8-2019

Advances in Big Data Analytics: Algorithmic Stability and Data Cleansing

Yuping Lu
University of Tennessee, ylu20@vols.utk.edu

Advances in Big Data Analytics: Algorithmic Stability and Data Cleansing

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Yuping Lu
August 2019

Acknowledgements

First and foremost, I would like to thank my advisor, Dr. Michael A. Langston, for his patience and guidance. Throughout my studies he encouraged me to develop independent thinking and research skills. Second, I would like to thank my research supervisor, Dr. Jitendra Kumar, for providing me the opportunity to do research at the ARM Data Science and Integration group of Oak Ridge National Laboratory and for always giving me the encouragement and support. I would also like to express my gratitude to my other exceptional doctoral committee members: Dr. Qing (Charles) Cao and Dr. Audris Mockus. I have been fortunate to work with my mentor Mr. John B. Rose for three years at the Office of Information Technology of University of Tennessee, where I learned web server configuration and optimization. I also extend my appreciation to Dr. George Ostrouchov for guiding me during my two summer internships at the Scientific Data Group of Oak Ridge National Laboratory. Former and present students I have worked with from Dr. Langston's research team whose friendship I will always value include Kai Wang, Charles Phillips, Ronald Hagan, Clarence Jackson, Carissa Bleker, Stephen Grady, Austin Wyer and Brett Hagan. In addition, I would like to thank Drs. Huanliang Xu, Daolong Dou and Jingdong Liang, as well as many other professors at Nanjing Agricultural University, where I completed my bachelor's degree. This dissertation could not have been written without the encouragement and help from my friends: Liang Wang, Lingyun Ren, Shaozhi Li, Yong Li, Xiaobing Li, Lipeng Wang, Cheng Chen, Katrina Schlum, Ben Ernest, Laxman Nathawat and many others. And last but not least, my gratitude and appreciation go to my family: my cousins Xueping Lu, Yunping Lu and Junping Lu, and especially my parents, Shuxin Lu and Xiaohong Pan, who were always there for me, guiding me to the right direction and supporting my decisions.

Abstract

Analysis of what has come to be called “big data” presents a number of challenges as data continues to grow in size, complexity and heterogeneity. To help addresses these challenges, we study a pair of foundational issues in algorithmic stability (robustness and tuning), with application to clustering in high-throughput computational biology, and an issue in data cleansing (outlier detection), with application to pre-processing in streaming meteorological measurement. These issues highlight major ongoing research aspects of modern big data analytics. First, a new metric, robustness, is proposed in the setting of biological data clustering to measure an algorithm’s tendency to maintain output coherence over a range of parameter settings. It is well known that different algorithms tend to produce different clusters, and that the choice of algorithm is often driven by factors such as data size and type, similarity measure(s) employed, and the sort of clusters desired. Even within the context of a single algorithm, clusters often vary drastically depending on parameter settings. Empirical comparisons performed over a variety of algorithms and settings show highly differential performance on transcriptomic data and demonstrate that many popular methods actually perform poorly. Second, tuning strategies are studied for maximizing biological fidelity when using the well-known paraclique algorithm. Three initialization strategies are compared, using ontological enrichment as a proxy for cluster quality. Although extant paraclique codes begin by simply employing the first maximum clique found, results indicate that by generating all maximum cliques and then choosing one of highest average edge weight, one can produce a small but statistically significant expected improvement in overall cluster quality. Third, a novel outlier detection method is described that helps cleanse data by combining Pearson correlation coefficients, K-means clustering, and Singular Spectrum Analysis in a coherent framework that detects instrument failures and extreme weather events in Atmospheric Radiation Measurement sensor data. The

framework is tested and found to produce more accurate results than do traditional approaches that rely on a hand-annotated database.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

What has come to be known as “big data” often includes large-scale information collected from many different sources. A key characteristic of big data is its volume and complexity, which exceed the storage and analysis capability of common database software and other management tools [1].

Review of Big Data

A Brief History of Big Data

The concept of big data was mentioned as early as 1997 by Michael Cox and David Ellsworth when they worked on the visualization of computational fluid dynamics [2]. In 2000, Francis X. Diebold attempted a formal definition of big data: "explosion in the quantity (and sometimes, quality) of available and potentially relevant data, largely the result of recent and unprecedented advancements in data recording and storage technology [3].” One year later, the famous three Vs for describing big data, volume, velocity and variety, were introduced by Doug Laney [4]. Volume refers to the massive size of data, which is often bigger than petabytes [5]. Issues like computational cost and algorithmic instability are commonly seen due to such large size. Velocity is a measure of the speed of data generation, which includes data generated from batch, near real time, real time and streams [6]. One of velocity’s main challenges is noise accumulation. Noise can stem from a variety of sources, including measurement errors, missing values and outliers. Variety refers to the source of data and usually is divided into structured data, semi-structured data and unstructured data [7]. The diversity of big data brings with it problems such as statistical bias, experimental variations and heterogeneity. Thus, robust algorithms are crucial to handle these issues [8]. A fourth V, value, is another important component frequently mentioned in big data analytics [9]. Value refers to insights gleaned from big data using tools such as graph algorithms, machine learning and other statistical methods [10]. Researchers have even proposed a fifth V, veracity, or certainty of data, to measure the credibility of big

data [11]. The copious amount and often high-dimensionality of data today presents both opportunities and challenges to modern big data analytics [6]. Thus, efficient algorithms and novel management tools are becoming a dominant focus for big data analytics. In a 2011 report, McKinsey Global Institute concludes that the two main factors of big data are as follows: 1) techniques for analyzing data, for example classification, cluster analysis, data mining and network analysis; and 2) big data technologies, such as Cassandra, cloud computing, distributed system and stream processing [9].

Examples of Big Data

Big data may come from a wide variety of fields. Examples include meteorology, genomics, neuroscience, social networks, public health, sensors, retail, financial services, transportation, web search, telecommunications and many other domains. In genomics alone, there are more than 500,000 microarray datasets publicly available due to the cheap price of genome sequencing [12]. Such a wealth of data has driven a trend where many researchers, instead of generating new data, are now concentrating on biological discovery in existing datasets [6]. Another application of big data is in the use of sensor networks. The Next Generation Weather Radar (NEXRAD), for example, collects data every five minutes over the entire U.S. (along with a few overseas locations) and makes it available to the public on Amazon S3 in real-time along with historical data dating back to June, 1991 [13]. In the field of public health, data from the U.S. healthcare system exceeded 150 exabytes in 2011 [11]. In social networks, 30 billion posts are shared on Facebook monthly. More than 100 million photos and videos are uploaded to Instagram daily. And 500 million tweets are posted on Twitter on a daily basis [14]. According to McKinsey Global Institute, projected growth in global data generated per year is 40% [9].

Big Data Analytics

Traditional techniques and technologies that perform well on conventional data cannot always be applied to big data. Thus, novel frameworks, tools and algorithms are needed to reach statistical accuracy and computational efficiency in modern big data analytics [6]. MapReduce is a good example. It is mainly a programming framework proposed by Google to process big data on computer clusters in parallel [15]. MapReduce consists of two steps: a map step that divides a task into many sub-tasks by a master node and assigns them to different worker nodes, and a reduce step that collects results from each worker node and analyze them together. Inspired by MapReduce and Google File System (GFS), Apache implemented Hadoop [16], its distributed file system. Hadoop is an open source cross-platform framework that contains an Hadoop Distributed File System (HDFS) to store big data with reliability and an Hadoop processing unit to form a MapReduce programming framework [17]. In 2010, Apache created Spark, a big data analytics engine, to outperform Hadoop in MapReduce [18]. In addition to these frameworks, big data issues are sometimes addressed using High Performance Computing (HPC) clusters. Unlike the frameworks mentioned previously, HPC clusters run faster, but require users to define their own data model using, for example, the Message-Passing Interface (MPI) because they have no upper level abstraction [1]. Representative clusters include Berkeley lab's the National Energy Research Scientific Computing Center (NERSC), and Oak Ridge National Laboratory (ORNL)'s the Compute and Data Environment for Science (CADES) and Summit, the fastest supercomputer in the world as of this writing [19]. To manage big data across many servers and even different data centers, Facebook developed a distributed NoSQL database system Cassandra [20]. Other popular database systems for big data are MongoDB, HBase, Neo4j and Hive. Cloud computing refers to the computing services provided by data centers without user maintenance [21]. According to vendors like Google App Engine, Microsoft Azure and Amazon AWS, these services can be classified into Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS)

depending on the type of products they provided. More and more big data is now generated, stored, and analyzed in the cloud.

Most techniques or algorithms on big data can be classified into one of several broad categories. Examples include cluster analysis, graph analytics, machine learning, data mining, natural language processing, neural networks, pattern recognition and spatial analysis [22]. These categories often overlap with no clear boundaries [23]. For instance, graph analytics includes graph partitioning, matchings, and graph clustering, each of which is also used for pattern recognition and data mining [24, 25]. In addition, each category itself has a wide range of applications in many different fields. For example, graph clustering algorithms have been applied to genomics, social networks and transportation. Effective scalable graph algorithms are especially important for big data.

Applications

The rise of big data promises many applications. In 2012, the Obama administration announced the "Big Data" initiative of $200 million to invest in research and development [26]. With the help of big data analytics, McKinsey estimates that more than $300 billion could be saved per year in U.S. healthcare [9]. Researchers have applied machine learning to big data to understand competitors and develop winning tactics in soccer [27]. Walmart detects patterns in their massive set of transaction data to help set prices and target advertisements [23].

Experimental Data

Perhaps the best algorithmic testbed available today is comprised of biological data, which includes data derived from experiments with DNA, RNA, proteins, metabolites and other sources. Due to its ease-of-access and diversity, we concentrate mainly on transcriptomic data, which simultaneously measures the

abundance of thousands of different mRNAs. DNA microarrays and next-generation sequencing (RNA-Seq) are two techniques for measuring transcript expression levels [28]. Here we focus only on publicly-available transcriptomic data downloaded from the Gene Expression Omnibus (GEO) [29], and use data from five species: baker's yeast (*S. cerevisiae*), fruit fly (*D. melanogaster*), bacteria (*E. coli*), mouse (*M. musculus*) and fungi (*P. chrysogenum*).

For another rich yet considerably different algorithmic testbed we turn to meteorological data, specifically Atmospheric Radiation Measurement (ARM) sensor data. This data includes observational measurements of Earth's climate from many ARM instruments distributed around the globe [30], and can be downloaded from the ADC website (https://www.arm.gov/data). ARM data comes in many forms, and includes readings from observation cameras, weather radars such as C-Band Scanning ARM Precipitation Radar (CSAPR), and satellite observations. In this work we will focus only on meteorological observation data from ARM's biggest facility located in Oklahoma, using these five core variables: air temperature, vapor pressure, atmospheric pressure, relative humidity and wind speed.

Graph Theoretical Basics and Related Algorithms

A graph $G = (V, E)$ is formed by a set of vertices $V(G)$ and a set of edges $E(G)$. Graphs mentioned in the dissertation are simple, finite, undirected and unweighted, unless otherwise stated. Two vertices u, v are said to be adjacent if $uv \in E(G)$. A graph $\hat{G} = (\hat{V}, \hat{E})$ is a subgraph of $G = (V, E)$, if $\hat{V} \subseteq V$ and $\hat{E} \subseteq E$. The neighborhood of a vertex u is a subgraph of G induced by a set of vertices adjacent to vertex u, and denoted $N(u)$. The cardinality of $N(u)$ is the degree of u.

A clique, or complete subgraph, is a subgraph in which each vertex is connected to every other vertex in that subgraph. A maximal clique is a clique to

which no vertex can be added to form a larger clique. A maximum clique is a largest maximal clique. The clique number, $\omega(G)$, is used to denote the number of vertices in a maximum clique. The classical clique decision problem, where one is given a graph G and an integer k and asked whether G contains a clique of size k, is *NP*-complete [31]. For the maximal clique enumeration problem, the Bron-Kerbosch algorithm and Tomita et al. are two popular choices [32, 33]. Eblen et al. presented an efficient way to enumerate all maximum cliques [34], which has made testing of the selection strategies in Chapter 3 computationally feasible.

A paraclique is a near-clique, that is, one that is missing a handful of edges [35]. It is designed to ameliorate the effects of noise, and is constructed by first finding a maximum clique, C, and then adding vertices adjacent to most but not all of C in a tightly controlled fashion.

Similarity Metrics

A graph can be formed by treating entities, for example genes or proteins, as vertices. We often wish to know how similar each entity is to others. Depending on the application, such similarity can represent physical characteristics, location, or how the entities respond to different conditions. Similarity metrics for this purpose yield a single score for each pair. That score is then used to weigh the graph's edges. Multiple methods are available to measure similarity. The selection of an appropriate similarity metric is highly dependent on the type and nature of the data and the goals of the analysis. When the data consists of measurements across multiple conditions, Pearson correlation is among the most commonly used similarity metrics [36]. It measures the linear relationship between entities. Spearman correlation is the Pearson correlation between the rankings of two entities and is resistant to outliers [37]. If one seeks non-linear relationships, mutual information is a good candidate [38]. Jaccard similarity is often used for similarity measurements of two sets [39]. Cosine similarity measures the similarity

between two non-zero vectors in vector space and is often applied to document comparison [40]. Euclidean distance is the straight-line distance between two entities in Cartesian space. Different from Euclidean distance, Manhattan distance is the sum of absolute differences of two entities' Cartesian coordinates. For mixed data, Goodall [41] and Gower [42] are good choices.

Thresholding

One technique for creating a graph is to let vertices in the graph represent entities and to weight the edges with the similarity between each pair of entities. This, of course, requires computation of all pairwise similarities. The result is a weighted graph which can then be transformed to an unweighted graph by picking a threshold and retaining only those edges at or above the threshold. Threshold selection is a topic of ongoing research. Many researchers pick a threshold, for example 0.875 Pearson correlation, based on their previous experience [43]. More rigorous methods for optimal threshold selection have been proposed, including the use of spectral graph theory [44]. In Chapter 2, we apply this technique to obtain rigorous thresholds for robustness comparisons.

Evaluation of Cluster Quality

Clustering is an important method for big data analytics. Chapters 2 and 3 in this dissertation each focus on a different aspect of clustering. In Chapter 3, we need a measure of cluster quality to test whether one clustering is “better” than another. Such a measure, however, can be difficult to quantify because often the ground truth is unknown. Cluster quality can be measured either by some theoretical standards or using a known classification scheme [39, 45, 46]. In the former case, commonly used statistical metrics include modularity [47], clustering coefficient [48, 49], silhouette coefficient [50], and adjusted mutual information [51]. In the latter case, domain-specific knowledge such as ontological enrichment [52, 53] is often applied to compare clusters extracted from transcriptomic data. In Chapter

3, we employ the latter method, using Gene Ontology (GO) [54, 55] categories to measure the quality of generated paracliques by comparing their enrichment p-values.

Contributions of this Dissertation

First, we concentrate on the ubiquitous clustering problem and introduce a robustness metric to measure the stability of a clustering algorithm when set to different parameters. Using transcriptomic data and a variety of commonly used clustering algorithms, we demonstrate how the robustness of the algorithms can be measured and compared. According to our tests, hierarchical methods and the paraclique algorithm have higher robustness scores than a host of other commonly-used clustering algorithms.

Second, we maintain our focus on clustering and evaluate tuning strategies for procedures such as the paraclique algorithm. Maximum clique methods typically return only the first one found, even though there may be many others [34]. We perform empirical testing on three different maximum clique selection strategies and find that selecting a maximum clique with highest average edge weight tends to produce superior results on transcriptomic data.

Third, we turn our attention to outlier detection, another foundational problem associated with big data, and concentrate on the analysis of time series data. We describe a novel automated framework for meteorological data collected via distributed sensors. We test the framework on ARM sensor data collected over an area of Oklahoma and stored in the database, where entries about outliers were inserted manually. Experimental results show that some 88.9% of outliers detected by the framework are not found in the database.

Chapter 2
A Robustness Metric for Biological Data Clustering Algorithms

A version of this chapter written by Yuping Lu, Charles A. Phillips and Michael A. Langston has been submitted for publication and is currently under review.

My contribution was to collect the data from GEO, run the clustering algorithms, and calculate each algorithms' robustness.

Abstract

Cluster analysis is a core task in modern data-centric computation. Algorithmic choice is driven by factors such as data size and heterogeneity, the similarity measures employed, and the type of clusters sought. Familiarity and mere preference often play a significant role as well. Comparisons between clustering algorithms tend to focus on cluster quality. Such comparisons are complicated by the fact that algorithms often have multiple settings that can affect the clusters produced. Such a setting may represent, for example, a preset variable, a parameter of interest, or various sorts of initial assignments. A question of interest then is this: to what degree do the clusters produced vary as setting values change? This work introduces a new metric, termed simply "robustness," designed to answer that question. Robustness is an easily-interpretable measure of the propensity of a clustering algorithm to maintain output coherence over a range of settings. The robustness of eleven popular clustering algorithms is evaluated over some two dozen publicly available mRNA expression microarray datasets. Given their straightforwardness and predictability, hierarchical methods generally exhibited the highest robustness on most datasets. Of the more complex strategies, the paraclique algorithm yielded consistently higher robustness than other algorithms tested, approaching and even surpassing hierarchical methods on several datasets. Other techniques exhibited mixed robustness, with no clear distinction between them. Robustness provides a simple and intuitive measure of the stability and predictability of a clustering algorithm. It can be a useful tool to

aid both in algorithm selection and in deciding how much effort to devote to parameter tuning.

Background

Clustering algorithms are generally used to classify a set of objects into subsets using some measure of similarity between each object pair. Comparisons between clustering algorithms typically focus on the quality of clusters produced, as measured against either a known classification scheme or against some theoretical standards [39, 45, 46]. In the former case, varying criteria for what constitutes a meritorious cluster are often applied, employing domain-specific knowledge such as ontological enrichment [52, 53], geographical alignment [56] or legacy delineation [57]. In the latter case, statistical quality metrics are most often used, with cluster density something of a gold standard. Examples include modularity [47], which measures the density of connections within clusters versus density of connections between clusters, clustering coefficient [48, 49], which gives the proportion of triplets for which transitivity holds, and silhouette coefficient [50], which is based on how similar a node is to its own cluster as compared to other clusters. Additional metrics include the adjusted rand index [58], homogeneity [59], completeness [60], V-measure [61], and adjusted mutual information [51]. No single algorithm is of course likely to perform best over every metric.

In this chapter, we consider algorithmic comparisons from another perspective. Rather than attempt to measure the quality or correctness of the clusters themselves, we focus instead on the sensitivity of an algorithm's clusters to changes in its various settings. The metric we introduce, which we term "robustness," provides a relatively simple measure of a clustering algorithm's stability over a range of these settings. We note that robustness should not be confused with other clustering appraisals such as correctness or resistance to

noise, which are studied elsewhere in the literature. And while it might seem tempting to try to combine multiple notions, such as accuracy and robustness, into some single metric, the resultant analysis is fraught with complexity and well beyond the scope of this work.

In order to demonstrate the utility of robustness, we chose transcriptomic data publicly available from the GEO [62]. This is a relevant and logical choice given current technology because of gene co-expression data's ready abundance, availability and standardized format, and because clustering of this sort of data is such an overwhelmingly common task in the research community's quest to discover and delineate putative molecular response networks.

Methods

Algorithms

Clustering algorithms typically have one or more adjustable settings. For instance, such a setting may denote a preset variable, a relevant parameter, or sets of initial assignments. Sometimes the only setting available is the number of clusters desired. To make the scope of this work manageable, and to keep comparisons as equitable as possible, we only consider algorithms that produce non-overlapping clusters, and that are unsupervised, in the sense that classes into which objects are clustered are not defined in advance. (We deviate from this very slightly in the case of NNN [63], which allows a pair of clusters to share a single element.) For each method considered we selected a range of settings commonly used in practice.

Different algorithms may produce (sometimes vastly) different clusters, as may different settings of the same algorithm. In a previous comparison of genome-scale clustering algorithms [39], we focused on cluster enrichment, using Jaccard similarity with known GO and KEGG annotation sets as a measure of cluster

quality. In that study, graph-theoretical methods outperformed conventional methods by a wide margin. A natural question then is whether something along the same line may hold for robustness.

Robustness

We seek to define a measure of robustness that can provide a single, easily-interpretable metric that captures the tendency of a clustering algorithm to keep pairs of objects together over a range of settings. Indeed, each algorithm may have its own optimum settings. We did not try to isolate such settings, but rather to measure an algorithm's sensitivity to parameter variations. Let us consider the results of a single clustering algorithm (ALG). If in any run ALG assigns a pair *P* of objects to at least one cluster, then we define *P*'s robustness to be the proportion of clustering runs in which *P* appears together in any cluster. Thus, for example, if genes *A* and *B* appear together (in any cluster) in 17 of 23 clustering runs, then the score for that pair is 17 / 23 = 0.7391. We extend this from *P* to ALG by defining ALG's robustness, *R*, as the average score of all such candidates for *P*. In this fashion, robustness is measured for one algorithm and for one dataset, but over multiple runs (setting values).

Formally, we therefore set $R = t / (dr)$, where t denotes the total number of (not necessarily distinct) pairs of objects that appear together in some cluster summed over all runs, d represents the number of distinct pairs of objects that appear together in some cluster produced by some run, and r is the number of times the clustering algorithm was run, each run using a different value for some setting of interest. In other words, robustness is the proportion of clustering runs in which a pair of entities appears together in some cluster, given that they appear together in a cluster in at least one run, averaged over all such pairs. R thus lies in the interval (0, 1] and, when all else is equal, we seek algorithms with R values as high as possible. Note that the effect of a pair appearing (or failing to appear) in a

cluster is typically minor as it only reduces by one the denominator in the above formula. In order to compare robustness values fairly, we were careful to select a range of values that produced clusters of the same scale. The number of clusters was not a consideration, except of course for algorithms such as K-means where the number of clusters is itself the parameter being varied.

We illustrate the notion of robustness with an elementary example based on three runs of some arbitrary clustering algorithm. As shown in Figure 1, pair (A, B) appears in some cluster in all three runs. Its robustness score is therefore 3/3. Pair (C, D), on the other hand, appears in some cluster in only two of three runs. Its score is thus 2/3. Robustness scores for all pairs that appear in at least one cluster are as follows: (A, B): 3/3; (A, C): 1/3; (A, D): 1/3; (B, C): 1/3; (B, D): 1/3; (C, D): 2/3; (C, E): 1/3; (D, F): 1/3; and (E, F): 2/3. We now simply average these scores to compute R, making the robustness of the algorithm that produced these clusters 0.481.

We tested several sorts of clustering algorithms, from conventional hierarchical clustering [64], to partitioning methods such as K-means [65] and QTClust [66], to graph-based methods such as paraclique [35, 67], CLICK [68], NNN [63] and WGCNA [69]. We also included SOM [70], a neural network method. Hierarchical clustering assigns items to clusters using a measure of similarity between clusters. Assignments are irrevocable; once an item has been placed in a cluster, it will remain in that cluster. Hierarchical clustering generally comes in two variants: bottom-up (agglomerative), which starts with size one clusters and iteratively combines clusters until only one is left, and top-down, which begins with all genes in one cluster, and then iteratively divides clusters until all clusters are size one. Agglomerative clustering is the simpler and more popular of the two, needing only a linkage criterion to compute cluster similarity. We therefore tested the agglomerative approach with four such criteria: average linkage [71], complete linkage [72], McQuitty [73], and Ward [74].

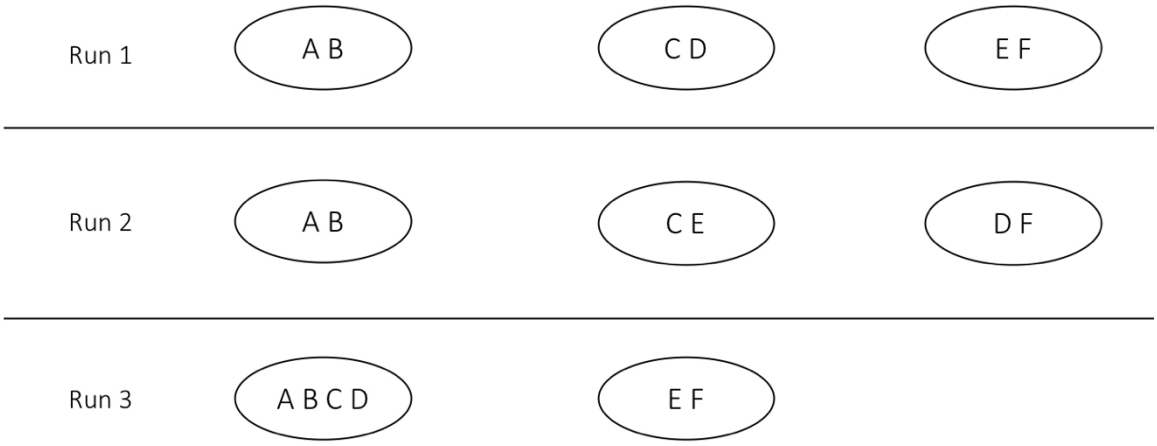

Figure 1. Clusters produced by three runs of a clustering algorithm.

Graph-based methods model items as vertices, with edges between items determined based again on some sort of similarity measure. To create graphs for transcriptomic data on which to run the paraclique method, we constructed co-expression networks as described in [43]. Genes were thus represented by vertices, while edges were weighted by Pearson product-moment correlation coefficients. A threshold was then applied to the network, so that an edge was retained if and only if its weight was at or above this threshold. In some circles, it has been fashionable to choose an arbitrary threshold, for example 0.85, based on previous experience [75-77]. We prefer a more mathematical and unbiased treatment based on spectral graph theory, whereby eigenvalues are computed over a range of potential thresholds, with the final threshold set using inflection points in network topology [44]. After thresholding, the paraclique method employs clique to help find extremely densely-connected subgraphs, but ones that may be missing a small number of edges [35, 67]. To generate such a cluster, paraclique isolates a maximum clique, then uses a controlled strategy to combine other vertices with high connectivity. Paraclique vertices are then removed from the graph, and the process repeated to find subsequent paraclique clusters. CLICK uses a graph-based statistical method to identify kernels and then expands them into full clusters with several heuristic approaches [68]. NNN, like paraclique, depends upon finding cliques, but only cliques of a specified (typically small) size. It edits a graph by connecting each vertex only to the k most similar other vertices according to some metric such as Pearson correlation, where k is a user-selected value. NNN merges overlapping cliques in the resulting graph to form an initial set of networks. It then divides the preliminary network at any existing articulation points, and ensures that no cluster is larger than half the number of input vertices. WGCNA operates on weighted networks using a soft threshold, raising the similarity matrix to a user-selected power in order to calculate extended adjacencies [69]. It then identifies gene modules using average linkage hierarchical clustering and dynamic tree cut methods. K-means clustering [65, 78] randomly selects k centroids and assigns genes to the nearest centroid, iteratively reassigning and recalculating centroids

until it converges. QTClust is a method developed specifically for gene expression data [66]. It builds a cluster for each gene, outputs the largest cluster, then removes these genes and repeats the process until no genes remain. SOM is a machine learning approach that groups genes using unsupervised neural networks. SOM repeatedly assigns genes to the most similar node until the algorithm converges [70].

In all, we tested four hierarchical methods, four graph-based methods, two partitioning methods, and one neural network method. We used publicly available versions of each technique. Most are available in R [79]. Table 1 provides a summary, along with the setting we varied for each algorithm.

Data

In previous work [39] we used *Saccharomyces cerevisiae* data from [80] to test cluster quality. In this chapter, we expand the test suite to 24 gene co-expression datasets from GEO, including the species *Drosophila melanogaster*, *Escherichia coli*, *Mus musculus* and *Penicillium chrysogenum*. Data from these organisms have been well-studied and annotated. All data are log2 transformed. Table 2 provides an overview of these datasets, along with the threshold selected using the aforementioned spectral techniques.

Comparisons

To compare algorithmic robustness, we altered a common setting for each method as specified in Table 1, selecting a range of values that produced clusters of the same scale. We transformed the myriad of output formats to simple cluster/gene membership lists. We also controlled r, the number of runs (values for each setting), to reduce its influence on our results. Runtime performance was not a consideration, although one algorithm, QTClust, never finished on dataset GDS5010, even after two weeks. We did not therefore obtain QTClust robustness

for that input. The robustness of each algorithm on each dataset was calculated for all runs over the range of settings.

Three algorithms (K-means clustering, hierarchical clustering and SOM) take the desired number of clusters as input. We thus selected this as the most appropriate setting to alter, and tested values from 200 to 300 so as to produce a range of average cluster sizes in line with the other algorithms. For example, hierarchical clustering produces a tree of clusters, and one obtains a list of disjoint clusters by choosing an articulation point in the tree. For SOM, we transformed the number of clusters to grid size. For example, when using 35 as the number of clusters (for dataset GDS344), the grid size was 5*7. We tested five grid sizes and two grid types (rectangular and hexagonal) for each dataset. We applied ten different powers (2, 4, 6, 8, 10, 14, 18, 22, 26 and 30) for WGCNA. For QTClust, we picked up ten different maximum cluster diameters from 0.05 to 0.5 with interval 0.05. For NNN, we chose ten different minimum neighborhood sizes ranging from 16 to 25. For CLICK, we applied nine homogeneity values (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9). For paraclique, we created graphs in the usual fashion, by calculating all pairwise correlations and placing edges between pairs correlated at or above a selected threshold. We controlled the number of paracliques generated so that they are in the same scale with other algorithms. We used the choice of maximum clique as the setting to vary. Dataset GDS772, for example, at threshold 0.94, resulted in a graph with nine maximum cliques. And so it was these nine cliques that provided variation. As can be seen from Table 2, over all inputs the threshold selected by spectral methods ranged from 0.8 to 0.95.

Table 1. Clustering methods tested for robustness.

Algorithm	Type	Setting	Implementation
Average	Hierarchical	Number of clusters	R 3.2.3
Complete	Hierarchical	Number of clusters	R 3.2.3
Mcquitty	Hierarchical	Number of clusters	R 3.2.3
Ward	Hierarchical	Number of clusters	R 3.2.3
CLICK	Graph-based	Cluster homogeneity	Expander4
NNN	Graph-based	Min neighborhood size	Java
Paraclique	Graph-based	Starting clique	C++
WGCNA	Graph-based	Power	R 3.2.3
K-means	Partitioning	Number of clusters	R 3.2.3
QTClust	Partitioning	Max cluster diameter	R 3.2.3
SOM	Neural network	Grid type/size	R 3.2.3

Results

Figure 2 shows robustness results for the four hierarchical algorithms, as tested across the 24 datasets previously described. Because all have robustness above 0.72, we averaged their scores to simplify Figure 3, which shows robustness results for all algorithms tested. As can be seen from this figure, hierarchical clustering and paraclique exhibit higher robustness than other algorithms. In fact, hierarchical clustering and paraclique have average robustness scores above 0.87, while all others are below 0.5. Figure 4 summarizes the results into an average robustness of each algorithm.

We also calculated the coefficient of variation (CV), the ratio of the standard deviation to the mean, as a measure of the stability of an algorithm's robustness. Hierarchical clustering exhibits the lowest CV, meaning that its robustness varies little across different datasets, whereas CLICK exhibits the highest CV. See Figure 5.

Discussion

It is not unexpected that hierarchical methods display the highest overall robustness. After all, results thereby produced form a hierarchical tree of successively merged clusters, so that varying the number of clusters simply cuts the tree at a different height, while the tree itself does not change. Once a pair of items appears together in some cluster, any decrease in the number of clusters on subsequent runs will continue to place that pair into the same cluster.

Table 2. Gene expression datasets tested in this study.

Dataset	Organism	Threshold	Edges	Vertices
GDS516	Drosophila melanogaster	0.89	3980	195322
GDS2485	Drosophila melanogaster	0.91	4604	30412
GDS2504	Drosophila melanogaster	0.81	7888	191715
GDS2674	Drosophila melanogaster	0.95	3334	5820
GDS1842	Drosophila melanogaster	0.91	2307	4589
GDS653	Drosophila melanogaster	0.95	1688	3368
GDS664	Drosophila melanogaster	0.8	14008	2298635
GDS1399	Escherichia coli	0.95	2880	5614
GDS5160	Escherichia coli	0.94	4826	74819
GDS5162	Escherichia coli	0.95	5038	293061
GDS5010	Mus musculus	0.9	10269	120907
GDS3870	Penicillium chrysogenum	0.94	6826	62431
GDS344	Saccharomyces cerevisiae	0.95	3071	6303
GDS772	Saccharomyces cerevisiae	0.94	1463	3785
GDS777	Saccharomyces cerevisiae	0.91	2244	11916
GDS1013	Saccharomyces cerevisiae	0.81	5312	555852
GDS1103	Saccharomyces cerevisiae	0.95	4215	38139
GDS1534	Saccharomyces cerevisiae	0.8	9335	1470003
GDS1674	Saccharomyces cerevisiae	0.93	3839	11904
GDS2267	Saccharomyces cerevisiae	0.83	4676	302104
GDS2508	Saccharomyces cerevisiae	0.9	3069	10485
GDS2663	Saccharomyces cerevisiae	0.8	9335	2617139
GDS3332	Saccharomyces cerevisiae	0.86	7290	572118
GDS2969	Saccharomyces cerevisiae	0.95	1679	5206

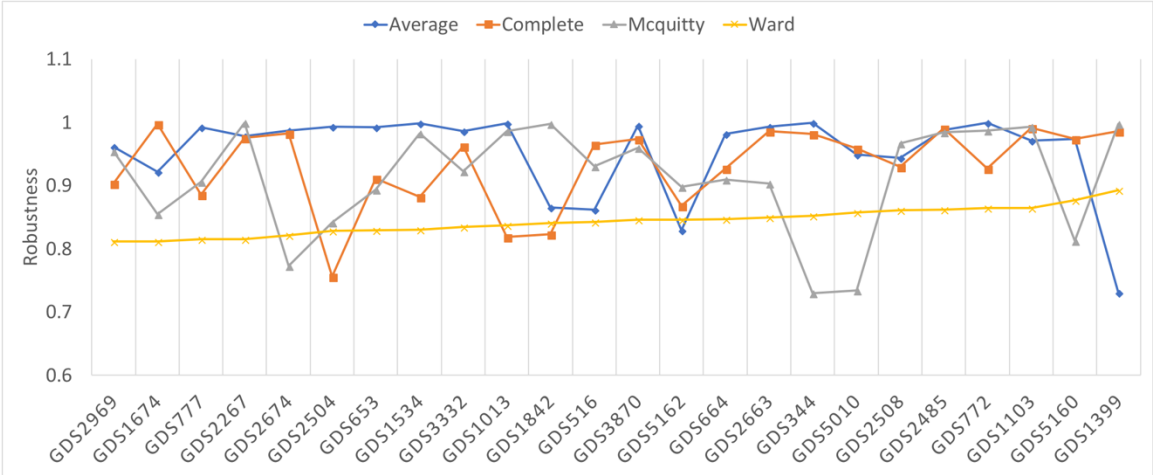

Figure 2. Robustness of four hierarchical algorithms on 24 transcriptomic datasets.

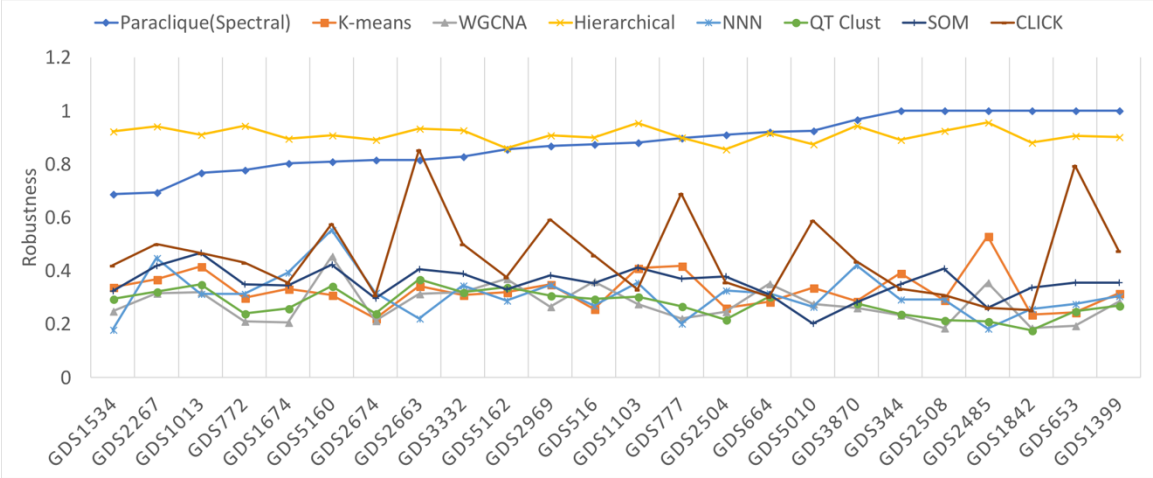

Figure 3. Robustness of all algorithms tested on 24 transcriptomic datasets.

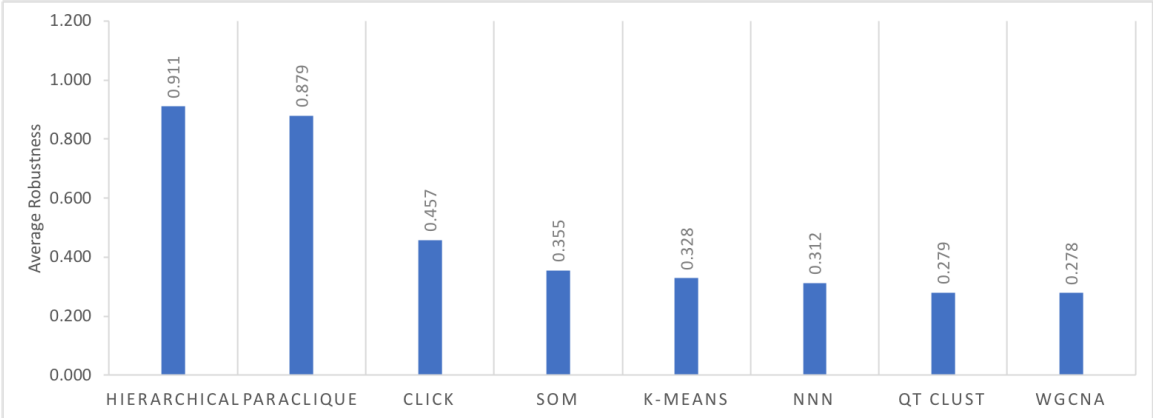

Figure 4. Average robustness of each algorithm.

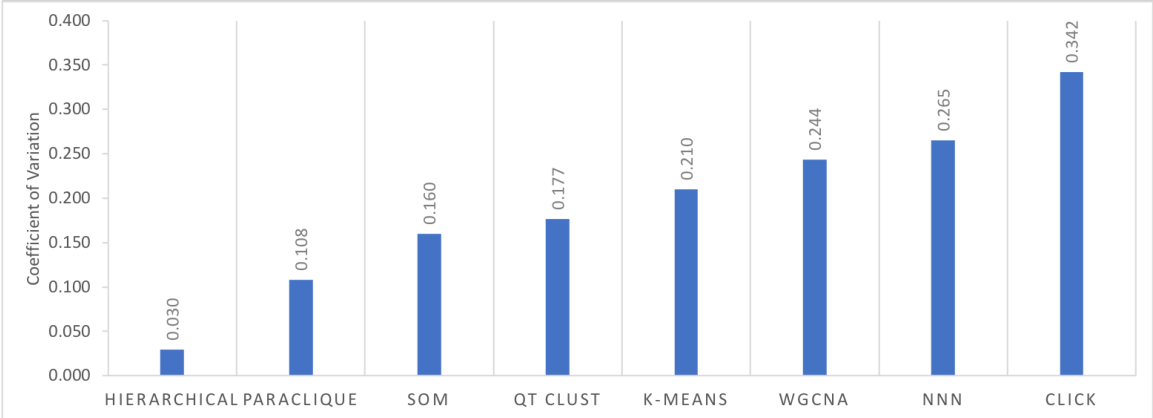

Figure 5. Coefficient of variation of each algorithm.

One might expect similar behavior from WGCNA, since it uses hierarchical clustering to identify modules. Because WGCNA uses soft-power to construct its network, however, the topology of each weighted network changes with different powers, so that item pairs are not at all stable. For K-means, as one alters the number of clusters (and hence centroids), the centroid with which a particular item is associated can change, while not changing an item's neighbors' centroids. Thus, items often shift to different clusters as the number of clusters changes. SOM and QTClust behave in similar fashion, in that grid size has a large effect on SOM while the partitioning performed by QTClust can divide pairs of formerly clustered items. Of the graph-based methods, CLICK and NNN first try to find a base cluster and then absorb other items into it. The absorbed items may change with different settings, affecting the clusters generated. For paraclique, the high robustness with different starting cliques is likely due in part to the fact that many of these cliques have significant overlap [34], at least on transcriptomic data. Many gene pairs may thus be included in a given cluster, no matter which maximum clique is selected. We have also observed quite similar overlap in graphs derived from many diverse types of data, including for example that derived from social and communications networks.

It is probably worth noting how robustness compares to accuracy and sensitivity [81], two popular clustering metrics. Accuracy measures faithfulness to ground truth. We make no assumptions, however, that ground truth is available or that it can even be known. Sensitivity most commonly refers to random noise or outliers. Robustness is not really related to either. A clustering algorithm could be highly sensitive to random noise, for example, and still have either high or low robustness.

This brings us to interpretation. How is the user to make sense of all this information? In our opinion, an algorithm with high robustness is generally preferable whenever it is difficult to determine optimum parameter settings. This

is of course because its results are unlikely to vary greatly across an entire range of these settings. As a case in point, if ground truth is largely unknown, or if hierarchical structure is implicit in the data under study, then hierarchical clustering can serve at least as a good starting candidate given its excellent robustness, relative simplicity and intuitive appeal. For more complex clustering tasks, however, we would endorse instead a graph-theoretical method such as paraclique due to its solid overall robustness and its much improved potential for biological fidelity [39].

Conclusions

We have introduced a new clustering metric, termed "robustness," in an effort to provide the research community with a simple, intuitive and informative measure of the stability and predictability of a clustering algorithm's behavior. To demonstrate its use, we have employed a suite of transcriptomic datasets as an unbiased testbed for algorithmic variation and evaluation. Widely-available data such as this provides a well-understood basis on which to introduce, explain and illustrate the use of the robustness metric. We hasten to add that robustness can, quite naturally, be applied to virtually any sort of omics data, or in fact to practically any sort of data on which clustering may be performed.

Simple hierarchical clustering displayed the highest overall robustness, due no doubt to the rigidly fixed tree structure of its clusters. Of the more sophisticated methods tested, only paraclique demonstrated similar robustness, thus demonstrating its resilience to the choice of starting maximum clique. In practice, one might expect that selecting such a clique with, say, the highest overall edge weight would be preferable. And certainly, that has much intuitive appeal. Nevertheless, our results show that it does not really much seem to matter, at least on data akin to those we've employed here.

Open questions abound. Note, for example, that robustness can be applied to virtually any non-overlapping clustering algorithm. All one needs is a reasonable settings range. What then of powerful clustering algorithms like clique? Clique is nonparametric and thus without settings. And one of its core strengths is actually its propensity to produce overlapping clusters on biological data (genes, for example, are very often pleiotropic, and thus likely to belong to multiple clusters). We are studying these and other related questions, and observe that for methods such as clique, in fact for essentially all clustering methods, an alternate notion of robustness might try to capture output predictability as the underlying network is perturbed.

Chapter 3
Clique Selection and its Effect on Paraclique Enrichment: An Experimental Study

A version of this chapter written by Yuping Lu, Charles A. Phillips, Elissa J. Chesler and Michael A. Langston has been submitted for publication and is currently under review.

My contribution was to write a suite of scripts to compare weighted paraclique enrichment p-values and to analyze the results.

Abstract

The paraclique algorithm provides an effective means for biological data clustering. It satisfies the mathematical quest for density, while fulfilling the pragmatic need for noise abatement on real data. Given a finite, simple, edge-weighted and thresholded graph, the paraclique method first finds a maximum clique, then incorporates additional vertices in a controlled manner, and finally extracts the subgraph thereby defined. When more than one maximum clique is present, however, deciding which to employ is usually left unspecified. In practice, this frequently and quite naturally reduces to using the first maximum clique found. In this chapter, maximum clique selection is studied in the context of well-annotated transcriptomic data, with ontological classification used as a proxy for cluster quality. Enrichment p-values are compared using maximum cliques chosen in a variety of ways. The most appealing and intuitive option is almost surely to start with the maximum clique having the highest average edge weight. Although there is of course no guarantee that such a strategy is any better than random choice, results derived from a variety of experiments indicate that, in general, this approach produces a small but statistically significant improvement in overall cluster quality.

Introduction

Clustering is a core task in biological network analysis, whereby a cluster is typically defined as a dense subnetwork extracted from high throughput omics data using some measure of pairwise similarity between genes, proteins, metabolites or other biological entities. Popular similarity metrics include Pearson's product-moment correlation, Spearman's and Kendall's rank correlations, and methods better suited for handling nonlinear relationships such as mutual information. An oft-used example is based on DNA microarray and gene co-expression analysis [82-84] in the context of the relevance network framework [85, 86]. In this setting, we begin with a complete graph whose vertices denote probe sets (gene surrogates), each of whose edges is assigned a weight equal to the similarity across all samples of the expression levels of its endpoints. Thresholding [44] produces an incomplete, unweighted graph on which scalable, state-of-the-art graph theoretical algorithms can be applied. The increased biological fidelity produced by these algorithms has previously been studied [39], further motivating their use. Well-known examples include clique-centric methods such as the bottom-up approach originally called k-clique communities [87] (now renamed clique percolation), and the more efficient top-down strategy known as paraclique first introduced in [35].

A main aim of the paraclique algorithm is to ameliorate the effects of noise, primarily by reducing type II errors (false negatives). It accomplishes this by expanding a maximum clique in a tightly-controlled manner with non-clique vertices that are adjacent to most, but not necessarily all, elements of the clique. We refer the reader to [67] for a density analysis and formal description of the paraclique method. A major motivation for such a strategy rests in the fact that clique-centric methods are highly sensitive to so-called "missing" edges, which may be lost due to noise, experimental data capture, the effects of thresholding, and a variety of other factors dependent on the problem at hand. The paraclique

algorithm has found utility in numerous network science domains. In the health sciences alone, it has been employed in the study of lung cancer [88] and the exposome [89], as well as in transcriptomics [90], proteomics [91], epigenetics [92, 93], diabetes [94], allergic rhinitis [95], obesity [96], community-acquired pneumonia [97] and even in studying the impact of low dose ionizing radiation [98].

The main feature of interest here is the selection criteria for the maximum clique chosen for expansion. This question may at first seem moot given the computational recalcitrance of finding even one maximum clique, a classic NP-complete problem [31]. But modern, practical algorithms make it feasible not only to find a single maximum clique, but to enumerate all of them [34]. With such capability now at hand, we created a test suite of graphs to measure the significance and consistency of maximum clique selection on cluster quality. For these we retained original edge weights, employed the well-known Gene Ontology (GO) [54, 55] as a proxy for a ground truth, and performed enrichment analysis [52] to determine how likely a cluster's contents are to occur by mere chance alone. For each graph thus constructed, we compared paracliques expanded from a maximum clique with the highest average edge weight, from another with the lowest average edge weight, and from one chosen at random. We note that, for a given graph, all maximum cliques have the same size, and thus a maximum clique with the highest (lowest) average edge weight will naturally also have the highest (lowest) total edge weight.

Main text

Experimental Data

We employed 28 *Saccharomyces cerevisiae* microarray expression datasets obtained from the Gene Expression Omnibus (GEO) [29, 62, 99]. S. cerevisiae is

one of the simplest and best-studied eukaryotic organisms, possessing numerous essential cellular processes analogous to those found in humans. The first column of Table 3 contains the GEO accession numbers for datasets used in this study. For each, we constructed 21 unweighted graphs using Pearson's product-moment correlations, with thresholds set at uniform increments of 0.01 over the interval 0.70 to 0.90. This produced a total of 588 graphs ranging in size from 1893 to 9335 vertices. Densities ranged from roughly 0.09% to 25%, where we define density in the usual way as the number of edges present divided by the maximum number of edges possible. On each such graph we tested the three aforementioned maximum clique selection strategies, and ran the paraclique algorithm using the ORNL CADES platform [100], a Cray CS400 with Intel Xeon E5-2698 v3 and 128–256 GB of RAM per node. We halted a run only if it failed to complete its task within 48 hours. All but 20 graphs were solved in this fashion. (These 20 were of course excluded from the analysis.) Over the remaining 568 graphs, we then performed GO functional enrichment using the tools at DAVID [53] on the first paraclique produced in each of the 1704 resultant paraclique listings. To produce a single score for each paraclique, we computed the p-value of its most significant GO term.

Results

In Table 3, we list results obtained for graphs constructed at a sample threshold 0.80. Often the choice between a highest, a lowest, and a randomly-chosen maximum clique makes little difference in p-value. On the other hand, this difference can sometimes be quite large, as is seen for example in the case of GDS2267. Of these 28 graphs, 11 had a better p-value in the paraclique constructed using a highest weight maximum clique versus a lowest weight maximum clique, nine exhibited no difference, and in eight a maximum clique of lowest weight produced a paraclique with a better p-value than did a maximum clique of highest weight. Thus, the ratio 11/8=1.375 denotes a measure of how often a better p-value was obtained by choosing a highest versus a lowest weight maximum clique. If this

ratio across all tests tends to be consistently greater than 1, then it may be viewed as a reliable indication that selecting a highest weight maximum clique generally produces more highly enriched paracliques, which may then result in improved average cluster quality.

Comparisons Between Highest and Lowest Weight Maximum Cliques

In Table 4, we summarize results comparing a highest weight paraclique to a lowest weight paraclique for all 21 thresholds under study. For each threshold, we list the number of graphs in which a highest weight maximum clique produced a lower p-value paraclique than did a lowest weight maximum clique, the number of graphs in which the reverse was true, the number of graphs in which the p-values were no different, and a ratio denoting the number of times highest weight was better to the number of times lowest weight was better. Overall, highest weight was better in 234 graphs, there was no difference in 177 graphs, and lowest weight was better in 157 graphs. Interestingly, the ratio was greater than one at all 21 thresholds, suggesting that it is generally beneficial to select a maximum clique of highest weight over one of lowest weight. Over the 1136 graphs tested, choosing a highest versus a lowest weight maximum clique resulted in improved cluster quality 1.490 times more often than it resulted in worse cluster quality. To estimate statistical significance, we employed two binomial tests. For the first test, shown in the last column of Table 4, we assumed an equal likelihood for each of three possible outcomes: a better, a worse, or an unchanged p-value. Overall, this test yielded a significant result, with $p = 0.0000163$. For the second test, we used the observed proportion of graphs for which there was no difference as an estimate of the proportion of "no difference" graphs in the population. This assumed that, for all other graphs, a paraclique constructed using a highest versus a lowest weight maximum clique had equal likelihood of producing a better p-value. This test was also significant, with $p = 0.00047$.

Table 3. Experimental results obtained at a threshold of 0.80.

Dataset	**Maximum Clique**		**Average Paraclique Edge Weights and Enrichment Scores**					
	Size	Number	Highest	P-value	Lowest	P-value	Random	P-value
GDS344	87	6	0.9111	1.10E-49	0.9099	5.30E-50	0.9105	5.30E-50
GDS362	304	75184	0.9267	2.60E-09	0.9259	1.90E-10	0.9267	1.90E-10
GDS600	1736	40	0.9584	1.50E-06	0.9584	1.40E-06	0.9584	1.50E-06
GDS772	78	6	0.9134	2.70E-26	0.9118	2.70E-26	0.9118	2.70E-26
GDS777	87	15	0.9101	2.00E-08	0.9096	2.00E-08	0.9096	2.00E-08
GDS922	450	2160	0.9235	5.20E-11	0.9230	5.80E-11	0.9232	6.50E-11
GDS991	317	2468	0.9245	1.10E-95	0.9224	1.70E-85	0.9243	6.90E-97
GDS1013	269	19152	0.9127	3.30E-127	0.9112	6.90E-123	0.9123	3.30E-127
GDS1103	312	672	0.9293	8.10E-20	0.9283	8.10E-20	0.9290	9.40E-20
GDS1534	154	180	0.9140	3.40E-08	0.9133	1.20E-06	0.9137	3.40E-08
GDS1550	361	240	0.9469	2.60E-05	0.9459	2.50E-05	0.9464	2.60E-05
GDS1551	453	48	0.9408	5.30E-06	0.9405	4.80E-06	0.9405	4.80E-06
GDS1611	182	258	0.8847	3.90E-05	0.8839	3.70E-05	0.8845	3.70E-05
GDS1674	93	160	0.9102	8.00E-14	0.9078	1.40E-13	0.9090	1.40E-13

Table 3. Continued.

Dataset	Maximum Clique		Average Paraclique Edge Weights and Enrichment Scores					
GDS2050	617	1152	0.9365	2.10E-32	0.9363	2.10E-32	0.9364	2.80E-32
GDS2079	1611	16	0.9563	8.30E-07	0.9563	8.30E-07	0.9563	4.50E-07
GDS2267	168	312	0.9058	7.50E-103	0.9035	3.00E-98	0.9058	2.60E-101
GDS2462	1351	13	0.9538	3.10E-46	0.9535	1.30E-43	0.9537	3.10E-46
GDS2508	49	11	0.9036	1.40E-03	0.8980	1.50E-03	0.9002	1.50E-03
GDS2522	428	13724	0.9321	1.40E-03	0.9313	1.50E-03	0.9318	2.10E-04
GDS2625	309	80	0.9191	3.00E-06	0.9187	2.80E-06	0.9189	2.80E-06
GDS2663	282	600	0.9283	5.80E-18	0.9269	4.40E-16	0.9273	4.40E-16
GDS2925	89	60	0.8940	1.10E-03	0.8930	3.80E-03	0.8934	4.40E-03
GDS2969	119	24	0.9161	1.80E-12	0.9143	1.80E-12	0.9148	1.80E-12
GDS3061	181	152	0.9218	2.80E-25	0.9198	2.80E-25	0.9208	2.80E-25
GDS3137	562	1088	0.9354	1.00E-04	0.9350	1.00E-04	0.9353	1.50E-04
GDS3198	383	2184	0.9333	3.50E-06	0.9327	1.70E-06	0.9331	2.80E-06
GDS3438	3424	2	0.9898	8.50E-11	0.9898	8.50E-11	0.9898	8.50E-11

Table 4. Paraclique with highest weight maximum clique vs paraclique with lowest weight maximum clique.

Threshold	Highest Better	No Difference	Lowest Better	Highest Better / Lowest Better	Binomial P-value
0.70	16	6	4	4	2.14E-03
0.71	10	7	6	1.667	9.96E-02
0.72	10	4	8	1.25	8.44E-02
0.73	11	6	9	1.222	9.96E-02
0.74	13	8	6	2.167	4.31E-02
0.75	14	5	8	1.75	2.15E-02
0.76	11	8	8	1.375	1.12E-01
0.77	13	8	7	1.857	5.36E-02
0.78	15	7	6	2.5	1.34E-02
0.79	9	10	8	1.125	1.61E-01
0.80	11	9	8	1.375	1.23E-01
0.81	12	7	8	1.5	7.47E-02
0.82	12	8	8	1.5	8.72E-02
0.83	9	12	7	1.286	1.58E-01
0.84	10	9	9	1.111	1.50E-01
0.85	11	8	9	1.222	1.23E-01
0.86	11	8	9	1.222	1.23E-01
0.87	8	13	7	1.143	1.42E-01
0.88	10	10	8	1.25	1.50E-01
0.89	10	10	8	1.25	1.50E-01
0.90	8	14	6	1.333	1.42E-01
Total	234	177	157	1.490	1.63E-05

Comparisons Between Highest and Random Weight Maximum Cliques

In Table 5, we list the results of testing whether choosing a highest weight maximum clique may be superior to choosing an arbitrary maximum clique, a process we simulated by selecting a maximum clique at random from among all maximum cliques enumerated. Once again, all ratios in the penultimate column are greater than or equal to one, and so we conclude that choosing a highest weight maximum clique tends to be wiser than merely making an arbitrary choice. Overall, the highest weight was better in 216 graphs, there was no difference in 225 graphs, and a random choice was better in 127 graphs. At first these differences may not appear as striking as did the differences between using a highest versus a lowest maximum clique. For example, the number of graphs for which there was no difference is noticeably larger in Table 5 than it was in Table 4. On the other hand, choosing a highest weight maximum clique resulted in improved cluster quality 1.701 times more often than it resulted in worse cluster quality, which is a slightly higher ratio than that computed from Table 4. Moreover, repeating the two binomial tests just described, we obtained significant results for both, with $p = 0.00219$ and $p = 0.0000278$, respectively.

Comparisons Between Random and Lowest Weight Maximum Cliques

Lastly, we used the same approach to compare paracliques constructed using random versus lowest weight maximum cliques. The results are shown in Table 6. A random choice was better in 191 graphs, there was no difference in 215 graphs, and a lowest choice was better in 162 graphs. Although the aforementioned ratio was still above one (at 1.179), neither binomial test reached the level of significance, with $p = 0.035$ and $p = 0.0876$, respectively.

Table 5. Paraclique with highest weight maximum clique vs paraclique with random maximum clique.

Threshold	Highest Better	No Difference	Random Better	Highest Better / Random Better	Binomial P-value
0.70	17	3	6	2.833	6.29E-04
0.71	9	12	2	4.5	1.42E-01
0.72	8	6	8	1	1.67E-01
0.73	10	8	8	1.25	1.37E-01
0.74	11	12	4	2.75	1.12E-01
0.75	11	6	10	1.1	1.12E-01
0.76	13	9	5	2.6	4.31E-02
0.77	15	11	2	7.5	1.34E-02
0.78	11	10	7	1.571	1.23E-01
0.79	11	11	5	2.2	1.12E-01
0.80	9	10	9	1	1.58E-01
0.81	9	11	7	1.286	1.61E-01
0.82	10	11	7	1.429	1.50E-01
0.83	8	14	6	1.333	1.42E-01
0.84	12	12	4	3	8.72E-02
0.85	9	12	7	1.286	1.58E-01
0.86	7	14	7	1	1.09E-01
0.87	11	12	5	2.2	1.23E-01
0.88	9	13	6	1.5	1.58E-01
0.89	9	13	6	1.5	1.58E-01
0.90	7	15	6	1.167	1.09E-01
Total	216	225	127	1.701	2.19E-03

Table 6: Paraclique with random maximum clique vs paraclique with lowest weight maximum clique.

Threshold	Random Weight is Better	No Difference	Lowest Weight is Better	Random Better / Lowest Better	Binomial P-value
0.70	12	7	7	1.714	6.23E-02
0.71	8	7	8	1	1.71E-01
0.72	10	6	6	1.667	8.44E-02
0.73	11	8	7	1.571	9.96E-02
0.74	10	7	10	1	1.45E-01
0.75	11	6	10	1.1	1.12E-01
0.76	9	7	11	0.818	1.61E-01
0.77	8	10	10	0.8	1.42E-01
0.78	13	6	9	1.444	5.36E-02
0.79	8	11	8	1	1.53E-01
0.80	7	13	8	0.875	1.09E-01
0.81	8	14	5	1.6	1.53E-01
0.82	9	10	9	1	1.58E-01
0.83	9	14	5	1.8	1.58E-01
0.84	8	11	9	0.889	1.42E-01
0.85	8	14	6	1.333	1.42E-01
0.86	9	13	6	1.5	1.58E-01
0.87	8	12	8	1	1.42E-01
0.88	11	10	7	1.571	1.23E-01
0.89	8	13	7	1.143	1.42E-01
0.90	6	16	6	1	6.91E-02
Total	191	215	162	1.179	3.50E-02

Discussion and Conclusions

As can be seen in Table 3, there is sometimes little difference in enrichment p-values. And indeed, as can be seen in Tables 4 and 5, there are instances for which the choice makes no difference at all. Close scrutiny reveals that this is usually due to significant overlap between maximum cliques. In GDS344, for example, it turns out that 84 (of 87) vertices appear in all maximum cliques at a threshold of 0.8. We also note that the number of maximum cliques can vary greatly between datasets, and even between graphs constructed at different thresholds from the same dataset. In Table 3, for instance, we witnessed from 2 to 75184 maximum cliques at a single threshold. And GDS2925 had but one maximum clique when thresholded at 0.89, but 95044 when thresholded at 0.74.

These issues are relevant because large numbers of maximum cliques can dramatically increase computational costs. Thus, we tested only the first paraclique produced under each criterion, else time requirements quickly become prohibitive. To see this, note that not only is clique extraction an expensive operation in its own right, but a sample graph with, say, 100 different maximum cliques will yield 100 different first paracliques that, once deleted, leave a set of 100 new graphs, each of which may again have 100 different maximum cliques, paracliques and so on ad infinitum.

In summary, these comprehensive tests provide convincing evidence that selecting a highest weight maximum clique tends to produce more functionally enriched paracliques than does choosing either a lowest weight or an arbitrary maximum clique. While this seems rather intuitive and to be expected, the effect size has been small, and so a large number of graphs has been required to confirm this relationship. Across Tables 4 and 5, for example, only two thresholds are significant at $p = 0.01$. Every other result, when analyzed alone, is non-significant. It is therefore only when results at many thresholds are combined that we reach a

large enough sample size for the maximum clique choice to meet the standards of statistical significance.

Limitations

Only transcriptomic data and Pearson correlations were considered. Testing was limited to the first paraclique in each graph. Tiebreakers were not employed should two or more paracliques have had the same average edge weight.

Chapter 4
Detecting Outliers in Streaming Time Series Data from ARM Distributed Sensors

A version of this chapter was originally published by Yuping Lu, Jitendra Kumar, Nathan Collier, Bhargavi Krishna, and Michael A. Langston:

Yuping Lu, Jitendra Kumar, Nathan Collier, Bhargavi Krishna, and Michael A. Langston. "Detecting outliers in streaming time series data from ARM distributed sensors." *In 2018 IEEE International Conference on Data Mining Workshops (ICDMW),* pp. 779-786. IEEE, 2018.

My contribution was to conceive, implement and test the framework of three algorithms to automatically detect outliers in ARM meteorological data.

Abstract

The Atmospheric Radiation Measurement (ARM) Data Center at ORNL collects data from a number of permanent and mobile facilities around the globe. The data is then ingested to create high level scientific products. High frequency streaming measurements from sensors and radar instruments at ARM sites require high degree of accuracy to enable rigorous study of atmospheric processes. Outliers in collected data are common due to instrument failure or extreme weather events. Thus, it is critical to identify and flag them. We employed multiple univariate, multivariate and time series techniques for outlier detection methods and studied their effectiveness. First, we examined Pearson correlation coefficient which is used to measure the pairwise correlations between variables. Singular Spectrum Analysis (SSA) was applied to detect outliers by removing the anticipated annual and seasonal cycles from the signal to accentuate anomalies. K-means was applied for multivariate examination of data from collection of sensors to identify any deviation from expected and known patterns and identify abnormal observation. The Pearson correlation coefficient, SSA and K-means methods were later combined together in a framework to detect outliers through a range of checks. We applied the developed method to data from meteorological sensors at ARM

Southern Great Plains site and validated against existing database of known data quality issues.

Introduction

The Atmospheric Radiation Measurement (ARM) user facility was founded by the U.S. Department of Energy (DOE) in 1989 [101]. Since then, its aim is to be the platform for the observation and study of Earth's climate. ARM facility collects large volume of datasets from instruments deployed in different ground stations across the globe [30]. The ARM Data Center (ADC) is responsible for ingesting the collected data and creating high level scientific data products for distribution and dissemination to scientific research community, especially to inform and improve the representation of atmospheric, cloud and aerosols processes in global climate models (GCMs) [102]. They also develop a large number of high-level data products, also called “Value Added Products” (VAPs), quality of which are highly dependent on the correctness of the raw data. Data are transferred from individual site to ADC in a streaming near-real-time fashion and the raw data is ingested, processed to produce VAPs and made available to users via a web-based data discovery interface with a lag time of less than an hour. Along with expediency, it is also essential to identify, address, and communicate any noise and outliers in the data to maintain high data quality. Thus an effective and efficient outlier and noise detection is crucial for ARM to provide scientific users with high quality data for research.

Outlier detection, also called anomaly detection or intrusion detection, is a common task in many application domains that include time series data, streaming data, distributed data, spatio-temporal data, and network data [103]. Common techniques for outlier detection include signal processing, classification, clustering, nearest neighbor, density, statistical, information theory, spectral

decomposition, and visualization. Among all these techniques, time series data outlier detection and temporal network outlier detection are especially useful for ARM data.

Outlier detection in time series data was first studied by Fox in 1972 [104]. Common types of outliers are additive outliers, level shifts, temporary changes, and innovative outliers. One common approach is the discriminative method which is based on a similarity function. For example, the normalized longest common subsequence (NLCS) is a similarity measurement widely used in the field of data mining [105-107]. Commonly used clustering methods such as K-means [78], dynamic clustering [107], single-linkage clustering [108], principal component analysis (PCA) [109], and self-organizing map (SOM) [110] are also popular.

Different from the methods mentioned above, window-based detection breaks the time series data into overlapping subsequences with fixed window size [111]. Each window is assigned an anomaly score, and then a final score for the times series data is calculated by aggregating the window scores. Subspace based analysis for univariate time series data is similar to window-based detection. The subspace-based transformation is to convert a univariate time series into a multivariate time series with fixed window size. It then transforms the multivariate time series back to univariate time series. Singular Spectrum Analysis is a widely used algorithm for such problem [112].

ARM data also belongs to the class of temporal data as we can sequentially create a time series of network changes or graph snapshots at different periods. Each period forms a graph snapshot using various graph distance metrics from a set of nodes. Many challenges exist for outlier detection for temporal data. First, the algorithm or model needs to be chosen carefully as the properties of each data and network are different. Second, the temporal data has space and time

dimensions which make it complex to analysis. Third, its scale is massive, and efficient algorithm is crucial for fast outlier detection. One common problem for temporal data is to detect outlier graph snapshots from a series of graph snapshots in temporal networks. Pearson correlation coefficient, which is explained in detail later, is a good candidate for such problem.

A number of approaches have been developed in literature for temporal outlier detection, especially for environmental sensor data. Birant et al. [113] discovered high wave heights values as outliers while studying the wave height values from the east of the Mediterranean Sea, the Marmara Sea, the Black sea, and the Aegean Sea. Hill et al. [114, 115] filtered out measurement errors in the wind speed data stream from Water and Environmental Research Systems (WATERS) Network Corpus Christi Bay testbed with dynamic Bayesian networks. Drosdowsky et al. [116] found anomalies from Australian district rainfall using rotated PCA. Wu et al. [117] detected precipitation outlier events while working on South American precipitation data set. Sun et al. [118] extracted locations which always have different temperature from their surroundings by exploring the South China area dataset from 1992 to 2002.

Within ARM program, the Data Quality Office (DQO) is charged with inspecting and assessing approximately 5,000 data fields on a daily to weekly basis. The objective of DQO is to quickly identify data anomalies and report them to site operators and instrument mentors so that corrective actions can be performed and thereby minimize the amount of unacceptable data collected. With focus on quick near real-time assessment of data, process relies heavily on univariate analysis and lacks rigorous detection of outliers. Objective of this study was to develop efficient and rigorous outlier detection technique for ARM time series data using univariate, multivariate and time series statistics techniques.

Datasets

ARM data are stored and distributed in the Network Common Data Form (NetCDF) format which is self-describing and machine-independent [119, 120] and has good performance and data compression. It is commonly used to handle scientific data, especially in climate and Earth sciences, meteorology, oceanography, and remote sensing etc. All ARM data are publicly available and can be downloaded from ARM Data Center (https://www.arm.gov/data) where a large range of datasets ranging from meteorology, to atmospheric profiles, to weather radars to satellite observations are available. Datasets are collected at a number of different locations using large number of diverse instruments are available within ARM.

In this study, we used the data from Surface Meteorology Systems (MET) collected at the ARM Southern Great Plains (SGP) site in Oklahoma, United States. SGP is ARM's largest facility that comprises of a network of core and extended facilities. In our study we used MET data from 24 extended facilities where surface meteorological observations have been collected continuously and independently. While MET instruments collect a large array of direct and indirect measurements, we focused our analysis on five core meteorological variables: air temperature (*temp_mean*), vapor pressure (*vapor_pressure_mean*), atmospheric pressure (*atmos_pressure*), relative humidity (*rh_mean*) and wind speed (*wspd_arith_mean*). These five core meteorological variables are inputs for a large number of derived datasets produced by the ARM and are often essential set of data for most atmospheric analysis, hence focus of our study. Table 7 provides details of sites and available time series for the datasets used.

Table 7. SGPMET datasets used in this study.

Facility	E1	E3	E4	E5	E6	E7
Begin Year	1996	1997	1996	1997	1997	1996
End Year	2008	2008	2010	2008	2010	2011
Facility	E8	E9	E11	E13	E15	E20
Begin Year	1994	1994	1996	1994	1994	1994
End Year	2008	2017	2017	2017	2017	2010
Facility	E21	E24	E25	E27	E31	E32
Begin Year	2000	1996	1997	2004	2012	2012
End Year	2017	2008	2001	2009	2017	2017
Facility	E33	E34	E35	E36	E37	E38
Begin Year	2012	2012	2012	2012	2012	2012
End Year	2017	2017	2017	2017	2017	2017

Methodology

From the many outlier detection methods introduced in the first section, we carefully selected Pearson correlation coefficient, Singular Spectrum Analysis and K-means for our study and applied them to ARM time series data.

Data Pre-processing

Raw time series data from MET instruments are available at temporal resolution of one minute for all variables considered in this study. Data were pre-processed for in various analysis in our study. One-minute temporal resolution time series was standardized with mean of zero and one standard deviation for Pearson Correlation analysis. A daily temporal resolution standardized time series was prepared for use with SSA based detection method. Multi-variate cluster analysis was conducted using standardized daily time series of all meteorological variables.

Pearson Correlation Coefficient

Co-located meteorological variables measure different aspect of the atmospheric conditions at any location, and driven by atmospheric physics are inherently correlated with each other. Any atmospheric phenomena at the location would affect all variables in an expected and correlated fashion. Analysis of historical time series data would provide us the baseline correlation structure and patterns for the location. Any abrupt change or break in correlation structure among meteorological behavior can be a sign of sensor malfunction and should be identified as an outlier. In addition, ARM SGP site comprise of multiple facilities making similar sets of measurement and any abrupt change in correlation structure not observed at other facilities will also indicate a potential outlier.

The Pearson correlation coefficient was first introduced by Karl Pearson [36] and can be used to measure the linear correlation between two variables. The Pearson correlation coefficient is calculated from the covariance of two variables divided by the multiplication of the standard deviation of those two variables. This normalization results in a value between [-1, 1]. If the value is close to -1, it means those two variables are highly negatively related. On the other hand, if the value is close to 1, then the two variables are strongly positively related. If the value is near 0, it means those two variables do not have linear relation.

We performed a pairwise comparison of the five variables using Pearson correlation using data from all 24 extended facilities. Atmospheric dynamics are strongly driven by seasons and the correlation patterns among meteorological variables can have season specific patterns. We performed our analysis seasonally by separating the data among Winter, Spring, Summer and Fall seasons. Figure 6 shows the distribution of pairwise correlation for Spring season. All variables show strong correlations which are normally distributed. The long tails of the distribution are potentially due to outlier data points. For example, the Pearson correlation between air temperature and vapor pressure is positively correlated with correlation mean close to 0.75. And the Pearson correlation between atmospheric pressure and air temperature is negatively correlated with correlation mean close to -0.60. These highly correlated Pearson correlation coefficients are stored as the expected values between two variables. We then compare each Pearson correlation of two variables from a specific season in a specific year from a specific instrument individually. If this pairwise Pearson correlation of two variables deviates far away from our expected historical correlation, we treat it as an outlier. This method would allow to check incoming datastream on near-real-time basis to identify outliers.

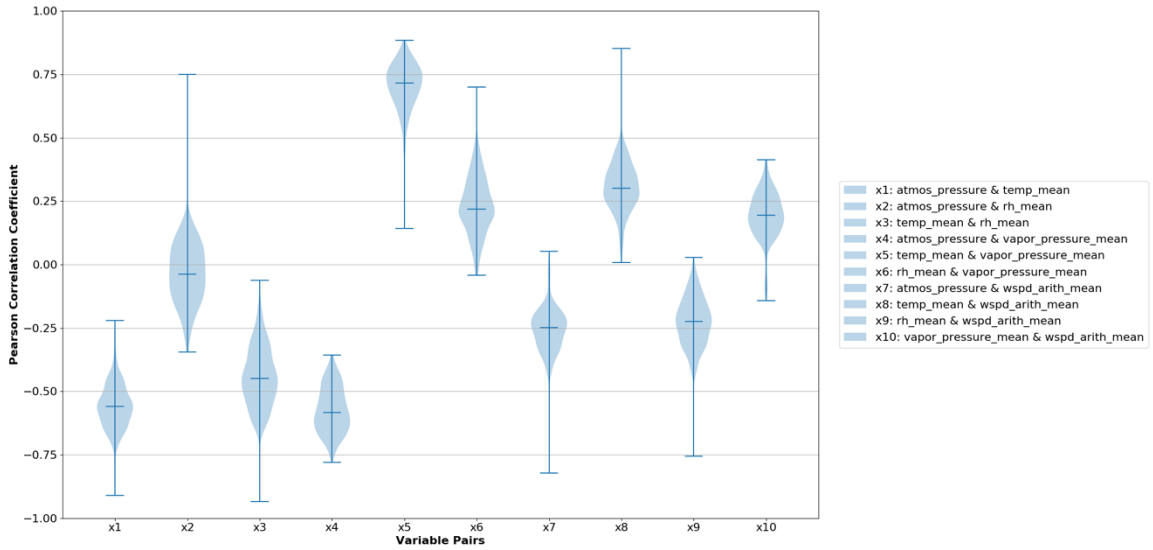

Figure 6. Pearson Correlation patterns for ten meteorological variable pairs during spring season across all the years.

Singular Spectrum Analysis

Univariate time series analysis of meteorological variables can be applied to identify any unexpected variability and extreme values observed by the instruments. These anomalous observations can be indicative of extreme atmospheric events at the site and are important to identify. However, a range of natural inter-annual and intra-annual variability in meteorological times series is also expected and it's important to not erroneously flag them as outliers. We applied Singular Spectrum Analysis for time series of analysis of meteorological observations to identify extreme events.

Singular Spectrum Analysis (SSA) is a popular method for time series data analysis [112, 121]. The general idea is to use a subset of the decomposition of trajectory matrix to approximate the original data. Many applications can be found in [112]. For example, SSA can be applied to monitor volcanic activity [122]. It can also be used to extract trend [123]. SSA method is designed to remove any number of modes of specified periodicity from the time series. This is meant to remove known seasonalities from the data in order to isolate true anomalous values more accurately.

Assume we have an ARM time series data Y of length T

$$\mathrm{Y} = (\mathrm{y}_1, \ \ldots, \mathrm{y}_\mathrm{T})$$

where $T > 2$ and y_i is not empty. Let L $(1 < L \leq T/2)$ be the window size and $K = T - L + 1$. In general, the algorithm contains two main parts: decomposition and reconstruction. The first step is to form the trajectory matrix **X** from vector Y by embedding subsets of Y. These subsets of Y X_i are lagged vectors of length L.

$$X_i = (y_i, \ \ldots, y_{L+i-1})^T \quad (1 \leq i \leq K)$$

$$X = [X_1, \ \ldots, X_K]$$

Thus the trajectory matrix is

$$\boldsymbol{X} = \left(x_{ij}\right)_{i,j=1}^{L,K} = \begin{pmatrix} y_1 & y_2 & y_3 & \cdots & y_K \\ y_2 & y_3 & y_4 & \vdots & y_{K+1} \\ y_3 & y_4 & y_5 & \vdots & y_{K+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_L & y_{L+1} & y_{L+2} & \cdots & y_T \end{pmatrix} \quad (1)$$

where $x_{ij} = y_{i+j-1}$. We can see from equation 1 that matrix **X** has equal elements on anti-diagonals and therefore it is a Hankel matrix. Then we perform the singular value decomposition (SVD) on $S = XX^T$ where the eigenvalues of S are denoted by $\lambda_1, \ldots, \lambda_L$ in the decreasing order of magnitude ($\lambda_1 \geq \cdots \geq \lambda_L \geq 0$) and the corresponding eigenvectors by $P_1, \ldots, P_L$. Let $d = rank\, \boldsymbol{X}$ and $V_i = \boldsymbol{X}^T P_i / \sqrt{\lambda_i} (i = 1, \ldots, d)$. Thus, the trajectory matrix **X** can then be written by its eigendecomposition,

$$\boldsymbol{X} = \boldsymbol{X_1} + \cdots + \boldsymbol{X_d} \quad (2)$$

where $\boldsymbol{X_i} = \sqrt{\lambda_i} P_i V_i^T$.

Next we choose a subset of eigenpairs to form an approximation of the trajectory matrix. It is at this point that our version of the algorithm differs. The Fast Fourier Transform (FFT) was first proposed by Cooley and Tukey [124] to compute the Discrete Fourier Transform faster, reducing the computation complexity from $O(n^2)$ to $O(n\, logn)$. FFT's are often used to convert data from the time domain to the frequency domain and vice versa. Given that the time series we are studying has seasonality at known frequencies, we use FFT to find the dominant frequency of each eigenvector. We then approximate the trajectory matrix by including modes which match the frequencies of the seasonality we wish to remove. For example, we anticipate that the temperature data will have an annual and possibly monthly cycle, as shown in Figure 7. SSA allows us to tease out these contributions in additive fashion. In this example, the signals from the year, month, and residual sum together to form the original raw data. This residual

is then the noise in the raw data with the seasonality removed as doing so exposes large anomalies which are possible outliers.

Once the eigenpairs are chosen, we proceed with the classical definition of the method. If I represents a set of indices corresponding to the eigenmodes to remove, we approximate the trajectory matrix

$$\boldsymbol{Xt} = \sum_{i \in I} \boldsymbol{X_i}$$

An approximation Yt to the original signal Y can be obtained from **Xt** by inverting the process used to form the trajectory matrix, Equation (1). Each column of **Xt** represents a shifted approximation to Yt, thus we average each shifted column. Finally the deseasonalized residual is the difference between the original signal and the reconstruction, $R = Y - Yt$.

We applied SSA for analysis of all five meteorological variables across all facilities (Table 7) to identify outliers in all meteorological observations.

Because SSA requires the time series data to be continuous, we corrected any missing values in the time series by replacing them with long term seasonality. We set $L = 400$ and isolated the signals corresponding to year and monthly frequency in the data. Thus $Yt = Yt[0] + Yt[1] + Yt[2]$. Figure 7 shows the result of SSA analysis for air temperature variable at facility E33. The first row of Figure 7 shows the raw daily time series (Yt) of air temperature, which shows no significant trend (orange line $Yt[0]$) at the site during period 2012 to 2017. The second and third rows show the annual ($Yt[1]$) and monthly ($Yt[2]$) frequencies of the temperature time series respectively. Temperature time series data shows strong annual and monthly frequencies at the sites which expected and reflective of long term weather patterns experienced at the SGP site. The last row shows the time series of residual after removing the tends, and annual and monthly

frequencies from the data. While some of the residuals may be reflective if natural variability, the anomalous positive or negative temperature residuals can be identified as outliers in the data. Multiple methods are available to set a threshold for extreme values in the residuals as outliers. We used the three sigma rule to extract outliers [125]. For example, the two peak points in Figure 7 are larger than three sigmas, thus are outliers.

K-means

Southern plains, where SGP site is located, are known to experience frequent extreme storms occurring most frequently during spring and early summer seasons. Identifying these extreme events is of interest for scientific users of the data to study and/or isolate these phenomena. However, meteorological variables during such events won't be captured by Pearson Correlation as they may still follow know correlation structure at seasonal scales or by SSA method since any individual variable may not show large deviation. Multivariate approach like K-means clustering have been widely used to identify weather and climate regimes [126, 127]. We used K-means clustering algorithm to delineate the weather regimes at SGP site. While extreme storms and weather events that often occur at sub-daily timescales may still fall within identified known weather regimes at the site, they often are out of norm extremes within the regime and of interest to us.

K-means is a partitioning clustering algorithm [65, 78]. It starts with user specified *k* centroids, and assigns the points to the nearest centroid. Then it computes new *k* centroids and assigns all data points to these centroids again. This process is repeated until convergence criteria is met.

Algorithm 1: K-means Outlier Detection

Input: *ARM time series data*

Output: *Outliers*

```
outliers ← ∅
df ← ARM time series data
data ← df[`atmos_pressure′,`temp_mean′,
`rh_mean′,`vapor_pressure_mean′,`wspd_arith_mean′]
number_of_clusters ← 4
clusters ← K − means(data, number_of_clusters)
distances ← Distance between each point and its centroid
mean ← arithmetic mean of distances
sigma ← standard deviation of distances
threshold ← mean + 3 ∗ sigma
For i in range(size of distances) do
        if distances[i] > threshold then
                outliers ← outliers ∪ distances[i]
        end
end
return outliers
```

We applied K-means clustering to ARM meteorological data set to defined weather regimes at SGP site. We then calculated the distance of each point within a cluster to its corresponding cluster centroid. Vector of distances within each cluster were used to identify points that are on fringes of the regime they belong to and considered outliers. All five meteorological variables were used in this analysis. Algorithm 1 describes the workflow.

Given known seasonal patterns at the site we set k to four to determine weather regimes for four seasons. Figure 8 shows the four regimes at facility E33 that representing spring (cluster 1), winter (cluster 2), summer (cluster 3) and fall (cluster 4). Data points within each weather regime (or cluster) that are at significant distance from their clusters (identified by red squares in Figure 8) were identified as outlier (and may correspond to extreme weather events).

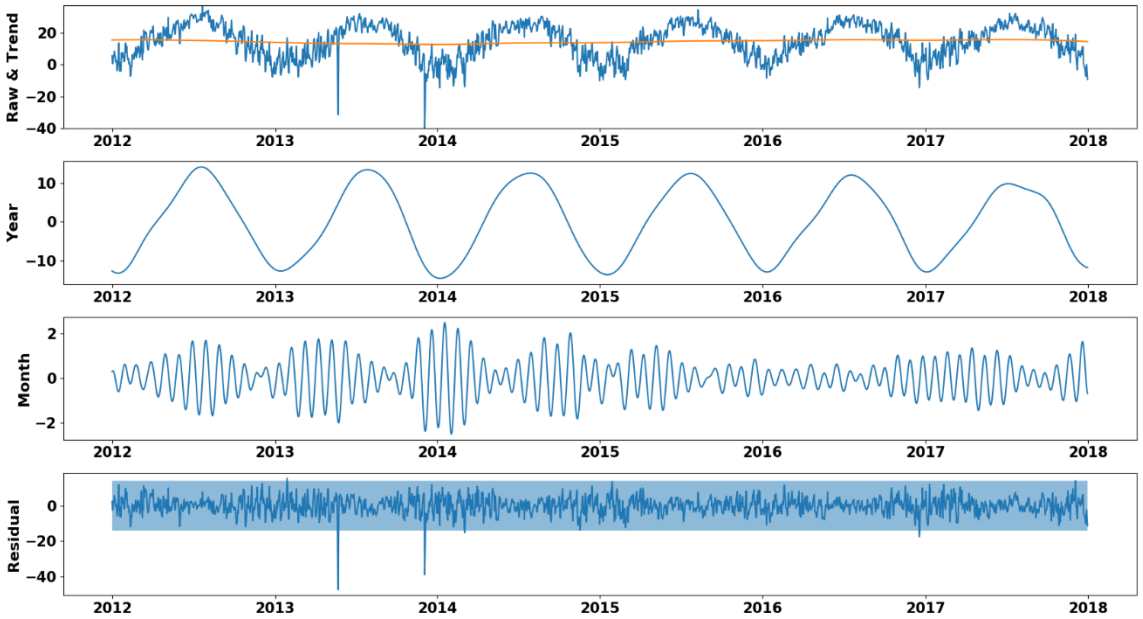

Figure 7. Decomposition of air temperature data from MET instrument at facility E33 using SSA method to isolate various frequencies.

Evaluation of Outlier Detection

ARM data quality assurance program maintains a database of outliers that has been identified, inspected and documented for all historical data. However, recorded data quality issues are added manually for historical data when an issue is identified or reported and are known to be incomplete [128]. A description of the outlier event is included in these Data Quality Reports (DQR) which often are temporary change in operating conditions such as power failures, frozen and snow-covered sensors, instrument degradation, or contamination. Most often extreme weather events are not captured and reported by the current system before. Each DQR entry also contains a specific time range affected, list of data projects, and specific measurements. And these entries are usually submitted by either the Data Quality Office [129] or the instrument mentor [130]. The DQRs are stored and available as PostgreSQL database (http://dq.arm.gov). During study period of 1994-2017, across 24 facilities studied at SGP site, a total of 181 DQRs were reported for MET variables analyzed, each often spanning multiple day time period totaling 8540 days. The reported data quality issues covered all five variables: air temperature (41 events; 8217 days), vapor pressure (42 events; 8194 days), atmospheric pressure (12 events; 76 days), relative humidity (32 events; 8108 days), and wind speed (52 events; 265 days). We evaluated outliers identified by methods developed in this study against the DQRs in the database through database queries and calculated *Precision* and *Recall* metrics [131]. We treated outliers detected in DQR database as True Positives. The equation 3 and 4 show the calculation of *Precision* and *Recall*.

$$\textbf{Precision} = \frac{\textbf{True Positives } (Outliers\ detected\ in\ DQR\ database)}{\textbf{True Positives } + \textbf{ False Positives } (Outliers\ detected\ not\ in\ DQR\ database)} \quad (3)$$

$$\textbf{Recall} = \frac{\textbf{True Positives } (Outliers\ detected\ in\ DQR\ database)}{\textbf{True Positives } + \textbf{ False Negatives } (Undetected\ not\ in\ DQR\ database)} \quad (4)$$

Results and Discussion

All three methods were applied to five meteorological variables across all facilities. The methods identified different sets of outlier events, with some events identified by more than one method (Figures 6,7,8).

Among three methods Pearson correlation was least effective with frequent false negatives. Pearson correlation is also an aggressive method that it may include many false positives. Those are all due to the fact that pairwise Pearson correlation method was applied at seasonal scale. Pearson correlation coefficient is a pairwise comparison method, however, if the two variables deviate in the same direction, their correlation may not change significantly and thus may go undetected. Due to seasonal nature of the analysis, it was not able to identify outliers that persisted at hours to days only. Univariate SSA method was very effective at identifying outliers with extreme high and low values in the time series but required the input data to be consistent with no missing values. K-means could be used to detect extreme storms and weather events, but it was hard to tell which variable mainly caused the abnormality. However, these drawbacks could be easily overcome by combining methods together to detect outliers from three different angles.

In our experiment, SSA method identified largest number of outlier events (922) (Table 8) across the entire dataset, while K-means identified 508 events. While 378 events were identified as outliers by both the methods (intersection), 674 events were only identified by one of the methods (Table 8). Figure 9 shows all the outliers detected by Pearson correlation, SSA and K-means methods at facility E33 for air temperature. We can see from Figure 9 that spring 2015 was treated as outlier season by Pearson correlation due to large temperature fluctuation due to spring frost event.

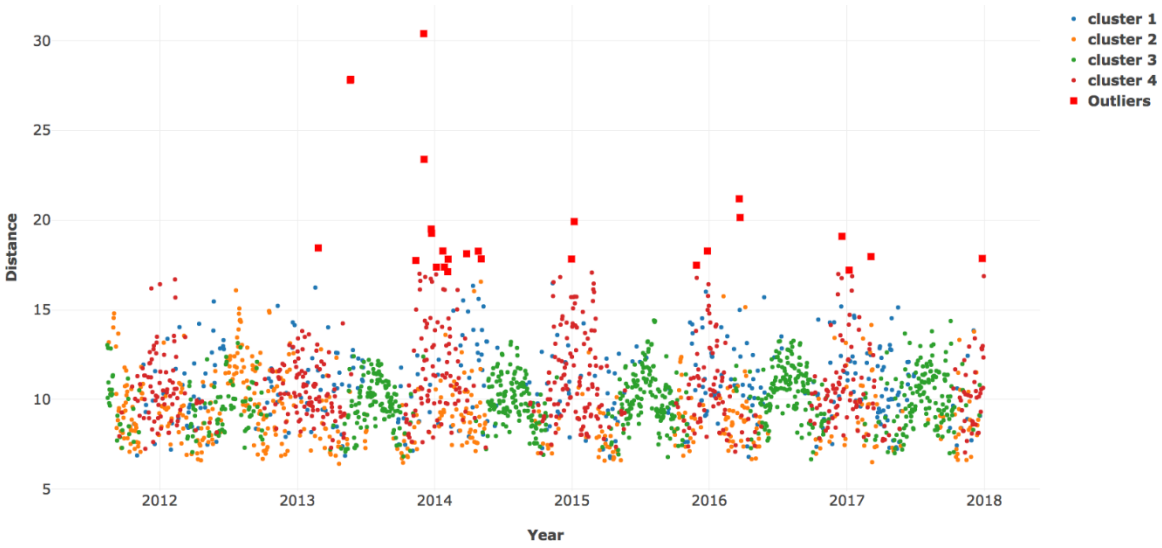

Figure 8. Outliers detected using K-means method at facility E33. X-axis represents the daily meteorological time series, colored by cluster (weather regime) they belong to, while Y-axis shows the distance of the data point from the centroid of its cluster (weather regime).

Table 8. Comparison of SSA and K-means Outlier Set Size.

	Outlier Set Size
SSA	922
K-means	508
Intersection	378
Symmetric Difference	674

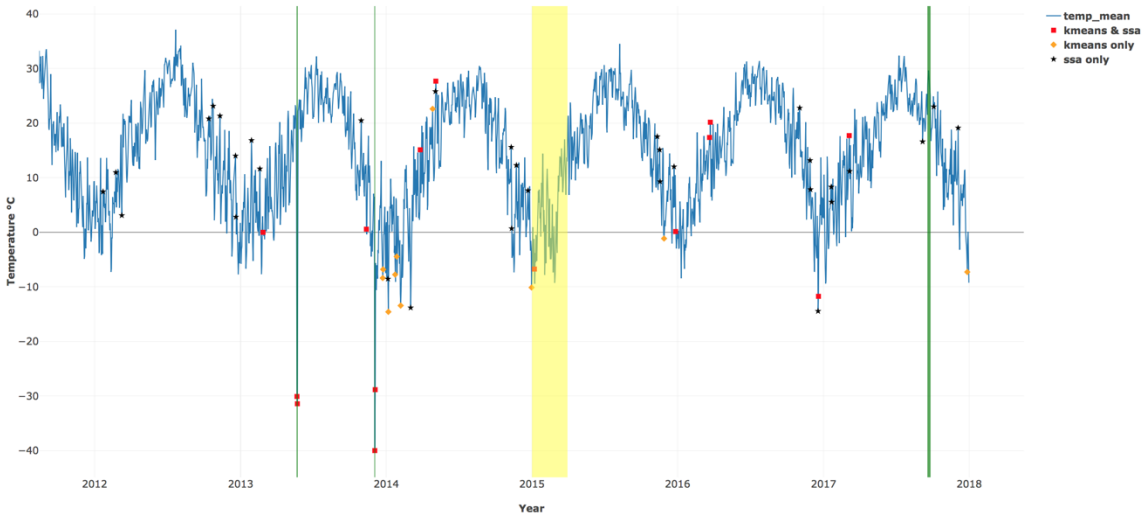

Figure 9. Outliers detected at facility E33 for air temperature by Pearson correlation, SSA and K-means algorithms. The yellow shaded areas are outliers detected by Pearson correlation. Outliers detected by both SSA and K-means algorithms are shown by red squares, while those identified by SSA and K-means only are indicated by black stars and orange diamonds respectively. DQR records are denoted by the vertical green shaded areas.

When using Pearson Correlation, we used the interquartile range (IQR) method to extract outlier seasons that is those values beyond Tukey's fences as the three sigma's rule is too aggressive for Pearson correlation [132]. However, since the Pearson Correlation was applied at seasonal scale it identified only a few outlier seasons in the data. For example, at facility E33 Pearson Correlation analysis of temperature time series identified spring 2015 that experienced a severe frost event as outlier season (Figure 9). When combined together SSA and K-means methods had *Precision* of 11.10% (Table 9) which shows that many of the outliers detected are not within ARM DQR database, which is a known limitation of the current records that this current study is trying to address. Detected outliers also had low *Recall* which in addition to small number of true positives can be due to fact that DQR database often records a wide affected date range for an identified outlier instead of a precise date thus leading to large false negatives, all of which leads to low *Recall* values.

Overall, when combined together within a framework, set of methods applied allows to capture outlier events caused by a wide range of conditions.

Table 9. Precision and Recall of SSA and K-means.

Method	Variable	Precision	Recall
SSA	Air Temperature	16.00%	1.20%
SSA	Vapor Pressure	20.70%	1.40%
SSA	Atmospheric Pressure	0.00%	0.00%
SSA	Relative Humidity	14.80%	0.50%
SSA	Wind Speed	0.60%	1.50%
K-means	All Variables	13.00%	1.90%
Combined	All Variables	11.10%	4.10%

Conclusions

In this chapter we tested pairwise Pearson correlation, univariate SSA and multivariate K-means based method for detection of outliers in the data at ARM meteorological observations at SGP site. Combining the approaches within a framework for streaming data within ARM provides a platform to detect outliers from a wide range of sensor failure scenarios to extreme events. While each of the methods developed and applied in this study has its strengths and limitations, our evaluation against existing database of data quality issue suggests that the framework is able to identify known outliers well. Although our current study focused on meteorological observations, it provides a framework for an efficient outlier detection of streaming datasets within ARM that can be extended to other classes of time series datasets not only tested MET data from SGP. In the future, we plan to analyze multiple classes of instruments like meteorological, radiometric, radar etc. simultaneously for improved detection of outliers. We also plan to develop multivariate SSA [133] and machine learning techniques to address this high dimensional problem in an operational data center environment.

The three algorithms and visualizations presented in this chapter were implemented in Python. All codes and results are available on GitHub (https://github.com/YupingLu/arm-pearson) and (https://github.com/YupingLu/arm-ssa).

Chapter 5
Conclusions

We have focused on a pair of topics central to big data analytics: algorithmic stability and data cleansing, developing new techniques and applying them to the analysis of transcriptomic data and meteorological data.

Summary of Contributions

Chapter 2 described robustness, a new metric to measure the stability of clustering algorithms over a range of different settings. The computation is straightforward: it is the proportion of clustering runs in which a pair of entities appears together in some cluster, given that they appear together in a cluster in at least one run, averaged over all such pairs. We found, somewhat surprisingly, that all four hierarchical clustering methods tested ranked at the top for robustness among all clustering algorithms tested. Further analysis showed that this is merely due to the rigid tree structure of such algorithms. Among non-hierarchical methods, the paraclique algorithm showed good performance in terms of robustness, in many cases matching or exceeding that of hierarchical methods. The robustness metric adds a useful tool for researchers to employ when selecting a clustering algorithm, since consistency of results is one of the cornerstones of good science.

We also investigated a method for choosing from among maximum cliques in unweighted graphs created by the thresholding of weighted graphs. The method can be applied to any unweighted graph created via thresholding, as long as the original edge weights are still available. Empirical tests on yeast transcriptomic data show that the method tends to produce clusters with improved enrichment p-values. Insofar as GO enrichment is a reliable surrogate for cluster quality, we conclude that the method tends to generate clusters of higher quality. This method provides an empirically tested means to select a maximum clique in graph-based clustering algorithms.

Lastly, we applied pairwise Pearson correlation coefficient, univariate SSA, and multivariate K-means to form a framework for outlier detection in ARM meteorological data collected from the SGP site. Although individually each method has its own limitations and drawbacks, the combined framework was found to be a highly efficient tool to filter out a wide range of sensor failures and extreme events. In experiments the framework demonstrated improved detection compared to the current manual method of flagging outliers in a database, both saving time and increasing accuracy.

Future Research Directions

In Chapter 2, we limited the scope of our tests of the robustness metric to non-overlapping clusters extracted from transcriptomic data from five species. In principle, the metric can be extended to overlapping clustering algorithms as well, so a comparison of such algorithms may be of interest. And naturally, determining whether the current results hold on other types of biological data and data from other domains such as communications, transportation and social networks would be of interest.

In Chapter 3, we only compared the first level paracliques in each graph. Future work might entail testing whether the selection strategies have the same effect on the second or deeper level paracliques. Other selection strategies are possible, too. For instance, instead of restricting the choice to maximum cliques, one might choose the (not necessarily maximum) clique with the highest total edge weight. Or, if the maximum clique has c vertices, one might choose a set of c vertices in the unweighted graph with maximum total edge weight to use as a paraclique core. We also observed a slight positive association between maximum clique weight and enrichment score of a resultant paraclique. The effect size appears so small, however, that it will likely require a much larger sample size to

confirm, using many more than the three maximum cliques per graph used in this work. Further analyses on larger and more diverse data sets may also reveal greater improvements based on the use of the additional information that the weights provide.

In Chapter 4, our framework works on meteorological data from the SGP site. It could be expected to incorporate data collected from other sites, and even other types of ARM data, for example, weather radar data and satellite observation data. Multivariate SSA methods and machine learning could also be explored in an effort to detect outliers more effectively.

References

[1] M. Kataria and M. P. Mittal, "Big data: A review," *International Journal of Computer Science and Mobile Computing,* vol. 3, no. 7, pp. 106-110, 2014.

[2] M. Cox and D. Ellsworth, "Application-controlled demand paging for out-of-core visualization," in *Proceedings. Visualization'97 (Cat. No. 97CB36155)*, 1997: IEEE, pp. 235-244.

[3] F. X. Diebold, "Big data dynamic factor models for macroeconomic measurement and forecasting," in *Advances in Economics and Econometrics: Theory and Applications, Eighth World Congress of the Econometric Society,"(edited by M. Dewatripont, LP Hansen and S. Turnovsky)*, 2003, pp. 115-122.

[4] D. Laney, "3D data management: Controlling data volume, velocity and variety," *META group research note,* vol. 6, no. 70, p. 1, 2001.

[5] S. Madden, "From databases to big data," *IEEE Internet Computing,* vol. 16, no. 3, pp. 4-6, 2012.

[6] J. Fan, F. Han, and H. Liu, "Challenges of big data analysis," *National science review,* vol. 1, no. 2, pp. 293-314, 2014.

[7] S. Sagiroglu and D. Sinanc, "Big data: A review," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*, 2013: IEEE, pp. 42-47.

[8] P. Zikopoulos and C. Eaton, *Understanding big data: Analytics for enterprise class hadoop and streaming data*. McGraw-Hill Osborne Media, 2011.

[9] J. Manyika *et al.*, "Big data: The next frontier for innovation, competition, and productivity," 2011.

[10] V. Borkar, M. J. Carey, and C. Li, "Inside Big Data management: ogres, onions, or parfaits?," in *Proceedings of the 15th international conference on extending database technology*, 2012: ACM, pp. 3-14.

[11] W. Raghupathi and V. Raghupathi, "Big data analytics in healthcare: promise and potential," *Health information science and systems,* vol. 2, no. 1, p. 3, 2014.

[12] L. D. Stein, "The case for cloud computing in genome informatics," *Genome biology,* vol. 11, no. 5, p. 207, 2010.

[13] NOAA, "NEXRAD on AWS." [Online]. Available: https://registry.opendata.aws/noaa-nexrad/

[14] J. Zote, "65 Social Media Statistics to Bookmark in 2019." [Online]. Available: https://sproutsocial.com/insights/social-media-statistics/

[15] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM,* vol. 51, no. 1, pp. 107-113, 2008.

[16] E. Begoli and J. Horey, "Design principles for effective knowledge discovery from big data," in *2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture*, 2012: IEEE, pp. 215-218.

[17] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *MSST*, 2010, vol. 10, pp. 1-10.

[18] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," *HotCloud,* vol. 10, no. 10-10, p. 95, 2010.

[19] "Top500 supercomputer sites." [Online]. Available: https://www.top500.org/

[20] A. Lakshman and P. Malik, "Cassandra: a decentralized structured storage system," *ACM SIGOPS Operating Systems Review,* vol. 44, no. 2, pp. 35-40, 2010.

[21] A. D. JoSEP, R. KAtz, A. KonWinSKi, L. Gunho, D. PAttERSon, and A. RABKin, "A view of cloud computing," *Communications of the ACM,* vol. 53, no. 4, 2010.

[22] N. Mehta and A. Pandit, "Concurrence of big data analytics and healthcare: A systematic review," *International journal of medical informatics,* vol. 114, pp. 57-65, 2018.

[23] C. P. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," *Information sciences,* vol. 275, pp. 314-347, 2014.

[24] M. U. Nisar, A. Fard, and J. A. Miller, "Techniques for graph analytics on big data," in *2013 IEEE International Congress on Big Data*, 2013: IEEE, pp. 255-262.

[25] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz, "Recent advances in graph partitioning," in *Algorithm Engineering*: Springer, 2016, pp. 117-158.

[26] R. Weiss and L.-J. Zgorski, "Obama administration unveils "big data" initiative: Announces $200 million in new R&D investments," *Office of Science and Technology Policy Executive Office of the President,* 2012.

[27] R. Rein and D. Memmert, "Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science," *SpringerPlus,* vol. 5, no. 1, p. 1410, 2016.

[28] C. A. Phillips, "Multipartite graph algorithms for the analysis of heterogeneous data," 2015.

[29] NCBI. "Gene Expression Omnibus." https://www.ncbi.nlm.nih.gov/geo/ (accessed.

[30] G. M. Stokes and S. E. Schwartz, "The Atmospheric Radiation Measurement (ARM) program: Programmatic background and design of the cloud and radiation test bed," *Bulletin of the American Meteorological Society,* vol. 75, no. 7, pp. 1201-1222, 1994.

[31] M. R. Garey and D. S. Johnson, "Computers and intractability: A guide to the theory of *NP*-Completeness," W. H. Freeman and Company, 1979, pp. 1-340.

[32] C. Bron and J. Kerbosch, "Algorithm 457: finding all cliques of an undirected graph," *Proceedings of the ACM,* vol. 16(9), pp. 575-577, 1973.

[33] E. Tomita, A. Tanaka, and H. Takahashi, "The worst-case time complexity for generating all maximal cliques and computational experiments," *Theoretical Computer Science,* vol. 363, pp. 28-42, 2006.

[34] J. D. Eblen, C. A. Phillips, G. L. Rogers, and M. A. Langston, "The maximum clique enumeration problem: algorithms, applications, and implementations," *BMC Bioinformatics,* vol. 13 Suppl 10, p. S5, Jun 25 2012, doi: 10.1186/1471-2105-13-S10-S5.

[35] E. J. Chesler and M. A. Langston, "Combinatorial genetic regulatory network analysis tools for high throughput transcriptomic data," in *Systems Biology and Regulatory Genomics*, vol. 4023, E. Eskin Ed.: Springer, 2006, pp. 150–165.

[36] K. Pearson, "Notes on Regression and Inheritance in the Case of Two Parents Proceedings of the Royal Society of London, 58, 240-242," ed, 1895.

[37] A. D. Well and J. L. Myers, *Research design & statistical analysis*. Psychology Press, 2003.

[38] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal,* vol. 27, no. 3, pp. 379-423, 1948.

[39] J. Jay *et al.*, "A systematic comparison of genome-scale clustering algorithms," *BMC Bioinformatics,* vol. 13, no. Suppl 10, p. S7, 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/S10/S7.

[40] A. Singhal, "Modern information retrieval: A brief overview," *IEEE Data Eng. Bull.,* vol. 24, no. 4, pp. 35-43, 2001.

[41] D. W. Goodall, "A new similarity index based on probability," *Biometrics,* pp. 882-907, 1966.

[42] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics,* pp. 857-871, 1971.

[43] B. H. Voy *et al.*, "Extracting gene networks for low-dose radiation using graph theoretical algorithms," *PLoS computational biology,* vol. 2, no. 7, p. e89, 2006.

[44] A. D. Perkins and M. A. Langston, "Threshold selection in gene co-expression networks using spectral graph theory techniques," *BMC Bioinformatics,* vol. 10, 2009.

[45] G. Chen, S. A. Jaradat, N. Banerjee, T. S. Tanaka, M. S. H. Ko, and M. Q. Zhang, "Evaluation and comparison of clustering algorithms in analyzing ES cell gene expression data," *Statistica Sinica,* vol. 12, no. 1, pp. 241-262, 2002. [Online]. Available: http://www.jstor.org/stable/24307044.

[46] S. Datta and S. Datta, "Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes," *BMC Bioinformatics,* vol. 7, p. 397, Aug 31 2006, doi: 10.1186/1471-2105-7-397.

[47] M. E. J. Newman, "Modularity and community structure in networks," *P Natl Acad Sci USA,* vol. 103, no. 23, pp. 8577-8582, Jun 6 2006, doi: 10.1073/pnas.0601602103.

[48] R. D. Luce and A. D. Perry, "A method of matrix analysis of group structure," *Psychometrika,* vol. 14, no. 2, pp. 95-116, Jun 1949. [Online]. Available: <Go to ISI>://WOS:000203877900003.

[49] S. Wasserman and K. Faust, *Social network analysis : methods and applications* (Structural Analysis in the Social Sciences, no. 8). Cambridge ; New York: Cambridge University Press, 1994, pp. xxxi, 825 p.

[50] P. J. Rousseeuw, "Silhouettes - a graphical aid to the interpretation and validation of cluster-analysis," *J Comput Appl Math,* vol. 20, pp. 53-65, Nov 1987, doi: Doi 10.1016/0377-0427(87)90125-7.

[51] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance," *J Mach Learn Res,* vol. 11, pp. 2837-2854, Oct 2010. [Online]. Available: <Go to ISI>://WOS:000284040000008.

[52] A. Subramanian *et al.*, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," *P Natl Acad Sci USA,* vol. 102, no. 43, pp. 15545-15550, Oct 25 2005, doi: 10.1073/pnas.0506580102.

[53] D. W. Huang *et al.*, "DAVID bioinformatics resources: expanded annotation database and novel algorithms to better extract biology from large gene lists," *Nucleic Acids Res,* vol. 35, pp. W169-W175, Jul 2007, doi: 10.1093/nar/gkm415.

[54] M. Ashburner *et al.*, "Gene ontology: tool for the unification of biology," *Nat Genet,* vol. 25, pp. 25–29, 2000.

[55] G. Antonazzo *et al.*, *Expansion of the Gene Ontology knowledgebase and resources*. 2017.

[56] G. K. D. d. Vries, W. R. v. Hage, and M. v. Someren, "Comparing vessel trajectories using geographical domain knowledge and alignments," in *2010 IEEE International Conference on Data Mining Workshops*, 13-13 Dec. 2010 2010, pp. 209-216, doi: 10.1109/ICDMW.2010.123.

[57] M. Liu and A. Samal, "Cluster validation using legacy delineations," *Image and Vision Computing,* vol. 20, no. 7, pp. 459-467, 2002/05/01/ 2002, doi: http://dx.doi.org/10.1016/S0262-8856(01)00089-0.

[58] W. M. Rand, "Objective criteria for evaluation of clustering methods," *J Am Stat Assoc,* vol. 66, no. 336, pp. 846-850, 1971, doi: Doi 10.2307/2284239.

[59] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Math Program,* vol. 79, no. 1-3, pp. 191-215, Oct 1 1997, doi: Doi 10.1007/Bf02614317.

[60] L. Hubert, "Min and max hierarchical clustering using asymmetric similarity measures," *Psychometrika,* vol. 38, no. 1, pp. 63-72, 1973, doi: Doi 10.1007/Bf02291174.

[61] A. Rosenberg and J. Hirschberg, "V-measure: a conditional entropy-based external cluster evaluation measure," presented at the Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning(EMNLP-CoNLL), 2007.

[62] R. Edgar, M. Domrachev, and A. E. Lash, "Gene expression omnibus: NCBI gene expression and hybridization array data repository," *Nucleic Acids Res,* vol. 30, no. 1, pp. 207-210, Jan 1 2002, doi: DOI 10.1093/nar/30.1.207.

[63] C. Huttenhower *et al.*, "Nearest neighbor networks: clustering expression data based on gene neighborhoods," *BMC Bioinformatics,* vol. 8, no. 1, pp. 250-250, 2007. [Online]. Available: http://www.biomedcentral.com/1471-2105/8/250.

[64] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika,* vol. 32, no. 3, pp. 241-54, Sep 1967. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/5234703.

[65] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *Applied Statistics,* vol. 28, pp. 100-108, 1979.

[66] L. J. Heyer, S. Kruglyak, and S. Yooseph, "Exploring expression data: identification and analysis of coexpressed genes," *Genome Research,* vol. 9, pp. 1106-1115, 1999.

[67] R. D. Hagan, M. A. Langston, and K. Wang, "Lower bounds on paraclique density," *Discrete Applied Mathematics,* vol. 204, pp. 208-212, 5/11/ 2016, doi: http://dx.doi.org/10.1016/j.dam.2015.11.010.

[68] R. Sharan, A. Maron-Katz, and R. Shamir, "CLICK and EXPANDER: a system for clustering and visualizing gene expression data," *Bioinformatics,* vol. 19, pp. 1787-1799, 2003.

[69] B. Zhang and S. Horvath, "A general framework for weighted gene co-expression network analysis," *Stat Appl Genet Mol Biol,* vol. 4, p. Article17, 2005, doi: 10.2202/1544-6115.1128.

[70] P. Tamayo *et al.*, "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation," *P Natl Acad Sci USA,* vol. 96, no. 6, 1999. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=15868.

[71] H. K. Seifoddini, "Single linkage versus average linkage clustering in machine cells formation applications," *Comput Ind Eng,* vol. 16, no. 3, pp. 419-426, 1989, doi: Doi 10.1016/0360-8352(89)90160-5.

[72] P. Dawyndt, H. De Meyer, and B. De Baets, "The complete linkage clustering algorithm revisited," *Soft Comput,* vol. 9, no. 5, pp. 385-392, May 2005, doi: 10.1007/s00500-003-0346-3.

[73] L. L. McQuitty, "Similarity analysis by reciprocal pairs for discrete and continuous data," *Educational and Psychological Measurement,* vol. 26, pp. 825-831, 1966.

[74] J. H. Ward, "Hierarchical grouping to optimize an objective function," *J Am Stat Assoc,* vol. 58, pp. 236-244, 1963.

[75] E. Willems *et al.*, "Differential expression of genes and DNA methylation associated with prenatal protein undernutrition by albumen removal in an avian model," *Sci Rep,* vol. 6, p. 20837, Feb 10 2016, doi: 10.1038/srep20837.

[76] I. Herrer *et al.*, "Gene expression network analysis reveals new transcriptional regulators as novel factors in human ischemic cardiomyopathy," *BMC Med Genomics,* vol. 8, p. 14, Mar 29 2015, doi: 10.1186/s12920-015-0088-y.

[77] R. C. Venu *et al.*, "Deep and comparative transcriptome analysis of rice plants infested by the beet armyworm (spodoptera exigua) and water weevil (lissorhoptrus oryzophilus)," *Rice,* vol. 3, no. 1, pp. 22-35, 2010/03/01 2010, doi: 10.1007/s12284-010-9037-8.

[78] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Berkeley, Calif., 1967: University of California Press, in Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297. [Online]. Available: http://projecteuclid.org/euclid.bsmsp/1200512992. [Online]. Available: http://projecteuclid.org/euclid.bsmsp/1200512992

[79] *R: a language and environment for statistical computing*. (2017). R Foundation for Statistical Computing, Vienna, Austria. [Online]. Available: http://www.R-project.org

[80] A. P. Gasch, M. Huang, S. Metzner, D. Botstein, S. J. Elledge, and P. O. Brown, "Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog Mec1p," *Mol Biol Cell,* vol. 12, no. 10, 2001. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=60150.

[81] A. Baratloo, M. Hosseini, A. Negida, and G. El Ashal, "Part 1: Simple definition and calculation of accuracy, sensitivity and specificity," *Emerg (Tehran),* vol. 3, no. 2, pp. 48-9, Spring 2015. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/26495380.

[82] E. Alm and A. P. Arkin, "Biological networks," *Curr Opin Struc Biol,* vol. 13, no. 2, pp. 193-202, Apr 2003, doi: 10.1016/S0959-440x(03)00031-9.

[83] A. J. M. Walhout, "Gene-centered regulatory network mapping," *Method Cell Biol,* vol. 106, pp. 271-288, 2011, doi: 10.1016/B978-0-12-544172-8.00010-4.

[84] J. M. Stuart, E. Segal, D. Koller, and S. K. Kim, "A gene-coexpression network for global discovery of conserved genetic modules," *Science,* vol. 302 (5643), pp. 249-255, 2003.

[85] C. J. Wolfe, I. S. Kohane, and A. J. Butte, "Systematic survey reveals general applicability of "guilt-by-association" within gene coexpression networks," *BMC Bioinformatics,* vol. 6, no. 227, 2005.

[86] A. J. Butte, P. Tamayo, D. Slonim, T. R. Golub, and I. S. Kohane, "Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks," *Proc Natl Acad Sci U S A,* vol. 97, no. 22, pp. 12182-6, Oct 24 2000. [Online]. Available: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=11027309

[87] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature,* vol. 435, no. 7043, pp. 814-818, 2005.

[88] P. D. Juarez *et al.*, "A novel approach to analyzing lung cancer mortality disparities: Using the exposome and a graph theoretical toolchain," *Environmental Disease,* vol. 2, pp. 33-44, 2017.

[89] M. A. Langston *et al.*, "Scalable combinatorial tools for health disparities research," *International Journal of Environmental Research and Public Health,* vol. 11, no. 10, pp. 10419-10443, 2014.

[90] M. A. Langston, A. D. Perkins, A. M. Saxton, J. A. Scharff, and B. H. Voy, "Innovative computational methods for transcriptomic data analysis: A case study in the use of FPT for practical algorithm design and implementation," *The Computer Journal,* vol. 51, pp. 26-38, 2008.

[91] A. Schoenrock *et al.*, "Efficient prediction of human protein-protein interactions at a global scale," *BMC Bioinformatics,* vol. 15, no. 383, pp. DOI: 10.1186/s12859-014-0383-1, 2014.

[92] D. Macartney-Coxson, M. C. Benton, R. Blick, R. S. Stubbs, R. D. Hagan, and M. A. Langston, "Genome-wide DNAMethylation analysis reveals Loci that distinguish different types of adipose tissue in obese individuals," *Clinical Epigenetics,* vol. 9, no. 48, pp. DOI 10.1186/s13148-017-0344-4, 2017.

[93] C. E. Nestor *et al.*, "DNA methylation changes separate allergic patients from healthy controls and may reflect altered CD4+ T-cell population structure," *PLoS Genetics,* vol. 10, p. e1004059, 2014.

[94] J. D. Eblen, I. C. Gerling, A. M. Saxton, J. Wu, J. R. Snoddy, and M. A. Langston, "Graph algorithms for integrated biological analysis, with applications to type 1 diabetes data," in *Clustering Challenges in Biological Networks*, W. A. Chaovalitwongse Ed.: World Scientific, 2009, pp. 207-222.

[95] S. Bruhn *et al.*, "Increased expression of IRF4 and ETS1 in CD4+ cells from patients with intermittent allergic Rhinitis," *Allergy,* vol. 67, pp. 33-40, 2012.

[96] L. S. Gittner, B. J. Kilbourne, R. Vadapalli, H. M. K. Khan, and M. A. Langston, "A multifactorial obesity model developed from nationwide public health exposome data and modern computational analyses," *Obesity Research & Clinical Practice,* vol. 11, no. 5, pp. 522-533, 2017.

[97] O. M. Peck-Palmer, G. Clermont, G. L. Rogers, S. Yende, D. C. Angus, and M. A. Langston, "Graph theoretical analysis of Genome-scale data: Examination of gene activation occurring in the setting of community-acquired Pneumonia," *Shock: Injury, Inflammation, and Sepsis: Laboratory and Clinical Approaches,* vol. 50, pp. 53-59, 2018.

[98] B. H. Voy *et al.*, "Extracting gene networks for low dose radiation using graph theoretical algorithms," *PLoS Computational Biology,* vol. 2, no. 7, p. e89, 2006.

[99] T. Barrett *et al.*, "NCBI GEO: archive for functional genomics data sets-update," *Nucleic Acids Res,* vol. 41, no. D1, pp. D991-D995, Jan 2013, doi: 10.1093/nar/gks1193.

[100] O. R. N. Laboratory. "CADES – Compute and Data Environment for Science " https://www.olcf.ornl.gov/olcf-resources/rd-project/cades-compute-and-data-environment-for-science/ (accessed.

[101] ARM, "ARM Research Facility." [Online]. Available: https://www.arm.gov/

[102] K. Gaustad *et al.*, "A scientific data processing framework for time series NetCDF data," *Environmental modelling & software,* vol. 60, pp. 241-249, 2014.

[103] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, "Outlier detection for temporal data: A survey," *IEEE Transactions on Knowledge and data Engineering,* vol. 26, no. 9, pp. 2250-2267, 2013.

[104] A. J. Fox, "Outliers in time series," *Journal of the Royal Statistical Society: Series B (Methodological),* vol. 34, no. 3, pp. 350-363, 1972.

[105] S. Budalakoti, A. N. Srivastava, and M. E. Otey, "Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 39, no. 1, pp. 101-113, 2008.

[106] V. Chandola, V. Mithal, and V. Kumar, "Comparative evaluation of anomaly detection techniques for sequence data," in *2008 Eighth IEEE international conference on data mining*, 2008: IEEE, pp. 743-748.

[107] K. Sequeira and M. Zaki, "ADMIT: anomaly-based data mining for intrusions," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002: ACM, pp. 386-395.

[108] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion detection with unlabeled data using clustering," in *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001*, 2001: Citeseer.

[109] M. Gupta, A. B. Sharma, H. Chen, and G. Jiang, "Context-aware time series anomaly detection for complex systems," in *Workshop Notes*, 2013, vol. 14.

[110] F. A. González and D. Dasgupta, "Anomaly detection using real-valued negative selection," *Genetic Programming and Evolvable Machines,* vol. 4, no. 4, pp. 383-403, 2003.

[111] D. Cheboli, "Anomaly detection of time series." [Online]. Available: http://hdl.handle.net/11299/92985

[112] N. Golyandina and A. Zhigljavsky, *Singular Spectrum Analysis for time series*. Springer Science & Business Media, 2013.

[113] A. Kut and D. Birant, "Spatio-temporal outlier detection in large databases," *Journal of computing and information technology,* vol. 14, no. 4, pp. 291-297, 2006.

[114] D. J. Hill, B. S. Minsker, and E. Amir, "Real-time Bayesian anomaly detection for environmental sensor data," in *Proceedings of the Congress-International Association for Hydraulic Research*, 2007, vol. 32, no. 2: Citeseer, p. 503.

[115] D. J. Hill and B. S. Minsker, "Anomaly detection in streaming environmental sensor data: A data-driven modeling approach," *Environmental Modelling & Software,* vol. 25, no. 9, pp. 1014-1022, 2010.

[116] W. Drosdowsky, "An analysis of Australian seasonal rainfall anomalies: 1950–1987. II: Temporal variability and teleconnection patterns," *International Journal of Climatology,* vol. 13, no. 2, pp. 111-149, 1993.

[117] E. Wu, W. Liu, and S. Chawla, "Spatio-temporal outlier detection in precipitation data," in *International Workshop on Knowledge Discovery from Sensor Data*, 2008: Springer, pp. 115-133.

[118] S. Yuxiang, X. Kunqing, M. Xiujun, J. Xingxing, P. Wen, and G. Xiaoping, "Detecting spatio-temporal outliers in climate dataset: A method study," in *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS'05.*, 2005, vol. 2: IEEE, p. 4 pp.

[119] R. Rew and G. Davis, "NetCDF: an interface for scientific data access," *IEEE computer graphics and applications,* vol. 10, no. 4, pp. 76-82, 1990.

[120] Unidata, "Network Common Data Form (NetCDF) version 4.1.1." [Online]. Available: https://doi.org/10.5065/D6H70CW6

[121] N. Golyandina and A. Korobeynikov, "Basic singular spectrum analysis and forecasting with R," *Computational Statistics & Data Analysis,* vol. 71, pp. 934-954, 2014.

[122] E. Bozzo, R. Carniel, and D. Fasino, "Relationship between Singular Spectrum Analysis and Fourier analysis: Theory and application to the monitoring of volcanic activity," *Computers & Mathematics with Applications,* vol. 60, no. 3, pp. 812-820, 2010.

[123] T. Alexandrov, "A method of trend extraction using singular spectrum analysis," *arXiv preprint arXiv:0804.3367,* 2008.

[124] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of computation,* vol. 19, no. 90, pp. 297-301, 1965.

[125] F. Pukelsheim, "The three sigma rule," *The American Statistician,* vol. 48, no. 2, pp. 88-91, 1994.

[126] F. M. Hoffman, W. W. Hargrove Jr, D. J. Erickson III, and R. J. Oglesby, "Using clustered climate regimes to analyze and compare predictions from fully coupled general circulation models," *Earth Interactions,* vol. 9, no. 10, pp. 1-27, 2005.

[127] W. W. Hargrove and F. M. Hoffman, "Potential of multivariate quantitative methods for delineation and visualization of ecoregions," *Environmental management,* vol. 34, no. 1, pp. S39-S60, 2004.

[128] R. McCord and J. Voyles, "The ARM data system and archive," *Meteorological Monographs,* vol. 57, pp. 11.1-11.15, 2016.

[129] R. A. Peppler, K. E. Kehoe, J. W. Monroe, A. K. Theisen, and S. T. Moore, "The ARM data quality program," *Meteorological Monographs,* vol. 57, pp. 12.1-12.14, 2016.

[130] T. S. Cress and D. L. Sisterson, "Deploying the ARM sites and supporting infrastructure," *Meteorological Monographs,* vol. 57, pp. 5.1-5.15, 2016.

[131] J. W. Perry, K. Allen, and M. M. Berry, "Machine literature searching x. machine language; factors underlying its design and development," *American Documentation (pre-1986),* vol. 6, no. 4, p. 242, 1955.

[132] J. W. Tukey, *Exploratory data analysis*. Reading, Mass., 1977.

[133] P. C. Rodrigues and R. Mahmoudvand, "The benefits of multivariate singular spectrum analysis over the univariate version," *Journal of the Franklin Institute,* vol. 355, no. 1, pp. 544-564, 2018.

Vita

Yuping (Allan) Lu was born and raised in Taizhou, Jiangsu, China. After graduating from Jiangyan High School, he attended Nanjing Agricultural University in Nanjing, Jiangsu, China, where he received a Bachelor of Engineering degree in 2011. He then joined Whale Cloud as an Implementation Engineer for a year. In 2013, Yuping started his doctoral study at the University of Tennessee majoring in Computer Science after taking a gap year break. He is currently studying under the direction of Dr. Michael A. Langston. During his Ph.D. study, he worked as a graduate research assistant at Dr Langston's lab from August 2013 to June 2014, and then at the Office of Information Technology of University of Tennessee from July 2014 to October 2017. He also did two summer internships at the Scientific Data Group of Oak Ridge National Laboratory in 2016 and 2017 respectively, where his mentor was Dr. George Ostrouchov. He is currently a graduate research assistant at the ARM Data Science and Integration group of Oak Ridge National Laboratory under the supervision of Dr. Jitendra Kumar starting October 2017.