In practice, multiple attribute outliers should be detected by first performing the single at-

tribute algorithms for all the q attributes, then performing the multiple attribute algorithm. The corresponding total complexity should be the aggregated cost of these two procedures. It follows from Section 2.1.4 that the complexity of the former is $O(qn)$ for the grid-based structure and $O(qnlogn)$ for the index-based structure. Since $q \ll n$, the complexity can be simplified as $O(n)$ and $O(nlogn)$, respectively. Adding this to the cost of Algorithm 1 (or 2), we see that in detecting spatial outliers with multiple attributes, the total complexity is $O(n)$ for the grid-based structure and $O(nlogn)$ for the index-based structure if n is large.

4.4 Experiments

In this section, we conduct experiments on two data sets, the U.S. census data and the West Nile virus data. First, Mahalanobis-distance-based algorithms, Algorithm 1 and Algorithm 2, are applied on Census data to evaluate their effectiveness. Next, based on West Nile virus data, we applied both the single attribute outlier algorithm and the Mahalanobis-distance-based algorithm to identify outliers.

4.4.1 Analysis of Census Data

In this section, we focus on the evaluation of Mahalanobis-distance-based algorithms, namely, Algorithm 1 and Algorithm 2. The experiment is based on the U.S. census data set, which has been introduced in Section 2.1.5. We empirically evaluated our detection algorithms by mining a real-life census data set. The experiment results indicate that our algorithms can effectively identify spatial outliers with multiple attributes.

In the experiment, we used the following 11 attributes: population in 2001, percentage of population change from April 1 2000 to July 1 2001, percentage of population change from 1990 to 2000, percentage of persons under 5 years old in 2000, percentage of persons under 18 years old in 2000, percentage of persons over 65 years old in 2000, percentage of persons under 5 years old in 2000, percentage of persons under 18 years old in 2000, percentage of White persons, percentage of Black persons, percentage of Asian persons, and percentage of American Indians. The experiment was conducted on data of all counties in the United States.

The multiple attributes may have different magnitudes. For example, population of a county is usually more than 10,000, but the percentage of population change is mostly less than 1. So population of a county may dominate the value of difference function. To avoid this negative

impact, we standardized the attribute values for each attribute. The experiment results are shown in Table 4.1 and Table 4.2.

Rank	County	M Dist	Pop 2001	00-01 change	90-00 change	≤ 5 %	≤ 18 %	≥ 65 %	Female %	White %	Black %	Indian %	Asian %
1	Los Angeles, CA	1142	32.15	0.37	0.23	1.28	0.78	1.24	0.06	2.24	0.06	0.12	7.08
2	Cook, IL	402	17.71	0.41	0.36	0.82	0.15	0.75	0.58	1.77	1.18	0.20	2.56
3	San Francisco, CA	395	2.28	0.56	0.23	2.06	3.44	0.27	0.664	2.18	0.07	0.18	19.11
4	Santa Clara, CA	266	5.31	0.56	0.08	0.72	0.24	1.29	0.61	1.93	0.41	0.14	15.80
5	Menominee, WI	262	0.29	0.42	0.05	2.96	4.20	1.53	0.11	4.56	0.60	13.41	0.49
6	Shannon, SD	244	0.26	0.98	0.92	4.26	6.21	2.43	0.19	5.00	0.60	14.49	0.49
7	Douglas, CO	190	0.36	6.19	11.2	3.05	1.91	2.58	0.19	0.50	0.53	0.18	1.10
8	Buffalo, SD	189	0.29	0.60	0.27	3.98	4.95	2.02	0.92	4.26	0.60	12.52	0.49
9	Rolette, ND	188	0.26	0.04	0.24	2.31	3.45	1.24	0.11	3.71	0.60	11.17	0.42
10	Sioux, ND	184	0.29	0.04	0.22	3.89	4.64	2.24	0.76	4.39	0.60	12.99	0.49

Table 4.1. The top 10 spatial outlier candidates detected by the Mean algorithm and their associated standardized attribute values

Rank	County	M Dist	Pop 2001	00-01 change	90-00 change	≤ 5 %	≤ 18 %	≥ 65 %	Female %	White %	Black %	Indian %	Asian %
1	Los Angeles, CA	1306	32.15	0.37	0.23	1.28	0.78	1.24	0.06	2.24	0.06	0.12	7.08
2	Cook, IL	441	17.71	0.41	0.36	0.82	0.15	0.75	0.58	1.77	1.18	0.20	2.56
3	San Francisco, CA	395	2.28	0.56	0.23	2.06	3.44	0.27	0.664	2.18	0.07	0.18	19.11
4	Santa Clara, CA	367	5.31	0.56	0.08	0.72	0.24	1.29	0.61	1.93	0.41	0.14	15.80
5	Shannon, SD	300	0.26	0.98	0.92	4.26	6.21	2.43	0.19	5.00	0.60	14.49	0.49
6	Menominee, WI	259	0.29	0.42	0.05	2.96	4.20	1.53	0.11	4.56	0.60	13.41	0.49
7	Sioux, ND	238	0.29	0.04	0.22	3.89	4.64	2.24	0.76	4.39	0.60	12.99	0.49
8	Douglas, CO	205	0.36	6.19	11.2	3.05	1.91	2.58	0.19	0.50	0.53	0.18	1.10
9	Buffalo, SD	198	0.29	0.60	0.27	3.98	4.95	2.02	0.92	4.26	0.60	12.52	0.49
10	Harris, TX	191	11.35	0.65	0.59	1.84	1.10	1.80	0.14	1.62	0.66	0.18	2.75

Table 4.2. The top 10 spatial outlier candidates detected by the Median algorithm and their associated standardized attribute values

Note that the attribute values for each county have been standardized. The tables show only top 10 counties which are most likely to be spatial outliers. As can be seen from Table 4.1 and Table 4.2, Los Angeles is selected as top spatial outlier by both algorithms, because it has the largest Mahalanobis distance, 1142 for the Mean Algorithm and 1306 for the Median Algorithm. Specifically, the largest distance mainly comes from the contribution of the corresponding attribute population (standardized value 32.15), compared with its neighboring counties, e.g., Orange Co. (9.43), Ventura Co.(2.28), San Bernardino Co. (5.64), and Kern Co. (1.97). The second spatial outlier, Cook Co, which encompasses the downtown of Chicago, is also identified due to its high population (standardized value 17.71), compared with its neighboring counties, e.g., Dupage Co. (2.76) Will Co. (1.5), Lake Co.(1.32), Kane Co. (1.12), and McHenry Co.(0.6). The third and fourth spatial outliers, San Francisco, CA and Santa Clara, CA, have high percentage of Asian population (standardized Value 19.11 and 15.80, respectively) compared with the Asian population of their neighboring counties. The remaining seven counties in both tables were detected as spatial outliers because the total contributions to the Mahalanobis distance from various attribute values

are significant. From Table 4.1 and Table 4.2, we can also see that the Mean Algorithm and the Median Algorithm have 9 outliers in common and the rankings of the top 4 outliers are in the same order. This shows that both of the algorithms are effective in detecting spatial outliers.

4.4.2 Analysis of West Nile Virus Data

The experiments in this section are to evaluate the effectiveness of multi-attribute outlier detection by combining the outliers identified by the single attribute algorithm and the outliers identified by the Mahalanobis-distance-based algorithm. We performed the experiments based on the U.S. West Nile virus data, which has been introduced in Section 2.1.5.

In conducting multiple attribute outlier detection, we considered all the counties in USA. All the 9 WNV attributes of each county will be considered, including Bird-2001, Mosquito-2001, Veterinary-2001, Bird-2002, Mosquito-2002, Veterinary-2002, Bird-2003, Mosquito-2003, and Veterinary-2003. Our detection process consists of three components: (a) the median single attribute detection algorithm (please refer to Section 2.1.3) was applied to all the attributes one by one; (b) the multiple attribute detection algorithm (Algorithm 2) was applied to all the attributes simultaneously; (c) the results from (a) and (b) were merged to form the final result set. Our experiment is based on two assumptions. First, we assumed that single attribute median algorithm and Algorithm 2 contributed equally to the final result. Second, we assumed that all the attributes contributed equally to outlier detection. Thus the total number of outlier candidates detected from the single-attribute median algorithm was set to be the number of outlier candidates detected from Algorithm 2. And when applying the single attribute algorithm, we selected an equal number of outlier candidates for each attribute.

The experiment follows the flow chart shown in Figure 4.1. Users can designate a number ν so that $\nu\%$ of the data will be identified as outlier candidates. We denote the number of outlier candidates, based on each attribute, from the single detection algorithm as N_s and the number of outlier candidates from the multiple attribute detection algorithm as N_m. Based on our first assumption, $q * N_s = N_m$, where q represents the number of attributes. This formula determines N_m once N_s is known. Since there are 9 attributes in the WNV data set, the total number of outlier candidates in part (a) equals $9N_s$. It is obvious that there will be duplicate outlier candidates detected in (a) and (b). Consequently, a natural question is how to decide the exact value of N_s according to the

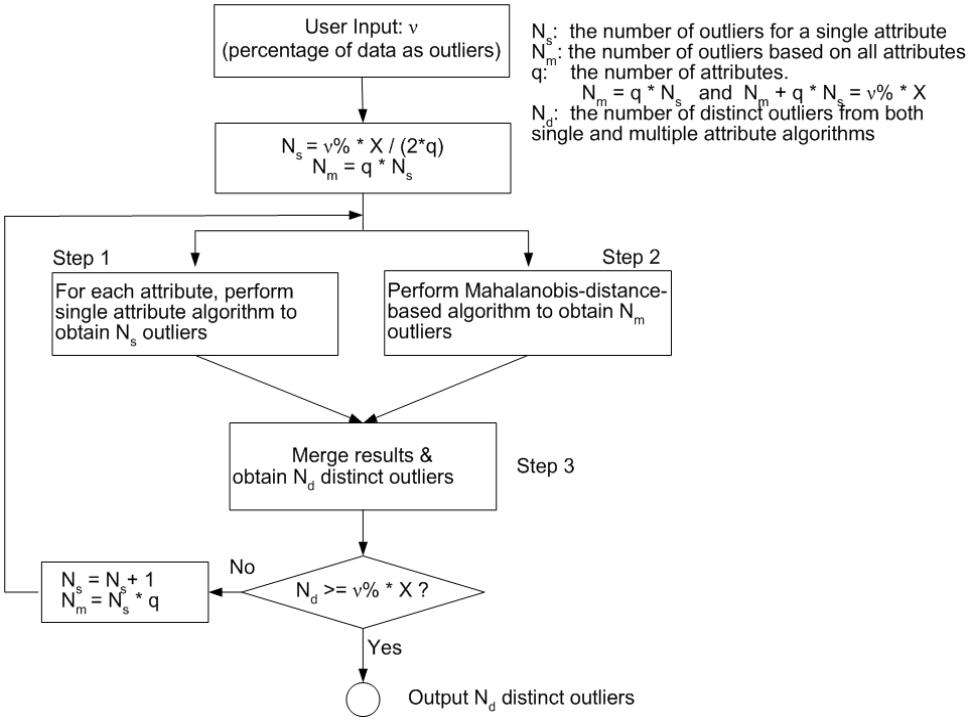


Figure 4.1. Experimental design for multiple attribute outlier detection.

user input ν. To resolve this issue, we adopted an iterative method as follows. We run the single attribute algorithm with an initial value of N_s and run the multiple attribute detection algorithm with an initial value of $9N_s(= N_m)$. If the number of distinct outlier candidates detected in (a) and (b) is less than $\nu\%$ of the data set, we increase the value of N_s by 1 and run the algorithms again. This process repeats until the number of detected outlier candidates is equal to or greater than $\nu\%$ of the data set. The approximate initial value of N_s is $\frac{\nu\%*|X|}{2*q}$, where $|X|$ denotes the power of the dataset. This is the smallest possible value for N_s, based on the assumption that there are no common outlier candidates in the results of (a) and (b).

Our experimental result is reported as follows. We choose $\nu = 5$, so at least 156 counties (5% of 3109) will be identified as outlier candidates. The initial value of $N_s \approx \frac{5\%*3109}{2*9} \approx 9$. The above iterative approach yields a final value of $N_s = 18$. Table 4.3 shows the result of the single attribute algorithm. In this table, both the original attribute values and standardized attribute values are listed for the 162 $(9 * 18)$ counties that are considered to be outlier candidates. Clearly, some counties appear as outlier candidates for more than one attribute. After removing the duplicate ones, we obtain 113 counties (not shown here).

When the multiple attribute algorithm was performed, all the 9 attributes were used. The top 162 counties with the largest Mahalanobis distances were selected as candidate outliers, as shown

Rank	Bird 2001	Vet 2001	Mosquito 2001	Bird 2002	Vet 2002	Mosquito 2002	Bird 2003	Vet 2003	Mosquito 2003
1	Dist. of Columbia, DC 359.0 (21.18)	Marion,FL 79.0 (32.21)	Richmond,NY 110.0 (31.43)	Fulton,GA 248.0 (14.95)	Holmes,OH 134.0 (14.30)	Cuyahoga,OH 351.0 (24.43)	Harris,TX 248.0 (20.01)	Lancaster,PA 168.0 (29.95)	Nueces,TX 342.0 (21.57)
2	Middlesex,MA 340.0 (20.05)	Plymouth,MA 38.0 (15.44)	Bergen,NJ 81.0 (23.12)	Harris,TX 208.0 (12.48)	Marion,FL 121.0 (12.86)	Hamilton,OH 292.0 (20.30)	Tulsa,OK 158.0 (12.63)	Chester,PA 98.0 (17.35)	Cook,IL 305.0 (19.22)
3	Suffolk,NY 257.0 (15.12)	Leon,FL 38.0 (15.44)	Monmouth,NJ 54.0 (15.39)	Suffolk,NY 180.0 (10.76)	Elkhart,IN 114.0 (12.09)	Cook,IL 289.0 (20.09)	Suffolk,NY 173.0 (13.86)	San Juan,NM 90.0 (15.91)	Shelby,TN 270.0 (17.00)
4	Baltimore city, MD 215.0 (12.63)	Alachua,FL 39.0 (15.85)	Philadelphia,PA 44.0 (12.52)	New Castle,DE 180.0 (10.76)	Weld,CO 99.0 (10.43)	Marion,IN 218.0 (15.12)	El Paso,CO 155.0 (12.39)	Fremont,WY 52.0 (9.06)	Dawson,NE 260.0 (16.36)
5	Monmouth,NJ 183.0 (10.73)	Jefferson,FL 42.0 (17.08)	Camden,NJ 50.0 (14.24)	Tulsa,OK 154.0 (9.15)	Wayne,OH 86.0 (8.99)	Franklin,OH 178.0 (12.32)	Albany,NY 165.0 (13.21)	Weld,CO 52.0 (9.06)	St. Louis,MO 179.0 (11.21)
6	Cuyahoga,OH 160.0 (9.36)	Clay,FL 39.0 (15.85)	Fulton,GA 32.0 (9.08)	Hartford,CT 181.0 (10.82)	Scotts Bluff,NE 92.0 (9.66)	Ouachita Parish, LA 125.0 (8.61)	Hennepin,MN 145.0 (11.57)	El Paso,CO 44.0 (7.62)	Hamilton,OH 160.0 (10.01)
7	Bristol,MA 177.0 (10.37)	Lee,MS 22.0 (8.90)	Jefferson,FL 29.0 (8.22)	Milwaukee,WI 157.0 (9.34)	LaGrange,IN 82.0 (8.55)	Dawson,NE 121.0 (8.33)	Bay,FL 108.0 (8.54)	Yellowstone,MT 35.0 (6.00)	Scotts Bluff,NE 159.0 (9.94)
8	Bucks,PA 153.0 (8.95)	Barnstable,MA 1.0 (0.31)	Cuyahoga,OH 25.0 (7.08)	Allegheny,PA 162.0 (9.65)	Allen,IN 69.0 (7.11)	Lorain,OH 121.0 (8.33)	Larimer,CO 125.0 (9.93)	Bucks,PA 34.0 (5.82)	Fairfax city,VA 0.0 (-0.16)
9	Fairfield,CT 138.0 (8.06)	Suwannee,FL 19.0 (7.67)	Suffolk,NY 52.0 (14.81)	Middlesex,MA 203.0 (12.17)	Taylor,TX 60.0 (6.12)	St. Louis,MO 119.0 (8.19)	Gwinnett,GA 93.0 (7.31)	Montrose,CO 28.0 (4.74)	Cuyahoga,OH 166.0 (10.39)
10	Plymouth,MA 257.0 (15.12)	Lowndes,GA 17.0 (6.85)	Passaic,NJ 31.0 (8.80)	Davidson,TN 138.0 (8.17)	Butte,SD 56.0 (5.67)	Arlington,VA 117.0 (8.05)	Rockingham,NH 101.0 (7.96)	Cascade,MT 29.0 (4.92)	Fairfax,VA 149.0 (9.31)
11	Washington,RI 153.0 (8.95)	Taylor,FL 23.0 (9.31)	Cook,IL 20.0 (5.65)	Albany,NY 137.0 (8.11)	Denton,TX 68.0 (7.00)	Shelby,TN 108.0 (7.41)	Henrico,VA 70.0 (5.42)	Bernalillo,NM 42.0 (7.26)	Lorain,OH 144.0 (8.99)
12	Camden,NJ 179.0 (10.49)	Suffolk,NY 17.0 (6.85)	Fairfield,CT 16.0 (4.50)	Escambia,FL 124.0 (7.31)	Wright,MN 67.0 (6.89)	Medina,OH 1.0 (-0.08)	Putnam,NY 109.0 (8.62)	Hot Springs,WY 4.0 (0.42)	Falls Church,VA 0.0 (-0.16)
13	Rockland,NY 153.0 (8.95)	Hamilton,FL 2.0 (0.72)	Queens,NY 32.0 (9.08)	Rockland,NY 138.0 (8.17)	Brookings,SD 51.0 (5.12)	Bergen,NJ 128.0 (8.82)	Los Angeles,CA 65.0 (5.01)	Los Alamos,NM 1.0 (-0.12)	Saginaw,MI 114.0 (7.08)
14	Mercer,NJ 18.0 (0.93)	Nassau,FL 16.0 (6.45)	Baltimore city, MD 14.0 (3.93)	Dist. of Columbia, DC 175.0 (10.45)	Calcasieu Parish, LA 45.0 (4.46)	Lycoming,PA 101.0 (6.92)	Teller,CO 1.0 (-0.23)	Park,WY 27.0 (4.56)	Weld,CO 145.0 (9.05)
15	Barnstable,MA 23.0 (1.23)	Duval,FL 27.0 (10.94)	Bronx,NY 15.0 (4.21)	Cobb,GA 119.0 (7.00)	Sangamon,IL 50.0 (5.01)	Harris,TX 99.0 (6.78)	Hartford,CT 105.0 (8.29)	Larimer,CO 32.0 (5.46)	Hall,NE 105.0 (6.51)
16	Suffolk,MA 31.0 (1.70)	Lafayette,FL 3.0 (1.13)	Mercer,NJ 1.0 (0.20)	DeKalb,GA 124.0 (7.31)	Lancaster,PA 42.0 (4.13)	Summit,OH 100.0 (6.85)	Goshen,WY 64.0 (4.93)	Carroll,MD 44.0 (7.62)	Dallas,TX 102.0 (6.32)
17	Leon,FL 115.0 (6.69)	Burlington,NJ 12.0 (4.81)	Ocean,NJ 6.0 (1.63)	Monmouth,NJ 157.0 (9.34)	Randall,TX 53.0 (5.34)	Richmond,NY 93.0 (6.36)	Oklahoma,OK 59.0 (4.52)	York,PA 41.0 (7.08)	York,PA 112.0 (6.95)
18	Ocean,NJ 18.0 (0.93)	Ocean,NJ 0.0 (-0.10)	Middlesex,MA 12.0 (3.35)	Bucks,PA 142.0 (8.42)	Harris,TX 45.0 (4.46)	Jefferson,OH 74.0 (5.03)	Ramsey,MN 88.0 (6.90)	Maricopa,AZ 29.0 (4.92)	Garfield,NE 93.0 (5.75)

Table 4.3. Top 18 spatial outlier candidates, detected by the Median single-attribute algorithm, and their original/standardized attribute values.

in Table 4.4. In fact, the top 78 counties detected by the multiple attribute algorithm were also identified by the single attribute algorithm. From the 79th county, there were 50 counties detected by the multiple attribute algorithm but not located by the single attribute algorithm. The rankings of these 50 counties were the 79th, 84th, 86th, 90th, 92th, $\cdots$, and 162th.

Rank	County	Mahalanobis Distance	Bird 2001	Vet 2001	Mosquito 2001	Bird 2002	Vet 2002	Mosquito 2002	Bird 2003	Vet 2003	Mosquito 2003
1	Marion,FL	1284.74	2.35	32.21	-0.08	2.13	12.86	-0.15	-0.15	3.30	-0.16
2	Richmond,NY	1182.91	2.89	-0.10	31.43	1.27	-0.52	6.36	-0.31	-0.30	-0.16
3	Lancaster,PA	999.38	0.39	-0.10	-0.08	4.10	4.13	3.91	1.08	29.95	6.83
4	Nueces,TX	688.40	-0.14	-0.10	-0.08	-0.33	0.15	4.26	-0.31	0.06	21.57
5	Cuyahoga,OH	675.38	9.36	-0.10	7.08	1.15	0.26	24.43	0.10	-0.30	10.39
...	...	...	...	...	...	...	...	...	...	...	...
79	Boulder,CO	59.58	-0.14	-0.10	-0.08	-0.02	-0.19	-0.15	3.78	2.94	7.34
...	...	...	...	...	...	...	...	...	...	...	...
84	Chase,NE	56.25	-0.14	-0.10	-0.08	-0.21	0.03	-0.15	0.01	-0.30	5.30
...	...	...	...	...	...	...	...	...	...	...	...
86	Middlesex,CT	55.70	1.17	0.31	-0.08	1.58	-0.52	-0.15	4.19	-0.30	0.34
...	...	...	...	...	...	...	...	...	...	...	...
90	Anne Arundel,MD	52.43	0.04	0.31	-0.08	3.43	0.03	0.55	0.10	2.76	0.47
...	...	...	...	...	...	...	...	...	...	...	...
92	Erie,PA	51.47	-0.14	-0.10	0.20	0.65	-0.52	4.96	0.92	-0.30	0.03
...	...	...	...	...	...	...	...	...	...	...	...
94	Wakulla,FL	50.37	3.72	4.40	-0.08	-0.33	-0.52	-0.15	-0.07	-0.12	-0.16
...	...	...	...	...	...	...	...	...	...	...	...
96	Lancaster,NE	49.48	-0.14	-0.10	-0.08	4.90	1.69	0.06	2.31	0.06	2.63
97	Montgomery,PA	48.77	3.13	0.72	0.20	8.42	-0.30	2.79	2.14	3.84	4.60
98	Baltimore,MD	48.74	5.15	2.36	0.20	4.04	-0.52	0.13	-0.31	3.30	0.79
...	...	...	...	...	...	...	...	...	...	...	...
100	Middlesex,NJ	47.91	8.41	0.31	6.79	2.81	-0.41	2.44	1.57	-0.30	1.11
101	Westchester,NY	47.87	2.18	-0.10	1.92	3.80	-0.52	3.00	-0.15	-0.30	0.79
103	Providence,RI	45.96	1.34	-0.10	1.92	4.90	-0.52	0.06	0.59	-0.12	-0.10
...	...	...	...	...	...	...	...	...	...	...	...
106	Lake,IL	45.25	0.22	-0.10	-0.08	0.47	1.14	4.82	0.01	-0.12	0.85
107	Montgomery,MD	43.57	2.77	-0.10	-0.08	6.07	-0.41	0.27	-0.31	2.58	0.85
...	...	...	...	...	...	...	...	...	...	...	...
109	Alexandria city,VA	43.06	3.54	-0.10	-0.08	0.96	-0.52	-0.15	0.34	-0.30	1.49
110	Otero,CO	42.71	-0.14	-0.10	-0.08	0.10	2.69	-0.15	0.18	0.06	4.16
111	Dauphin,PA	41.88	-0.08	-0.10	-0.08	2.81	-0.41	3.49	2.31	0.78	6.83
...	...	...	...	...	...	...	...	...	...	...	...
114	Newport,RI	40.13	1.70	-0.10	1.63	1.15	-0.52	-0.15	0.26	-0.30	-0.04
...	...	...	...	...	...	...	...	...	...	...	...
116	Okaloosa,FL	39.41	1.34	0.31	-0.08	-0.09	-0.30	-0.15	4.44	0.42	-0.16
...	...	...	...	...	...	...	...	...	...	...	...
118	Morris,NJ	37.87	2.06	-0.10	4.50	4.53	-0.52	2.30	1.90	-0.12	0.47
119	Travis,TX	36.50	-0.14	-0.10	-0.08	-0.27	0.81	-0.15	1.49	0.78	4.09
...	...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...	...
161	DuPage,IL	24.05	1.11	-0.10	-0.08	0.84	0.59	1.95	1.08	-0.30	4.86
162	Muskogee,OK	23.71	-0.14	-0.10	-0.08	3.55	-0.19	-0.15	1.33	0.06	-0.16

Table 4.4. Top 162 spatial outlier candidates, detected by the multiple attribute algorithm, and their associated standardized attribute values. The rank is based on the magnitude of the Mahalanobis distance.

After merging the results from the single attribute detection algorithm and multiple attribute detection algorithm, we obtained $113+50 = 163$ counties that were considered to be spatial outliers.

We conclude this section by making several remarks regarding the above experiments.

Remark 1. The merging is done on the basis of two result sets, one from (a) and one from (b). Within each result set, the outlierness of the counties can be ranked. However, in general the outliers from these two different result sets can not be compared as the underlying detection rules are totally different.

Remark 2. Further examination coupled with specific knowledge of epidemiology of the WNV can be conducted to indicate whether a detected county is a true spatial outlier. For instance,

Marion County (Florida) is selected as the top outlier candidate on the basis of attribute Vet-2001. Figure 4.2 shows that Marion County has 79 veterinary WNV cases while its neighboring counties have attribute values of 9,39,9,9,1,5 and 1. Clearly, Marion County is an outlier, since its attribute value is much higher than those of its neighbors.

Remark 3. The multiple detection algorithm is indispensable in detecting spatial outliers. This is because it is able to identify outliers ignored by the single attribute outlier algorithm. For example, when using the single attribute outlier algorithm, Montgomery County(MD), the 107th candidate detected by the multiple attribute algorithm, ranks the 201th, 1212th,84th, 49th, 1080th, 46th, 254th, 37th, and 185th for the 9 attributes respectively. Obviously, Montgomery County is not likely to be detected by the single attribute algorithm, since the algorithm stops for $N_s = 18$. Although based on each single attribute Montgomery County does not look much different from its neighbors, it is indeed an outlier since its high Mahalanobis distance reveals that the overall property of its 9 attributes is much different from those of its neighbors.

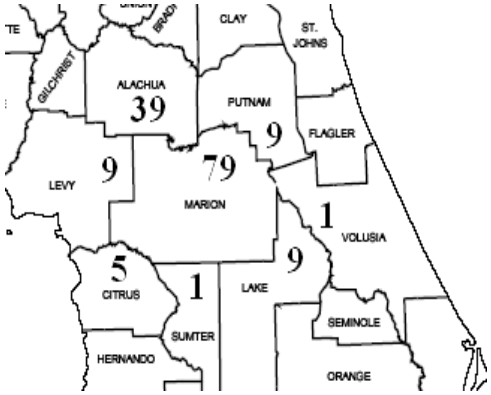


Figure 4.2. Distribution of the values of attribute vet-2001 in the vicinity of Marion County.

Chapter 5

Graph-based Outlier Detection

The outlierness of a spatial object can be evaluated by comparing the nonspatial attribute values of this object with those of its k-nearest neighbors (kNN). The entire data set can be represented with a number of intra-connected subgraphs based on the kNN relationships. k directed edges can be drawn from each object to its k-nearest spatial neighbors, and the weight of an edge denotes the absolute nonspatial difference between two neighboring nodes. In this paper, we propose two algorithms to detect both point outliers and region outliers based on the kNN graph. The proposed algorithms have two major advantages compared with the existing spatial outlier detection methods: accurate in detecting point outliers and capable of identifying region outliers.

The rest of this chapter is organized as follows. Section 5.1 provides the motivation of our approaches; Section 5.2 presents two spatial outlier detection algorithms based on kNN graph; Section 5.3 illustrates the experimental design and results; and finally Section 5.4 summarizes our work and points out future research directions.

5.1 Motivation and Problem Formulation

The existing spatial outlier detection algorithms have several deficiencies. **First**, As described in [59], if an object has exceptionally large or small nonspatial attribute values, it will have negative impact on its spatial neighbors, such that some of them may falsely marked as outliers and the "true" outliers may be overlooked. Figure 5.1 shows an example spatial data set, where x and

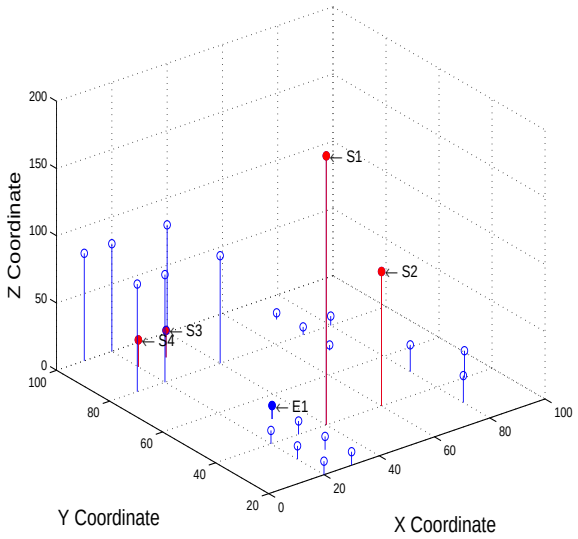
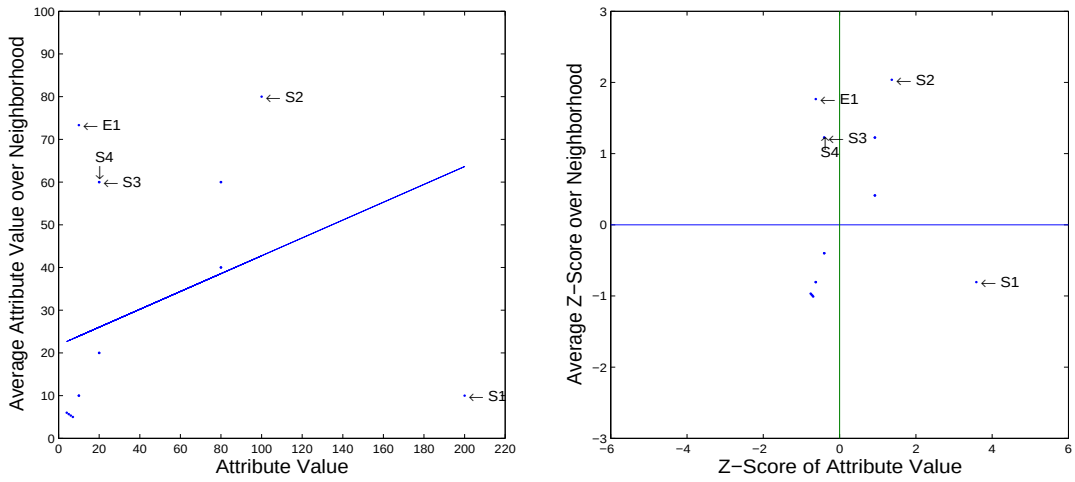


Figure 5.1. An example data set, where X and Y coordinates denote the spatial locations and Z coordinate represents the value of nonspatial attribute.

y coordinates denote the 2D spatial locations and z coordinate represents the value of nonspatial attributes. In this data set, four spatial outliers are expected, $S1$, $S2$, $S3$, and $S4$, because their nonspatial attribute values are much larger or smaller than their spatial neighbors. Table 5.1 shows the outlier detection results of several existing algorithms on the data set in Figure 5.1, including z-value algorithm, Moran-scatterplot, and scatterplot. We can observe that none of them can accurately identify all the four "true" outliers. Moreover, these three algorithms identify a "false" outlier, $E1$, which is in fact a normal point. $E1$ is erroneously detected because the nonspatial attribute value of its neighbor $S1$ is extremely high (200), and therefore dominates the neighborhood average of $E1$. Figure 5.2(a) and Figure 5.2(b) demonstrate scatterplot and Moran scatterplot of the example data set. In Figure 5.2(a), the points far from the regression line are marked as outliers. In Figure 5.2(b), the points located in the upper left and lower right quadrants are identified as outliers. We can see that both scatterplot and moran-scatterplot mistakenly detect $E1$ as an outlier.

Rank	Methods			
	Scatter-plot	Moran Scatterplot	z Alg.	Graph-based
1	S1	S1	S1	S1
2	E1	E1	E1	S2
3	S2	S3	S3	(S3,S4)

Table 5.1. Top three spatial outliers detected by scatterplot, Moran scatterplot, z-value, and graph-based method.



(a) Scatterplot (b) Moran-Scatterplot

Figure 5.2. The Scatterplot and Moran-scatterplot of the data set shown in Figure 5.1.

Second, most of the existing algorithms focus on evaluating the outlierness of single object and are not suitable for detecting region outliers. If a group of outlying objects cluster together and have similar nonspatial attribute values, they might be erroneously marked as normal points. As shown in Table 5.1, neither scatterplot, Moran-scatterplot, nor z-value approach can identify the region outlier (S3,S4). In many applications, spatial outliers appear in the form of connected components, or regions. For example, a hurricane zone consists of a group of points whose wind speed, air pressure, and water vapor density are much different from the surrounding areas.

Third, some existing methods such as z-value approach and Moran-scatterplot identify spatial outliers by calculating the normalized non-spatial attribute difference between each object and the average of its spatial neighbors. This normalization is across the entire data set, which may not be appropriate for certain conditions. For example, if the data set consists of a number of spatial clusters, the objects in the same cluster are spatially correlated to each other and the objects in different clusters have no direct correlations. Therefore, normalization based on the entire data set may not be suitable. One potential solution for this issue is to first identify the object clusters in the data set, and then consider statistic characteristics of each cluster individually to detect outliers.

To address the above deficiencies, we propose a set of graph-based algorithms to accurately identify both point outliers and region outliers. Based on the example data set in Figure 5.1, we first construct a kNN graph as shown in Figure 5.3. Each circle represents an object whose

nonspatial attribute value is marked within the circle. An arrow pointing from a node x_1 to another node x_2 represents that x_2 is one of x_1's k nearest neighbors. If x_1 and x_2 are connected by a double directed edge, they are k-nearest neighbors of each other. The edge weight is the absolute difference between the nonspatial attribute values of two neighboring nodes. For example, the weight of the edge connecting $S1$ and $S2$ is $100(200 - 100)$. Note that there are 3 subgraphs in the kNN graph. To make the edges comparable among different subgraphs, a standardization process can be performed for each subgraph independently. The proposed graph-based algorithms then continuously cut the longest edge, i.e., the edge with the largest weight until certain number of points or connected regions have been isolated. The isolated points will be marked as point outliers since their nonspatial attribute values are much different from those of their neighbors. The regions will be marked as region outliers once the verification procedure validates that their degrees of outlierness are beyond thresholds.

Figure 5.4 shows the edge-cut result of Figure 5.3, where three outliers are identified, $S1$, $S2$, and ($S3$, $S4$). One of them consists of two spatial objects, forming a region outlier. ($S3$, $S4$) is labelled as a region outlier because they have been isolated from other points and their nonspatial attribute values are similar to each other but dissimilar to those of their neighbors. A region is marked as "isolated" when there are no edges going out from each point in that region. The neighbors of a region consist of all points which have directed edges going out of that region. Table 5.1 shows that the graph-based method can detect all expected spatial outliers and is capable of identifying the outlier region. In summary, the graph-based algorithms have the following advantages compared with the existing spatial outlier detection methods. (1) Detect both point and region outliers. Based on graph connectivity, neighboring points with similar nonspatial attribute values can be grouped together conveniently; (2) Avoid identifying "false" outliers and correctly locate "true" outliers; (3) More "local". We only consider the outlierness of an object within a subgraph where this object belongs to, instead of normalizing data across the entire data set.

5.2 Detection Algorithms

In this section, we propose two graph-based algorithms. The first algorithm POD is designed for point outlier detection, and the second algorithm ROD is to identify region outliers. Computational

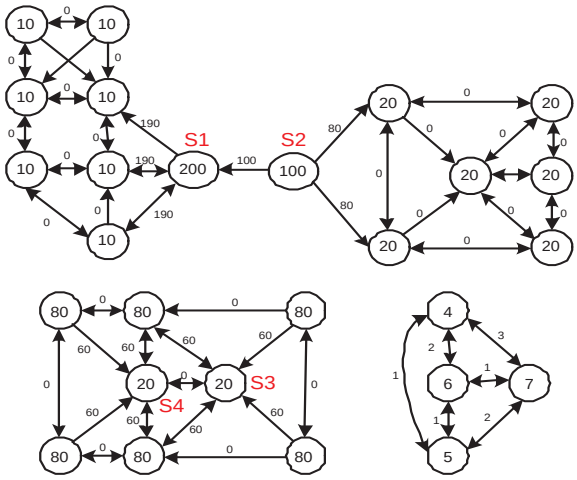


Figure 5.3. The kNN based graph representation of a small data set, $k = 3$.

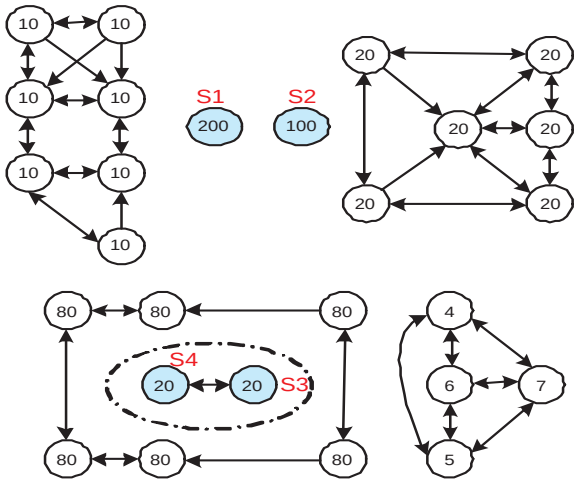


Figure 5.4. The result of graph-based outlier detection: 2 point outliers and 1 region outlier are identified.

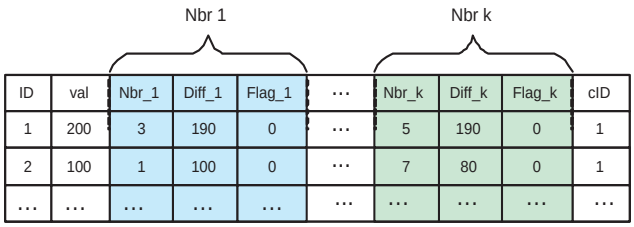


Figure 5.5. The data structure for graph representation.

complexities of both algorithms are examined as well.

The major step of spatial outlier detection is to compare the nonspatial values between each object and its spatial neighbors. Thus, a graph $G(X, E)$ can be created based on the k-nearest neighbors (kNN) relationship. Each node represents an object x_i, and k edges direct from x_i to each of its k-nearest neighbors. The weight of an edge represents the dissimilarity between two neighboring points. For single (nonspatial) attribute data set, we can use the absolute difference of the nonspatial attribute as the weight. For a data set with multiple (nonspatial) attributes, we can use the Mahalanobis distance between two nodes as the weight. The outgoing degree of each node is k, the number of neighbors. Note that the graph may consist of a set of subgraphs which are not connected to each other. We call each subgraph as a cluster and consider their different statistic characteristics in spatial outlier detection.

The data structure for the graph is shown in Figure 5.5. Each node (object) x_i is represented by a tuple with $3k + 3$ elements, where x_i=$\langle$ id, val, nbr_1, $diff_1$, $flag_1$, ..., nbr_k, $diff_k$, $flag_k$, cID $\rangle$. id denotes the unique identification of a node; val is the nonspatial attribute value of x_i; $nbr_j(1 \leq j \leq k)$ represents the j-th neighbor of x_i; $diff_j$ stands for the weight of the edge $\overrightarrow{x_i, x_j}$; $flag_j$ indicates the status of edge $\overrightarrow{x_i, x_j}$ ($flag_j = 1$ denotes that the edge has been cut and $flag_j = 0$ represents that x_i and x_j are still connected); cID refers to the subgraph which x_i belongs to. The neighbor list is sorted by their edge weights in descending order, *i.e.*, $diff_1 \geq diff_2 \geq \ldots \geq diff_k$. The reverse k nearest neighbors($rKNN$) of each object also need to be stored. The $rKNN$ of an object x_i is a set of objects whose k nearest neighbors contains x_i. We can use a data structure similar to Figure 5.5 to store the $rKNN$ of each node. The only difference is that the number of objects in $rKNN$ may not be fixed as k, so the tuple for each node is a variant-length array. The reverse nearest neighbor table can be contructed from the nearest neighbor table and used for extracting connected regions from the graph.

In addition, a priority queue *EdgeQueue* is created to maintain all edges, which are sorted based on their weights in descending order. An array *clusterArr* contains a set of subgraphs. Each subgraph is represented as a four element tuple $cluster_i$, where $cluster_i$=<cID, $mean$, std, $size$>. cID is the unique identification of a subgraph; $mean$ is the average nonspatial attribute value of all objects in the subgraph; std is the standard deviation of nonspatial attribute values in this subgraph; and $size$ is the number of objects in the subgraph. These statistics can be used for normalizing edge weights in each subgraph.

5.2.1 Algorithm 1: Point Outlier Detection

This section introduces an algorithm to detect single point outliers based on kNN graph. We assume that all $k(x_i)$ are equal to a fixed number k. (The algorithm can be easily generalized by replacing the fixed k by a dynamic $k(x_i)$.)

Algorithm 3 : Point Outlier Detection (POD)

Input:
X is the set of n spatial objects;
Y is the set of nonspatial attribute values for X;
k is the number of neighbors;
m is the number of requested spatial outliers;
Output:
O_s is a set to store detected spatial outliers

```
for(i=1; i ≤ n ; i++) {
   /* calculate the neighborhood relationship */
   NN_k(x_i) = GetNeighbors(X, x_i); }
/* Generate a graph G based on NN_k */
G = createGraph(X, Y, NN_k);
/* standardize the attribute values by subgraph */
G = standardize(G);
/* create the edge priority queue */
EdgeQueue = createEdgeQueue(G);
O_s = empty; /* initialize O_s */
while ( sizeOf(O_s) ≤ m ) {
   edge = DeQueue(EdgeQueue); /* select the longest edge */
   /* cut this edge and return the starting node */
   o = CutEdge(G, edge);
   /* check if o has become an isolated point*/
   if ( isIsolated(o) ) {
      MarkOutlier(O_s, o); } }
/* output the outliers */
Output(O_s);
```

The proposed algorithm has four input parameters. X is a set of n objects containing spatial attributes such as location, boundary, and area. The non-spatial attributes are contained in another set Y. In many applications, these nonspatial attribute values can be the results of preprocessing procedure like dimension reduction or data standardization. k is the number of neighbors. m is the number of requested outliers. Generally, m should not be greater than 5% of n, assuming the nonspatial attribute follows normal distribution with confidence interval of 95%.

For each data object, the first step is to identify its k nearest neighbors. Calculating the Euclidean distance between the centers of two objects is the most commonly used approach. Next, based on the k-nearest neighbor set NN_k, a graph G is constructed. G may contain multiple subgraphs which are disconnected with each other. Since different subgraphs may have distinct characteristics, the nonspatial attribute values need to be standardized first within each subgraph. In this way, the edge weights are comparable among different subgraphs. Next, a priority queue *edgeQueue* is created to arrange all edges in descending order based on their weights. To partition the graph, a "longest-edge-cut" strategy is employed and the partition will be conducted with multiple iterations. In each iteration, the longest edge $\overrightarrow{x_i, x_j}$ (the edge with the highest weight) is dequeued from *edgeQueue* and cut. Here, cutting a edge is equal to set the $flag_j$ of the node x_i as "1." The $CutEdge$ function returns the starting node of the edge. After each cutting operation the algorithm will check if this starting node o has become isolated, i.e., no outgoing edges. If true, o is marked as a spatial outlier and inserted into the outlier set O_s. Finally, when all m spatial outliers are identified, the partition loop stops and O_s is output. The ranking of outliers is determined by their identification sequence.

5.2.2 Algorithm 2: Region Outlier Detection

One major benefit of the graph-based method is that it can detect region outliers. A region outlier consists of a group of adjoining spatial objects whose nonspatial attributes are similar to each other but quite different from the surrounding objects of this group. In a partitioned graph, a connected component can be deemed as a region outlier if the variance in the component is small and the variance between the component and its neighborhood is large. Algorithm 2 is designed especially for discovering region outliers. For simplicity, we call it *ROD* algorithm. *ROD* has 6 input parameters: X contains the spatial attributes of the data set; Y contains the nonsptial attribute values of the data set; k is the number of neighbors; T_{EDGE} denotes the threshold for edge-cutting; T_{SIM} represents the threshold for evaluating the evenness within a region; T_{DIFF} is

the threshold to measure the difference between a region and its neighbors. The detailed algorithm is described as follows.

Algorithm 4 : Region Outlier Detection (ROD)

Input:
X is a set of n spatial objects;
Y is the set of nonspatial attribute values for X;
k is the number of neighbors;
T_{EDGE} is the threshold for the edge weights to be cut;
T_{SIM} is the threshold of within region similarity;
T_{DIFF} is the threshold of between-region difference;
Output:
O_s is a set of region outliers

```
for(i=1; i ≤ n ; i++) {
   /* calculate the neighborhood relationship */
   NN_k(x_i) = GetNeighbors(X, x_i); }
/* Generate a kNN graph G */
G = createGraph(X, Y, NN_k);
/* create a priority queue based on edge weight */
EdgeQueue = createEdgeQueue(G);
edge = DeQueue(EdgeQueue); //select the longest edge
while ( weightOf(edge) > T_EDGE ) {
   G = CutEdge(G, edge); // cut edge in the graph
   /* select the longest edge */
   edge = DeQueue(EdgeQueue); }
/* find all regions with size less than k*/
regions = findCandidateRegions(G);
for(i=1; i < sizeOf(regions) ; i++) {
   if ( regionEvenness(regions[i]) > T_SIM ) {
      nbrs = getNbrOfRegion(G, regions[i]);
      if ( computeDiff(regions[i], nbrs > T_DIFF) {
         markOutlier(O_s, regions[i]); } } }
/* output the outliers */
Output(O_s);
```

Similar to Algorithm 1, a graph first needs to be created based on the kNN relationship. Then a priority queue *edgeQueue* is to rank all edges in descending order based on their weights. Next, the algorithm continuously selects the longest edge from the *edgeQueue* and cuts it if the weight of this edge is larger than threshold T_{EDGE}. After cutting all edges with weight above T_{EDGE}, a function is called to detect the candidate region outliers, *regions*. Each of these candidates is a connected region which contains less than k objects. *regions* is only a candidate set, and each region in this set needs further verification. The verification includes three steps: (1) for a region

$region[i]$, determine the set of its spatial neighbors, $nbrs$, which contains all the distinct k-nearest neighbors of each point in $regions[i]$. If the size of $nbrs$ is smaller than the size of $regions[i]$, then $region[i]$ is not a region outlier; (2) check the evenness of nonspatial attribute values in $regions[i]$. If the degree of evenness is smaller than the threshold T_{SIM}, $regions[i]$ should not be identified as an outlier; (3) compute the nonsptial attribute value difference between $region[i]$ and $nbrs$. If the difference is less than the threshold T_{DIFF}, $regions[i]$ is not an outlier. Once $regions[i]$ has been verified as a region outlier through the above procedures, it will be inserted into O_s.

There are two essential issues in setting the ROD algorithm: how to evaluate the evenness within a region and how to measure the difference between a region and its neighborhood. The evenness of a region can be evaluated by "Coefficient of Variation" (dividing standard deviation by mean), by "Inter-Quartile Range" (difference between the first and third Quartiles), or by "MinMax Difference" ($\frac{Max-Min}{Mean}$). The variation between two regions can be measured using $diff = \frac{|m_r - m_{nbr}|}{(m_r + m_{nbr})/2}$, where m_r is the median nonspatial attribute value of a region r and m_{nbr} is the median nonspatial attribute value of r's neighbors. A high $diff$ value represents large difference between a region and its neighbors.

5.2.3 Time Complexity

In the POD algorithm, a k nearest neighbor (kNN) query is issued for each spatial point. It will take $O(n)$ to perform kNN query for n objects if a grid-based indexing is used and the grid directory resides in memory. The cost of generating a kNN graph is $O(kn)$, and edge weight standardization takes $O(kn)$ computation. The priority queue generation has time complexity of $O((kn)log(kn))$ based on heap sort. The number of edge cuts is indeterministic, so we assume there are at most n edge-cuts. The cost of checking "isolated" status of a point is $O(k)$, thus the cost for checking all edge cuts is $O(kn)$. In summary, assuming $n \gg k$ and $n \gg m$, the total time complexity of POD algorithm is $O(n) + O(kn) + O(kn) + O((kn)log(kn)) + O(kn) \approx O(nlogn)$ for grid-base indexing. The computation cost is primarily determined by priority queue generation.

For the ROD algorithm, the time complexity of kNN computation, graph generation, and edge weight priority queue creation is the same as that of the POD algorithm. For edge cutting cost, we can assume the total number of edge-cuts is n, leading to complexity of $O(n)$. In addition, it takes $O(n)$ to identify candidate regions. The verification procedure costs $O(mk)$, where m is the number of detected region outliers and $m \ll n$. Finally, the total time complexity of the ROD algorithm is $O(n) + O(kn) + O((kn)log(kn)) + O(n) + O(n) + O(mk) \approx O(nlogn)$, which is also

primarily determined by priority queue generation.

5.3 Experiments

In this section, we discuss our experiments on a real data set, Fair Market Rents data provided by the $PDR - DHUD$ (Policy Development and Research, U.S. Department of Housing and Urban Development), to validate the effectiveness of our proposed algorithms. Experimental design is introduced first, followed by result analysis.

5.3.1 Experiment Design

The Fair Market Rent data contain the 50th percentile rents for fiscal year 2005 at county level. This data include the rental prices for efficiencies, one-bedroom apartments, two-bedroom apartments, three-bedroom apartments, and four-bedroom apartments in 3000+ counties across the United States. The proposed algorithms can help administrative personnel identify and then investigate outlier counties whose rental prices are much different from their neighboring counties. Apartment renters can also explore the rent data to find a place where the rent is abnormal.

The location of each county is determined by the county center, in the form of longitude and latitude. The proposed algorithms can facilitate the discovery of abnormal rent rates by comparing reports from neighboring counties. The number of neighbors was chosen to be $k = 8$, which represents eight different directions from the centering county: East, West, North, South, Northeast, Northwest, Southeast, and Southwest. The distance between a county and its neighbors is obtained by calculating the Euclidean distance between their centers. The experiments contain two parts: point outlier detection and region outlier detection. For the point outlier detection methods, top ten outliers will be identified from 3095 counties.

5.3.2 Result Analysis

Point outlier Detection

We applied four algorithms to the apartment rent data set, including z-algorithm, Scatterplot, Moran-scatterplot, and the proposed graph-based point outlier detection algorithm POD. Table 5.2 shows the top ten outlier counties detected by these algorithms based on one-bedroom rent in year 2005. The number in the parenthesis denotes the average rent for one bedroom apartments in the

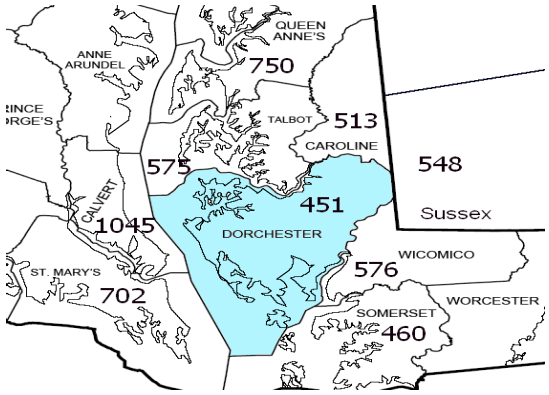


Figure 5.6. Dorchester Co.(MD) is detected by Moran-scatterplot algorithm based on one-bedroom rent data in 2005.

given county. *POD* algorithm identifies seven common counties as the z algorithm, which is a commonly used approach.

Rank	Methods			
	z Alg.	Scatterplot	MoranScatterplot	*POD* Alg.
1	Nantucket Co.,MA(1250)	Nantucket Co.,MA(1250)	Blaine Co.,ID(801)	Blaine Co.,ID(801)
2	Pitkin Co.,CO(1095)	Caroline Co.,VA(495)	Teton Co.,WY(748)	Nantucket Co.,MA(1250)
3	Frederick Co.,MD(1045)	Blaine Co.,ID(801)	Elbert Co.,CO(444)	Teton Co.,WY(748)
4	Suffolk Co.,MA(1120)	Teton Co.,WY(748)	Surry Co.,VA(446)	Summit Co.,UT(901)
5	Blaine Co.,ID(801)	Elbert Co.,CO(444)	La Paz Co.,AZ(471)	Suffolk Co.,MA(1120)
6	Summit Co.,UT(901)	Plymouth Co.,MA(636)	Moffat Co.,CO(435)	**Dukes Co.,MA(941)**
7	Teton Co.,WY(748)	Worcester Co.,MA(549)	**Dorchester Co.,MD(451)**	Fairfield Co.,CT(1239)
8	Ventura Co.,CA(1093)	Summit Co.,UT(901)	Sumter Co.,FL(415)	Dane Co.,WI(660)
9	Fairfield Co.,CT(1239)	Barnstable Co.,MA(729)	Polk Co.,WI(465)	Centre Co.,PA(610)
10	Clarke Co.,VA(956)	Pitkin Co.,CO(1095)	Polk Co.,GA(414)	Pitkin Co.,CO(1095)

Table 5.2. Top 10 spatial outliers detected by z-value, scatterplot, Moran scatterplot, and graph-partition algorithms based on 1-bedroom rent.

POD algorithm can avoid identifying "false" outliers. For example, Dorchester Co.(MD) is identified as the 7th outlier by MoranScatterplot algorithm. However, Dorchester Co. is not very outlying if we take a closer look. Figure 5.6 illustrates the rental prices of Dorchester Co. (451) and its 8 neighbors (575,576,513,1045,460,750,548,702). The number in each county denotes the average rent in US dollar. Generally, the rent difference within 140 dollar is viewed as normal. More than half of the neighboring counties have normal rent difference with Dorchester Co., so Dorchester Co. should not be identified as a spatial outlier. Dochester Co. is identified by Moran-scatterplot mainly because it has a neighbor, Calvert Co.(MD), which has a very high rent (1045) and significantly raises the average rent of the neighborhood. The *POD* algorithm does not have this problem, because the edge-cutting makes each neighboring county contribute equally to the

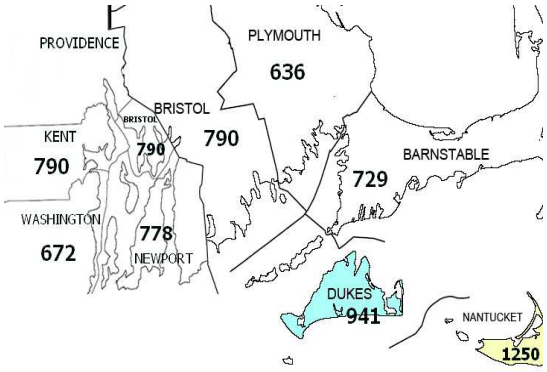


Figure 5.7. Dukes Co.(MA) identified by POD algorithm based on one-bedroom rent data in 2005.

outlierness of the centering county.

In addition, POD algorithm can detect outliers that are not identified by other three methods. For example, Dukes Co.(MA) is ranked as the sixth outlier by POD algorithm. As shown in Figure 5.7, the one-bed rent in Dukes Co. is 941, which is much higher than its 7 neighbors (difference > 150) except Nantucket Co.(MA). Nantucket Co. has been identified as the 2nd outlier which has very high attribute value, 1250. If the arithmetic average is used to represent the overall characteristics of Dukes Co.'s neighbors in other three algorithms, Nantucket Co. will significantly increase the neighborhood average. Therefore, the difference between Dukes Co. and its neighborhood average will be small, thus causing Dukes Co. not being detected by other three algorithms. However, it is a "true" outlier, since its attribute value is much different from most of its neighboring counties.

Region Outlier Detection

Outliers	Counties in the region	Neighboring counties
1	Wake,NC(829) Orange,NC(829) Johnston,NC(829) Franklin,NC(829) Durham,NC(829) Chatham,NC(829)	Wilson,NC(602) Wayne,NC(549) Warren,NC(522) Vance,NC(507) Randolph,NC(645) Person,NC(546) Nash,NC(593) Moore,NC(616) Harnett,NC(539) Greene,NC(467) Granville,NC(573) Caswell,NC(526) Sampson,NC(431) Lee,NC(576) Alamance,NC(645)
2	Hinsdale,CO(1033) Ouray,CO(1033) Mineral,CO(1033) San Miguel,CO(1060)	San Juan,CO(692) Saguache,CO(494) Rio Grande,CO(494) Montrose,CO(634) La Plata,CO(776) Gunnison,CO(757) Dolores,CO(692) Delta,CO(578) Conejos,CO(494) Archuleta,CO(742) Montezuma,CO(582)

Table 5.3. 2 region outliers detected by ROD Alg. based on two-bedroom rent data in 2005.

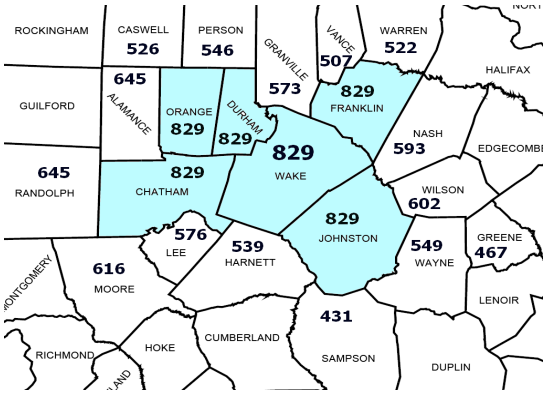


Figure 5.8. A region outlier detected by ROD algorithm based on two-bedroom rent data in 2005.

We also applied ROD algorithm to the 2-bedroom rent data in 2005. The evenness within a region is evaluated by "MinMax" difference, which is calculated by $\frac{Max-Min}{Mean}$. The threshold T_{SIM} was chosen to be 0.15, which means the difference between the minimum and the maximum nonspatial attribute values should be less than 15% of the average value. The dissimilarity between a region and its neighbor set is measured with $diff = \frac{|m_r - m_{nbr}|}{(m_r + m_{nbr})/2}$. The threshold T_{DIFF} was set to be 0.4, which means that a region will be identified as an outlier if the difference with its neighbors is more than 40% of the average of both sets. Table 5.3 shows two identified region outliers: one consists of 6 counties and the other consists of 4 counties.

Figure 5.8 shows the first region outlier with 6 counties, which have the same rent values 829 and marked in light gray. This region has 15 neighboring counties(rent ranging from 431 to 645), whose rents are much lower than the outlier region. Figure 5.9 presents the second region outlier with 4 counties, whose average two-bedroom rents are 1060, 1033, 1033, and 1033 respectively. This region has 11 neighboring counties whose rents are significantly lower than the four counties within the region. The existing algorithms are not capable of detecting a group of counties as outliers. In fact, many counties in the region can not be detected by point outlier detection methods, because their non-spatial attribute values are similar to most of their neighbors.

5.4 Summary

In this chapter, we propose two spatial outlier detection algorithms based on kNN graph. One is to detect point outliers and another is to identify region outliers. The construction of kNN graph makes it possible to connect adjacent points to a region if their nonspatial attribute values

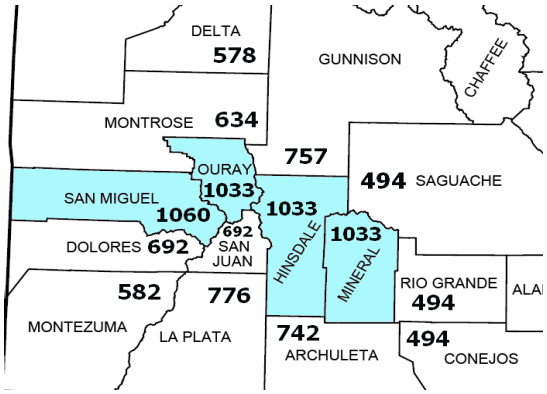


Figure 5.9. A region outlier detected by ROD algorithm based on two-bedroom rent data in 2005.

are similar. Therefore, the ability of region outlier identification distinguishes our method from the existing spatial outlier detection algorithms. In addition, the edge cut strategy can reduce the negative impact of the objects with very large/small values on their neighbors. Thus the graph-based approach can detect "true" outliers ignored by other methods and avoid identifying "false" outliers. The experimental results on the US House Rent data set validate the effectiveness of the proposed methods. This paper focuses on the single nonspatial attribute outlier detection. We are working on multi-attribute algorithms to identify point and region outliers with multiple nonspatial attributes based on kNN graphs. The research is on the way and further result will be reported in the near future.

Chapter 6

Conclusion and Future Work

This chapter summarizes our contributions in detecting various abnormal patterns in spatial data. In addition, several promising directions are discussed for future research work.

6.1 Contributions

This dissertation focuses on effective and efficient mechanisms to recognize abnormal patterns (outliers) in spatial data. Formal definitions were presented for various types of spatial outliers and a suite of algorithms were proposed to identify them accurately. In general, the contributions of this paper can be summarized as five parts: (1) single attribute outlier detection; (2) multi-attribute outlier detection; (3) region outlier detection; (4) outlier detection in continuous data sequences; and (5) several surveys and demo systems that are related to spatial data analysis. Detailed summarization is described as follows.

6.1.1 Single Attribute Outlier Detection

Single attribute outlier detection concentrates on those spatial data sets with a single non-spatial attribute. A number of methods have been developed to identify single attribute outliers. However, these methods suffer from several potential deficiencies: "Masking" and "Swamping" problems, focusing on single attribute outlier detection, and not considering spatial relationships when performing neighborhood comparison. We developed a set of algorithms to address the above issues, including median-based algorithm, iterative-based algorithms, and spatial weighted algo-

rithm [47, 59]. Their performance has been validated through extensive experiments on various spatial data sets such as US Census data, West Nile virus data, and US Housing data.

- **Median-based algorithm** uses a robust estimator median to represent the overall behavior of a set of objects, and then employs z-value to measure the outlierness. Median is used because it will not be impacted by the smallest or the largest value in a sample. Therefore, it can alleviate the negative impact of the clustering outliers. Using median to represent the average is a common method. We apply it to the spatial outlier detection to reduce the masking and swamping effect.

- **Iterative-based algorithms** are designed to reduce "masking" and "swamping" effect. The idea is to detect multiple outliers in multiple iterations and each iteration identifies only one outlier. After each iteration, the non-spatial attribute value of the detected outlier is replaced with the average of the non-spatial attribute values of its neighbors, thus mitigating the negative impact of the identified outliers on the following iterations. Based on the different ways of neighborhood comparison, we designed two iterative-based algorithms, iterative-R and iterative-Z. Iterative-R uses the ratio of the non-spatial attribute values between an object and the average of its neighbors to measure the degree of outlierness, while iterative-Z employs statistical z-value to represent the difference between an object and its neighbors.

- **Spatial weighted algorithm** is the extension of point spatial outlier detection. The existing methods usually abstract spatial objects as points and spatial attributes are only used for determining the neighborhood relationship. The outlierness of a spatial object is then solely based on the non-spatial attributes of this object and its neighbors. However, in many real applications, spatial objects can not be simply abstracted as isolated points. They have different location, area, contour, and volume, which may influence the impact of a spatial object on its neighbors and should not be ignored. Based on this observation, we propose a spatial weighted outlier detection method, which assigns different weights for different neighbors when evaluating the outlierness of the central object. The weight is determined by spatial relationships such as the center distance and common border length.

Graph-based method is mainly designed for detecting region outliers, but it can also be used to identify point outliers. Since we have a number of algorithms available for point outlier detection, one intriguing question is that which one to choose. The selection of algorithms depends on the

requirements of different applications. Generally, if the users have clear knowledge about the impact of spatial properties based on a particular application, spatial weighted algorithms should be used. Otherwise, median, iterative-based or graph-based method can be selected. If users have strict requirement about processing time, median algorithm should be selected since it computes the rankings of all objects in one single iteration. Iterative-based methods have higher computation time since it need multiple iterations. It can be used when the computation time is not a major concern or the number of expected outliers is not large. Graph-based method has comparatively high computation cost since constructing a kNN graph requires additional time (O(knlog(kn)), where k is the number of neighbors and n is the size of data set. However, if some applications require to detect both region outliers and point outliers, this method is the only choice.

Table 6.1 shows the time complexity of the above methods. m denotes the number of iterations (outliers to be identified) and k represents the number of neighbors. We can see that if grid based indexing is used, iterative-based methods and graph-based methods have higher computational cost. However, if tree-based indexing is used, the time complexities of Median algorithm, iterative-based algorithms, and spatial weighted algorithms are similar, because their computation cost is mainly determined by the k-nearest neighbor search. The time complexity of graph-based methods is determined by the graph construction and edge sorting, which is $O(knlog(kn))$.

	Methods			
Time complexity	Median alg.	Iterative z value (or ratio)	spatial weighted alg.	graph-based alg.
grid indexing	$O(n)$	$O(mn)$	$O(n)$	$O(knlog(kn))$
R-tree indexing	$O(nlogn)$	$O(nlogn)$	$O(nlogn)$	$O(knlog(kn))$

Table 6.1. The time complexity comparison between Median algorithm, Iterative-based algorithms, spatial weighted algorithms, and graph-based algorithms.

6.1.2 Multi-attribute Outlier Detection

In many applications, there are more than one non-spatial attributes associated with each spatial location. For example, in the Census data, each census tract contains several non-spatial attributes, including population, population density, income, poverty, housing, education, and race. Detecting outliers from these spatial data with multiple attributes will help demographers and social workers to identify local anomalies for further analysis. We proposed a spatial outlier detection method based on Mahalanobis distance [57,58]. The objects with extremely large Mahalanobis distances will be identified as spatial outliers. This method considers the correlations among multiple attributes

and can use for wide applications by extending the definition of inverse covariance matrix.

6.1.3 Region Outlier Detection

Most of the existing spatial outlier detection methods identify individual objects as outliers. However, in real geospatial applications an outlier can consist of a group of spatially adjacent objects with similar nonspatial attribute values. This type of outliers can be defined as region outliers [113]. We proposed two approaches to detect them. The first approach is based on image processing techniques and the second approach is based on k-NN graph. Experiments have been conducted on several data sets (e.g., NOAA water vapor data and US housing data) to validate their performance.

- **Image-based algorithm** models spatial data as digital images and applies wavelet transformation and image segmentation to identify connected components with abnormal values [61]. The image-based method first applies wavelet transformation to filter noise and present data with appropriate resolution. Then, an efficient image segmentation method, lambda-connectedness, is applied to identify regions whose non-spatial attribute values are above certain thresholds. Since fast wavelet transformation and lambda-connectedness segmentation have linear computation time, they can also be applied to detect region outliers in continuous data sequences such as meteorological data streams from satellites.

- **Graph-based algorithm** builds a graph based on the spatial locations, where each node represents a spatial object and each edge denotes the neighborhood relationship. The edge weight stands for the non-spatial dissimilarity between the two nodes. By iteratively cutting the longest edge in the graph, the isolated nodes or isolated small connected components can be identified as spatial outliers. This method does not use the neighborhood average, thus do not have the negative impact between neighbors. One prominent advantage of this method is its ability to detect both region outliers and point outliers. The adjacent objects with similar non-spatial attribute values naturally form connected regions.

Image-based approach has linear computation time, which is very suitable for detecting region outliers in continuous data sequences. However, it can not be used for detecting point outliers. Graph-based method is favorable if users need to identify both region outliers and point outliers. Its disadvantage is that the cost of constructing a k nearest neighbor graph is expensive, which make it unsuitable for stream data processing.

6.1.4 Outlier Detection in Continuous Data Sequences

Spatial data can be dynamic, arriving in the format of continuous data sequences. An interesting and challenging task is to identify abnormal patterns in the spatio-temporal data. In this dissertation, a framework was proposed to detect region outliers in continuous meteorological data and track their movements [61]. By using efficient wavelet transform and image segmentation methods, the data sequence is processed frame by frame. The center, boundary, and moving speed are calculated to record the trajectory. In addition, wavelet-based methods have been designed to detect significant changes in spatio-temporal data [114]. By applying wavelet transformation on spatial domain or temporal data, significant data variations can be detected along locations or temporal durations. Experiments on NOAA water vapor data illustrate the effectiveness and efficiency of our methods.

6.1.5 Related Surveys and Demo Systems

In addition to the above contents, we have conducted extensive work closely related to data mining, geospatial data representation, and spatio-temporal data analysis and visualization. Fraud detection is often closely related to spatial data. For example, an unusual large expenditure at a place far from the billing address may indicate a credit card fraud. Therefore, a comprehensive survey was conducted concerning the application of data mining techniques in fraud detection [49]. In addition, we investigated how *GML (Geography Markup Language)* can be used to support spatial database management and geospatial services [64]. As for demo systems, two web-based spatial data mining tools *MapView* and *MapCube* were developed to integrate spatial anomaly detection algorithms with US Census data and VDOT traffic data [60]. Another web-based visualization and analysis system was implemented for managing the surface water data of Occoquan watershed in northern Virginia [48].

6.2 Future Work

The current research work can be extended in several directions, including complex region outlier detection, statistical performance evaluation on large synthetic data sets, and parameter-free spatial outlier detection.

For region outlier detection, the current proposed algorithms focus on a single (non-spatial) attribute. However, in many real applications, multiple (non-spatial) attributes should be con-

sidered together in order to determine the outlierness. It presents a challenging task since these attributes may have inter-correlations and follow different distributions. For single attribute data, it is possible to view them as images and employ image segmentation methods to extract region outliers. However, for multi-dimensional data, the traditional image segmentation methods may not be applicable. One possible solution is to revise current image segmentation methods and take the correlations among multiple attributes into consideration. 3D region outlier detection is another important topic which can find applications in bioinformatics. For example, cell and gene data are usually not 2D-images but 3-D objects. Their 3D structures and spatial relationships may indicate important biological characteristics. Therefore, it would be interesting to develop algorithms to analyze the similarity and dissimilarity between different 3D objects and identify interesting but hidden patterns.

In this dissertation, a set of new algorithms are proposed for effective spatial outlier detection. These methods can be further improved as follows. First, better detection performance may be achieved if we combine spatial weights and multiple iterations to generate a new algorithm. Extensive experiments are needed to validate its effectiveness. Second, the experiments in this dissertation are based on several real data sets and the results are mainly evaluated by domain experts. It would be more convincing if a benchmark data set is tested to quantify the accuracy of the proposed algorithms. Unfortunately, there is no such a benchmark data set in spatial outlier detection community. We are working with domain experts trying to fully examine West Nile virus data and make it a benchmark. Third, statistical experiments and analysis should be conducted to further validate the accuracy of the proposed methods. One possible approach is to generate synthetic spatial data to simulate various distributions and evaluate the outlier detection performance.

Finally, an interesting and important direction is parameter-free spatial outlier detection. All the current methods require various parameters such as detection thresholds and spatial weights. The determination of these parameters is a subjective process, dependent on the experience of domain experts. If the algorithms can determine the parameters themselves, it will greatly reduce the burden of users. One possible approach is to design spatial outlier detection algorithms based on machine learning. One possible solution is to combine neural network with a set of labeled data to train the algorithms and obtain optimal parameters. For example, we can try different spatial weight values and select an appropriate one based on the feedback from the outlier examples.

Bibliography

[1] Nabil R. Adam, Vandana Pursnani Janeja, and Vijayalakshmi Atluri. Neighborhood based detection of anomalies in high dimensional spatio-temporal sensor datasets. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 576–583, Nicosia, Cyprus, March 14-17, 2004.

[2] Charu C. Aggarwal. Redesigning distance functions and distance-based applications for high dimensional data. *SIGMOD Record*, 30(1):13–18, March 2001.

[3] Charu C. Aggarwal. A framework for diagnosing changes in evolving data streams. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 575–586, San Diego, California, USA, June 9-12, 2003.

[4] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia Procopiuc, and Jong Soo Park. Fast algorithms for projected clustering. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 61–72, Philadelphia, Pennsylvania, United States, June 1-3, 1999.

[5] Charu C. Aggarwal and Philip S. Yu. Outlier detection for high dimensional data. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, pages 37–46, Santa Barbara, California, United States, May 21-24, 2001.

[6] M. Alexiuk, N. Pizzi, P C. Li, and W. Pedrycz. Classification of volumetric storm cell patterns. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 4(3):206–211, 2000.

[7] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley, New York, 1994.

[8] Stefan Berchtold, Christian Böhm, and Hans-Peter Kriegal. The pyramid-technique: Towards breaking the curse of dimensionality. In *Proceedings of the 1998 ACM SIGMOD International*

Conference on Management of Data, pages 142–153, Seattle, Washington, United States, June 1998.

[9] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104, Dallas, Texas, United States, May 14-19, 2000.

[10] Andrea Cerioli and Marco Riani. The ordering of spatial data and the detection of multiple outliers. *Journal of Computational and Graphical Statistics*, 8(2):239–258, June 1999.

[11] Philip K. Chan, Wei Fan, Andreas L. Prodromidis, and Salvatore J. Stolfo. Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems*, 14(6):67–74, 1999.

[12] Tony Chan and Luminita Vese. An active contour model without edges. In *Proceedings of the Second International Conference on Scale-Space Theories in Computer Vision*, pages 141–151, Corfu, Greece, September 26-27, 1999.

[13] W. S. Chan and W. N. Liu. Diagnosing shocks in stock markets of southeast asia, australia, and new zealand. *Mathematics and Computers in Simulation*, 59(1-3):223–232, 2002.

[14] Li Chen, Heng-Da Cheng, and Jianping Zhang. Fuzzy subfiber and its application to seismic lithology classification. *Information Sciences: Applications*, 1(2):77–95, 1994.

[15] Wen-Tsuen Chen, Chia-Hsien Wen, and Chin-Wen Yang. A fast two-dimensional entropic thresholding algorithm. *Pattern Recognition*, 27(7):885–893, 1994.

[16] Tai Wai Cheng, Dmitry B. Goldgof, and Lawrence O. Hall. Fast fuzzy clustering. *Fuzzy Sets and Systems*, 93(1):49–65, 1998.

[17] Tao Cheng and Zhilin Li. A Hybrid Approach to detect spatial-temporal outliers. In *Proceedings of the 12th International Conference on Geoinformatics - Geospatial Information Research: Bridging the Pacific and Atlantic, University of Gavle, Sweden*, June 7-9, 2004.

[18] M. Cheriet, J. N. Said, and C. Y. Suen. A recursive thresholding technique for image segmentation. *IEEE Transactions on Image Processing*, 7(6):918–921, June 1998.

[19] Aura Conci and Claudia Belmiro Proença. A system for real-time fabric inspection and industrial decision. In *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*, pages 707–714, Ischia, Italy, July 15-19, 2002.

[20] Thierry Denoeux and P. Rizand. Analysis of rainfall forecasting using neural networks. *Neural Computing and Applications*, 3(1):50–61, 1995.

[21] Nassima Djafri, Alvaro Fernandes, Norman W. Paton, and Tony Griffiths. Spatio-temporal evolution: querying patterns of change in databases. In *Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems*, pages 35–41, McLean, Virginia, USA, November 8-9, 2002.

[22] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 71–80, Boston, Massachusetts, USA, August 20-23, 2000.

[23] Guozhu Dong and Jinyan Li. Efficient mining of emerging patterns: discovering trends and differences. In *Proceedings of the Fifth ACM International Conference on Knowledge Discovery and Data Mining*, pages 43–52, San Diego, California, USA, Aug 15-18, 1999.

[24] Nicolae Duta and Milan Sonka. Segmentation and interpretation of mr brain images using an improved knowledge-based active shape model. In *IPMI '97: Proceedings of the 15th International Conference on Information Processing in Medical Imaging*, pages 375–380, Poultney, Vermont, USA, June 9-13, 1997. Springer-Verlag.

[25] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, Oregon, United States, August 2-4, 1996.

[26] M. Farge. Wavelet transforms and their applications to turbulence. *Annual Review of Fluid Mechanics*, 24:395–457, 1992.

[27] E. Foufoula-Georgiou and Eds. P. Kumar. *Wavelets in Geophysics*. Acadamic Press, 1995.

[28] Venkatesh Ganti, Johannes Gehrke, and Ramakrishnan Ramakrishnan. A framework for measuring changes in data characteristics. In *Proceedings of the Eighteenth ACM SIGMOD-*

SIGACT-SIGART Symposium on Principles of Database Systems (PODS '99), pages 126–137, May 31 - June 2,.

[29] C. Giannella, J. Han, J. Pei, X. Yan, and P.S. Yu. Frequent patterns in data streams at multiple time granularities. *NSF Workshop on Next Generation Data Mining*, AAAI/MIT, 2003.

[30] Jian Gong, Liyuan Li, and Weinan Chen. Fast recursive algorithms for two-dimensional thresholding. *Pattern Recognition*, 31(3):295–300, 1998.

[31] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (Second Edition)*. Prentice Hall, New Jersey, 2002.

[32] Tony Griffiths, Alvaro Fernandes, Norman W. Paton, Keith T. Mason, Bo Huang, and Michael F. Worboys. Tripod: A comprehensive model for spatial and aspatial historical objects. In *ER '01: Proceedings of the 20th International Conference on Conceptual Modeling*, pages 84–102, Yakohama, Japan, Nov 27-30, 2001.

[33] Sudipto Guha, Adam Meyerson, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams: theory and practice. *IEEE Transactions on Knowledge and Data Engineering*, 15(3):515–528, 2003.

[34] I. Guttman. *Linear Models: An Introduction*. John Wiley, New York, 1982.

[35] R.P. Haining. *Spatial Data Analysis in the Social and Environmental Sciences*. Cambridge University Press, 1993.

[36] Jiawei Han and Micheline Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, CA, USA, 2001.

[37] J. Haslett, R. Brandley, P. Craig, A. Unwin, and G. Wills. Dynamic Graphics for Exploring Spatial Data With Application to Locating Global and Local Anomalies. *The American Statistician*, 45:234–242, 1991.

[38] D. Hawkins. *Identification of outliers*. Chapman and Hall, Reading, Massachusetts, 1980.

[39] Alexander Hinneburg, Charu C. Aggarwal, and Daniel A. Keim. What is the nearest neighbor in high dimensional spaces? In *Proceedings of 26th International Conference on Very Large Data Bases*, pages 506–515, Cairo, Egypt, September 10-14, 2000.

[40] L. Hudgins, C. Friehe, and M. Mayer. Wavelet transforms and atmospheric turbulence. *Physical Review Letters*, 71:3279–3282, 1993.

[41] G. Hulten, L. Spencer, and P. Domingos. Mining time-changing data streams. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 97–106, San Francisco, CA, August 26-29, 2001.

[42] Wen Jin, Anthony K. H. Tung, and Jiawei Han. Mining top-n local outliers in large databases. In *KDD '01: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 293–298, San Francisco, California, August 26-29, 2001.

[43] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, 2002.

[44] George Karypis, Eui-Hong Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer*, 32(8):68–75, 1999.

[45] Edwin M. Knorr and Raymond T. Ng. Algorithms for mining distance-based outliers in large datasets. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 392–403, New York City, NY, USA, August 24-27, 1998.

[46] K. Koperski, J. Adhikary, and J. Han. Spatial data mining: Progress and challenges. In *Workshop on Research Issues on Data Mining and Knowledge Discovery(DMKD'96)*, pages 1–10, Montreal, Canada, 1996.

[47] Yufeng Kou, Chang-Tien Lu, and Dechang Chen. Spatial weighted outlier detection. In *SIAM International Conference on Data Mining*, pages 614–618, Bethesda, Maryland, April 20-22, 2006.

[48] Yufeng Kou, Chang-Tien Lu, Adil Godrej, Thomas Grizzard, and Harry Post. A web-based data visualization and analysis system for watershed management. *Journal of Geographic Information Science Special Issue on Distributed GIS*, 11(1):40–49, June 2005.

[49] Yufeng Kou, Chang-Tien Lu, S. Sirwongwattana, and Y.P. Huang. Survey of fraud detection techniques. In *Proceedings of the 2004 International Conference on Networking, Sensing, and Control*, pages 749–754, Taipei, Taiwan, March 21-23, 2004.

[50] K. M. Lau and H. Y. Weng. Climate signal detection using wavelet transform: how to make a time series sing? *Bulletin of the American Meteorological Society*, 76(12):2391–2402, 1995.

[51] S. Lee, S.Y. Chung, and R. H. Park. A comparative performance study of several global thresholding techniques for segmentation. *Computer Vision, Graphics and Image Processing*, 52(2):171–190, 1990.

[52] P. Li, N. Pizzi, W. Pedrycz, D. Westmore, and R. Vivanco. Severe storm cell classification using derived products optimized by genetic algorithm. *Proceedings of the 2000 IEEE Canadian Conference on Electrical and Computer Engineering*, pages 445–448, March 2000.

[53] T. Li, Q. Li, S. Zhu, and M. Ogihara. A survey on wavelet applications in data mining. *SIGKDD Explorations*, 4(2):49–67, 2002.

[54] Yifan Li, Jiawei Han, and Jiong Yang. Clustering moving objects. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 617–622, Seattle, Washington, USA, Aug 22-25, 2004.

[55] Ping-Sung Liao, Tse-Sheng Chen, and Pau-Choo Chung. A fast algorithm for multilevel thresholding. *Journal of Information Science and Engineering.*, 17(5):713–727, 2001.

[56] Hongxing Liu, Kenneth C. Jezek, and Morton E. O'Kelly. Detecting outliers in irregularly distributed spatial data sets by locally adaptive and robust statistical analysis and gis. *International Journal of Geographical Information Science*, 15(8):721–741, 2001.

[57] Chang-Tien Lu, Dechang Chen, and Yufeng Kou. Detecting spatial outliers with multiple attributes. In *Proceedings of the 15th International Conference on Tools with Artificial Intelligence*, pages 122–128, Sacramento, California, USA, November 3-5, 2003.

[58] Chang-Tien Lu, Dechang Chen, and Yufeng Kou. Multivariate spatial outlier detection. *International Journal on Artificial Intelligence Tools, World Scientific*, 13(4):801–812, 2004.

[59] Chang-Tien Lu, Dechang Chen, and Yufeng Kou. Algorithms for spatial outlier detection. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 597–600, Melbourne, Florida, USA, November 19-22, 2003.

[60] Chang-Tien Lu, Yufeng Kou, H Wang, P Zhang, and Rulin Liu. Two web-based spatial data visualization and mining systems: Mapcube and mapview. In *International Workshop on Next Generation Geospatial Information*, pages 749–754, Cambridge (Boston), Massachusetts, USA, Oct. 19-21, 2003.

[61] Chang-Tien Lu, Yufeng Kou, Jiang Zhao, and Li Chen. Detecting and tracking regional outliers in meteorological data sequences. *Informatics and Computer Science: An International Journal, to be appeared in 2007.*

[62] Chang-Tien Lu and Lily R. Liang. Wavelet fuzzy classification for detecting and tracking region outliers in meteorological data. In *Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems*, pages 258–265, Washington DC, United States, November 12-13, 2004.

[63] Chang-Tien Lu, Hongjun Wang, and Yufeng Kou. *MapView: A web-based outlier detection tool.* http://europa.nvc.cs.vt.edu/˜ctlu/Project/ MapView/index.htm.

[64] C.T. Lu, R. Santos, L. Sripada, and Y. Kou. Advances in gml for geospatial applications. *GEOINFORMATICA - An International Journal on Advances of Computer Science for Geographic Information System (Springer)*, to be appeared in 2007.

[65] Anselin Luc. Exploratory Spatial Data Analysis and Geographic Information Systems. In *New Tools for Spatial Analysis*, pages 45–54, 1994.

[66] Anselin Luc. Local Indicators of Spatial Association: LISA. *Geographical Analysis*, 27(2):93–115, 1995.

[67] Prasanta Mahalanobis. On the generalized distance in statistics. *Proceedings of Natural Institute of Sciences*, 2:49–55, 1936.

[68] Mankin Mak. Orthogonal wavelet analysis: Interannual variability in the sea surface temperature. *Bulletin of the American Meteorological Society*, 76(11):2179–2186, 1995.

[69] S. G. Mallat and W. L. Hwang. Singularity detection and processing with wavelets. *IEEE Transactions on Information Theory*, 38(2):617–643, 1992.

[70] Nancy E. Miller, Pak Chung Wong, Mary Brewster, and Harlan Foote. A wavelet-based text visualization system. In *Proceedings of the Conference on Visualization '98*, pages 189–196, Research Triangle Park, North Carolina, USA, October 18-23, 1998.

[71] Abdallah Mkhadri. Shrinkage parameter for the modified linear discriminant analysis. *Pattern Recognition Letters*, 16(3):267–275, 1995.

[72] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Communications of Pure and Applied Mathematics*, 42:577–685, 1989.

[73] Raymond T. Ng and Jiawei Han. Efficient and effective clustering methods for spatial data mining. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 144–155, Santiago de Chile, Chile, September 12-15, 1994.

[74] N R Pal and S K Pal. Entropy: A new definition and its applications. *IEEE Transactions on Systems, Man and Cybernetics*, 21(5):1260–1270, 1991.

[75] Themistoklis Palpanas, Dimitris Papadopoulos, Vana Kalogeraki, and Dimitrios Gunopulos. Distributed deviation detection in sensor networks. *ACM SIGMOD Record: Special Section on Sensor Network Technology and Sensor Data Managment*, 32(4):77–82, 2003.

[76] Y. Panatier. *VARIOWIN: Software for Spatial Data Analysis in 2D.* Springer-Verlag, New York, 1996.

[77] J. F. Peters, Z. Suraj, S. Shan, S. Ramanna, W. Pedrycz, and N. Pizzi. Classification of meteorological volumetric radar data using rough set methods. *Pattern Recognition Letter*, 24(6):911–920, 2003.

[78] Marcel Prastawa, Elizabeth Bullitt, Sean Ho, and Guido Gerig. A brain tumor segmentation framework based on outlier detection. *Medical Image Analysis*, 9(5):457–466, 2004.

[79] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry - An Introduction.* Springer, 1985.

[80] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient Algorithms for Mining Outliers from Large Data Sets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 427–438, Dallas, Texas, USA, May 14-19, 2000.

[81] L Ramirez, W Pedrycz, and N Pizzi. Severe storm cell classification using support vector machines and radial basis function approaches. In *Proceedings of Canadian Conference on Electrical and Computer Engineering*, pages 87–92, Toronto, Canada, May 13-16, 2001.

[82] Philippe Rigaux, Agnes Voisard, and Michel O. Scholl. *Spatial Databases: With Application to Gis*. Morgan Kaufmann Publishers, San Francisco, CA, United States, 2 edition, May 2001.

[83] A Rosenfeld. The fuzzy geometry of image subsets. *Pattern Recognition Letters*, 2(5):311–318, 1983.

[84] T. A. Runkler, J. C. Bezdek, and L. O. Hall. Clustering very large data sets: the complexity of the fuzzy c-means algorithm. In *Proceedings of EUNITE 2002*, pages 420–425, Aachen, Gemany, September 19-21, 2002.

[85] I. Ruts and P. Rousseeuw. Computing Depth Contours of Bivariate Point Clouds. In *Computational Statistics and Data Analysis, 23:153–168*, 1996.

[86] Cyrus Shahabi, Seokkyung Chung, Maytham Safar, and George Hajj. 2d TSA-tree: A wavelet-based approach to improve the efficiency of multi-level spatial data mining. In *Proceedings of the 13th International Conference on Scientific and Statistical Database Management*, pages 59–68, Fairfax, Virginia, USA, July 18-20, 2001.

[87] Cyrus Shahabi, Xiaoming Tian, and Wugang Zhao. TSA-tree: A wavelet-based approach to improve the efficiency of multi-level surprise and trend queries on time-series data. In *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*, pages 55–68, Berlin, Germany, July 26-28, 2000.

[88] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In *VLDB '98: Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 428–439, New York, NY, August 24-27, 1998.

[89] S. Shekhar and S. Chawla. *A Tour of Spatial Databases*. Prentice Hall, 2002.

[90] S. Shekhar, S. Chawla, S. Ravada, A. Fetterer, X. Liu, and C.T. Lu. Spatial Databases: Accomplishments and Research Needs. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):45–55, 1999.

[91] S. Shekhar, Chang-Tien Lu, and P. Zhang. A Unified Approach to Spatial Outliers Detection. *GeoInformatica, An International Journal on Advances of Computer Science for Geographic Information System*, 7(2):139–166, June 2003.

[92] Shashi Shekhar, Chang-Tien Lu, and Pusheng Zhang. Detecting graph-based spatial outliers: algorithms and applications (a summary of results). In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 371–376, San Francisco, California, August 26-29, 2001.

[93] Shashi Shekhar, Chang-Tien Lu, Pusheng Zhang, and Rulin Liu. Data mining for selective visualization of large spatial datasets. In *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, pages 41–48, Washington, DC, USA, November 4-6, 2002.

[94] Zbigniew R. Struzik and Arno Siebes. The haar wavelet transform in the time series similarity paradigm. In *PKDD '99: Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery*, pages 12–22, September 15-18, 1999.

[95] Z. Suraj, J. F. Peters, and W. Rzasa. A comparison of different decision algorithms used in volumetric storm cells classification. *Fundamenta Informaticae*, 51(1):201–214, 2002.

[96] Michael E. Tipping and Christopher M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[97] W Tobler. Cellular geography. In *Philosophy in Geography*, Dordrecht, Holland, 1979. Dordrecht Reidel Publishing Company.

[98] C Torrence and G Compo. A practical guide to wavelet analysis. *Bulletin of the American Meteorological Society*, 79(1):61–78, January 1998.

[99] U.S. Census Bureau, United Stated Department of Commence. http://www.census.gov/.

[100] Olli Virmajoki and Pasi Franti. Fast pairwise nearest neighbor based algorithm for multilevel thresholding. *Journal of Electronic Imaging*, 12(4):648–659, 2003.

[101] Y. Wang. Jump and sharp cusp detection by wavelets. *Biometrika*, 82(2):385–397, 1995.

[102] B. J. Whitcher, P. Guttorp, and D. B. Percival. Multiscale detection and location of multiple variance changes in the presence of long memory. *Journal of Statistical Computation and Simulation*, 68(1):65–88, 2000.

[103] Weng-Keen Wong, Andrew Moore, Gregory Cooper, and Michael Wagner. Rule-based anomaly pattern detection for detecting disease outbreaks. In *the Eighteenth National Conference on Artificial Intelligence*, pages 217–223, Edmonton, Alberta, Canada, July 28 - August 1, 2002.

[104] Weng-Keen Wong, Andrew Moore, Gregory Cooper, and Michael Wagner. Bayesian network anomaly pattern detection for disease outbreaks. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 808–815, Menlo Park, California, August 21-24, 2003.

[105] Lei Xu. Bayesian ying-yang machine, clustering and number of clusters. *Pattern Recognition Letters*, 18(11-13):1167–1178, 1997.

[106] Kenji Yamanishi, Jun-Ichi Takeuchi, Graham Williams, and Peter Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. *Data Mining and Knowledge Discovery*, 8(3):275–300, 2004.

[107] Kenji Yamanishi and Junichi Takeuchi. A unifying framework for detecting outliers and change points from non-stationary time series data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 676–681, Edmonton, Alberta, Canada, July 23-26, 2002.

[108] Jeffrey Xu Yu, Weining Qian, Hongjun Lu, and Aoying Zhou. Finding centric local outliers in categorical/numerical spaces. *Knowledge and Information Systems*, 9(3):309–338, 2006.

[109] Stefano Zanero and Sergio M. Savaresi. Unsupervised learning techniques for an intrusion detection system. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 412–419, Nicosia, Cyprus, March 14-17, 2004.

[110] Mingrui Zhang, Lawrence O. Hall, and Dmitry B. Goldgof. A generic knowledge-guided image segmentation and labeling system using fuzzy clustering algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 32(5):571–582, 2002.

[111] Mingrui Zhang, Lawrence O. Hall, Dmitry B. Goldgof, and Frank E. Müller-Karger. Knowledge-guided classification of coastal zone color images off the west florida shelf. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 14(8):987–1007, 2000.

[112] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 103–114, Montreal, Quebec, Canada, June 4-6 1996.

[113] Jiang Zhao, Chang-Tien Lu, and Yufeng Kou. Detecting region outliers in meteorological data. In *Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems*, pages 49–55, New Orleans, Louisiana, United States, November 7-8, 2003.

[114] Jiang Zhao, Chang-Tien Lu, and Yufeng Kou. Change detection in meteorological data. In *Proceedings of the 7th Joint Conference on Information Sciences*, pages 509–512, Research Triangle Park, North Carolina, USA, September 26-30, 2004.