Synchronization Inspired Data Mining

Junming Shao

München 2011

Synchronization Inspired Data Mining

Junming Shao

Dissertation

an der Fakultät für Mathematik, Informatik und Statistik

der Ludwig–Maximilians–Universität

München

vorgelegt von

Junming Shao

aus Neijiang, China

München, den August 5, 2011

Erstgutachter: Prof. Dr. Christian Böhm

Zweitgutachter: Prof. Dr. Ira Assent

Tag der mündlichen Prüfung: November 10, 2011

Contents

List of Figures

List of Tables

Acknowledgement

Three years of PhD study in the University of Munich is a precious asset in my life. I would like to take this opportunity to express my deep and sincere appreciation to all the people who supported and motivated me during my study and my life in Germany.

First and foremost, I would specially like to express my sincere gratitude to my supervisor Prof. Dr. Christian Böhm for his expert supervision, invaluable advice, endless support, and opportunities. Your research style, expert supervision, conscientiousness and friendship encouraged me to further develop as a scientific researcher. Many thanks are due to Dr. Claudia Plant for her excellent guidance, pleasant cooperation, and friendly encouragement. Furthermore, I warmly thank Prof. Dr. Ira Assent and Prof. Dr. Claudia Linnhoff-Popien, for their interest in my work and their willingness to act as the second and third referee on this thesis.

My warmest thanks also go to all my current and past colleagues at the data mining group. In particular, I want to thank Annahita Oswald, Bianca Wackersreuther, Jing Feng, Xiao He, Nikola Müller, Katrin Haegler, Bettina Konte, Frank Fiedler, Peter Wackersreuther, Michael Plavinski and Son Mai Thai for many inspiring and encouraging discussions and long help and support. In this group I found a very positive and supportive environment giving me much freedom for my research.

I am also very grateful to the scientists in the group of Neuroscience of Technical University of Munich, especially for Dr. Afra Wohlschläger, Dr. Christian Sorg, Dr. Velentin Riedl, Nickolas Myers, Andrew Zherdin. Thanks a lot for the pleasant discussion, cooperation and friendship. They are enthusiastic and attentive, and I really enjoy working with them.

Many thanks are due to Dr. Klaus Hahn, for his expert guidance and constructive comments. His serious attitude, patience and passion on research questions inspired me and he would be a good model for me.

I would wish to acknowledge the China Scholarship Council and The University of Munich for providing me the financial support (CSC-LMU joint Scholarship) during my PhD study.

I am very thankful to many of my friends in Munich. I am particularly indebted to Wen, Liqun, Tianfan, Huanhuan, Wei, for bringing me heartwarming care, supports and happiness.

Finally, special thanks to my parents and my wife Qinli, who are always my powerful backing with their endless love, understanding and support. Without their support, this thesis would not have seen its conclusion.

Abstract

Advances of modern technologies produce huge amounts of data in various fields, increasing the need for efficient and effective data mining tools to uncover the information contained implicitly in the data. This thesis mainly aims to propose innovative and solid algorithms for data mining from a novel perspective: *synchronization*.

Synchronization is a prevalent phenomenon in nature that a group of events spontaneously come into co-occurrence with a common rhythm through mutual interactions. The mechanism of synchronization allows controlling of complex processes by simple operations based on interactions between objects.

The first main part of this thesis focuses on developing the innovative algorithms for data mining. Inspired by the concept of synchronization, this thesis presents *Sync* (Clustering by Synchronization), a novel approach to clustering. In combination with the Minimum Description Length principle (MDL), it allows discovering the intrinsic clusters without any data distribution assumptions and parameters setting. In addition, relying on the different dynamic behaviors of objects during the process towards synchronization, the algorithm *SOD* (Synchronization-based Outlier Detection) is further proposed. The outlier objects can be naturally flagged by the definition of Local Synchronization Factor (LSF). To cure the *curse of dimensionality* in clustering, a subspace clustering algorithm *ORSC* is introduced which automatically detects clusters in subspaces of the original feature space. This approach proposes a weighted local interaction model to ensure all objects in a common cluster, which accommodate in arbitrarily oriented subspace, naturally move together. In order to reveal the underlying patterns in graphs, a graph partitioning approach *RSGC* (Robust Synchronization-based Graph Clustering) is presented. The key philosophy of *RSGC* is to consider graph clustering as a dynamic process towards synchronization. Inherited from the powerful concept of synchronization, *RSGC* shows several desirable properties that don't exist in other competitive methods. For all presented algorithms, their efficiency and effectiveness are thoroughly analyzed. The benefits over traditional approaches are further demonstrated by evaluating them on synthetic as well as real-world data sets.

Not only the theory research on novel data mining algorithms, the second main part of the thesis focuses on brain network analysis based on Diffusion Tensor Images (DTI). A new framework for automated white matter tracts clustering is first proposed to identify the meaningful fiber bundles in the Human Brain by combining ideas from time series mining with density-based clustering. Subsequently, the enhancement and variation of this approach is discussed allowing for a more robust, efficient, or effective way to find hierarchies of fiber bundles. Based on the structural connectivity network, an automated prediction framework is proposed to analyze and understand the abnormal patterns in patients of Alzheimer's Disease.

Zusammenfassung

Durch den technologischen Fortschritt werden in verschiedenen Anwendungen riesige Datenmengen gewonnen. Daher gibt es einen steigenden Bedarf an effektiven und effizienten Data-Mining-Techniken um aus den Daten die implizit enthaltenen Information zu extrahieren. In dieser Arbeit werden vor allem innovative und grundlegende Algorithmen vorgestellt, die Data Mining aus einer neuen Perspektive betrachten: *Synchronisation*.

Synchronisation ist ein in der Natur weit verbreitetes Phänomen. Gruppen von Ereignissen kommen spontan in einen gemeinsamen Rhythmus durch wechselweise Interaktion. Das Prinzip der Synchronisation erlaubt es, komplexe Prozesse durch einfache Operationen, die auf Interaktion zwischen Objekten beruhen, zu steuern.

Der erste Teil der Arbeit beschäftigt sich mit innovativen Algorithmen für synchonisationsbasiertes Data Mining. Zunächst wird die Technik *Sync* (Clustering by Synchronisation) vorgestellt. Diese Technik kombiniert das Synchronisationsprinzip mit dem Prinzip der minimalen Beschreibungslänge (Minimum Description Length, MDL). Dadurch wird die intrinsische Clusterstruktur gefunden, ohne einschränkende Annahmen über die Datenverteilung oder Parametrisierung zu treffen. Der Algorithmus *SOD* (Synchronization

based Outlier Detection) beruht auf den Unterschieden in der Objektdynamik im Synchronisationsprozess. Ausreißer können mit dem lokalen Synchronisationsfaktor (Local Synchronization Factor, *LSF*) auf natürliche Weise erkannt werden. Um den "*Curse of Dimensionality*" ("*Fluch der Dimensionalität*") zu lindern, wird der Subspace-Clustering-Algorithmus *ORSC* vorgestellt, der automatisch Cluster in Unterräumen des ursprünglichen Merkmalsraums entdeckt. Dieser Ansatz beinhaltet ein lokal gewichtetes Interaktionsmodell, um zu gewährleisten, dass sich alle Objekte in einem beliebig orientierten Subspace-Cluster auf natürliche Weise aufeinander zubewegen. Um Muster in Graphen zu entdecken, wird ein Graph-Clustering-Ansatz (*RSGC*, Robust Synchronization-based Graph Clustering) vorgestellt. Die zentrale Idee von *RSGC* ist es, Graph Clustering als einen dynamischen Synchronisationprozess aufzufassen. Durch das mächtige Konzept von Synisation hat *RSGC* einige wünschenswerte Eigenschaften, die andere Vergleichsverfahren nicht haben. Alle vorgestellten Algorithmen werden hinsichtlich ihrer Effektivität und Effizienz gründlich evaluiert. Die Vorteile über traditionelle Verfahren sind nachgewiesen durch Evaluierung an synthetischen und Realdaten.

Der zweite Teil dieser Arbeit beschäftigt sich mit der Analyse von Netzwerken im Gehirn basierend auf Diffusionstensor-Bildern (Diffusion tensor Imaging, DTI). Es wird ein neues Framework fr automatisches Clustering der Nervenfaserbündel vorgestellt. Dieses Framework kombiniert Ideen von Data Mining auf Zeitreihen mit dichtebasiertem Clustering. Anschließend wird eine verbesserte Variante dieses Ansatzes vorgestellt, die eine effiziente und effektive Möglichkeit bietet, eine hierarchische Clusterstruktur von Ner-

venbündeln zu finden. Außerdem wird basierend auf der strukturellen Konnektivität von Netzwerken im Gehirn ein automatisches Framework vorgestellt, das es erlaubt, Patienten mit der Alzheimerschen Krankheit zu erkennen und die veränderten Muster in Gehirnnetzwerken zu verstehen.

Part I

Preliminaries

Chapter 1

Introduction

Nowadays, we are overwhelmed with data. As information technology becomes ever more prevalent in nearly every aspect of our lives, the amount of data generated and stored continues to grow at an astounding rate in various fields, such as medicine, biology, society, etc. This increase in the amount of data calls for a need to automatically understand and analyze the data effectively and efficiently. The interdisciplinary field of Knowledge Discovery in Databases (KDD) has thus emerged in recent years to unveil the potential patterns or rules contained implicitly in the data automatically. During last decades, the study of knowledge extraction and discovery in databases has attracted huge attention, with multiple books, e.g., surveys and papers.

In this chapter, the overview on the field of Knowledge Discovery in Databases and Data Mining is first generally described in Section. Afterwards, the current open challenges of clustering are illustrated in Section 1.2. The chapter concludes with an outline of the thesis in Section 1.3.

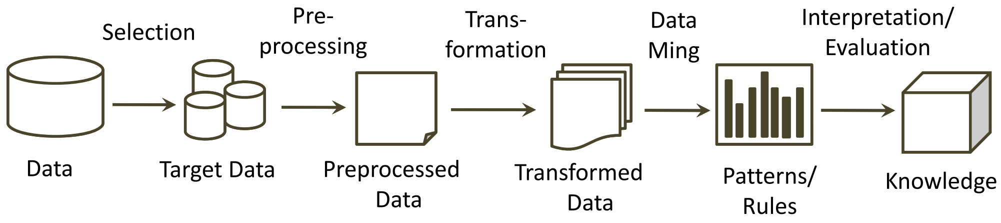


Figure 1.1: The KDD process.

1.1 Knowledge Discovery in Databases

> *"Knowledge Discovery in Databases is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data."* [57]

According to this definition, KDD extracts the meaningful patterns from data, which should be interpretable by a human user. In addition, KDD is a process, indicating that it consists of an iterative sequence of steps, which is further illustrated as follows (cf. Figure 1.1).

1. **Selection**. The first step of KDD process is to create a target data set by selecting a data set or focusing on a subset of attributes or data samples. The criteria of data selection often include data availability, quality, type, format, semantics, etc.

2. **Preprocessing**. The main goals of this step are cleaning and integrating the target data set. The data from multiple sources should be combined together and strategies to handle the noise and inconsistent data have to be selected and applied.

3. **Transformation**. To help the goal of the KDD task, the useful attributes are first identified to represent the data, e.g. using feature selection approaches or transformation methods (like PCA) to reduce the number of attributes or to find compact representation for the data.

4. **Data Mining**. The goal of this essential step is to identify the relevant data mining task, e.g. clustering, classification and association mining. In this step, efficient and effective algorithms are used to extract novel, unknown and useful patterns from the transformed or original data.

5. **Interpretation and Evaluation**. The interesting patterns extracted in the data mining step are presented using knowledge visualization and representation techniques. In addition, the extracted patterns are evaluated by domain experts according to the task definition. If the results are not satisfactory, we need to go back one or more steps.

Among all these steps of the KDD process, data mining is the core step. Therefore, in most literatures, the terms "data mining" and "KDD" are often used in a synonymous way. Formally, *data mining* in [57] is defined as follows:

Data Mining is *a step in the KDD process consisting of applying data analysis and discovery algorithms that, under acceptable computational efficiency limitations, produce a particular enumeration of patterns over the data.* [57]

In the following, the most important data mining methods are presented.

- **Clustering**: Clustering is the task of finding a grouping of the objects of the data sets into groups (clusters) while maximizing the similarity of the objects in a common cluster and minimizing the similarity of the objects in different clusters.

- **Classification**: Classification is the task of learning a function that maps data objects to one or several classes in a predefined class set. To learn this function, classification methods need a training set, containing data objects that are already mapped to the class they belong to. After learning the training set, classification methods can map new unknown objects to the classes.

- **Association Rules**: Discover association rules that occur frequently together in a given data set. Association rules express as a specified set of items appearing together in the same transaction with a certain support/probability.

- **Outlier Analysis**: Find irregular objects in the data set, e.g. which do not correspond to the general characteristics or model of the data set.

- **Characterization and Discrimination**: Finding a compact description for a subset of the data or comparing a particular subset of the data with comparative subsets.

This thesis mainly focuses on proposing new clustering and outlier detection techniques and the practical applications of data mining techniques in neuroscience. Clustering is essential for knowledge discovery in a large variety of applications. During the last decades, clustering has attracted a huge volume of attention and many algorithms have been proposed. The most typical algorithms will be elaborated in Section 2.1.1. As we will see, many established approaches have some specific limitations. To deal with the limitations of existing methods, a new cluster notion: "Synchronization" for cluster analysis is introduced (cf. Part II). Furthermore, the practical applications of data mining methods on brain network analysis in the field of neuroscience are demonstrated in Part III. A series of frameworks are proposed for brain network mining by the integration with the techniques of clustering, feature selection and graph mining.

In the following section, the current open challenges in clustering are given, and then a broad overview of the main contributions of this thesis is provided in Section 1.3.

1.2 Challenges in Clustering

During last several decades, although many clustering algorithms have been proposed, there also exist some open challenges in clustering as follows. This thesis tries to address most of these challenges in Part II and III by proposing different data mining algorithms.

- **High-Dimensional Data**: In real-world, high-dimensional data are always encountered in various areas. These real-world data sets are often represented as sparse, high-dimensional feature vectors, contaminated by outliers and noisy objects. Clustering such high-dimensional data becomes difficult for traditional clustering methods due to the problem called "curse of dimensionality". With increasing dimensionality, more and more objects are located at the boundaries of the feature space. In very high dimensions, it is common for all of the data objects to be nearly equidistant from each other and objects do not cluster anymore in the full-dimensional feature space. Global dimensionality reduction like Principal Component Analysis (PCA) can be applied to transform the data to a lower dimensional space. However, for clustering, Global PCA is not the best way to cure the curse of dimensionality. The reason is that PCA preserves the global variance between objects. It is thus impossible to yield good results if many irrelevant attributes hide the cluster structure. Moreover, it is a non-trivial task to interpret the new features (principal components).

- **Large Data**: Nowadays, the size of databases has been increasing dramatically. Most traditional clustering algorithms have been proposed to handle a relatively small size of data set.

- **Noise Handling**: In real data sets, there often exist some noise or outliers due to the erroneous measurement, or human error. For many proposed clustering algorithms, such as K-Means, EM, Spectral Clustering, the performance of these algorithms are very sensitive to noise or outliers.

- **Parametrization**: For many clustering algorithms, the results strongly depend on suitable parameter settings. For instance, the number of clusters or cluster density needs to be specified as an input parameter. Such parameters are not easy to be estimated without prior knowledge of the data set.

- **Complex data**: In recent years, a large amount of data sets have been generated in the field of multimedia, neuroscience, biology, etc. Most of these data sets are represented as complex, application specific data objects. Since clustering algorithms build on a similarity of data objects, a specific similarity measure must be defined. However, introducing a new similarity measure for handling complex objects is a non-trivial task.

1.3 Outline of the Thesis

In this thesis, novel clustering and outlier detection algorithms are proposed based on the synchronization principle. Furthermore, the practical applications of data mining techniques in neuroscience are developed. The content of this thesis is organized as follows:

Part I deals with the preliminaries.

Chapter 1 presents an overview on the field of Knowledge Discovery in Databases and Data Mining in a very general manner, providing the reader with the broader context of this thesis.

Chapter 2 introduces basic notions of clustering and outlier detection, and surveys some fundamental algorithms in more detail.

Part II presents new techniques to clustering and outlier detection inspired by synchronization.

Chapter 3 gives a brief introduction of synchronization, including the concept of synchronization, Kuramoto model and its applications in diverse fields.

Chapter 4 proposes *Sync*, a novel approach to clustering. By combining with the Minimum Description Length principle (MDL), it allows discovering the intrinsic clusters without any data distribution assumptions and parameters setting.

Chapter 5 presents the algorithm *hSync*, by extending the concept of synchronization to hierarchical data analysis. It robustly reveals the natural cluster hierarchy regardless of size and data distribution of the clusters.

Chapter 6 proposes a subspace clustering algorithm *ORSC* to find all synchronized clusters in arbitrarily oriented subspaces. Relying on a weighted local interaction model, all objects in a common cluster which accommodated in arbitrarily oriented subspace can naturally move together.

Chapter 7 presents the graph partitioning approach *RSGC*, which considers graph clustering as a dynamic process towards synchronization. As the graph connection information is naturally integrated in the interaction model, *RSGC* shows several desirable properties and can find the natural partitions of a graph without any insensitive parameters.

Chapter 8 introduces a new outlier detection algorithm *SOD* from a new perspective: "*Out of Synchronization*". By simulating the dynamical behavior of each data object towards synchronization, outlier objects are naturally flagged by local synchronization factor (LSF).

Part III presents new techniques on brain network mining in the field of neuroscience.

Chapter 9 provides the background information on brain network mining, especially for Diffusion Tensor Image (DTI) and fiber tracking.

Chapter 10 presents a framework for automated white matter tracts clustering in the Human Brain by combining ideas from time series mining with density-based clustering.

Chapter 11 proposes the enhancement and variation of hierarchical fiber clustering approach which allowing for a more robust, efficient, or effective way to find hierarchies of fiber bundles.

Chapter 12 gives an automated prediction framework to analyze and understand the abnormal patterns in patients of very early Alzheimer's disease based on individual structural connectivity networks.

Part IV Conclusion.

Chapter 13 concludes the thesis and gives the outlook for the future work.

Chapter 2

Clustering and Outlier Detection

As the main focus of this thesis is to develop new algorithms for data mining especially for clustering and outlier detection, this chapter first gives a general background to both of them. Section 2.1 starts with a general introduction to clustering. As there is a huge variety of clustering algorithms, only a brief survey on different representatives of various clustering paradigms is provided (cf. Section 2.1.1). Subsequently, Section 2.1.2 presents the validation of the clustering results for comparing clustering algorithms. Afterwards, Section 2.2 introduces outlier analysis. Finally, the chapter ends with a short conclusion in Section 2.3.

2.1 Cluster Analysis

Clustering is the process of grouping a given data set into classes or clusters. The objects within a cluster should be similar to each other and the objects of different clusters should be dissimilar to each other. To determine the similarity between objects, an appropriate similarity measure is needed.

Given two d-dimensional objects $x = \{x_1, \cdots, x_d\}$ and $y = \{y_1, \cdots, y_d\}$ of a data set D, to measure the similarity between them, a distance function *dist* is used. Formally, let *dist* be one of the L_p norms. The similarity of the two objects is defined as follows for an arbitrary $p \in N$:

$$dist(x, y) = \sqrt[p]{\sum_{i=1}^{d} (x_i - y_i)^p}$$

Currently, the widely-used distance functions are the Euclidean, Hamming, Manhattan, and Mahalanobis distance. Whenever not otherwise specified, throughout this thesis, Euclidean distance measure is used ($p = 2$). However, for complex data objects, such as images, protein structures or text data, it is a non-trivial task to define domain specific similarity measures. In Chapter 10, a new similarity measure for white matter tracts in the human brain is introduced.

During the last decades, clustering has attracted a huge volume of attention, with multiple books, e.g. [82], surveys, e.g. [115] and research papers, e.g. [6, 8, 47, 179] to mention a few. As there is a huge variety of clustering algorithms, only methods which are fundamental for the techniques described in this thesis are surveyed in the following. For a comprehensive survey on basic clustering algorithms, please refer to [82, 75].

2.1.1 Clustering Algorithms

In general, clustering algorithms can be classified into partitioning and hierarchical methods. In the following, the common algorithms of each type are described.

Partitioning Clustering

Partitioning clustering refers to a class of methods which group objects of a data set into a set of clusters and assign each object to exactly one cluster. Here, four prominent representatives of partitioning clustering methods and their related approaches are briefly illustrated.

K-Means. The most well-known and commonly used partitioning clustering algorithm is K-Means [101]. The algorithm starts with an arbitrary initial partitioning of the data set into k clusters. Afterwards, the algorithm iteratively improves the initial clustering by minimizing the sum of squared distances of the data objects to the mean vector of respective cluster.

Formally, given a data set D, $x \in D$ and the initial set of k cluster centers are $(m_1, \cdots, m_k)$, which can be specified randomly or by some heuristics, the K-Means algorithm proceeds by alternating between two following steps.

Assignment step: Assign each point to the closest cluster center.

$$S_i = \{x | dist(x, m_i) \leq dist(x, m_j)(\forall j = 1, \cdots, k)\}$$

where $dist(\cdot)$ is a distance function.

Update step: Update the new cluster center to be the centroid of the objects in the current cluster.

$$m_i = \frac{1}{|S_i|} \sum_{x \in S_i} x$$

The algorithm terminates when no changes in the cluster assignment of the objects. The main advantage of K-Means is its efficiency. However, for K-Means, the number of clusters k has to be specified in advance, which is a non-trivial task on real-world data sets. The clustering result is also sensitive to noise and outliers. In addition, K-Means implicitly assumes a Gaussian data distribution, and is thus restricted to detect spherically or Gaussian clusters. To handle these problems, many extended algorithms have been proposed. For example, to estimate the reasonable number of clusters k, the X-means [125] is proposed in combination with the information theory. The K-Medoid methods [86] are presented to alleviate the noise effect on clustering.

Expectation-Maximization (EM) Clustering. Generally, the EM-algorithm [47] is a soft version of the K-means algorithm. It is an iterative optimization method to estimate the unknown parameters. EM algorithm proceeds by alternating an expectation (E) step, which computes the expectation of the log-likelihood under the current estimate of the parameters, and a maximization (M) step, which finds parameters maximizing the expected log-likelihood.

Specifically, the EM algorithm can be outlined as follows:

1. Choose some random values to the estimated parameters.

2. E-step: Calculate the expected value of the log-likelihood function with respect to the given parameter values.

3. M-step: Find a better estimate for the parameters by the maximization of the log-likelihood obtained in the E-step.

4. Assess convergence (with respect to some criterion) and repeat the E and M-steps until convergence is reached.

For clustering, the data set is modeled as a finite mixture of k distributions. To calculate the cluster membership of each data point, the mixture model of the k distributions is optimized until no improvement of the log-likelihood of the data is reached. Like K-Means, the result heavily depends on the initialization and is very sensitive to noise and outliers.

DBSCAN. The DBSCAN algorithm [55] considers the clustering problem from a density-based point of view: Clusters are regarded as areas of high object density which are separated by areas of lower object density. DBSCAN is based on three main concepts: core object, density reachability and density connectability. These concepts depend on two input parameters ϵ and $MinPts$. $MinPts$ specifying a minimum number of objects, and ϵ , the radius of a hyper-sphere. An object is called a core object if there are at least $MinPts$ objects in the ϵ-neighborhood. If one object P is in the ϵ-neighborhood of a core-object Q, then P is said to be directly density-reachable from Q. The density-connectivity is the symmetric, transitive closure of the direct density reachability, and a density based cluster is defined as a maximal set of density connected objects. Except for DBSCAN, many other density-based clustering have been proposed, such as DENCLUE [78] and OPTICS [8].

In contrast to many other partitioning clustering methods such as K-Means and EM methods, the density-based notion has several attractive benefits. The user does not have to specify the number of clusters. Density-based clustering algorithms are able to detect clusters of arbitrary shape.

It is robust to noise and outliers. Finally, density-based clustering methods are not restricted to vector data but applicable on general metric spaces. In Chapter 10, the density-based clustering and dynamic time wrap are combined to identify meaningful fiber bundles in the human brain. However, for DBSCAN, the clustering result strongly depends on an appropriate choice of the density parameters.

Spectral Clustering. Spectral Clustering refers to a class of well-known techniques which rely on the Eigenvector decomposition of a similarity matrix to partition objects into disjoint clusters [165]. One of the most popular algorithms is normalized spectral clustering [116] proposed by Ng et al. The basic idea of this algorithm is first to map the data points into a new space and then apply the traditional methods (like K-Means) in this new space for clustering. Formally,

1. Given a set of data points A, construct a similarity matrix S, where S_{ij} represents a measure of the similarity between points $i, j \in A$.

2. Define D to be the diagonal matrix whose (i,i)-element is the sum of S's i-th row, and construct the matrix $L = D^{-1/2} S D^{-1/2}$.

3. Find $x_1, \cdots, x_k$: the k largest eigenvector of L and form a new matrix $X = [x_1, \cdots, x_k]$ by stacking the eigenvectors in columns.

4. Normalize each of X's rows to have unit length to form a matrix Y.

5. Regard each row of Y as a point and cluster them as k clusters by K-Means or other algorithms.

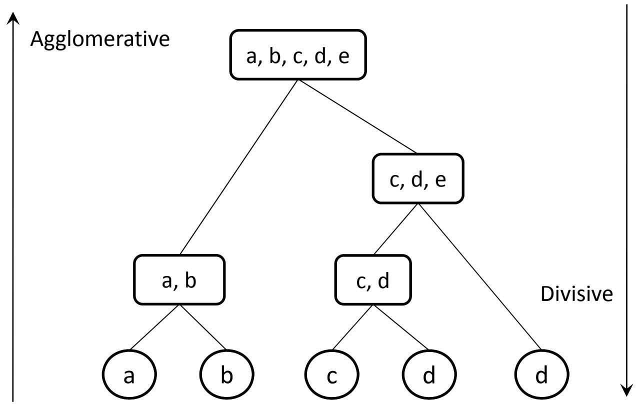


Figure 2.1: Hierarchical clustering with agglomerative and divisive strategies.

6. Assign the original point i to cluster j if and only if row i of the matrix Y is assigned to cluster j.

Hierarchical Clustering

In hierarchical clustering, the data are not partitioned into a particular cluster but a series of partitions in different levels. Currently, there are two strategies to explore the hierarchical clusters: agglomerative and divisive. Agglomerative strategies perform in a bottom-up way. Starting with each data object regarding as a cluster, they gradually merge the most similar pair of clusters until all data objects are in a common cluster. In contrast, divisive strategies work in a top-down way. Starting with all points in a common cluster, they recursively split the clusters when all data objects are in different clusters (See Figure 2.1).

Single Link. Single link [82] is a very simple agglomerative hierarchical clustering method. The algorithm starts with n singleton clusters, each containing only one data object. The algorithm then merges the pair of clusters which have the minimal distance in each step until all objects are merged in one cluster. The hierarchical cluster structure can be then visualized in a dendrogram (cf. Figure 2.1). The leaves of the dendrogram represent data objects and the levels of the dendrogram demonstrate the cluster merging procedure.

During the merging process, a distance function between two groups of objects should be specified. Single Link uses the smallest minimum pair-wise distance between the two clusters of objects. Except for Single Link, many other variants of Single Link have been proposed based on different distance functions. The variant Complete Link applies the smallest maximum pair-wise distance between two clusters, while the Average Link uses the average among the distances between both clusters of objects. Through the dendrogram, the clusters can be obtained at any level by horizontally cutting the dendrogram. However, the dendrogram of real data is often very complicated and difficult to visualize the data structure. The reason lies in the generated dendrogram has $N-1$ levels with N objects, which reflects the merging process according to the distances among pair of clusters. For real data, it is not easy to find the optimal splitting of the dendrogram to identify clusters. Another drawback of Single Link and its variants is the single-link effect. Moreover, the clustering result is also sensitive to noise and outliers, which can be demonstrated in Section 10.5.4.

OPTICS. OPTICS [8] is a hierarchical variant of the density-based clustering algorithm DBSCAN. In contrast to DBSCAN, OPTICS does not assign cluster membership, but orders objects of a data set D into a hierarchi-

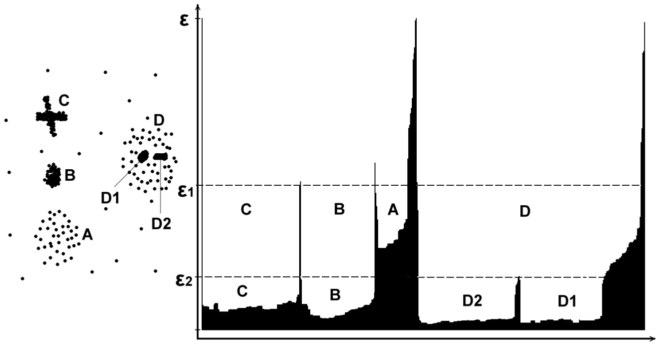


Figure 2.2: Illustration of the reachability plot of OPTICS. Left: 2D sample data set, where different capital letters indicate hierarchical clusters with different densities, shapes and sizes. Right: the reachability plot of the sample data set, where the different valleys of this plot indicate the clusters.

cal cluster structure. This structure contains clustering information, which could also be achieved by all possible DBSCAN clusterings with respect to distances ϵ' that are smaller than the generating distance ϵ. To describe the hierarchical cluster structure, each object is characterized by two measures, its core-distance and its reachability-distance. The output of OPTICS is a linear order of the data objects according to their hierarchical cluster structure which is visualized in the so-called reachability-plot. This reachability plot is a 2D plot showing the objects' position index on the x-axis and the objects' reachability-distance on the y-axis. The reachability plot provides a visual representation of the cluster structure of the data from the density-based point of view and enables an intuitive way to find clusters. Valleys in this plot indicate clusters, the deeper the valley, the denser the cluster. Clusters can be obtained by cutting the reachability plot with a horizontal line.

An example of a reachability plot for a 2D data set is presented in Figure 2.2. The reachablity plot provides information about the general number of clusters, the densities of different clusters, and the hierarchical structure of the data.

2.1.2 Validation of Clustering Results

Several quantitative measures have emerged to compare the results of different clustering algorithms. In the following, different measures that are used in this thesis are described.

Encoding Cost

To compare clustering results for different approaches with a ground truth, an information-theoretic external cluster-validity measure in [50] is proposed. The evaluation scheme measures how useful the calculated cluster labels are as predictors of the ground truth cluster labels.

In contrast to measures like the Rand Index [72] or the Cluster Purity [167], different numbers of clusters can be compared. The clustering quality measure is defined as the entire encoding cost, $Q(C, K)$, which is the sum of the empirical conditional entropy $H(C|K)$ and of the code length for the number of clusters $CL(C|K)$:

$$Q(C, K) = H(C|K) + CL(C|K) \tag{2.1}$$

Where C is the set of ground truth labels and K equals the set of calculated labels.

$$H(C|K) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{h(c,k)}{n}log\frac{h(c,k)}{h(k)}$$

$$CL(C|K) = \frac{1}{n}\sum_{k=1}^{|K|}log\binom{h(k)+|C|-1}{|C|-1}$$

where $h(c,k)$ is the number of fibers labeled within class C with label c and within class K with label k. $h(k) \equiv \sum_c h(c,k)$. $|K|$ and $|C|$ are the number of calculated clusters and the number of ground truth clusters respectively. The smaller the value of the entire encoding cost, the better is the quality of the clustering results.

Mutual Information

Besides the entire coding cost used to evaluate the clustering results, more and more information theoretic measures are proposed based on the Mutual Information. Formally, the mutual information between two discrete random variables X and Y is defined as:

$$MI(X,Y) = \sum_{y\in Y}\sum_{x\in X}p(x,y)log(\frac{p(x,y)}{p(x)p^{'}(y)})$$

where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p^{'}(y)$ are the marginal probability distribution functions of X and Y.

Intuitively, mutual information measures the information that X and Y share. This property indicates that the mutual information can be used to measure the information shared by two clusters. Based on this idea, many mutual information-based measures are proposed.

Normalized Mutual Information (NMI): The most popular and intuitive way to use mutual information for clustering evaluation is the normalized version of the Mutual Information (NMI) [156]. Formally, given two clusterings $U = \{U_i, \cdots, U_M\}$ and $V = \{V_1, \cdots, V_N\}$, the NMI is defined as:

$$NMI(U,V) = \frac{MI(U,V)}{\sqrt{H(U)H(V)}}$$

where $H(U)$ and $H(V)$ are the entropies associated with the clustering U and V respectively. The Normalized Mutual Information of two clusterings U and V has fixed upper and lower bounds and takes a value of 1 when two clusterings are identical and 0 when the two clusterings are independent.

Adjusted Mutual Information (AMI): To correct the effect of chance agreement in clustering evaluation, AMI [164] is proposed based on the expected mutual information. Given two clusterings U and V, the adjusted measure for the mutual information can be:

$$AMI(U,V) = \frac{MI(U,V) - E\{MI(U,V)\}}{max\{H(U), H(V)\} - E\{MI(U,V)\}}$$

where $H(U)$ and $H(V)$ are the entropies associated with the clustering U and V respectively, and $E\{MI(U,V)\}$ is the expected mutual information of clusterings U and V. AMI takes a value of 1 when the two clusterings are identical and 0 when the mutual information between the two clusterings equals its expected value.

Adjusted Variation of Information (AVI): In addition, based on the variation of information, the Adjusted Variation of Information (AVI) [164] is given by:

$$AVI(U,V) = \frac{2MI(U,V) - 2E\{MI(U,V)\}}{H(U) + H(V) - 2E\{MI(U,V)\}}$$

Like AMI, AVI takes a value of 1 when the two clusterings are identical and 0 when the mutual information between the two clusterings equals its expected value.

2.2 Outlier Detection

As illustrated in Section 1.1, outlier detection is a primary step in KDD. In data analysis, one of the preliminary steps is to detect the outlying observations in the data sets. According to the definition of Hawkins [76], an outlier is defined as an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. Similarly, Grubbs [67] indicates that an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs. Detecting such irregular observations in data sets is very curial for further data analysis. In addition, in this thesis, outliers also refer to noise objects. The terms “outlier” and “noise” are used in a synonymous way.

During last several decades, outlier detection has attracted huge attention in diverse applications, such as credit card fraud detection, clinical trials, voting irregularity analysis, network intrusion detection, fault detection, and athlete performance analysis. Many algorithms for outlier detection have thus been proposed [30, 91, 92, 24]. In the following, some widely-used outlier detection algorithms are generally described.

Mahalanobis distance: The most of the earliest methods for outlier detection rely on the statistical analysis. For a given data set, it often assumes to follow a known distribution, such as Gaussian. If the data objects of the data set badly fit the data distribution and then they are considered as outliers. One of a well-known method is Mahalanobis distance. Given n data objects in the d-dimensional data set, Mahalanobis distance for each data object x_i is defined as [19]:

$$Mahal_{x_i} = \sqrt{\sum_{i=1}^{n}(x_i - \bar{x}_n)^T \cdot \Sigma^{-1} \cdot \sum_{i=1}^{n}(x_i - \bar{x}_n)} \tag{2.2}$$

Where Σ is the covariance matrix of n data objects and $\bar{x}_n$ denotes mean vector.

Since Mahalanobis distance reflects the distribution of the data set, data objects with a large Mahalanobis distance are indicated as outliers.

CoCo: Based on information-theoretic principle, recently, Böhm, et al. proposed CoCo [24], a parameter-free outlier detection with coding cost. Based on the MDL principle, outliers are flagged as those objects which need more coding cost than regular objects. For coding each object, the optimal neighborhood size is heuristically determined. Independent Component Analysis and Exponential power distribution (EPD) are combined to estimate the probability and the corresponding coding cost. Like most distribution-based methods, CoCo tends to fail if the estimated distribution does not fit the data model well. It is also time consuming to find the optimal neighborhood to estimate the coding cost for each object by screening for suitable neighborhood sizes.

LOF: To handle the issue of varying densities in the data set, Breunig, et al. [30], introduce a notion of local outlier from a density-based perspective.

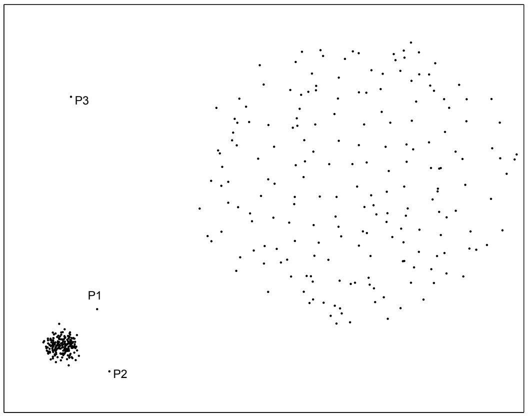


Figure 2.3: LOF: local outlier vs. global outlier [30].

An object is regarded as an outlier if its local density does not fit well into the density of its neighboring objects. The local outlier factor (LOF) is then proposed to capture the degree to which the object is an outlier. It is defined as the average of the ratio of the local reachability density of the object and those of the objects in its neighborhood. A LOF value of approximately 1 indicates the object is located inside a cluster, while the objects with higher LOF values are more rather considered as outliers. In Figure 2.3, LOF shows the advantage of outlier detection for local outliers in the data set with varying densities. It can obtain the local outliers by computing the relative degree of isolation from its surrounding neighborhood.

LOCI: The Local Outlier Integral (LOCI) [120] flags outliers, based on probabilistic reasoning and motivated from the concept of a multi-granularity deviation factor (MDEF). Similar to LOF, the LOCI outlier model takes the local object density into account, but differently, the MDEF of LOCI uses ϵ-neighborhoods rather than $MinPts$ nearest neighbors. The local neighborhood in LOCI model is defined by two parameters: the counting and the sampling neighborhood. The counting neighborhood specifies some volume of the feature space which is used to estimate the local object density. The sampling neighborhood is larger than the counting neighborhood and contains all points which are used to compute the average object density in the neighborhood. Objects which deviate in their local object density more than three times of the standard deviation are regarded as outliers. The flagging scheme of LOCI thus assumes the object densities follow a Gaussian distribution.

In the Chapter 8, the advantages and drawbacks will be further illustrated and discussed by comparing with the new proposed outlier detection algorithm based on synchronization principle.

2.3 Conclusion

In this chapter, a brief introduction to clustering and outlier detection is given. Some wide spread algorithms which are fundamental for the techniques described in the following chapters have been briefly characterized. In view of the open challenges of clustering (cf. Section 1.2), a new concept in data mining analysis: *synchronization*, is introduced in the following part.

Part II

Synchronization-based Data Mining

Chapter 3

Synchronization

Synchronization, a ubiquitous phenomenon in nature, has attracted a large volume of interest in physics, biology, ecology, sociology, communication and other fields of science and technology. In this chapter, the concept of synchronization is first introduced. In Section 3.2, the widely used Kuramoto model to describe the mechanism of synchronization is provided. Finally, the applications based on synchronization are reviewed.

3.1 The Concept of Synchronization

> "*Synchronization means an adjustment of rhythms of oscillating objects due to their weak interaction.*" [126]

Synchronization is an emerging phenomenon of a population of dynamically interacting units, has fascinated humans from ancestral times [10]. In 1665, the great Dutch physicist and mathematician, inventor of the pendulum clock, Christiaan Huygens discovered an odd "sympathy of two clocks", where a couple of pendulum clocks hanging from a common support were swinging in perfect synchrony [80]. He explained this effect as mutual synchronization and suspected that the conformity of the rhythms of two clocks had been caused by air movements or imperceptible vibrations in the common support [126]. From that moment on, synchronization has been observed in biological, chemical, physical, and social systems, and it has attracted the interest of scientists for centuries. For example, in the middle of the nineteenth century, Rayleigh [131] described the interesting phenomenon of synchronization in acoustical systems in his famous treatise ”The Theory of Sound”. Eccles and Vincent [52] applied for a British Patent confirming their discovery of the synchronization property of a triode generator. Appleton [9] and van der Pol [162] replicated and extended the experiments of Eccles and Vincent and made the first step in the theoretical study of this synchronization effect.

Actually, synchronization is one of the most captivating collective phenomena in nature [2]. A paradigmatic example is the synchronous flashing of fireflies. At the beginning, each firefly rests in the trees. And suddenly, some fireflies start emitting flashes of light. Firstly they emit flash incoherently, but after some time all the fireflies are flashing with the same frequency. Moreover, it is known that synchrony is rooted in human life from the metabolic processes in our cells to the highest cognitive tasks we perform as a group of individuals [10]. For instance, synchronization seems to be a basic mechanism for organizing neuronal information processing within a brain area, such as visual cortex [66] and sensorimotor cortex [133]. However, as yet, little is known about the basic mechanism of synchronization. Therefore, it is important to analyze synchronization processes to achieve a better understanding of it.

3.2 The Kuramoto model

To capture what are the basic mechanisms responsible for the collective synchronous behavior, the pioneering work was done by Winfree [170], which called for mathematical approaches to tackle the problem. Based on this, Kuramoto proposed the simplest and most successful model for the analysis of synchronization: Kuramoto model (KM) [94, 95] in 1975. Its formulation was motivated by the behavior of systems of chemical and biological oscillators. Kuramoto model considers a system consists of a large population of weakly-coupled, nearly identical, interacting limit-cycle oscillators. For each oscillator, it interacts with others and therefore changes its rhythm according to a given coupling function. Specially, when the natural frequencies of the oscillators are too different, they are difficult to synchronize together. However, if the coupling strength is strong enough, all oscillators can achieve synchrony.

Formally, the KM model consists of a population of N coupled phase oscillators where the phase of the *i-th* unit, denoted by θ_i, evolves over time according to the following dynamics:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i), (i = 1, \ldots, N), \tag{3.1}$$

where ω_i stands for its natural frequency and K describes the coupling strength between units. There are two major tendencies in this model: the coupling strength K tends to synchronize the oscillators through mutual interaction while the variance of these natural frequencies, ω_i, tends to drives them to stay away from each other. When the coupling strength $K = 0$, each oscillator tries to run independently at its own frequency. However, if the coupling is strong enough, all oscillators will freeze into synchrony, which called global synchronization.

To characterize the level of synchronization between oscillators, a global order parameter r is defined as [94]:

$$re^{i\psi} = \frac{1}{N}\sum_{j=1}^{N} e^{i\theta_j}, \tag{3.2}$$

where r changes over time and $0 \leq r(t) \leq 1$, measures the coherence of the oscillator population at time t, and $\psi(t)$ is the average phase at time t. In the thesis, r will instead $r(t)$ for simplification.

In the following chapters (Chapter 4 to Chapter 8), the Kuramoto model will be reformulated in different ways to handle the clustering and outlier detection problems in various aspects, e.g. subspace clustering, hierarchical clustering.

3.3 Applications

Synchronization phenomena in large populations of interacting elements are the subject of intense research efforts in physical, biological, chemical, and social systems. Seliger et al. [141] discuss mechanisms of learning and plasticity in networks of phase oscillators through a generalized Kuramoto model. Arenas et al. [11] apply the Kuramoto model for network analysis, and study the relationship between topological scales and dynamic time scales in complex networks. This analysis provides a useful connection between synchronization dynamics, network topology and spectral graph analysis. In bioinformatics, Kim et al. [89] propose a strategy to find groups of genes by analyzing the cell cycle specific gene expression with a modified Kuramoto model. Aeyels et al. [3] introduce a mathematical model for the dynamics of chaos system. They characterize the data structure by a set of inequalities in the parameters of the model and apply it to a system of interconnected water basins. In summary, previous approaches mainly focus on the synchronization phenomena of a dynamic system from a global perspective. In this case, the local structure of the system cannot be well explored.

3.4 Conclusion

In this chapter, a brief introduction to synchronization is given. The basic concept of synchronization is illustrated. After that, the Kuramoto model which is used to understand the collective synchronization is provided. The Kuramoto model is the basis for the following proposed interaction models in this thesis. Finally, the widely applications based on synchronization are briefly surveyed. In next chapters, the local synchrony effect and new approaches to clustering and outlier detection will be presented.

Chapter 4

Clustering by Synchronization

As illustrated in Chapter 3, synchronization is a powerful basic concept in nature regulating a large variety of complex processes ranging from the metabolism in the cell to social behavior in groups of individuals [10]. Based on this powerful concept, *Sync*, a novel approach to clustering, is proposed in this chapter. Inherited from synchronization principle, *Sync* has several desirable properties over existing approaches: (i) The clusters revealed by dynamic synchronization truly reflect the intrinsic structure of the data set; (ii) *Sync* does not rely on any distribution assumption and allows detecting clusters of arbitrary number, shape and size; (iii) In combining with the Minimum Description Length (MDL) principle, *Sync* allows fully automatic clustering.

The remainder of this chapter is organized as follows: in Section 4.1, it starts with an introduction. A brief summary of related work is given in Section 4.2. In Section 4.3, the algorithm of *Sync* is elaborated. Section 4.4 contains an extensive experimental evaluation of synthetic and real data sets. Finally, Section 4.5 gives a summary and concludes this chapter. Parts of the material presented in this chapter have been published in [27], where Junming Shao (J.S.) was mostly responsible for the development of the main concept and wrote the largest parts of the paper. C.P. developed the idea of MDL-based parameter estimation. C.B., C.P. and Q.Y. also contributed to the conceptual development and paper writing.

> "*Christian Böhm, Claudia Plant, Junming Shao, Qinli Yang. Clustering by synchronization. The 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining 2010: 583-592.*"

4.1 Introduction

Clustering is essential for knowledge discovery in a large variety of applications. During the last decades, clustering has therefore attracted a huge volume of attention, with multiple books, e.g. [82], surveys, e.g. [115] and research papers, e.g. [6, 8, 47, 179] to mention a few. Many approaches have been proposed to address the clustering problem from different points of view, and each cluster notion comes with specific advantages and drawbacks. As an example, consider the wide-spread Expectation Maximization (EM) algorithm [47]. The result of EM, a mixture model of multivariate Gaussians, is very useful for interpretation in many applications. However, EM only yields good results if the data at least approximately follows the model assumption, i.e. consists of Gaussian clusters. Moreover, the number of clusters needs to be specified as an input parameter and the result of EM is very sensitive w.r.t. outliers.

In this chapter, a novel point of view for clustering: *synchronization* is proposed. Before demonstrating many attractive benefits of clustering based on synchronization, the basic idea will be first illustrated.

4.1.1 Basic Idea

Synchronization phenomena are prevalent in daily life, as an example considering opinion formation. In the beginning, each person has their own view about a certain problem. After *interaction* by conversation or discussion, some people with similar background, such as education or hobby, will easily group together and form a common opinion. As time evolves, in the phase of *local synchronization*, groups with diverse opinions will be formed. After further conversation with different groups, all people may finally achieve a uniform opinion (*global synchronization*).

Based on these synchronization phenomena, the synchronization dynamics is exploited to obtain a natural clustering of a given data set. The key idea is to regard each data object as a phase oscillator which interacts dynamically with similar objects. The formal description of the dynamical object interaction patterns by a reformulated Kuramoto model will be elaborated in Section 4.3.2 but now the intuition of clustering by synchronization is first provided. The process of the dynamical clustering involves the following stages: First, starting from initial conditions, each object runs independently at its own phase. As time evolves, those objects with highest similarity will synchronize first. Then, in a sequential process, more and more objects synchronize together and clusters are produced driven by the intrinsic structure of the data set. Finally, the whole population is split into several stable distinct

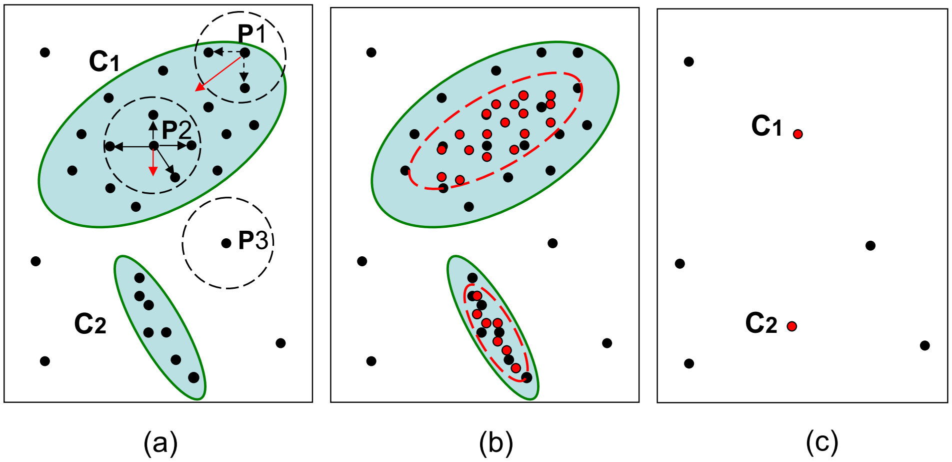


Figure 4.1: Illustration of Clustering by synchronization. (a) The initial state of the objects: the black arrows indicate the mutual interaction and red arrows indicate the direction of movement for some sample objects. (b) The comparison of objects states before and after one time step: black points represent the initial state, red points indicate the new state. (c) The final state of objects in local synchronization: synchronized clusters and outliers.

synchronized clusters. Outliers are effectively detected due to their different dynamic behaviors hardly synchronizing with any of the cluster objects.

To illustrate cluster formation by synchronization, Figure 4.1 displays 3 snapshots of the simulated dynamical movements of objects. For simplicity, 2-dimensional data are displayed. Consider the 3 objects ($P1$, $P2$, $P3$), where $P1$ and $P2$ are cluster objects and $P3$ is an outlier. For border object $P1$, its neighborhood contains less objects and all neighbors are located at the inner side of the cluster. Therefore, the border object is driven towards the inside of the cluster through the non-linear dynamical interaction with its neighbors (Figure 4.1 (a)). The larger the distance between an object and its neighbors, the stronger is the interaction imposed, which implies that neighbors with larger distances have a stronger impact on the object. For the center objects, e.g. $P2$, its neighborhood contains more objects, however, all these neighbors locate around the object, causing a relatively slow movement of $P2$ in comparison to the border objects, such as $P1$. The movements of objects depend on the structure of their overall neighborhood, which implies that the objects will move towards the main direction of their neighborhood. Through the non-linear interaction, all objects with similar attributes finally synchronize together. In contrast, outlier objects remain isolated from all other objects, e.g. $P3$. Figure 4.1 (b) displays the old and the new positions of the objects for comparison. The objects with high similarity gradually synchronize together through mutual coupling. The initial points (black color) are replaced by the red points after one time step. Figure 4.1 (c) depicts the final states of the objects. The data set is split into clusters and outliers, where cluster objects are synchronized together and have the same phase while outlier objects remain isolated over time.

4.1.2 Contributions

The major benefits of the algorithm *Sync*: (Clustering by **Sync**hronization), can be summarized as follows:

1. *Clusters detected by Sync truly reflect the intrinsic data structure.* Rooted in the concept of synchronization, *Sync* allows detecting clusters of arbitrary number, shape, size and object density without requiring any distribution assumptions.
2. *Sync is a powerful technique for clustering and outlier detection.* Based

on the different dynamical behaviors of objects during the process of synchronization, outliers are naturally and effectively detected.

3. *Sync is robust against parameter settings and fully automatic by the combination with Minimum Description Length.* The basic version of the algorithm is relatively robust against parameter settings. For automatic clustering, *Sync* is combined with the Minimum Description Length principle.

To the best of our knowledge, no other clustering method meets all of the above properties. Table 4.1 gives a list of symbols used in the following.

4.2 Related Work

As mentioned, during the past several decades, many algorithms have been proposed for clustering, such as EM [47], CURE [69], CLIQUE [6], BIRCH [179], CLARANS [117], and DBSCAN [55]. Here, only a very brief survey on some important major research directions is provided. In addition, the related work on kernel density estimation is briefly introduced.

PDF-based Clustering. The key idea of these methods is to detect clusters by using a model of probability density functions (PDFs) to describe the data structure. The most fundamental techniques in this line are K-Means [101] and EM [47]. These methods are suitable to detect spherically Gaussian clusters and the number of clusters k needs to be specified by the user. The algorithm X-Means [125] extends K-Means by a technique for automatically detecting k founded on information theory. The algorithm G-Means [74] additionally provides detection of non-spherical Gaussian clus-

Table 4.1: Table of symbols.

Symbol	Definition
KM	Kuramoto model.
$\mathcal{D}$	The data set.
x	A data object in the data set $\mathcal{D}$.
x_i	The i-th dimension of the data object x .
N	The number of objects in the data set.
d	The dimensionality of the data set.
k	The number of clusters in a data set.
$x(t)$	The renewal value of object x at time step t during dynamic clustering.
K	The coupling strength among objects in the Kuramoto model.
$N_\epsilon(x)$	ϵ-neighborhood of the object x.
r_c	Cluster order parameter.
C_i	The i-th cluster of the data set.
$\|C_i\|$	The number of object in the i-th cluster.
$\|p_i\|$	The number of free parameters.

ters. Another information-theoretic method, RIC [22], has been designed as a postprocessing step to improve an initial clustering of an arbitrary conventional clustering algorithm. The cluster model comprises a rotation matrix determined by PCA and a PDF assigned to each coordinate selected from a set of predefined PDFs. Clusters with similar characteristics are finally merged. However, the result strongly depends on the quality of the initial clustering and the cluster model is limited to linear attribute correlations and a predefined set of PDFs. Recently, OCI [23] is proposed to detect clusters using Independent Components. This algorithm uses the Exponential Power Distribution as cluster model and applies the Independent Component Analysis for determining the main directions inside a cluster as well as for finding split planes in a top-down clustering approach. All methods in this category tend to fail if the data distribution does not correspond to the cluster model. An alternative largely avoiding restricting assumptions on the data distribution is the idea of Parzen Windows. Local information on the object density is collected by histograms over local windows in the feature space or by local kernel functions, e.g. Gaussian. Thereby, at a global level, arbitrary data distributions can be captured. The algorithm MeanShift [40] combines this idea with clustering: during the run of the algorithm, the windows move towards the direction of the highest object density until convergence. However, the window size needs to be suitably selected and the experiments demonstrate that this algorithm tends to degrade in performance in the presence of outliers and noise points.

Density-Based and Spectral Clustering. In density-based clustering, clusters are regarded as regions of high object density in the data space which are separated by regions of low object density (noise). The algorithm DBSCAN [55] formalizes this idea using two parameters : the neighborhood of a given radius ϵ has to contain at least a minimum number of objects ($MinPts$). Arbitrarily shaped clusters can be easily detected by DBSCAN, however the choice of a suitable settings of ϵ and $MinPts$ is often difficult, especially since the parameters are correlated. Conceptually closely related, spectral clustering refers to a class of techniques which rely on the Eigenstructure of a similarity matrix to partition objects into disjoint clusters. These algorithms, e.g. [116], [14] detect arbitrarily shaped clusters by considering the clustering problem from a graph-theoretic perspective. A clustering is obtained by removing the weakest edges between highly connected subgraphs, which is formally defined by the normalized cut or similar objective functions. Similar to K-Means, the problem of these approaches is to choose a suitable number of clusters and they are sensitive w.r.t. outliers. The approach [14] partially overcomes these problems by proposing a new cost function for spectral clustering based on the error between a given graph partitioning and a clustering result. However, this approach requires sample data with a known partitioning which is not available in most cases. The approach of Affinity Propagation [60] is closely related to spectral and density-based clustering. Each data point is regarded as a node in a network. The algorithm performs clustering by letting the data points exchange messages along the edges of the networks. By message passing, certain data points emerge as so-called exemplars (cluster representatives) and the other data points establish their cluster membership. As input parameter for each data point a real value must be specified characterizing the probability that this particular point is selected as an exemplar. The number of clusters is not only controlled by this parameter but also by the structure of the network.

Kernel density estimation. The kernel density estimation technique (also called Parzen window method) is a non-parametric way of estimating the probability density function of a random variable, which is widely used in various applications, such as machine learning, data mining, pattern recognition, and computer vision [124]. In comparison to parametric estimation where the density is assumed to come to from a given probability distribution function, kernel density estimation has no fixed distribution assumption and depends upon all the data points to reach an estimate. The extent of the influence of each data point depends on the shape of the kernel function adopted and the bandwidth of the kernel. The choice of bandwidth is important since it affects the roughness or smoothness of the kernel histogram that is ultimately generated. A variety of techniques have been developed for bandwidth selection, such as plug-in and cross-validation methods [84].

4.3 Clustering by Synchronization

In this section, *Sync*, is introduced to explore clusters by synchronization. It starts with an introduction of the Kuramoto Model and then develops a variant suitable for clustering. In Section 4.3.3 the algorithm *Sync* is discussed in detail.

4.3.1 Kuramoto Model

As stated in Section 3.2, to understand basic mechanisms responsible for synchronization phenomena, the most successful approach is *Kuramoto model* (KM), first proposed by Kuramoto [94, 95]. The KM model consists of a population of N coupled phase oscillators where the phase of the *i-th* unit, denoted by θ_i, evolves over time according to the following dynamics:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i), (i = 1, \ldots, N), \tag{4.1}$$

where ω_i stands for its natural frequency and K describes the coupling strength between units.

The Kuramoto model well describes the global synchronization behavior of all coupled phase oscillators, which implies that all the oscillators will be in phase finally through mutual coupling. This situation rarely occurs in real-life systems. Phase locking or local synchronization is more often observed. This means a local ensemble of oscillators are synchronized together, where the whole oscillators are divided into several clusters of mutually synchronized oscillators. Moreover, it is observed that the sets of oscillators with high similarity synchronize more easily than those with large variance.

4.3.2 Reformulation of the Kuramoto Model for Clustering

To apply KM for clustering, it is necessary to extensively reformulate $Eq.(4.1)$. Here, the concept of *local synchronization* of phase oscillators needs to be formally introduced to reflect the intrinsic data structure. To introduce the local synchronization, the notion of ϵ-neighborhood is defined.

Definition 4.1 (ϵ-neighborhood of an object x): The ϵ- neighborhood of object x, which denoted by $N_\epsilon(x)$, is defined as:

$$N_\epsilon(x) = \{y \in \mathcal{D} | dist(y, x) \leq \epsilon\}, \tag{4.2}$$

where $dist(y, x)$ is a metric distance function.

Definition 4.2 (Extensive Kuramoto model for Clustering): Let $x \in \mathcal{R}^d$ be an object in the data set $\mathcal{D}$ and x_i be the i-th dimension of the data object x respectively. Each object x is regarded as a phase oscillator. According to Eq.(4.1), with an ϵ-neighborhood interaction, the dynamics of each dimension

x_i of the object x is governed by:

$$\frac{dx_i}{dt} = \omega_i + \frac{K}{|N_\epsilon(x)|} \sum_{y \in N_\epsilon(x)} \sin(y_i - x_i). \tag{4.3}$$

Let $dt = \Delta t$, then:

$$x_i(t + \Delta t) = x_i(t) + \Delta t \omega_i + \frac{\Delta t \cdot K}{|N_\epsilon(x(t))|} \sum_{y \in N_\epsilon(x(t))} \sin(y_i(t) - x_i(t)) \tag{4.4}$$

It is essential that all objects have a common frequency ω since different individual frequencies would disturb or even prevent cluster formation. The term $\Delta t \cdot \omega_i$ can thus be safely ignored. $\Delta t \cdot K$ is a constant and simply set to 1. Finally the dynamics of each dimension x_i of the object x over time is provided by:

$$x_i(t + \Delta t) = x_i(t) + \frac{1}{|N_\epsilon(x(t))|} \cdot \sum_{y \in N_\epsilon(x(t))} \sin(y_i(t) - x_i(t)). \tag{4.5}$$

The object x at time step $t = 0$: $x(0) = (x_1(0), ..., x_d(0))$ represents the initial phase of the object (the original location of object x). The $x_i(t+\Delta t)$ describes the renewal phase value of i-th dimension of object x at the $t = t + \Delta t$ time evolution.

To characterize the level of synchronization between oscillators during the synchronization process, an order parameter needs be defined. The order parameter of Eq.(3.2) is suitable for global synchrony. However, it is not effective to identify local dynamic effects. In particular it cannot bring the deep information of the degree of local synchronization. For this reason, instead of a global observation, a cluster order parameter r_c is defined, measuring the degree of synchronization of the local oscillator population.

Definition 4.3 (Cluster Order Parameter): The cluster order parameter r_c characterizing the degree of local synchronization is provided by:

$$r_c = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|N_\epsilon(x)|}\sum_{y\in N_\epsilon(x)}\mathbf{e}^{-||y-x||}. \tag{4.6}$$

The value of r_c increases as more neighbors synchronize together over time evolution. The dynamic clustering will terminate when $r_c(t)$ converges, which indicates the local phase coherence, also called *local perfect synchronization*. At this moment, all local objects (within in a cluster) have the same phase (location).

4.3.3 The Sync Algorithm

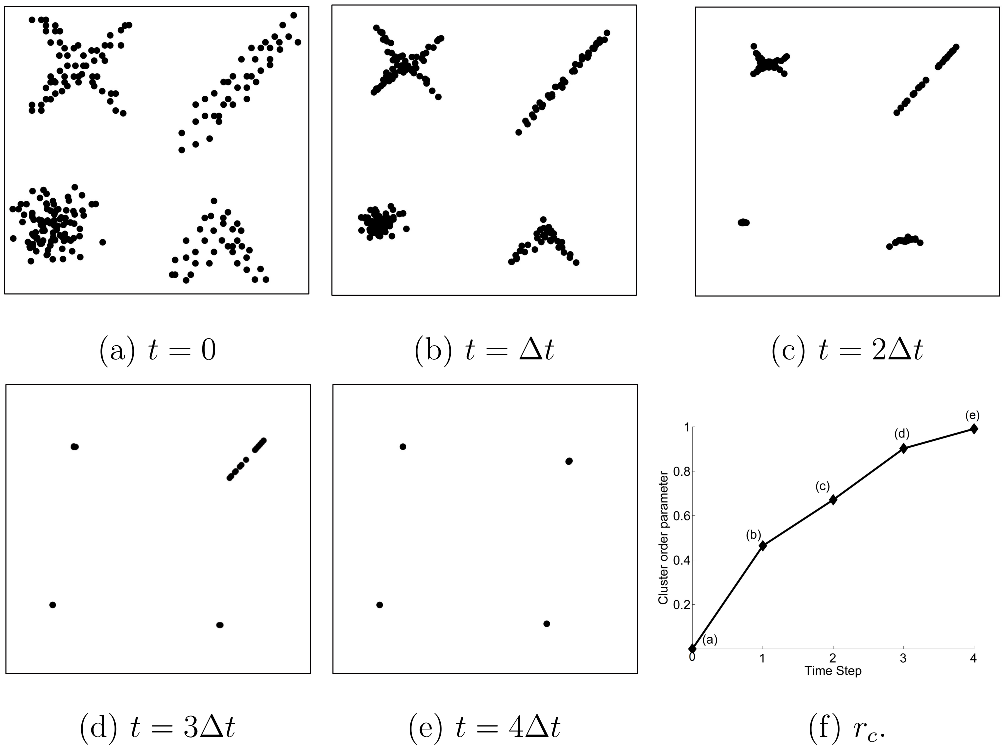
In this section, the *Sync* algorithm based on the reformulated extensive Kuramoto model is presented.

Dynamic Clustering

The basic idea of *Sync* is to regard each object as a non-identical phase oscillator which changes its phase (location) dynamically over time evolution according to *Eq.*(4.5). The process of the dynamic clustering involves the following steps:

1. At initial time ($t = 0$), without any interaction, all objects in the data set have their own phase, which are represented by different values (feature vectors). Thus, there are N disconnected sets (N clusters).

2. As time evolves, each object interacts with its ϵ- neighborhood, cf. Definition 4.1. The interaction strength which each neighbor is deter-



(a) $t = 0$ (b) $t = \Delta t$ (c) $t = 2\Delta t$

(d) $t = 3\Delta t$ (e) $t = 4\Delta t$ (f) r_c.

Figure 4.2: The process of dynamical clustering over time evolution and corresponding cluster order parameter: Diagrams (a)-(e) describe the detailed process of dynamical clustering towards synchronization. Here, the states of objects at the time step $t = (0, 1, 2, 3, 4) \cdot \Delta t$ are presented. Diagram (f) illustrates the corresponding cluster order parameter during dynamical clustering.

mined proportional to the similarity (Eq.(4.5)). Objects with similar attributes synchronize together and form several synchronized clusters gradually following the intrinsic structure of a data set.

3. Finally, the cluster order parameter (Definition 4.3), which character-

izes the level of local synchronization, is used to determine the termination of the dynamic clustering (r_c converges).

During the dynamic process toward synchronization, all objects change their locations through the interaction with similar objects according to $Eq.$(4.5). This means that also the neighborhood of each object changes dynamically over time evolution. The traces of all objects are in line with the main direction of the local data structure. Finally, all objects in each cluster synchronize at a certain common phase. Figure 4.2 (a)-(e) show the detailed process of dynamic clustering of 2-dimensional points from $t = 0$ to $t = 4\Delta t$. $t = 0$ indicates the original data set at the initial time. From that moment on, all objects with similar attributes start to synchronize together through the dynamic interaction and finally, all objects in the data set synchronize at 4 different phases after 4 time steps. The level of local synchronization over time evolution is characterized by the cluster order parameter, which indicated in Figure 4.2 (e). When r_c converges, all objects in a cluster synchronize together (coherence or phase locking). According to empirical studies, in most cases, r_c will converge in 20 time steps. The dynamic clustering is illustrated in Algorithm 1.

Outlier Handling

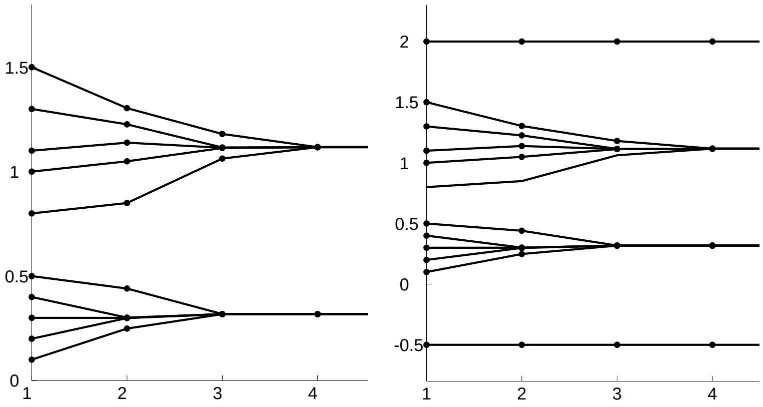
Depending on their dynamic behavior over time evolution, all objects can be divided into two types: *synchronized objects* (objects in several synchronized clusters) and *non-synchronized objects* (outliers). Most objects synchronize with other objects and form synchronized clusters. Objects which remain isolated all the time are regarded as outliers. In order to illustrate the process

Algorithm 1 DynamicClustering

```
Input: Data Set $D$, Interaction Range $\epsilon$

while (loopFlag = True) do
  for (Each object $p \in D$) do
    Compute $N(p)$;
    //Update value
    Obtain new value of object $p$ using Eq.(4.5);
  end for
  //Order parameter
  Compute local order $r_c$ of all objects using Eq.(4.6);
  if $r_c$ converges OR $loopNum > 50$ then
    loopFlag = False;
    $C = synCluster(D')$; //Find synchronized clusters
    $O = setDiff(D', C)$; //Outliers
    $R = \{C, O\}$;
  end if
end while
return $R$;
```

of handling outliers, two simple one-dimensional data sets are used: Without outliers $D_1 = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0, 1.1, 1.3, 1.5\}$, and with outliers $D_2 = \{-0.5, 0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.0, 1.1, 1.3, 2.0\}$. As displayed in Figure 4.3, in D_1, all objects synchronize to different clusters and achieve the phase lock. However, in D_2, owing to the different movement dynamics



(a) (b)

Figure 4.3: Object dynamics plot displaying the different dynamics of cluster and outlier objects. (a): data set without outliers, (b): data set with outliers.

and directions, outlier objects cannot synchronize with other objects and keep their original location over time. Moreover, the plots of dynamic object movement are a concise visualization for the intrinsic data structure including cluster points and outliers.

MDL-based Clustering Model Selection

To explore clusters based on synchronization, an appropriate initial size of the neighborhood for mutual interaction needs to be determined. Although the experiments demonstrate that the clustering result is quite robust against the initial neighborhood size ϵ, the Minimum Description Length [68] principle is also used for fully automatic clustering. More specifically, the MDL principle is applied to compress a set of candidate clustering models M^l, where in our case different models correspond to the clustering results with various ϵ.

To generate a suitable set of models, the following heuristic is applied:

To guarantee a stable interaction of each object, ϵ is initiated with the average value of the k-nearest neighbor distance determined from a sample from the data set for a small k. The ϵ is increased stepwise until all objects synchronize in a cluster. A reasonable step size is determined by the difference between average $(k+1)$-nearest neighbor distance and average k-nearest neighbor distance. Choosing k sufficiently small is recommended and $k = 3$ nearest neighbors are applied in all experiments. From all candidate models, the model resulting in the minimum description length is selected. In the following, how to compress a candidate model is explained in detail.

Given a data set $\mathcal{D}$ and a clustering model M, the fundamental idea of MDL is to exploit the regularities in the data described by M for effective compression of the data. To avoid overly complex models, the overall description length includes not only the cost for coding the data exploiting the model, denoted by $L(\mathcal{D}|M)$, but also the cost $L(M)$ for coding the model M itself.

Assuming there are k clusters, the coding cost or description length for the data $L(D|M)$ is provided by:

$$L(D|M) = -\sum_{i=1}^{k} \sum_{x \in C_i} \log_2(pdf(x)). \tag{4.7}$$

where the $pdf(x)$ means the probability of object x in each cluster. To allow for arbitrarily shaped clusters, we propose to estimate $pdf(x)$ by non-parametric kernel density estimation, a widely used technique with numerous applications in machine learning, pattern recognition and computer vision [124].

For coding the cluster model $L(M)$, the cluster assignment as well as the model parameters need to be specified, which are given as:

$$L(M) = \sum_{i=1}^{k} \sum_{j=1}^{|C_i|} \log_2(\frac{N}{|C_i|}) + \sum_{i=1}^{k} \frac{p_i}{2} \log_2(|C_i|). \tag{4.8}$$

where the first term represents the coding cost for the cluster assignment and the second term is dedicated for coding the parameters. Here p_i is the number of bandwidths for kernel density estimation, which equals the dimensionality of the data set.

Finally, the total coding cost for a clustering model M is:

$$L(D, M) = L(M) + L(D|M). \tag{4.9}$$

The objective is to find the best cluster model minimizing the total coding cost.

$$M^j = \operatorname*{argmin}_{M^j} L(D, M^j). \tag{4.10}$$

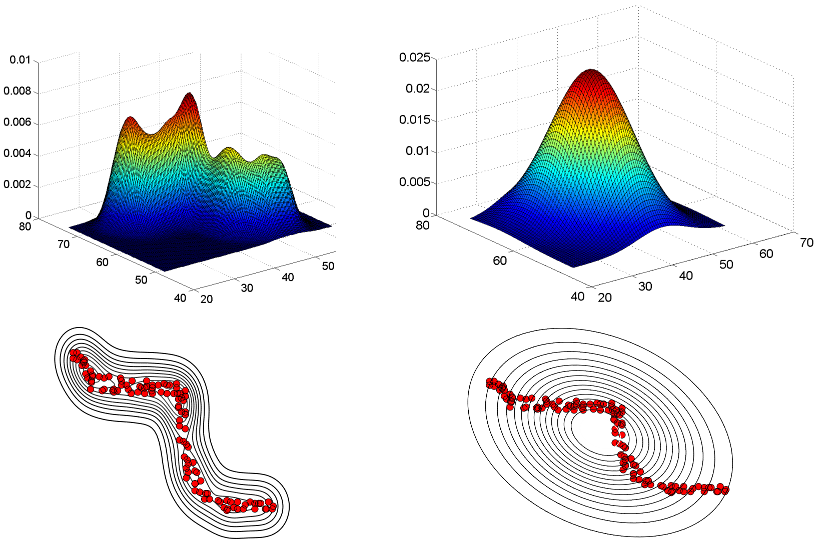
Kernel Density Estimation. The multivariate kernel density estimation (KDE) is defined as follows:

$$\hat{f}(x) = \frac{1}{N} \sum_{y \in \mathcal{D}} \frac{1}{h^d} K(\frac{x - y}{h}). \tag{4.11}$$

where $h = (h_1, \cdots, h_d)^T$ is the bandwidth and the term $K(\cdot)$ is a $d-$dimensional kernel function that is non-negative and integrates to one. As a common choice, the Gaussian kernel with mean 0 and variance 1 is used. More specifically:

$$\hat{f}(x) = \frac{1}{N} \sum_{y \in \mathcal{D}} \left(\prod_{i=1}^{d} \frac{1}{h_i} K\left(\frac{x_i - y_i}{h_i}\right) \right), \tag{4.12}$$

where $K(x) = (2\pi)^{-1/2} \exp(-x^2/2)$. The bandwidth h is selected using an established heuristic which is proven to work well in various applications,



(a) Kernel density estimation (density and contour plot).

(b) Gaussian estimation (density and contour plot).

Figure 4.4: Kernel density estimation vs. Gaussian estimation.

even in the case of outliers and bi-modal distributions [151]: $h_i = 0.9 \cdot N^{-1/(d+4)} \min(\sigma_j, IQR_i/1.34)$, where σ_i is the variance and IQR_i is the interquartile range of dimension i.

Finally, the probability of each object x is given as:

$$pdf(x) = \frac{\hat{f}(x)}{\sum_{y \in \mathcal{D}} \hat{f}(y)}. \tag{4.13}$$

Unlike parametric statistics, KDE allows to robustly estimate the probability for any unknown distribution. Consider for instance the cluster in Figure 4.4, KDE provides an exact and concise representation of the data. In contrast, the Gaussian model is inappropriate for this cluster.

4.3.4 Runtime Complexity

For each dynamic clustering by synchronization, the runtime complexity with respect to the number of data objects is $O(T \cdot N^2)$, where N is the number of objects and T is the time evolution. In most cases, T is small with $5 \leq T \leq 20$. If there exists an efficient index, the complexity reduces to $O(T \cdot N \log N)$. With clustering model selection based on MDL principle, the final complexity of *Sync* is $O(L \cdot T \cdot N \log N)$ and L is the number of different clustering models.

4.4 Experimental Evaluation

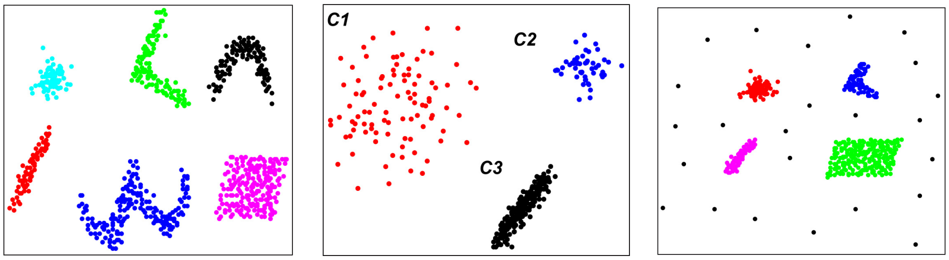
To extensively study the performance of *Sync*, its performance is compared to representatives of various clustering paradigms on synthetic and real-world data. The selected algorithms include iterative partitioning algorithm K-Means [101], the density-based algorithm DBSCAN [55], spectral clustering (SC) [116], the algorithm MeanShift [40] and the affinity propagation algorithm (AP) [60]. For the comparison methods, several parameter settings are tried and the best result is presented. Moreover, *Sync* is compared to several state-of-the-art parameter-free clustering algorithms: To X-Means [125], which extends K-Means by estimation of the number of clusters based on MDL, and to RIC [22] and OCI [23] which have been recently proposed for parameter-free and outlier-robust clustering. For X-Means, RIC and OCI, no parameter settings are required. In all experiments, *Sync* is used in combination with MDL as introduced in Section 4.3.3 which also does not require any input parameters.

Spectral clustering and *Sync* are implemented in Java and the source code of RIC and OCI are obtained from the authors. The source code of the MeanShift algorithm is available at: `http://www.mathworks.com/matlabcentral/fileexchange/10161-mean-shift-clustering` and that of AP at: `http://www.psi.toronto.edu/affinitypropagation/`. K-Means, DBSCAN and X-Means are implemented in WEKA available at `http://www.cs.waikato.ac.nz/ml/weka`. All experiments have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM.

Comparing the results of different clustering algorithms is a non-trivial problem, especially if different algorithms produce results with different numbers of clusters. To provide an objective comparison of effectiveness, EC, an information-theoretic cluster validity measure proposed in [50], is reported. Intuitively, EC corresponds to the number of bits required to encode the class labels when the cluster labels are known, the smaller EC the better is the clustering. Recently, 3 more robust information-theoretic measures for cluster quality have been proposed in [164]: Adjusted Mutual Information (AMI), Adjusted Variation of Information (AVI), and Normalized Variation of Information (NVI). For these measures, higher values represent better clusterings. For more information, please refer to Section 2.1.2.

4.4.1 Synthetic Data

It starts with the evaluation of 2-dimensional synthetic data to facilitate presentation and follows to demonstrate the benefits of *Sync* mentioned in Section 4.1.2.



(a) Arbitrarily shaped. (b) Various densities. (c) Outlier-robust.

Figure 4.5: Clustering driven by the intrinsic data structure.

Clusters Reflecting the Intrinsic Structure

The performance of *Sync* to detect natural clusters in difficult settings is first evaluated, starting with clusters of arbitrary shape and data distribution. The data set displayed in Figure 4.5 (a) consists of 6 clusters: one Gaussian cluster, one correlation cluster and 3 other arbitrarily shaped clusters. *Sync* successfully detects all types of clusters without requiring any input parameters. Moreover, *Sync* allows detecting clusters of very different object density, as Figure 4.5 (b) demonstrates. The performance of *Sync* also does not degrade in the presence of outliers or noise points. Regardless of whether the data set contains noise, such as in Figure 4.3(b) and Figure 4.5 (c) the clustering result remains stable.

Performance on High-dimensional Data

To assess the impact of the dimensionality of the data space on *Sync*, a 5-dimensional synthetic data set (1,000 objects, 5 clusters) is created and successively the dimensionality is increased by adding noise dimensions with uniform data distribution. Table 4.2 presents the results. The performance

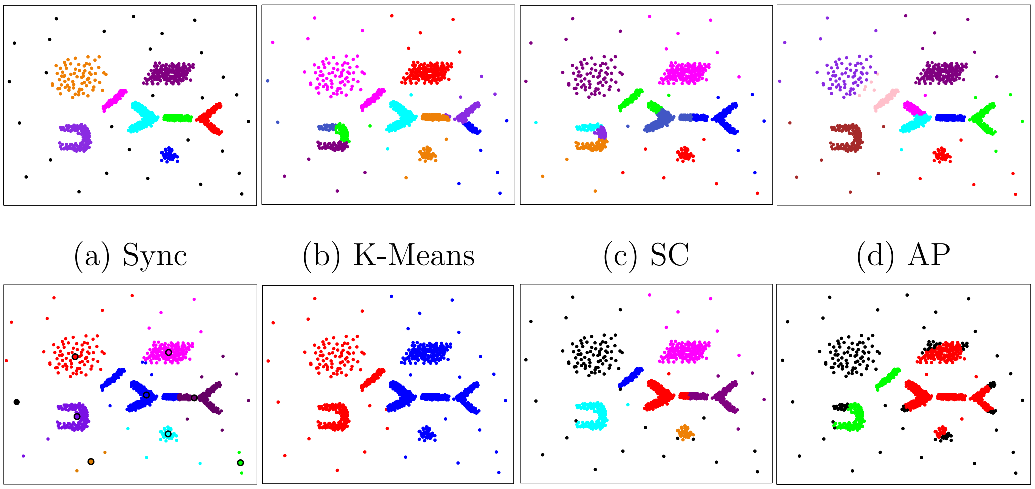
Table 4.2: Performance of *Sync* on high-dimensional data.

Data set	#Dim./#Noise Dim.	#Clusters	#Clusters Detected	NMI	AMI	AVI	EC
DS1	5/0	5	5	1	1	1	0.090
DS2	8/3	5	5	1	1	1	0.090
DS3	10/5	5	5	0.996	0.996	0.998	0.092
DS4	12/7	5	5	0.785	0.783	0.811	0.439
DS5	15/10	5	4	0.557	0.555	0.569	0.801

of *Sync* does not degrade up to a dimensionality of 8 as demonstrated by a constant EC of 0.09 (NMI, AMI, AVI = 1). Also on the 10-dimensional data set, which corresponds to a fraction of one third noise dimensions, 5 clusters are detected but one instance is regarded as a noise point resulting in a slight increase of EG to 0.092 (NMI = 0.996, AMI = 0.996, AVI = 0.998). When the dimensionality increases to 12, more and more objects are viewed as noise but *Sync* still obtains the correct clusters (EC:0.44, NMI:0.785, AMI:0.783, AVI:0.811). *Sync* detects 4 clusters with dimensionality of 15. Two clusters are merged together and some objects are regarded as noise (EC:0.8, NMI:0.557, AMI:0.555, AVI:0.569). In summary, the *Sync* performs very well on high-dimensional data with a fraction of approximately up to 50% noise dimensions.

Comparison to State-of-the-art Clustering

For comparison, a 2-d data set is created consisting of 8 arbitrarily shaped clusters plus noise points, cf. Figure 4.6. For K-Means and spectral clustering, best results have been achieved for $K = 9$ clusters. K-Means cannot

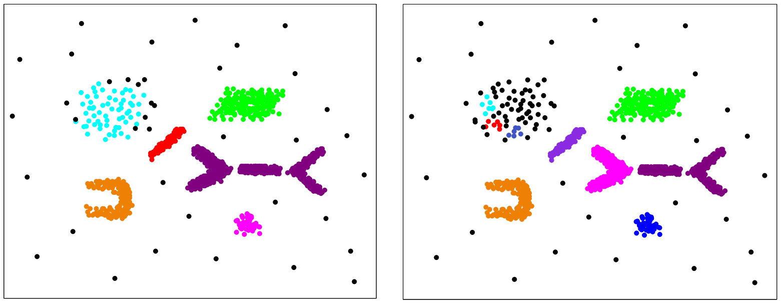


(a) Sync (b) K-Means (c) SC (d) AP

(e) MS (f) X-Means (g) RIC (h) OCI

Figure 4.6: Comparison with various clustering algorithms: K-Means (K=9), Spectral Clustering (SC) (K=9), Affinity Propagation (AP) (K=9), Mean-Shift (MS) (bandwidth = 6.3), X-Means, RIC and OCI.

successfully detect the clusters in this data set which are not limited to Gaussian or spherical data distribution, cf. Figure 4.6 (a). Spectral clustering, which in principle has the potential to detect arbitrarily shaped clusters also performs poorly on the data set cf. Figure 4.6 (b). The reason lies in the fact that the performance of spectral clustering is severely affected by outliers or noise objects. Parameterized as recommended in [60], affinity propagation yields 341 clusters. However, the implementation also allows manually selecting the number of clusters. For comparison with the other algorithms, the result with $K = 9$ clusters is displayed in Figure 4.6 (c). As for spectral clustering, the performance of AF degrades in the presence of noise and outliers. For MeanShift clustering, best results have been achieved setting the window size to 6.3. In Figure 4.6 (d), the areas of estimated major object density are marked by black circles. Similar to SC and AF, the density estimation in Meanshift is disturbed by outliers. For DBSCAN, a wide range of different settings for the parameters $MinPts$ and ϵ are tried. However, to find two suitable parameters is not a trivial task as they are correlated. Fixing $MinPts = 6$ (the default value in Weka) and varying ϵ yielded the best clustering results, displayed in Figure 4.7 (a-b). Since the data includes clusters of various object densities, DBSCAN cannot identify all these clusters at the same time. In comparison with these established clustering algorithms, *Sync* automatically and correctly detects all clusters and noise points driven by the synchronization principle, cf. Figure 4.6 (e).



(a) DBSCAN(ϵ=0.035) (b) DBSCAN(ϵ=0.025)

Figure 4.7: The result of DBSCAN with different parameters on the synthetic data.

In the experiments displayed in Figure 4.6 (f-h), the performance of *Sync* is further compared to various parameter-free clustering algorithms. As evident from Figure 4.6 (f), X-Means fails to detect the clusters since they are not Gaussian and the data set contains noise points. The cluster model of RIC is limited to linear attribute correlations and a predefined set of PDFs,

Table 4.3: Performances on synthetic data.

Algorithms	SynC	K-Means	SC	AP	MS
NMI	1	0.8493	0.7702	0.8837	0.7034
AMI	1	0.8472	0.767	0.8822	0.6999
AVI	1	0.8527	0.7732	0.9103	0.7957
Algorithms	X-MEANS	OCI	RIC	DBSCAN(0.035)	DBSCAN(0.025)
NMI	0.2882	0.3241	0.8342	0.7318	0.9
AMI	0.287	0.3219	0.8325	0.729	0.8985
AVI	0.4422	0.4427	0.8828	0.8364	0.9338

which does not fit to our example data set. Therefore, RIC does not perform very well on this example, cf. Figure 4.6 (g). Also the performance of OCI is not satisfying since again, the assumption of the data distribution, in this case Exponential Power Distribution, does not fit to our example data set, cf. Figure 4.6(h).

4.4.2 Real-world Data

In this section, the performance of *Sync* is evaluated on real-world data publicly available at the UCI machine learning repository (`http://archive.ics.uci.edu/ml`). Due to difficult parametrization, only the comparisons to the parameter-free clustering algorithms are provided.

Wisconsin Data

The Wisconsin data set deriving from a study on breast cancer consists of 683 instances which are labeled to the classes malignant (M: 239 instances) and benign (B: 444 instances) (16 instances with missing values have been removed from the original data set). Each instance is described by 9 numerical attributes.

Sync detects two clusters successfully. The first cluster with 433 objects represents the class benign, the second cluster with 250 objects the class malignant. In total, 23 instances have been wrongly clustered which results in an EC of 0.154. X-Means detects 3 clusters (C1:197(M:23,B:174), C2:261(B:261), C3:225(M:216 B:9)). 32 instances have been wrongly clustered with EC of 0.183. With RIC, five clusters are obtained with EC of 0.182. On the data set, also the OCI algorithm obtains a good result comprising 13 clusters with good class purity, as reported in [23]. This clustering result has an equally good EC(0.154) as the result of *Sync*. However, the detected 2 clusters by *Sync* correspond to the expert rating that there are 2 classes in the data set.

In total, *Sync* achieves all aims that people want: (1) *Sync* automatically finds the correct number of clusters; (2) *Sync* detects natural clusters in the data (with high EC value); (3) *Sync* discovers almost all objects of each cluster with high recall (96.2% and 97.5%); (4) all instances in each cluster match with corresponding type (with highest precision of 98.6% and 93.2%). Moreover, the other three measures also demonstrated its advantages in Table 4.4.

Diabetes Data

The Pima Indian diabetes database is a collection of medical diagnostic reports of 768 samples from a population living near Phoenix, Arizona, USA.

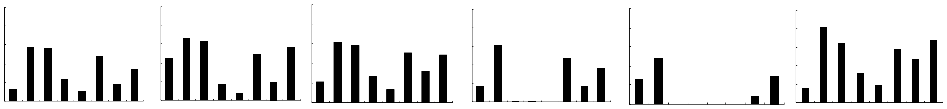


Figure 4.8: Illustration of the result of *Sync* on diabetes data: each bar in each of the 6 clusters indicates the mean value of different factors and is scaled to [0,1]. The eight factors correspond to: number of pregnancies, plasma glucose concentration, diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin, body mass index, diabetes pedigree function and age, respectively.

Table 4.4: Performances on Wisconsin data and diabetes data.

Algorithms	Wisconsin Data				Diabetes Data			
	SynC	X-Means	OCI	RIC	SynC	X-Means	OCI	RIC
NMI	0.7767	0.3238	0.2742	0.3441	0.0514	0.0512	0.0319	0.0109
AMI	0.7765	0.3223	0.2716	0.3428	0.0481	0.0503	0.0307	0.009
AVI	0.7821	0.464	0.4112	0.4747	0.0582	0.0508	0.038	0.0111

Each sample consists of eight significant risk factors which were chosen for forecasting the onset of diabetes, including e.g. the number of pregnancies, the diastolic blood pressure, the diabetes pedigree function, age, etc. These samples are labeled to 2 classes (Positive:268; Negative:500), namely whether the patient is tested positive for diabetes or not according to World Health Organization criteria. *Sync* detects 6 clusters on the data set. Figure 4.8 summarizes the cluster contents. Each bin represents the scaled average of each attribute within the cluster. The first cluster consists of 477 samples and mainly hosts the healthy subjects (365 samples). This group is characterized with the relative low value of these eight risk factors, especially for number of pregnancies and the body mass index which is characteristic for young people. *Sync* assigns 118 out of 216 samples of the patient group to cluster 2, who are mainly middle-aged people who have a high number of pregnancies, and a high plasma glucose concentration. Cluster 3 and Cluster 6 are mainly hosting subjects with diabetes, who are characterized by high diabetes pedigree function, skin fold thickness and age. Subjects in cluster 4 are characterized by a high plasma glucose concentration a 2 hours in an oral glucose tolerance test and missing measurements for blood pressure, skin fold thickness and 2-hour serum insulin. Moreover, Cluster 5 hosts people who have no measurement of blood pressure, skin fold thickness, 2-hour serum insulin and body mass index. In total, the clustering results in an EC of 0.625. X-Means merges all instances in one big cluster. RIC and OCI detect 3 and 4 clusters, respectively and corresponding EC values are 0.661 and 0.635. In summary, *Sync* detects meaningful clusters and yields best results with the lowest EC value. For a further comparison of the clustering results, Table 4.4 provides a summary of the quality criteria introduced in [164]. As with EC, *Sync* clearly outperforms the comparison methods on most data sets.

4.5 Conclusion

In this chapter, the partitioning clustering algorithm *Sync* is introduced based on the synchronization principle. The basic idea is to regard each data object as a phase oscillator and simulate the dynamical object interaction over time. To characterize the object interaction, an adapted variant of the Kuramoto model suitable for clustering is proposed. The extensive experiments demonstrate that the *Sync* algorithm shows several desirable properties and outperforms most of the art-of-state clustering algorithms.

Chapter 5

Exploring Natural Hierarchies of Synchronized Cluster

In Chapter 4, the flat structure of a data set is analyzed using *Sync*. In real-life, the structure of a data set may be complex and can be better represented as a hierarchy. It is not sufficient to analyze the cluster structure only on one scale. Hierarchical clustering algorithms have thus been proposed to find the compact representations with different scales and provide deeper insights into the data structure.

In nature, synchronization is a powerful and inherently hierarchical concept regulating a large variety of complex processes. In the beginning of the dynamical synchronization process, each object interacts with its local neighborhood only. As time evolves, small distinct groups of similar objects emerge which are gradually combined to form larger groups. In this chapter, the algorithm *hSync*, is introduced by extending the concept of synchronization to the hierarchical data analysis. Similarly, each data object is regarded as a phase oscillator and the dynamical behaviors of the objects over time are simulated. By interaction with similar objects, the phase of an object gradually aligns with its neighborhood, resulting in a non-linear object movement naturally driven by the local cluster structure. The inherently hierarchical nature of object movement allows exploring the hierarchical cluster structure at several levels of abstraction. Compared with existing hierarchical clustering algorithms, *hSync* robustly reveals the natural cluster hierarchy regardless of size and data distribution of the clusters. The MDL principle guides *hSync* to identify meaningful levels of the cluster hierarchy corresponding to high-quality clusterings.

The remainder of this chapter is organized as follows: After an introduction to the basic idea of the novel hierarchical clustering in Section 5.1, the related work on hierarchical clustering is given in Section 5.2. In Section 5.3, the algorithm of *hSync* is elaborated. Section 5.4 contains extensive experimental evaluations on synthetic and real data sets. Finally, Section 5.5 concludes this chapter. Parts of the material presented in this chapter have been submitted in [147], where J.S. was mostly responsible for the development of the main concept, implemented the main algorithms and wrote the largest parts of the paper; X.H. implemented the main part of hierarchical algorithm; C.P., C.B. and Q.Y. also contributed to the conceptual development and paper writing.

> "*Junming Shao, Xiao He, Christian Böhm, Qinli Yang, and Claudia Plant. Synchronization-inspired Partitioning and Hierarchical Clustering. Submitted for publication.*"

5.1 Introduction

As stated in Section 2.1.1, partitioning clustering is efficient and conceptually simple, but is often difficult to unveil the natural cluster structure of complex hierarchical data. Hierarchical clustering outputs a hierarchy, a structure that is more informative than the unstructured set of clusters returned by partitioning clustering. During the last decades, hierarchical clustering has become very popular in various scientific disciplines, such as molecular biology, medicine, or economy. However, well-known hierarchical clustering algorithms like Single Link [82] often fail to detect the true clusters for a real data set. The dendrogram generated by Single Link is usually very complex and difficult to interpret. Moreover, the performance of many hierarchical clustering algorithms are very sensitive to noise and outliers.

How can we find a natural, compact and meaningful representation of a given hierarchical data set? In this chapter, *hSync*, a new clustering algorithm to hierarchial data analysis, is proposed. Based on natural synchronization phenomena, the synchronization dynamics is exploited to obtain a natural hierarchical representation of a given data set. The key idea is to regard each data object as a coupled phase oscillator and each object interacts dynamically with similar objects on different levels. The process of the hierarchical clustering inspired by synchronization involves the following stages: Starting from initial conditions, each object runs independently with its own phase. As time evolves, those objects with highest density forming local clusters will synchronize together with a small interaction range. Then, in a sequential process, more and more objects synchronize together and clusters are produced with a larger interaction range. Finally, the whole population will synchronize together and have a common phase. Thus, with different interaction ranges, the dynamical process of synchronization reveals the whole structure of the data set on all scales, from the micro-scale at an early stage up to the macro-scale. At each scale, outliers are effectively detected since they exhibit differently and hardly synchronize with any of the cluster objects. The principle of synchronization thus allows detecting a natural clustering of complex multi-scale data sets with outliers.

To illustrate cluster formation by synchronization at each scale, Figure 5.1 displays 6 snapshots of the simulated dynamical object movement. For simplicity, 2-dimensional data are displayed. Specifically, for illustration, the 3 objects ($P1$, $P2$, $P3$) are considered in detail, where $P1$ is an object in cluster $C1$ having high object density, $P2$ is an object in cluster $C2$ having relatively low density and $P3$ is a global outlier, respectively. At the first level, given a small interaction range, each object interacts with its neighbors. For example $P1$ interacts with the objects highlighted by a blue dash circle (See Figure 5.1(a)). The movement of the object is determined by oscillation phases within of its neighborhood. Through the non-linear interaction, all objects in $C1$ finally synchronize together. In contrast, for objects with less local density such as objects $P2$ and $P3$, they are not likely interacting with other objects and therefore keep the original phases corresponding to their original position over time. Figure 5.1(b) displays the old and the new positions of the objects for comparison. The initial points (black color) are replaced by the red points after one time step. The objects with highest similarity gradually synchronize together through mutual coupling. Figure 5.1(c) depicts the final states of the objects. The objects in cluster $C1$ are

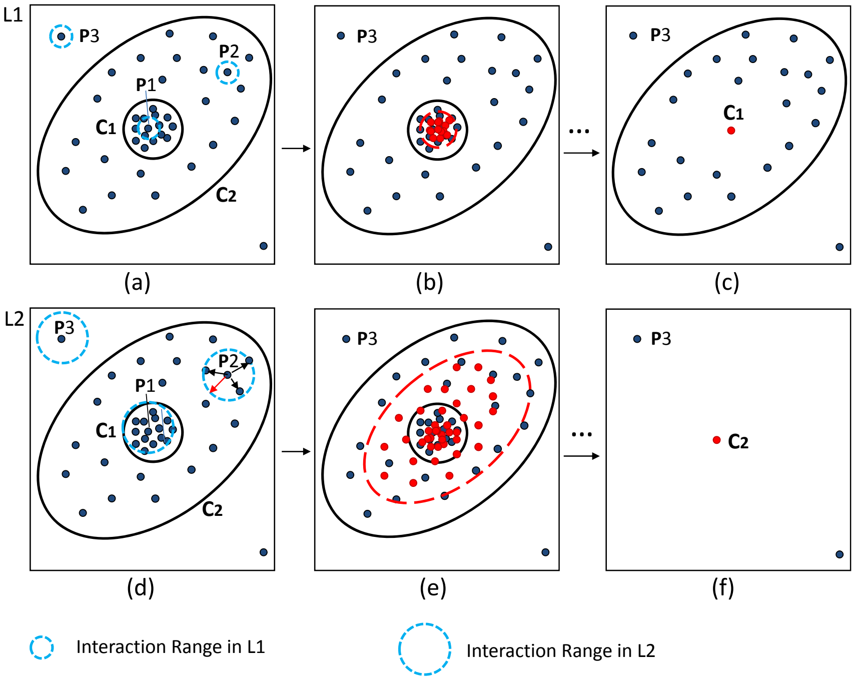


Figure 5.1: Illustration of hierarchical cluster structure exploring by synchronization. (a)-(c). The dynamics of objects during the process towards synchronization in the first level. (a) The initial state of the objects: the blue dashed circles indicate the interaction range for each object. (b) The comparison of objects states before and after one time step: black points represent the initial state, red points indicate the new state. (c) The final state of objects: a synchronized cluster $C1$ and noise points. (d-f). The dynamics of objects in the second level with larger interaction range. (d) The initial state of the objects. Here, for illustration, the black arrows indicate the mutual interaction and red arrow indicates the direction of movement for the sample object $P2$. (e) The comparison of objects states before and after one time step at the second level. (f). The final state of objects in second level: a new synchronized cluster $C2$ and two outliers.

synchronized and have the same phase (location) while outlier objects remain isolated over time, e.g. $P3$. Similarly, with a larger interaction range, more and more objects will synchronize together. Take an object $P2$ for example, in contrast with the first level, it can interact with its neighbors (Figure 5.1(d)). Due to the interactions with its neighbors, $P2$ will move towards the center of the cluster driven by the local data structure. Finally, the cluster of lower density will gradually synchronize as indicated by a common oscillation phase ((Figure 5.1(f))). Since $P3$ is isolated, it keeps its original phase and location during the time revolution and can be easily identified as a global outlier object. In Chapter 4, the idea of synchronization is introduced for flat clustering. In this chapter, this idea is extended to a general framework not only supporting partitioning but also hierarchical clustering.

Contributions

The powerful paradigm of synchronization has many attractive benefits for hierarchical clustering, most importantly:

1. *Clustering driven by the intrinsic data structure.* The simulated non-linear object movement faithfully follows the intrinsic data structure. Thereby our synchronization-based hierarchical algorithm *hSync* reliably detects clusters of arbitrary number, shape, size and data distribution, even in difficult settings with large amounts of noise points and outliers.

2. *Robust discovery of natural cluster hierarchies.* The inherent hierarchical nature of synchronization allows an intuitive and effective approach

for hierarchical clustering. The algorithm *hSync* explores the hierarchical cluster structure from micro-scale to macro-scale by simulating the way to synchronization over different levels of locality.

3. *Compact and interpretable cluster hierarchies.* In combination with MDL, the algorithm *hSync* generates an interpretable cluster tree only consisting of meaningful levels, each representing a clustering of high quality. Besides the cluster tree, the output of *hSync* includes the locality-quality diagram, a visualization which allows the user to comprehensively assess the quality of the cluster hierarchy over all levels.

5.2 Related Work

Hierarchical clustering algorithms decompose a data set into several levels of partitions, representing by a dendrogram. One of the most well-known hierarchical clustering approaches is Single-Link [82]. Starting with the clustering obtained by placing every object in a unique cluster, in every step two closest clusters are merged until all objects are in a whole cluster [115]. For the merging criterion several alternatives have been proposed, such as Average-Link and Complete-Link (for a detailed survey see [115]). The hierarchy obtained by the merging order is visualized as a dendrogram. For a real data set, the dendrogram is often very complex. If a large data set has N objects, the generated dendrogram contains $N-1$ layers and thus it is difficult to find optimal splitting levels that correspond to meaningful clusters. Outliers may also cause the so-called single-link effect that two clusters are difficult to be separated if there is a chain between the two clusters.

The technique CURE [69] utilizes multiple representative points to evaluate the distance between clusters to exploit arbitrary shaped clusters and avoid the so-called single-link effect. However, for a given data set, it is still difficult to define appropriate splitting levels which correspond to meaningful clusters. Furthermore, the dendrogram is created to indicate the clustering process, and does not display the true hierarchical structure of a data set.

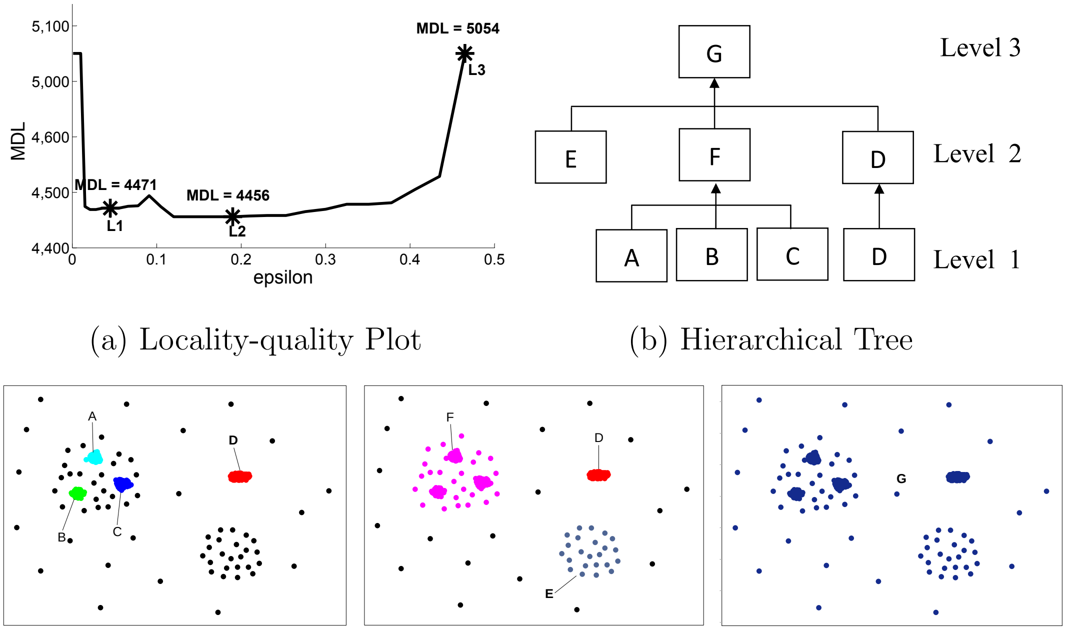
The well known OPTICS algorithm [8] analyzes the hierarchical data from the perspective of density (cf. Section 2.1.1). It provides the reachability plot to give a more intuitive and transparent way to visualize the hierarchical cluster structure for large data sets. However, for many real data sets, the reachability plot is very smooth and cannot find the hierarchal clusters.

5.3 hSync: Discovering Interpretable Cluster Hierarchies

In the section, relying on the concept of synchronization and the Minimum Description Length (MDL) principle, a new algorithm *hSync* is introduced for detecting meaningful levels of the hierarchical cluster structure.

5.3.1 Key Observation

For the partitioning clustering algorithm *Sync*, it starts the dynamical interaction among objects with a small value of ϵ and then increases it stepwise until all objects synchronize in a cluster. Minimum Description Length (MDL) is used to find the best cluster structure: whenever the clusters are



(a) Locality-quality Plot (b) Hierarchical Tree

(c) Level 1 (d) Level 2 (e) Level 3

Figure 5.2: Exploring hierarchical structural data.

good representation of the data structure, they can be used for efficient coding (or compression) of the data set, which results in the minimal MDL value. Actually, the MDL principle can also be linked to hierarchical data analysis, not linking the global minimal MDL value, but all local stable minimal MDL values. The key observation is that if a data set exhibits a hierarchical cluster structure, the MDL values show several distinct stable local minima.

Specifically, when the interaction range ϵ starts with a small value, objects with highest density will synchronize together and are regarded as a cluster (See Figure 5.1(a - c) in Introduction Section 5.1). If the synchronized objects form reasonable clusters reflecting the data structure at the micro-scale, the coding costs of the clusters will result in a local relatively low MDL value. By increasing of the ϵ with the step size $\Delta\epsilon$, if there exists a hierarchal structure of the data, a period of the interaction ranges ϵ will result in the similar clustering results, which thus result in a period of stable MDL values. Then, in a sequential process, with the further increase of the ϵ, more and more objects with less local density will tend to synchronize together (c.f. Figure 5.1(d - f)). Equally, if these new synchronized clusters indicate a meaningful level of the hierarchical structure of data, it will result in a new period of relatively low and stable MDL values for a range of ϵ. Finally, all objects may merge together with enough interaction range ϵ.

5.3.2 Exploring the Hierarchical Cluster Structure

Based on the above observation, the MDL values obtained with different interaction ranges ϵ indicate meaningful levels of the cluster hierarchy. Since the MDL values specify the coding costs or compression ratio of a clustering, the local minima of the MDL values indicate high-quality clusterings representing meaningful levels of the hierarchy. Multiple periods of stable ranges of MDL values strongly support the hierarchical nature of a cluster structure. The *locality-quality plot* visualizes the MDL values for increasing ϵ. From this plot, for each local and stable range of minimal MDL values, the corresponding clusters can be easily determined. To robustly explore the cluster structure, the central ϵ associated with a local stable range of MDL values is selected as a representative denoted by ϵ^{KEY}. After determining these representatives ϵ^{KEY}, the corresponding clusters can be easily determined. These clusterings form a compact and interpretable hierarchical tree. Formally, a representative ϵ^{KEY} and the locality-quality plot are defined as follows.

Definition 5.1 (Representative ϵ^{KEY}): Let $\mathcal{J}$ be the set of all intervals $J = (\check{\epsilon}, \hat{\epsilon})$ where $MDL(\epsilon)$ is sufficiently constant, i.e. where $\frac{\partial}{\partial\epsilon} MDL(\epsilon) \leq \delta$. Then the mean of each of these intervals defines a representative ϵ^{KEY} with $\epsilon^{KEY} = \frac{1}{2}(\check{\epsilon} + \hat{\epsilon}) \quad \forall J \in \mathcal{J}$.

Definition 5.2 (Locality-quality Plot): The x-axis of this plot represents clustering results generated by increasing ϵ from a small start value with step size $\Delta\epsilon$. The y-axis displays the corresponding MDL values. Furthermore, representatives ϵ^{KEY} corresponding to meaningful hierarchy levels are included.

For illustration, Figure 5.2 shows a 2-dimensional data set with hierarchical structure. Figure 5.2(a) displays the locality-quality plot. It is obvious that there exist two stable and local minimal ranges of MDL values. The representative ϵ^{KEY} are detected as in Definition 5.1 (see the star points in Figure 5.2(a)). The corresponding hierarchical tree of the data set is shown in Figure 5.2(b). Figure 5.2(c)-(d) further illustrates the detected clusterings with the the representatives ϵ^{KEY} at different levels. Finally, the pseudocode of hierarchical clustering algorithm *hSync* is displayed in Algorithm 2.

5.3.3 Parameter Selection

In order to explore data structure inspired by synchronization, the interaction range needs to be determined. The MDL principle is thus used to compress a set of candidate clustering models M^l which correspond to the clustering results with various ϵ. The ϵ is initiated with a small value and then increased with a reasonable step size $\Delta\epsilon$. The question is: what is the impact of the

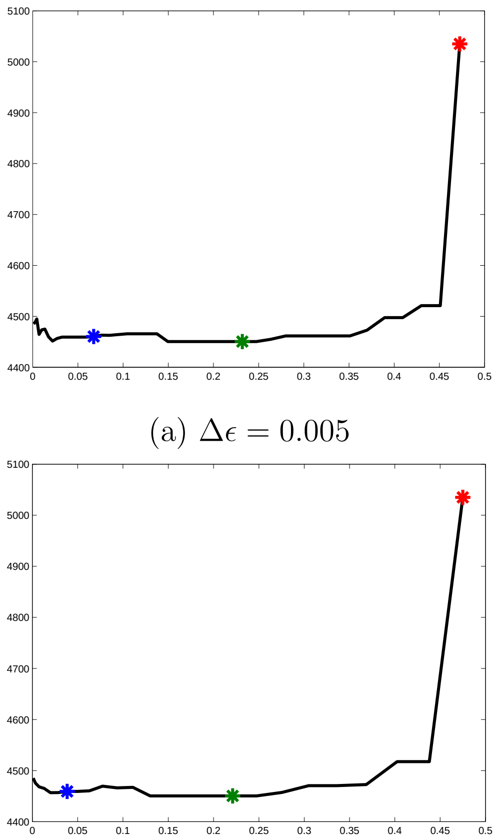
Algorithm 2 Hierarchical Clustering Algorithm hSync

```
Input: Data Set $D$

$l = 0, \epsilon^0$=NN(k); //Initialization
while (No global synchronization) do
    [$R^l$]=DynamicClustering($D, \epsilon^l$);
    if ($R^l$.clusterSize==1) then
        global synchronization = True;
    end if
    Compute the coding cost $L(D, R^l)$;
    $\epsilon^l = \epsilon^l + \Delta\epsilon$; //Increase $\epsilon$: $\Delta\epsilon$ = (NN(k+1) - NN(k))
    $l = l + 1$;
end while
Calculate the representative $\epsilon^{KEY}$ with Definition 5.1;
Generate the Hierarchical tree;
return $R^{\epsilon^{KEY}}$;
```

step size $\Delta\epsilon$ on clustering?

As illustrated in Section 4.3.3, the step size $\Delta\epsilon$ can be heuristically determined by the difference between average $(k+1)$-nearest neighbor distance and average k-nearest neighbor distance. To systematically examine the impact of the step size on clustering, the variation of MDL values with different $\Delta\epsilon$ is studied. Figure 5.3 shows different locality-quality plots obtained by increasing with different step size (from $\Delta\epsilon = 0.005$ to $\Delta\epsilon = 0.02$) on the data set as indicated in Figure 5.2. Obviously, the hierarchical pattern of MDL



(a) $\Delta\epsilon = 0.005$

(b) $\Delta\epsilon = 0.01$

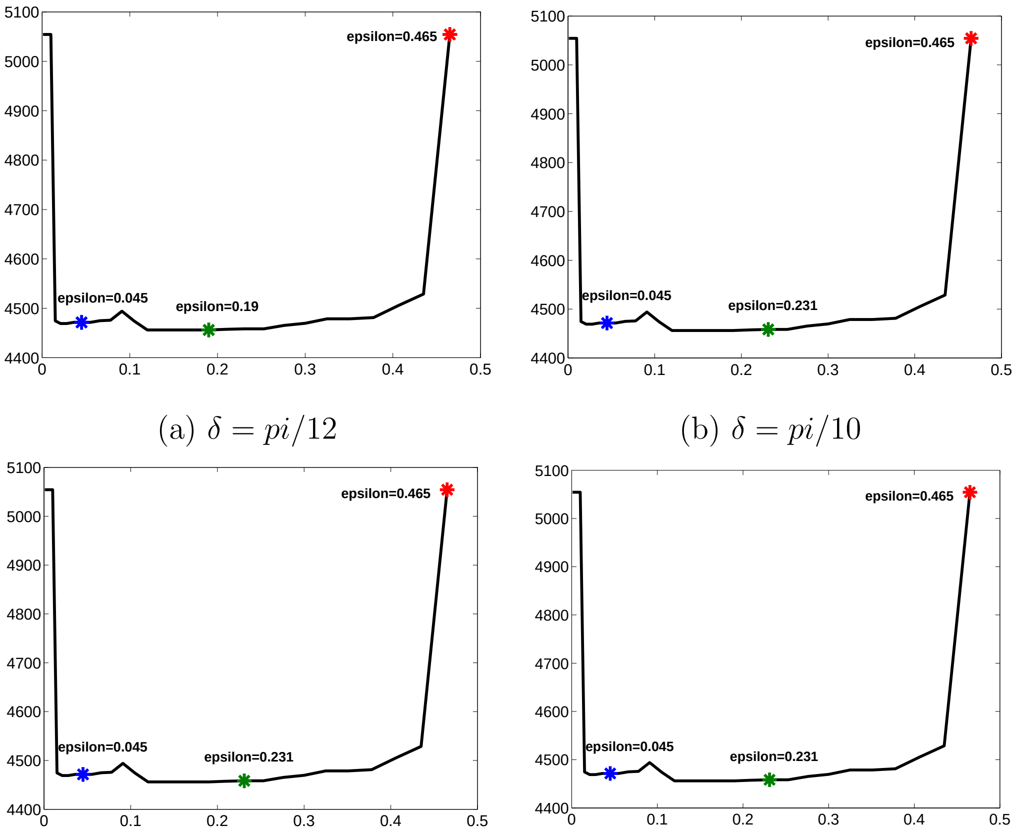
(c) $\Delta\epsilon = 0.015$

(d) $\Delta\epsilon = 0.02$

Figure 5.3: The impact of different step size $\Delta\epsilon$ on clustering by investigating the corresponding locality-quality plots.

values is very similar for different step sizes. Therefore, the corresponding representative ϵ^{KEY} indicating the clustering structures are the same. Thus, the result is very robust against $\Delta\epsilon$ which shows that our heuristic is sufficient to provide accurate results.

Furthermore, the impact of different slope thresholds on clustering is evaluated. Figure 5.4 displays three locality-quality plots obtained with different



(a) $\delta = pi/12$

(b) $\delta = pi/10$

(c) $\delta = pi/8$

(d) $\delta = pi/6$

Figure 5.4: The impact of different slope threshold δ on clustering by checking the locality-quality plots.

slope thresholds (from $\delta = pi/12$ to $\delta = pi/6$) on the same data set. It is interesting to note that the slope threshold is very robust in relation to the clustering results since the corresponding representative ϵ^{KEY} are nearly the same in all plots.

5.3.4 Runtime Complexity

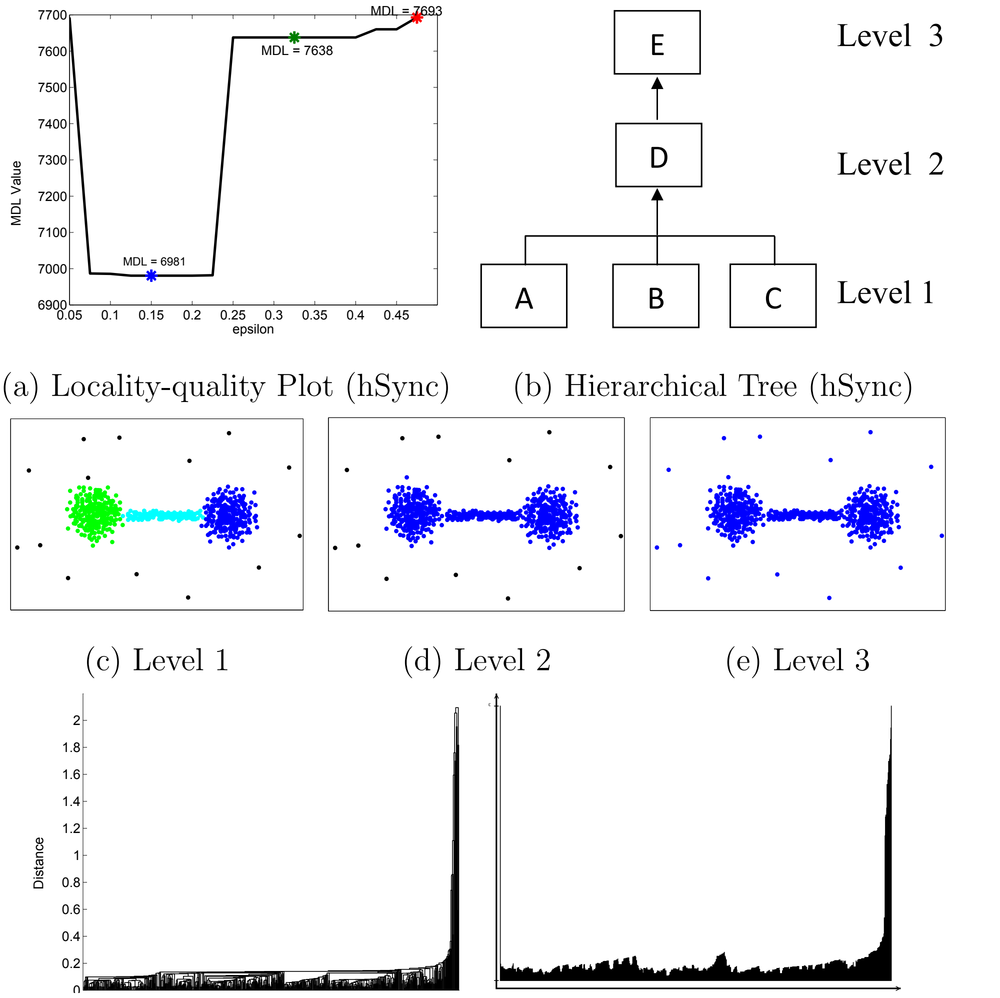
For each dynamical clustering by synchronization, the runtime complexity with respect to the number of data objects is $O(T \cdot N^2)$, where N is the number of objects and T is the time step. In most cases, T is small with $5 \leq T \leq 20$. If there exists an efficient index, the complexity reduces to $O(T \cdot N \log N)$. With meaningful hierarchical clustering exploring based on MDL principle, the final complexity of *hSync* is $O(L \cdot T \cdot N \log N)$ and L is the number of different clustering models.

5.4 Experimental Evaluation

To extensively study the performance of the algorithm *hSync*, they are compared to two representatives of various hierarchical clustering paradigms on synthetic and real-world data: Single-Link [82] and OPTICS [8]. *hSync* is implemented in Java. OPTICS is implemented in WEKA available at `http://www.cs.waikato.ac.nz/ml/weka`. Single Link is implemented in Matlab. All experiments have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM.

5.4.1 Synthetic Data

To evaluate the performance of *hSync* on hierarchical data set, a simple two-dimensional data is generated with the so-called single-link effect and noise, which are two known problems with hierarchical clustering. Here, *hSync* successfully copes with both problems and finds the clear hierarchy, which is illustrated in Figure 5.5(b). It is obvious that two stable ranges of local



(a) Locality-quality Plot (hSync) (b) Hierarchical Tree (hSync)

(c) Level 1 (d) Level 2 (e) Level 3

(f) Dendrogram (Single-Link) (g) Reachability-Plot (OPTICS)

Figure 5.5: Exploring the Single-Link effect data with different hierarchical clustering algorithms.

minimal MDL values exist in the Locality-quality Plot (cf. Figure 5.5(a)). Therefore, the cluster structure and corresponding clusters in different levels are easily obtained with the three representative ϵ^{KEY} (see Figure 5.5(c)-(e)). For comparison, the two popular hierarchical clustering algorithms

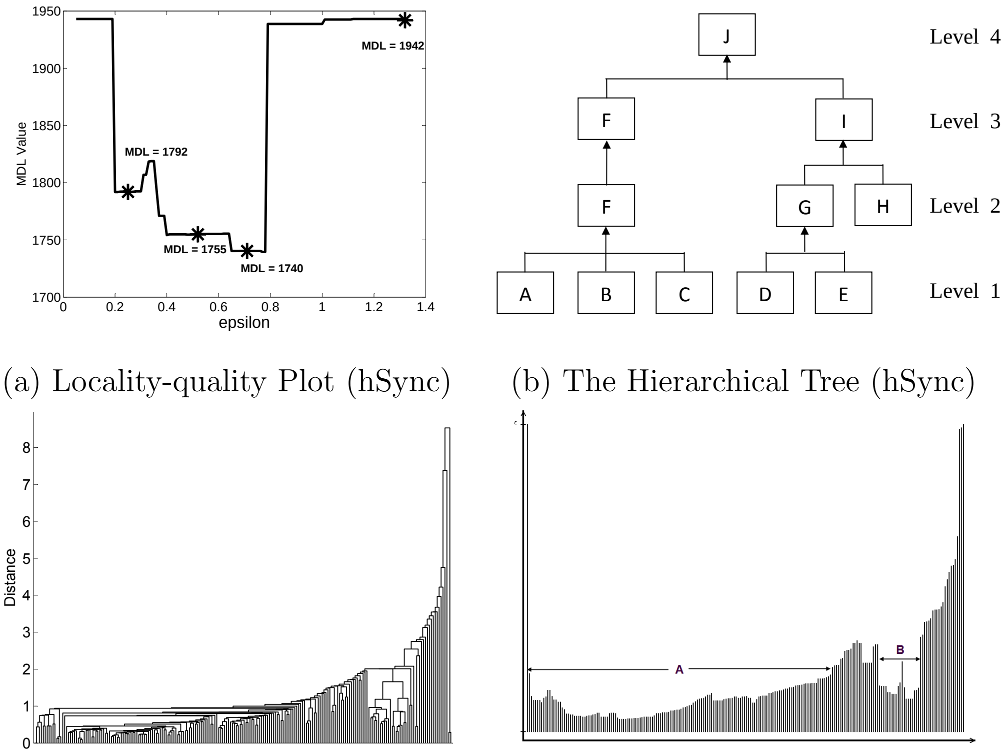
Single Link and OPTICS are also applied to the data set. For Single Link, there is no obvious cluster structure visible in the dendrogram (cf. Figure 5.5(f)) due to the massive single-link effect. Also with OPTICS, it is not easy to detect the cluster hierarchy, especially because of the noise points, see Figure 5.5(g). Best results have been obtained using $\epsilon = 0.8$ and $MinPts$ = 6. Note that *hSync* is especially robust against outliers and noise points, since those points do not synchronize with any of the clusters but keep their original phases during time evolution.

5.4.2 Real-world Data

In this section, we evaluate the performance of *hSync* on real-world data publicly available at the UCI machine learning repository (`http://archive.ics.uci.edu/ml`).

Glass Data

The data set comprises 9 attributes representing different physical and chemical properties. 214 instances are labeled to 7 classes (no instance for class of vehicle windows) representing various types of glass of this data set. Since the real data set exhibits a hierarchical structure, the hierarchical clustering algorithm *hSync* is performed and the results are further illustrated in Figure 5.6. At the first scale, clusters (A)-(C) start to emerge from the data set when ϵ ranges from 0.2 to 0.3, which represent the classes "building windows float processed" and "building windows non float processed" and "vehicle windows float processed" respectively. At the same time, Cluster (D) and Cluster (E) are detected at this level, which mainly represent in-



(a) Locality-quality Plot (hSync)

(b) The Hierarchical Tree (hSync)

(c) Dendrogram (Single Link)

(d) Reachability-plot

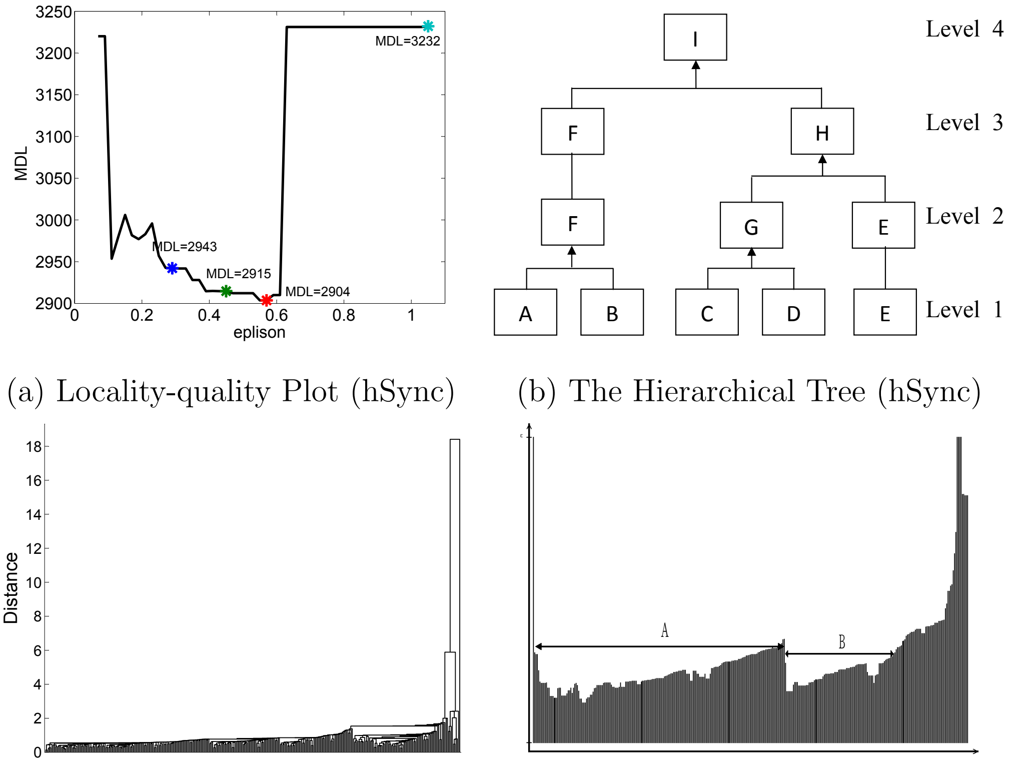
Figure 5.6: Hierarchical Structure of Glass Data detected by different approaches. (a) The plot of the variation of MDL values w.r.t. ϵ with *hSync*. (b) The hierarchical cluster structure generated by *hSync* according to the MDL values explored. (c) The dendrogram with Single Link, in which it is hard to detect hierarchy of the data. (d) The reachability-plot of OPTICS algorithm. Approximately two clusters (A, B) can be detected from the plot.

stances of the classes "non-window glass containers" and "non-window glass tableware". Since the three clusters (A)-(C) are very similar, they are further merged together as the cluster [F] ("window glass") when ϵ at the ranges from

0.4 to 0.62. Similarly, the cluster (D) and cluster (E) are also synchronized together with cluster [G] at the second level. In addition, another cluster [H] is formed, which mainly belongs to the class ”head-lamps”. With further increase of ϵ, over the range from 0.64 to 0.78, the cluster [G] and cluster [H] are merged as a cluster [I], which is characterized as the ”non-window glass”. Finally, most instances (except a few outliers) are formed as a cluster when $\epsilon \geq 1.2$ at the fourth level. For comparison, Figure 5.6(c) displays the reachability-plot generated by OPTICS with $MinPts = 7, \epsilon = 1.5$ of this data set. Only two clusters are visible and the class ”tableware” is well separated forming a distinct cluster (marked by B). The other clusters are much less pronounced. For the dendrogram of Single Link only the cluster ”headlamps” can be identified clearly and most other instances are wrongly clustered (See Figure 5.6(d)).

Ecoli Data

This Ecoli data set derives from a study on protein locations and consists of 336 instances. It includes 8 highly unbalanced classes having from 2 to 142 objects per class and each instance is described by 7 attributes. The real data set is also represented as a hierarchical structure [26], therefore, the hierarchical analysis is performed. For algorithm *hSync*, the plot of the MDL values w.r.t. ϵ is indicated in Figure 5.7(a). Based on the plot, the hierarchical structure of the data set can be easily generated (see Figure 6.9). *Sync* detects a three-level hierarchy. At the first level, five stable clusters (A - E) start to emerge when ϵ ranges from 0.26 to 0.32. Cluster A is composed of 155 instances and 135 out of them belong to “cytoplasm



(a) Locality-quality Plot (hSync) (b) The Hierarchical Tree (hSync)

(c) Dendrogram (Single Link) (d) Reachability-plot

Figure 5.7: Exploring the hierarchical cluster structure of Ecoli data with different algorithms.

(cp)", which results in P = 87.1% and R = 94.4%. Cluster B includes 30 instances and mainly represents the type "perisplasm (pp)" with the $P = 90.0\%$. The types of "inner membrane without signal sequence (im)" and "inner membrane, uncleavable signal sequence (imu)" form as the cluster C (77 instances) as they both belong to "inner membrane" and thus are very similar. Cluster D contains 16 instances of "inner membrane without signal sequence (im)" ($P = 100.0\%$). Cluster E consists of 20 instances and 17 out of them belong to the type “outer membrane (om)”. In addition, 38 instances are viewed as outliers at this level. Subsequently, at the second level with ϵ ranging from 0.39 to 0.53, the cluster A (type ”cp”) and cluster B (type ”pp”) merge together and form cluster F. At the same time, the cluster C (”imu”) and cluster D (’”im”) form a new cluster G, which represent the ”inner membrane”. At the third level, cluster G (”imu” and ”im”) and cluster E (”om”) further group together and are represented as cluster H with both inner and outer membrane. Finally, cluster F and cluster H merge to cluster I. The corresponding hierarchical cluster structure is illustrated in Figure 5.7(b). For Single Link, the dendrogram is created and it is difficult to find the cluster structure (See Figure 5.7(c)). Figure 5.7(d) shows the reachability- plot of OPTICS. From the plot, like the dendrogram of Single Link, it is hard to identify the cluster structure. However, only two general clusters (A, B) can be visualized.

5.5 Conclusion

In this chapter, the novel dynamic hierarchical clustering algorithm *hSync* based on synchronization and MDL principle is introduced. The extensive experiments demonstrate that the algorithm shows several desirable properties and outperforms most state-of-the-art hierarchical clustering algorithms.

Chapter 6

Finding Arbitrarily Oriented Synchronized Subspace Clusters

How to address the challenges of the curse of dimensionality in clustering? Clustering is a powerful data mining technique for structuring and organizing vast amounts of data. However, the high-dimensional data space is usually very sparse and meaningful clusters can only be found in lower dimensional subspaces. In many applications the subspaces hosting clusters provide valuable information for interpreting the major patterns in the data. Detection of subspace clusters is challenging since usually many of the attributes are noisy, some attributes may exhibit correlations among each other and only few of the attributes truly contribute to the cluster structure. In this chapter, ORSC (Arbitrarily ORiented Synchronized Clusters), a novel effective and efficient method for subspace clustering, is proposed. Relying on the proposed weighted local interaction model, the approach ORSC naturally detects correlation clusters in arbitrarily oriented subspaces, including arbitrarily-shaped non-linear correlation clusters. ORSC is robust against noise points and outliers. In contrast to previous methods, ORSC is easy to parameterize, since there is no need to specify the subspace dimensionality and all interesting subspace clusters can be detected. Finally, ORSC outperforms most comparison methods in terms of runtime efficiency and is highly scalable to large and high-dimensional data sets.

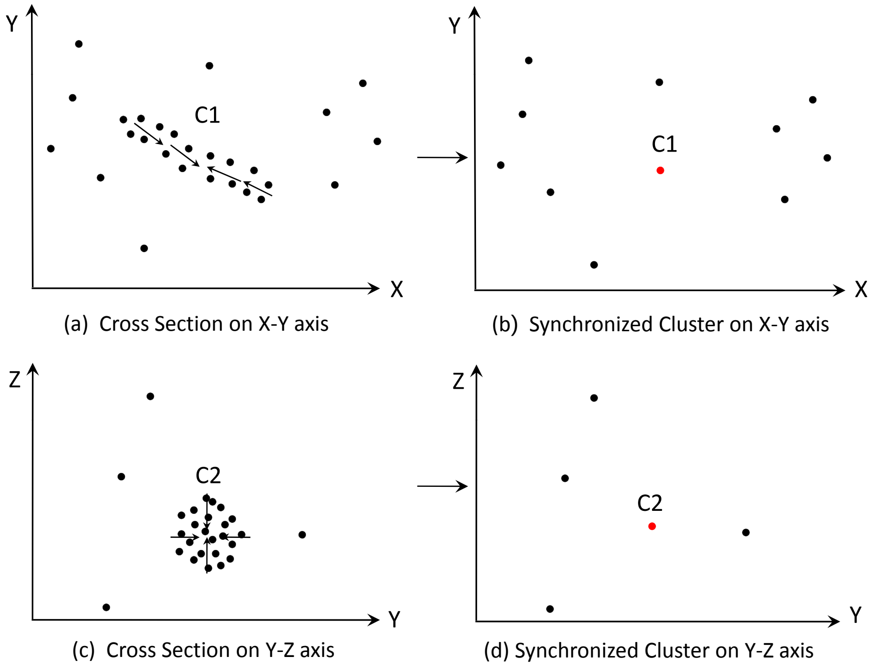
The remainder of this chapter is organized as follows: In the following section, the introduction is given. Section 6.2 briefly surveys related work of subspace clustering. Section 6.3 presents our new concept and algorithm in detail. Section 6.4 contains an extensive experimental evaluation and Section 6.5 concludes the chapter. Parts of the material presented in this chapter have been published in [150], where J.S. proposed the main idea, implemented the algorithms and wrote the largest parts of the paper; C.P., C.B. and Q.Y. also contributed to the conceptual development and paper writing.

> "*Junming Shao, Qinli Yang, Christian Böhm, and Claudia Plant. Detection of Arbitrarily Oriented Synchronized Clusters in High-dimensional Data. ICDM 2011, in press.*"

6.1 Introduction

Owing to the modern technology advance, nowadays, tremendous amounts of data in a large variety of application domains have been produced. These real-world data sets are often represented as sparse, high-dimensional feature vectors, contaminated by outliers and noisy objects. Clustering such high-dimensional data becomes difficult for traditional clustering methods due to the problem called “curse of dimensionality”. With increasing dimensionality, more and more objects are located at the boundaries of the feature space. In very high dimensions it is common for all of the data objects to be nearly equidistant from each other and objects do not cluster anymore in the full-dimensional feature space. Instead, clusters of data objects are often embedded in subsets of features. Another reason why traditional clustering algorithms often deteriorate in high-dimensional spaces is that many of the dimensions may be irrelevant for the clustering problem. These irrelevant dimensions can confuse clustering algorithms by hiding clusters in noisy data. To reduce the effect of “curse of dimensionality”, a classic approach is to map the whole high-dimensional feature space onto a relatively lower-dimensional subspace using dimensionality reduction techniques like principal component analysis (PCA). Such dimensionality reduction approaches attempt to transform a high-dimensional data set into fewer dimensions by creating linear combinations of the original attributes. However, since they preserve the global variance between objects, it is impossible to yield good results if many irrelevant attributes hide the cluster structure. It only provides a single, global reduction that does not account for local variances in subspaces for different patterns. Moreover, it is very difficult to interpret the new features in the context of the domain.

To cure the curse of dimensionality in clustering, the new concept of *subspace clustering* has been introduced which automatically detects clusters in subspaces of the original feature space. A number of subspace clustering algorithms have already been proposed, which can be mainly distinguished as axis-parallel subspaces clustering, e.g. [6], [4], [93] and arbitrarily oriented subspace clustering (also called generalized subspace clustering, or correlation clustering), e.g. [5], [25], [161]. However, most of these algorithms fail to discover clusters of complex shapes and different densities. Take CLIQUE [6] for example. This approach detects dense units by using a grid of regular size to divide each dimension into bins. Adjacent dense units are then combined to form clusters. Obtaining meaningful results is dependent on the proper tuning of the grid size and the density threshold parameters. Even after well tuning both parameters, the algorithm will likely fail if at least two clusters with quite different densities are present in the data set. This problem also exists in other approaches, such as density-based subspace clustering SUBCLU [93]. Another challenge for subspace clustering is the search strategy for the suitable subspaces. The number of possible axis-parallel subspaces is exponential (2^d) in the number of dimensions d, and the number of arbitrarily oriented subspaces is even infinite. Therefore, a complete enumeration of all possible subspaces to be checked for clusters is not feasible. Consequently, all previous solutions rely on specific assumptions and heuristics, and try to find promising subspaces during the clustering process, for instance in an iterative optimization. It can be seen that these previous approaches of learning suitable subspaces work well if (but only if) subspace clusters are locally well separated and no outlier objects (belonging to no cluster) exist. In the presence of outliers in the local neighborhood of cluster points or cluster representatives in the entire feature space, most previous subspace clustering algorithms fail to detect subspace clusters, because the algorithms



(a) Cross Section on X-Y axis

(b) Synchronized Cluster on X-Y axis

(c) Cross Section on Y-Z axis

(d) Synchronized Cluster on Y-Z axis

Figure 6.1: Illustration of the Synchronized Subspace Clusters.

try to find suitable subspaces for each cluster from the local neighborhood of cluster points or cluster representatives in the entire feature space.

To deal with these problems, in this chapter, subspace clustering is considered from a different point of view, *synchronization*. Each object interacts with similar objects and the strength of the interaction is determined by the local structure of objects. Through non-linear weighted interaction among objects, objects with similar attributes in arbitrarily oriented subspaces tend to synchronize together. It will be demonstrated that the synchronization-based subspace clustering has many attractive benefits, but some of its fundamental concepts are first illustrated.

6.1.1 Synchronized Subspace Cluster

Based on these synchronization phenomena and models, each dimension of objects is considered as a phase oscillator and the dynamics of objects towards synchronization is exploited to obtain clusters in arbitrarily oriented subspaces. The concept is demonstrated with the help of a simple 3-dimensional data set. Fig. 6.1(a) and Fig. 6.1(c) represent the X-Y and Y-Z projection of the data, respectively. In Cluster $C1$, the X and Y dimensions are strongly correlated and the Z-dimension is not cluster-specific white noise. Cluster $C2$ only exists in the Y-Z space and the X dimension is noise. To find such clusters, in this context, like oscillators, each object interacts with similar objects in axis-parallel or arbitrarily oriented subspaces. Objects will gradually change their states and move to other objects driving by its intrinsic structure, which will be elaborated in detail in Section 6.3. The arrows in Fig. 6.1(a) and Fig. 6.1(c) illustrate the main directions of movements for objects. Fig. 6.1(b) and Fig. 6.1(d) further indicate the final states of objects (red points) after synchronization. Finally, all objects of a common subspace cluster move together and form as synchronized clusters (cf. Fig. 6.1(b) and Fig. 6.1(d)).

6.1.2 Contributions

Inherited by the concept of synchronization, in this chapter, a new subspace clustering approach, ORSC (arbitrarily ORiented Synchronized Clusters) is proposed. The major benefits of the algorithm ORSC can be summarized as follows:

1. *Natural data structure exploring.* The synchronized cluster formation is driven by the mutual interactions among objects relying on intrinsic data structure.

2. *Detection of arbitrarily shaped correlation clusters.* Without any assumption of the data distribution, ORSC allows detecting clusters with different densities as well as arbitrary numbers, shapes and sizes in subspaces.

3. *Outlier robustness.* Since the outliers cannot easily synchronize with other objects, they can be effectively distinguished from cluster objects.

4. *Efficient subspace searching.* Inherited by the properties of synchronization, the subspace search strategy of ORSC does not need scan possible subspaces but simply find all subspace clusters by searching all synchronized phases.

5. *Easy parametrization.* ORSC requires only one single user specified parameter which is more flexible and robust to clustering results.

6.2 Related Work

As most traditional clustering algorithms fail to detect clusters embedded in high-dimensional data, various subspace clustering approaches have been studied. Subspace clustering algorithms like CLIQUE [6] and its extensions ENCLUS [38], OptiGrid [79], projected clustering algorithms such as PROCLUS [4], DOC [128], and SUBCLU [93] only find axis-parallel clusters. Pattern based subspace clustering methods (e.g. [166],[97]) aim to group objects that exhibit a similar trend in a subset of attributes into clusters rather than objects with low distance. However, they limit themselves to finding only clusters that represent pairwise positive correlations in the data set. Here, the focus is the more recent correlation clustering algorithms which can find clusters in arbitrarily oriented subspaces. Recently, many arbitrarily oriented subspace clustering algorithms such as ORCLUS [5], 4C [25], Curler [161] are proposed to find clusters with arbitrarily oriented principle axes.

ORCLUS [5] is one of the iterative top-down search methods for arbitrarily projected subspaces. It integrates PCA into k-means and includes three steps: assign clusters, determinate subspaces and merge them. However, it requires users to specify the number of clusters and the size of the subspace dimensionality in advance. If the estimation does not match with the actual number of clusters the result of ORCLUS tend to fail. Another problem is that ORCLUS cannot handle the noisy and sparse data efficiently. Moreover, due to using random sampling to improve computation speed and scalability, it may suffer from missing smaller clusters. 4C [25] combines PCA and density-based clustering (DBDSCAN) to identify local subgroups of the data objects sharing a uniform but arbitrarily complex correlation. It formalizes the correlation connected cluster as a dense region of points in the d-dimensional feature space having at least one principal axis with low variation along this axis. Like ORCLUS, 4C assumes the clustering structure is dense in the entire feature space, thus it can not handle sparse data properly. Moreover, both ORCLUS and 4C limit them to identifying linear correlation clusters without considering nonlinear correlation clusters. Actually, in real-life data sets, the correlation between attributes could however be nonlinear. To address the nonlinear issue, Curler [161] is proposed which allows to detect both global and local orientations of the clusters. This method merges the micoclusters generated by the EM variant algorithm according to their co-sharing level. Therefore, the resulting clusters may represent a more complex, not necessarily liner correlation. However, like ORCLUS, the performance of Curler is very sensitive to noise and strongly depends on the suitable parametrization.

6.3 The Algorithm ORSC

In this section, ORSC is presented to discover all possible clusters in arbitrarily oriented subspaces. First the overview of the novel perspective of subspace clustering is given and then the weighted interaction model is constructed to simulate the dynamics of objects towards synchronization. Finally an efficient searching strategy is used to find all synchronized clusters.

6.3.1 A Novel Perspective of Subspace Clustering

As introduced in Section 6.1, the subspace analysis is considered from a novel perspective: synchronization. The key point is to view each object as an oscillator and it moves dynamically according to a weighted local interaction model (cf. Section 6.3.2). For each object, its similar objects are determined as interaction partners. The strength of interactions from these similar objects is determined by its local structure. To ensure all objects of a common arbitrarily oriented subspace cluster can move together, PCA is integrated into the interaction model to determine the main directions of the local cluster structure. Through the weighted interactions, all objects in arbitrarily oriented subspace clusters can easily synchronize together and form as synchronized clusters. In the following, it starts to elaborate how to construct the weighted local interaction model.

6.3.2 Interaction Model

Currently, the most successful way to explore the synchronization phenomena is Kuramoto Model [94, 95] (cf. Section 3.2). However, in order to introduce the Kuramoto model into subspace clustering, it is reconsidered in a different way.

1. **Local Interaction Fashion**. To exploit the hidden clusters or patterns in arbitrarily oriented subspaces, the local structure of data should be investigated. Therefore, it focuses on the dynamics of objects in a local way.

2. **Weighted Interaction**. In high dimensional space, the correlations in the dimensions are often specific to data locality, which means some objects are correlated with respect to a given set of dimensions and others are correlated with different dimensions. Thus, the coupling strengths of interactions of objects in relevant or irrelevant dimensions should be considered with different weights.

In the following, the interaction model will be reformulated based on the above two criteria.

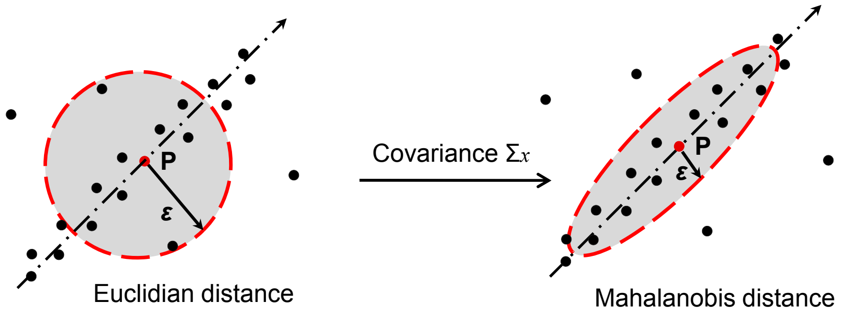


Figure 6.2: ϵ-Range search with different distance functions.

Local Interaction

To formalize the local interaction for each object, the intuitive way is to consider its ε-neighborhood according to Equation 4.2.

However, such ε-neighborhood search can not fit the goal well since it does not consider the local data distribution. To look for objects which are close in the local subspace cluster structure, therefore, the Mahalanobis distance instead of Euclidean distance is used to determine similar objects.

Definition 6.1 (ε-Neighborhood with mahalanobis distance) Given a $\varepsilon \in \mathcal{R}$ and $x \in \mathcal{D}$, the ε-neighborhood of an object x with Mahalanobis distance, $N_\epsilon^m(x)$, is defined as:

$$N_\varepsilon^m(x) = \{y \in \mathcal{D} | \sqrt{(y-x) \cdot \Sigma_x^{-1} \cdot (y-x)^T}\} \tag{6.1}$$

where Σ_x is the covariance matrix of ε-neighborhood of x. Since the Mahalanobis distance considers the local data distribution, it can better search similar objects considering the local cluster structure and is also less sensitive to noise. Fig. 6.2 illustrates the similar objects determination of an object P with different distance functions.

According to Definition 6.1, the Kuramoto model is extended in a local fashion, where each object interacts with its ε-Neighborhood with mahalanobis distance during time revolution. Moreover, since without additional attributes of each object, all objects are assumed to be the same frequency ω, which well fits the condition of Kuramoto model. Here, each dimension of an object is viewed as a phase oscillator and its original value represents the initial phase.

Formally, let $x \in R^d$ be an object in the data set $\mathcal{D}$ and x_i be the i-th dimension of the data object x. $N_\varepsilon^m(x)$ is the ε-neighborhood of object x. According to Eq.(3.1), the dynamics of each dimension x_i of the object x with a local interaction is further written as:

$$\frac{dx_i}{dt} = \omega + \frac{K}{|N_\varepsilon^m(x)|} \sum_{y \in N_\varepsilon^m(x)} \sin(y_i - x_i) \tag{6.2}$$

Let $dt = \Delta t$, then:

$$x_i(t + \Delta t) = x_i(t) + \Delta t \cdot \omega + \frac{\Delta t \cdot K}{|N_\varepsilon^m(x(t))|} \cdot \sum_{y(t) \in N_\varepsilon^m(x(t))} \sin(y_i(t) - x_i(t)) \tag{6.3}$$

Since the term $\Delta t \cdot \omega$ is the same for each dimension of all objects and thus can be ignored. Let $C = \Delta t \cdot K$, the dynamics of each dimension x_i of an object x over time is written as:

$$x_i(t + \Delta t) = x_i(t) + \frac{C}{|N_\varepsilon^m(x(t))|} \cdot \sum_{y(t) \in N_\varepsilon^m(x(t))} \sin(y_i(t) - x_i(t)) \tag{6.4}$$

Weighted Interaction Determination

For most existing interaction models, e.g. [94], [11], [3], the coupling strength of the object interactions is constant. However, it is not appropriate for subspace clustering since clusters exist in different subspaces. Thus, to ensure all cluster objects can synchronize in corresponding subspaces, the strength of interactions is considered followed by the local data structure of the objects. For an object x, it is expected that the interactions along the main directions of the local cluster structure (relevant and potentially correlated dimensions) are imposed much higher weights while those in irrelevant dimensions have lower weights. Therefore, to determine the main directions of the local cluster structure, the PCA is used to decompose the covariance matrix Σ of objects $N_{\varepsilon}^{m}(x)$, which is denoted by $\Sigma = VEV^T$. The orthogonal Matrix V called eigenvector matrix and the diagonal matrix E called eigenvalue matrix. The eigenvectors represent the principal directions of these similar objects and the eigenvalues represent the variance along these directions. The eigenvalues are normalized into the interval (0, 1) by dividing each eigenvalue λ_i with the sum of all eigenvalues. Then, the normalized eigenvalues are viewed as the interaction weights along with the corresponding principal directions. Finally, the difference vector between two objects is projected onto these orthogonal eigenvectors and the difference with corresponding normalized eigenvalues is coupled. Formally, the weighted interaction between two objects are defined as follows.

Definition 6.2 (Weighted Interaction) Let $x \in R^d$ be an object in the data set $\mathcal{D}$. $N_{\varepsilon}^{m}(x(t))$ is the ε-Neighborhood of the object x and $y \in N_{\varepsilon}^{m}(x(t))$. $\vec{v_1}, ..., \vec{v_d}$ and $\lambda_1, ..., \lambda_d$ are the eigenvectors and eigenvalues by PCA decomposition of the covariance matrix of $N_{\varepsilon}^{m}(x(t))$. The weighted interaction between the object $y \in N_{\varepsilon}^{m}(x(t)$ and the object x, donated by $WI_{(y \bowtie x)}$, is defined as:

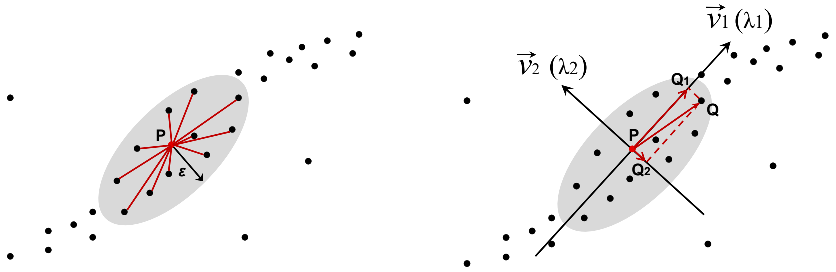


Figure 6.3: Weighted interaction between two objects.

$$WI_{(y \bowtie x)} = \sum_{k=1}^{d} \lambda_i \cdot sin(proj(\Delta(y,x), \vec{v_k})) \tag{6.5}$$

where $\Delta(y,x) = y - x$ means the difference vector between y and x, $proj(\Delta(y,x), \vec{v_i})$ means the projection of vector $\Delta(x,y)$ onto $\vec{v_i}$. Since the eigenvectors $\vec{v_i}$ are unit vectors, therefore,

$$proj(\Delta(y,x), \vec{v_i}) = (\Delta(y,x) \odot \vec{v_i}) \cdot \vec{v_i} \tag{6.6}$$

where $\odot$ means the inner product.

To illustrate the weighted interaction, Fig. 6.3 gives an example with a 2-dimensional data set. Given an object P, first, the ε-neighborhood of object P with Mahalanobis distance are obtained. Then the covariance matrix of these objects is decomposed by PCA and the eigenvectors ($\vec{v_1}$, $\vec{v_2}$) and eigenvalues (λ_1, λ_2) are obtained respectively. For each interaction with object P, e.g. $Q \bowtie P$ interaction, the difference vector $\overrightarrow{QP}$ is projected to the first direction $\vec{v_1}$ with $\overrightarrow{Q_1P}$ and the second direction $\vec{v_2}$, denoted by $\overrightarrow{Q_2P}$. The interaction between objects Q and P is finally determined with $\lambda_1 \cdot sin(\overrightarrow{Q_1P}) + \lambda_2 \cdot sin(\overrightarrow{Q_2P})$.

Finally, the dynamics of each dimension x_i of the object x is governed by:

$$x_i(t + \Delta t) = x_i(t) + \frac{1}{|N_\varepsilon^m(x(t))|} \cdot \sum_{y(t) \in N_\varepsilon^m(x(t))} \cdot \sum_{k=1}^{d} \lambda_k \cdot sin(proj_{(i)}(\Delta(x(t), y(t)), \vec{v_k})) \tag{6.7}$$

where $proj_{(i)}$ means the i-th dimension of project vector. The object x at time step $t = 0 : x(0)(x_1(0); \cdots ; x_d(0))$ represents the initial state of the object. The $x_i(t + \Delta t)$ describes the renewal state value of i-th dimension of object x at time point $(t + \Delta t)$.

To determine the termination of the dynamic process, a synchronization order parameter r is defined as measuring the degree of synchronization of objects.

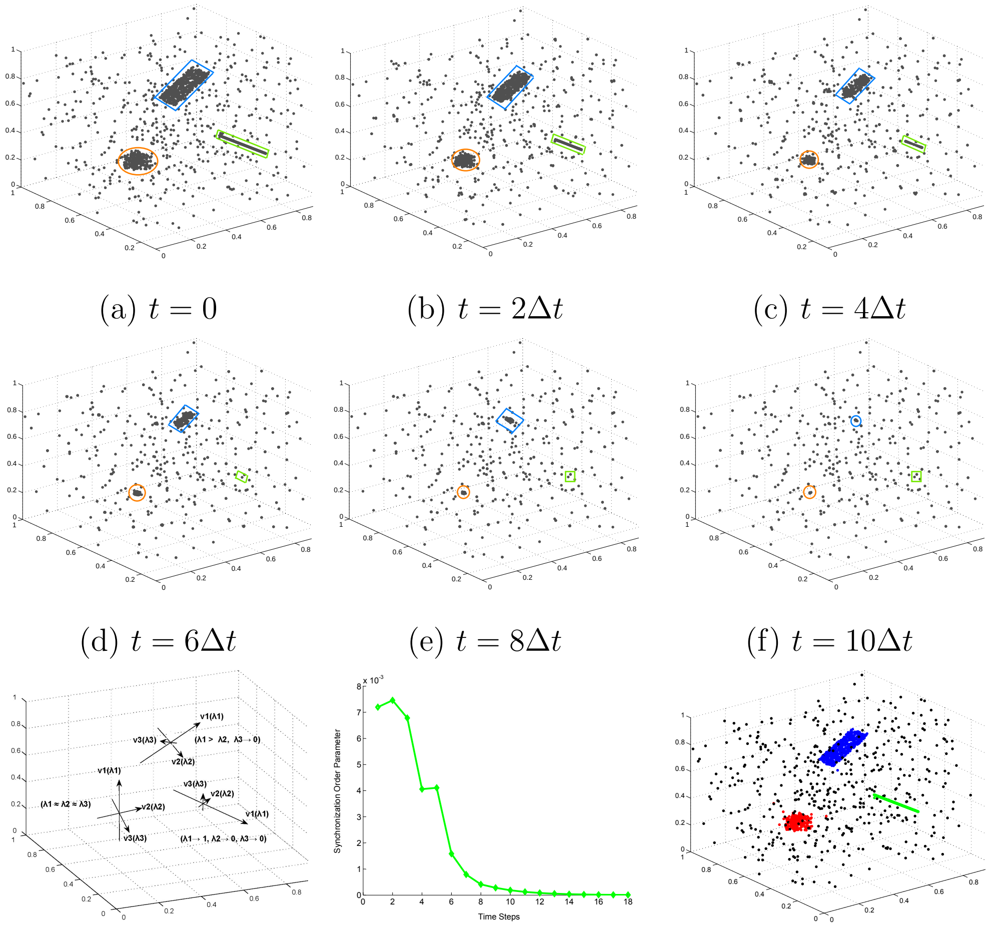
Definition 6.3 (Synchronization order parameter) The synchronization order parameter r characterizing the degree of synchronization is defined as the average movements of objects over time:

$$r = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{|N_\varepsilon(x)|} \sum_{y \in N_\varepsilon(x)} WI_{(y-x)} \right) \tag{6.8}$$

The value of r decreases as more and more objects synchronize over time. The process towards synchronization terminates when r converges, which indicates there is no further change of objects.

6.3.3 Simulation of the Object Dynamics

After the formulation of the interaction model, the dynamics of objects are simulated to investigate all clusters in arbitrarily oriented subspaces. Generally, the dynamics of objects involve the following steps:



(a) $t = 0$ (b) $t = 2\Delta t$ (c) $t = 4\Delta t$

(d) $t = 6\Delta t$ (e) $t = 8\Delta t$ (f) $t = 10\Delta t$

(g) Main directions (h) Order Parameter (i) Synchronized Clusters

Figure 6.4: Illustration of dynamics of objects.

1. Initially ($t = 0$), all objects in the data set have their own states (feature vectors).

2. As time evolves ($t > 0$), for each object $x(t)$, it searches its similar objects with Mahalanobis distance $N_\varepsilon^m(x(t))$ and then applies PCA to decompose the covariance matrix of $N_\varepsilon^m(x(t))$ to obtain the corresponding eigenvectors and eigenvalues. The renewal state of object $x(t+\Delta t)$

is then determined by Eq. 6.7.

3. During the process towards synchronization, the order parameter $r(t)$ (Eq. 6.8) is calculated to terminate the simulation when r converges.

To illustrate the dynamics of objects according to our interaction model, for the ease of illustration, a 3-dimensional data set is provided as an example. Fig. 6.4(a) (t=0) shows the initial states of objects, where three clusters exist in the data set: a 3-dimensional Gaussian cluster in full dimensions, a 2-dimensional linear correlation cluster in arbitrarily oriented subspace and 1-dimensional linear cluster in Z axis. When $t > 0$, for each object, it starts to interact with similar objects according to the local cluster structure. For revelent dimensions, the object interaction is imposed much higher strength while there is lower impact for irrelevant dimensions. E.g., in Fig. 6.4(b)-(f), the objects in the Gaussian cluster gradually move together from all directions in the three dimensions since these objects belong to a common 3-d subspace cluster. We can see that the eigenvalues along with main directions of these objects (eigenvectors) decomposed by PCA are very similar ($\lambda_1 \approx \lambda_2 \approx \lambda_3$, see Fig. 6.4(g)). Therefore, the objects interactions are imposed similar strengths and gradually group together. This situation is different from objects in the other two clusters. For the objects in the 2-dimensional correlation cluster, the main directions of the cluster structure are not along the coordinate axis but arbitrarily oriented. The eigenvalues on the corresponding eigenvectors decomposed by PCA are thus very different, where ($\lambda_1 > \lambda_2$ and $\lambda_3 \to 0$). These cluster objects therefore mainly move towards two directions ($\vec{v_1}$) and ($\vec{v_2}$) with weights λ_1 and λ_2 respectively (Fig. 6.4(g)). Similarly, the 1-dimensional linear cluster objects tend

Table 6.1: Illustration of the subspace search strategy.

Obj.	$d1$	$d2$	$d3$	$d4$	Syn. Dim.	New Subs.	Cluster
1	0.1	0.2	0.1	0.3	1,2	(1,2)	(1 2 3 4)
2	0.1	0.2	0.2	0.2	1,2,4	(1,2,4)	(2 3 4)
3	0.1	0.2	0.7	0.2	1,2,3,4	(1,2,3,4)	(3,4)
4	0.1	0.2	0.7	0.2	1,2,3,4	-	-
5	0.3	0.4	0.3	0.3	3	(3)	(5,6)
6	0.9	0.5	0.3	0.1	3	-	-
7	0.7	0.6	0.4	0.5	Null	-	Noise

to move towards one direction ($\vec{v_1}$). Through such weighted interactions, finally, all cluster objects synchronize together in the corresponding subspaces (Fig. 6.4(f) and Fig. 6.4(i)). During the process of synchronization, the synchronization order parameter will decrease and finally converge. (Fig. 6.4(h)).

6.3.4 Synchronized Clusters Search

After the simulation of dynamics of objects by our interaction model, cluster objects in arbitrarily oriented subspaces synchronize together. To find these synchronized clusters, the intuitive way is to find all synchronized phases and corresponding objects. The principle of our strategy is to consider the subspace search from objects instead of dimensionality. Begin that, let's first define $\mathcal{L}$-Dimensional Synchronized Cluster and Synchronized Dimension.

Definition 6.4 ($\mathcal{L}$-Dimensional Synchronized Cluster)

Let $\mathcal{S} \subseteq \mathcal{D}, \mathcal{L} \subseteq \mathbb{N}$ and $\mathcal{L} = \{\ell_1, \cdots, \ell_i\}, \ell_i \in \mathbb{N}, \ell_i < d$. If all objects in $\mathcal{S}$ are synchronized in $\mathcal{L}$ dimensions, $\mathcal{S}$ forms a $\mathcal{L}$-Dimensional Synchronized Cluster. Formally,

$$\textsc{SynClu}^{\mathcal{L}}(\mathcal{S}) \Leftrightarrow \{x_{\ell_i} = y_{\ell_i} | x, y \in \mathcal{S}, \ell_i \in \mathcal{L}\}$$

Definition 6.5 (Synchronized Dimensions)

Let $\mathcal{S} \subseteq \mathcal{D}$ be a $\mathcal{L}$-Dimensional Synchronized Cluster, $\mathcal{L} \in \mathbb{N}$ are called synchronized dimensions, denoted by SynDim($\mathcal{S}$).

To search all subspace synchronized clusters, each object is investigated over whether other objects synchronize with it in any dimension. If it does not synchronize with any dimension of other objects (with same phase), this object is viewed as noise. If it synchronizes with some dimensions of other objects, these synchronized dimensions are regarded as a synchronized subspace and these synchronized objects are formed as a synchronized cluster. All synchronized subspaces are extracted and the synchronized objects are assigned in corresponding subspaces. This process is repeated for each object and finally all synchronized subspaces and corresponding synchronized clusters are obtained.

To illustrate the search strategy, please look at Table 6.1. Supposing there are 7 objects with 4 dimensions, after weighted interaction among objects, the final states of objects are outlined in the left part of Table 6.1. For each object, its synchronized dimensions and objects are found. For instance, for object #1, it synchronizes with objects #2 to #4 in dimensions of 1 and 2. Therefore, a synchronized subspace (1,2) is created and the corresponding objects #1to #4 are added to a cluster in this subspace. Similarly, for object #2, it synchronizes with objects #3 and #4 in dimensions (1,2,4) and a new subspace is thus created and obtains corresponding cluster. This process is repeated until all objects are investigated. In addition, since object #7 does not synchronize with any other object in any dimension, it is thus viewed as noise in the full dimensional space.

Through this search strategy, all synchronized subspaces and corresponding clusters are obtained by iterating all objects. Once all synchronized clusters are detected, their dimensionality can be further determined. The reason is that objects in synchronized clusters moving together do not need to be parallel with an axis but arbitrarily orientated of the synchronized cluster objects. Therefore, if a cluster is located in axis-parallel subspace, the synchronized subspace is the exact subspace. However, if a cluster is embedded in an arbitrarily oriented subspace, the synchronized subspace only stands for the original feature space that these clusters accommodated.

Definition 6.6 (Dimensionality of a Synchronized Cluster) Let $\mathcal{S} \subseteq \mathcal{D}$ be a synchronized cluster and Σ is the covariance matrix of objects in the clustering and $\Sigma = VEV^T, \lambda_1, \cdots \lambda_d$ are the eigenvalues of $\mathcal{S}$ in descending order. Given a $\delta \approx 0$, the dimensionality of $\mathcal{S}$ $w.r.t$ δ is η if $d - \eta$ eigenvalues of $\mathcal{S}$ are close to zero ($\Phi(\lambda_i) \leq \delta$ and $\Phi(\lambda_i) = \lambda_i/\lambda_1$), which is denoted by $\text{DimSynClu}_{\delta}^{\eta}$.

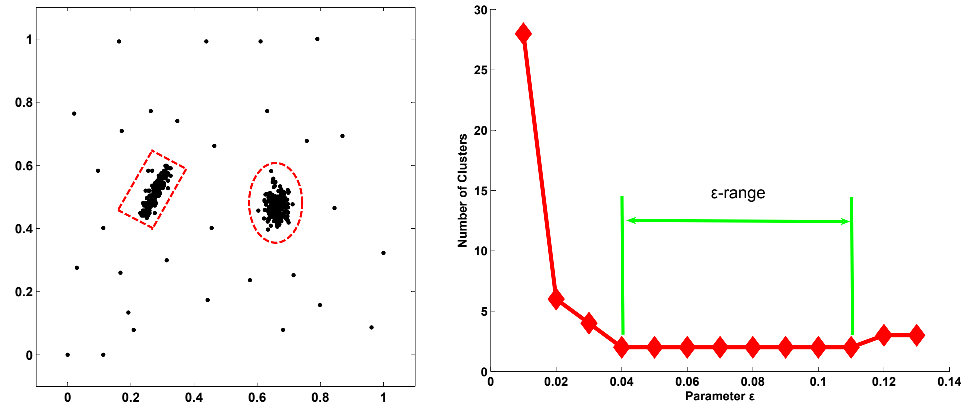
Finally, the Pseudocode of the ORSC Algorithm is illustrated in Fig. 6.5.

6.3.5 Parameter Setting

To simulate the dynamics of objects, an interaction range (ε) needs to be specified in the Interaction Model. The question is: how to determine the ε value and how does the clustering results change when the ε value is adjusted?

```
**algorithm** $[S, C] = ORSC(D, \varepsilon)$
    $D^{'}$= Simulation($D, \varepsilon$);
    $[S, C]$ = SearchCluster($D^{'}$)
 **Return** $S, C$;
Function $D^{'}$=Simulation($D, \varepsilon$); **//Objects' dynamics Simulation.**
    **while**($r$ converges)
      **for**(each object $x \in D$)
        Search $N_\varepsilon(x)$ of object $x$;
        Compute $N_\varepsilon^m(x)$ by Definition 1 in $N_\varepsilon(x)$;
        Determine the interaction directions and weights by PCA;
        Obtain new state of object $x$ with interaction model ($Eq$.6.7);
     **end for**
     Set $D^{'}$ to be the new states of all objects;
     Compute synchronization order parameter $r$;
   **end while**
 **Return** $D^{'}$;
 Function $[S, C]$=SearchCluster($D^{'}$);
    $S = null; C = null$; //$S$ saves subspaces and $C$ saves clusters
    **for**(each object $x \in D^{'}$)
      **for**(each object $y \in D^{'}$)
        If $y$ synchronized with $x$ in dimensions $L$;
          If ($L$ exists in $S$) and synchronized phases are same
            Add $y$ to the exist corresponding cluster;
          Else
            Create a new synchronized subspace $L$ and a new cluster $CL$;
            Add $x$ and $y$ to the new cluster $CL$ and $S.add(L)$, $C.add(CL)$;
          End If
        Else
          $y$ is viewed as noise object;
     End For
   End For
 **Return** $S, C$;
```

Figure 6.5: Pseudocode of the ORSC Algorithm.



(a) (b)

Figure 6.6: Impact of parameter setting for clustering.

In order to generate a stable interaction among objects, a heuristic way is to use the average value of the k-nearest neighbor distance determined by a sample from the data set for a small k. Fig. 6.6(a) shows a simple data set which consists of 2 clusters plus outliers. It starts with a very small value and then gradually increases it to see the change of clustering results. With different parameter ε, the number of detected clusters is computed. It can be seen that the same clustering results (2 clusters) are obtained with a fairly long stable range (see Fig. 6.6(b)). In contrast to other subspace clustering algorithms, the parameter of our algorithm is much more flexible and more robust to noise. The reason is that our clustering is a dynamic process, in which each object moves during the process towards synchronization. With a small interaction range, in the beginning, a few objects interact with each other. But after several time steps, these objects in a cluster will gradually move closer and thus more and more objects can interact with each other and finally synchronize together. For relatively larger interaction range, the only difference is that more objects interact with each other at the beginning and thus tend to synchronize much faster.

6.3.6 Complexity Analysis

For the simulation of the dynamics of objects at each time step, ε-Neighborhood of each object with Mahalanobis distance needs to be searched. The covariance matrix with Euclidean distance query computation can be evaluated in $O(N \cdot d)$ time. The covariance matrix is then used to calculate the Mahalanobis distance and the time complexity approximately requires $O(d^3)$ time. PCA is further used to decompose the covariance matrix with Mahalanobis distance and it thus requires $O(d^3)$ time. For each object's interaction, it requires $O(d^3)$ time. Therefore, for all objects together, the simulation of dynamics at one time step is $O(N^2 \cdot d + N \cdot d^3)$. If there exists an efficient index, the complexity reduces to $O(T \cdot N \log N \cdot d + \log N \cdot d)$. For all time steps towards synchronization, the time complexity of objects' simulation is $O(T \cdot (N^2 \cdot d + N \cdot d^3))$ in worst case. T is the number of time steps. In most cases, T is small with $5 \leq T \leq 20$.

For the time complexity of subspaces and the corresponding clusters search, the most bottom-up establishing algorithms of subspace clustering need $O(2^d)$ time. For the search, it is not necessary to iterate all subspaces but instead finding all synchronized subspaces by investigating the synchronized dimensions of each object. Therefore, the time complexity becomes $O(N^2 \cdot d)$. Finally, the time complexity of the algorithm ORSC in worst case is $O(T \cdot (N^2 \cdot d + N \cdot d^3) + N^2 \cdot d)$.

6.4 Experimental Evaluation

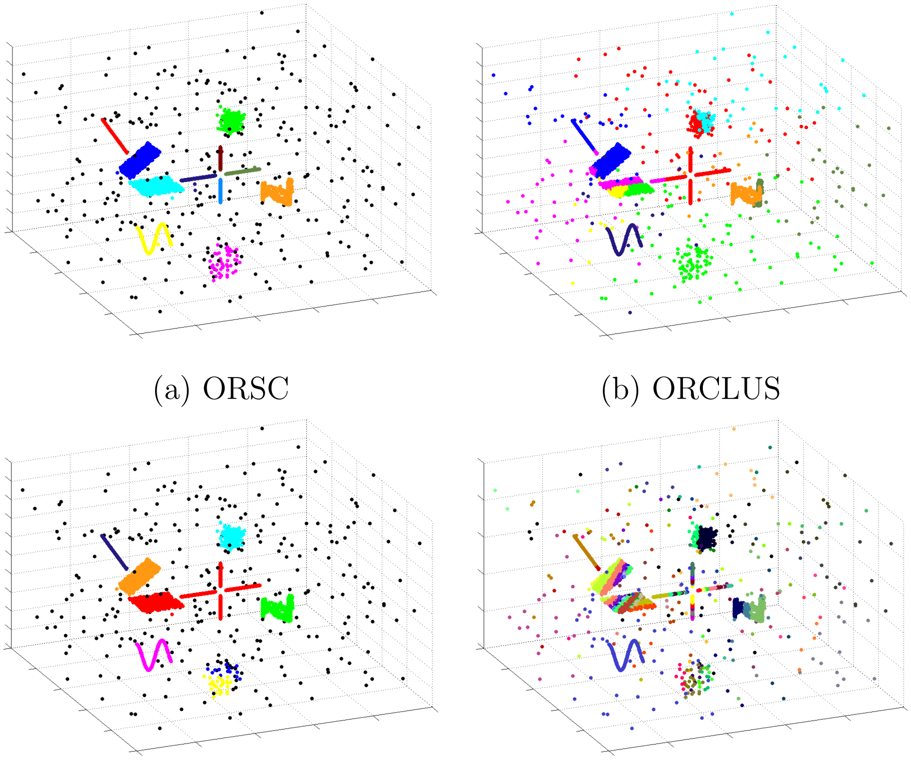
To extensively study the performance of ORSC, experiments are performed on several synthetic and real world data sets. The performance of ORSC is compared to ORCLUS [5], 4C [25] and Curler [161]. These particular algorithms are selected because they are representatives of different algorithmic paradigms: ORCLUS is a K-means style iterative partitioning algorithm; 4C is a local density-based method; Curler tries to find non-linear correlation clusters based on EM clustering. All experiments have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM.

Moreover, two established measures for cluster quality [50] are reported: Normalized Mutual Information (NMI), Adjusted Mutual Information (AMI) to evaluate different clustering results. For both measures, higher values represent better clustering performance. The precision (P) and recall (R) are also provided as validity measures to analyze each individual cluster.

6.4.1 Effectiveness

Synthetic Data

The evaluation of ORSC starts with several synthetic data sets to facilitate presentation and demonstrate its benefits. Fig. 6.7 displays the clustering results on a 3-dimensional synthetic data with all subspace clustering algorithms to be compared. The data consists of 11 clusters of different dimensionality, object density and correlation strength plus noise (Fig. 6.7(a)). For ORSC with parameter $\varepsilon = 0.06$, it successfully detects all these correlation clusters, including five 1-dimensional linear correlation clusters, two



(a) ORSC

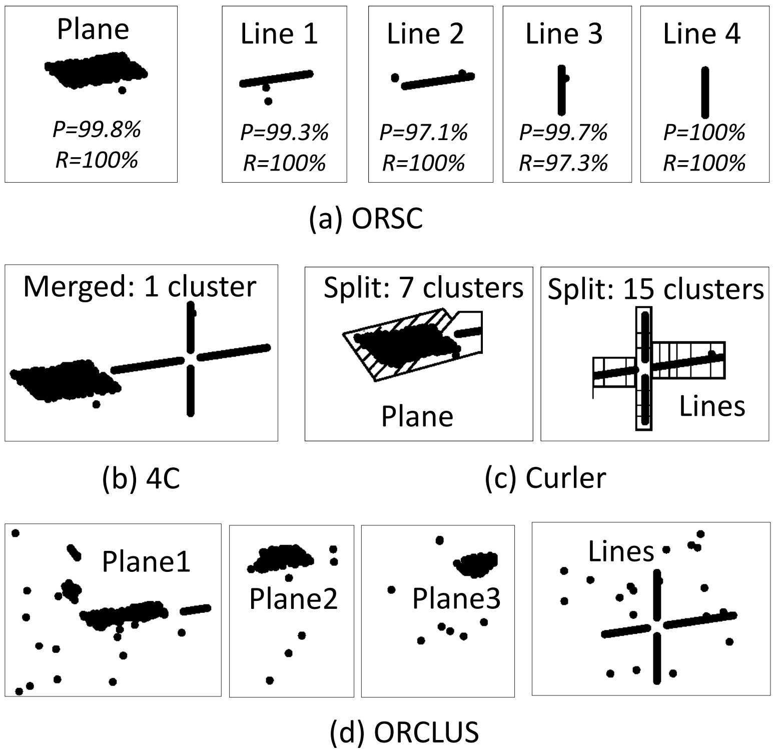
(b) ORCLUS

(c) 4C

(d) Curler

Figure 6.7: Comparison with different algorithms on a 3-d synthetic data.

2-dimensional linear plane clusters, two 2-dimensional non-linear clusters and two 3-dimensional clusters (Fig. 6.7(a)). ORCLUS requires the number of cluster K and the average subspace dimensionality l as input parameters. Specifying $K = 11$, the best results are obtained with $l = 3$, which are indicated in Fig. 6.7(b). Fig. 6.7(c) displays the best clustering results of 4C with parameters $\epsilon = 0.03$, $MinPts = 6$ and $\lambda = 3$. For Curler, all default parameter values suggested by authors are used. It obtains as many as 150 clusters (Fig. 6.7(d)). For better investigating different clustering results, it focuses on the detailed views of the 2-d linear plane cluster and four 1-d



(a) ORSC

(b) 4C

(c) Curler

(d) ORCLUS

Figure 6.8: Detailed view of clustering results with different algorithms on part of 3-d synthetic data.

linear clusters in the center of the data set (cf. Fig. 6.8). It is obvious that ORSC outperforms the competitors, where all correlation clusters are successful detected with high precision and recall. The evaluation of the clustering results is further illustrated in Table 6.4.

To further evaluate the algorithm ORSC, five high-dimensional data sets are generated (See Table 6.2). In each data set, several clusters are hidden in subspaces of varying dimensionality plus noise. For comparison, it is checked whether each algorithm can detect these clusters with suitable parameters. The precision and recall for each cluster are reported. The results with

Table 6.2: Data Description.

Data	d	#Clu.	Dim. of Clu.
DS1	5	1	3
DS2	10	1	5
DS3	15	2	10,5
DS4	20	2	10,10
DS5	30	3	20,15,10

Table 6.3: Comparative evaluation of different subspace approaches on high-dimensional synthetic data sets.

Data	True clusters found by			
	ORCLUS	4C	Curler	ORSC
DS1	1 (Dim.: 3) (P=32.7%;R=91.2%)	1 (Dim.: 3) (P=100%;R=100%)	1 (Dim.: 3) (P=99.0%;R=20.4%)	1 (Dim.: 3) (P=100%;R=97.0%)
DS2	1 (Dim.: 5) (P=27.8%;R=62.2%)	1 (Dim.: 5) (P=100%;R=94.2%)	1 (Dim.: 5) (P=48.4%;R=11.8%)	Dim.: 5) (P=100%;R=97.4%)
DS3	2 (Dim.: 10,5) (P=16.9%,74.4%) (R=14.0%,61.6%)	1 (Dim.: 10) (P=100%,R=99.6%)	2 (Dim.: 10,5) (P=14.5%,12.0%) (R=19.3%,16.0%)	2 (Dim.: 10,5) (P=100%,99.6%) (R=99.6%,100%)
DS4	2 (Dim.: 10,10) (P=17.9%,100%) (R=13.2%,74.0%)	2 (Dim.: 10,10) (P=100%,100%) (R=100%,100%)	2 (Dim.: 10,10) (P=12.5%,100%) (R=12.5%,100%)	2 (Dim.: 10,10) (P=100%,99.6%) (R=100%,99.6%)
DS5	3 (Dim.: 20,15,10) (P=21.7%,12.3%,13.2%) (R=100%,67.5%,72.5%)	1 (Dim.: 20) (P=100%; R=98.5%)	3 (Dim.: 20,15,20) (P=100%,99.5%,76.9%) (R=99.0%,100%,5%)	3 (Dim.: 20,15,10) (P=100%,98.5%,99.6%) (R=100%,99.5%,100%)

Table 6.4: Evaluation on different data sets.

Methods	Synthetic data		Ecoli data		Wine data	
	NMI	AMI	NMI	AMI	NMI	AMI
ORSC	**0.981**	**0.980**	**0.682**	**0.670**	**0.701**	**0.695**
4C	0.830	0.829	0.338	0.328	0.474	0.469
ORCLUS	0.598	0.596	0.452	0.430	0.191	0.182
Curler	0.583	0.561	0.060	0.049	0	0

different subspace clustering algorithms are depicted in Table 6.3. In all these data sets, ORSC finds the synthetic clusters in corresponding subspaces with both high recall and precision. In contrast to the comparing algorithms, there is no need to specify the subspace dimensionality and all interesting subspace clusters are detected. For 4C and ORCLUS, the subspace dimensionality is manually specified although it is difficult to know in real-world. 4C performs very well on the first two low-dimensional data sets. But it fails to find 5-dimensional cluster in 15-dimensional data set. This is also the same for the fifth data set, where it can not find the 10-dimensional cluster and 15-dimensional cluster because these two clusters are not dense in the full 30-dimensional feature space any more. The algorithm of ORCLUS is sensitive to noise and usually most noisy objects are included in the clusters identified. In addition, ORCLUS often merges these subspace clusters together, which thus results in low precision. The algorithm of Curler is also sensitive to noise and cannot yield good results for all data sets. However, in contrast to ORCLUS, it tends to splits true clusters into several distinct clusters and the obtained clusters are usually with low recall.

Real-world Data

In the following, the performance of ORSC is evaluated on two real-world data sets which are publicly available at the UCI machine learning repository.

Ecoli Data: This Ecoli data deriving from a study on protein location consists of 336 instances. It includes eight highly unbalanced classes having from 2 to 142 objects per class and each instance is described by 7 attributes. ORSC detects 6 meaningful clusters for this data set. Each cluster is mainly represented as one type, see Fig. 6.9 in detail. Cluster 1 is composed of 147 instances and 140 out of them belong to cytoplasm, which results in $P = 95.24\%$ and $R = 97.90\%$. Cluster 2 includes 56 instances and mainly represents perisplasm with $P = 85.71\%$ and $R = 92.31\%$. The type of inner membrane without signal sequence is identified by cluster 3 and cluster 4 together. Similarly, the cluster 5 and cluster 6 represent the type outer membrane and outer membrane lipoprotein with high precision 92.86% and 100% respectively. For 4C algorithm, it obtains best results with parameters $Minpts = 6$, $\epsilon = 0.15$ and $\lambda = 7$. It detects 6 clusters but many instances are wrongly clustered. ORCLUS obtains much better results with parameters $k = 8$ and $l = 7$. However, like 4C, many instances are wrongly assigned. The algorithm of Curler has similar results like 4C and ORCLUS, which cannot predict the protein location effectively. The evaluation of the clustering results is further illustrated in Table 6.4.

Wine Data: The well-known wine data set is the results of a chemical analysis of wine grown in the same region in Italy but deriving from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wine. The class distribution is as follows:

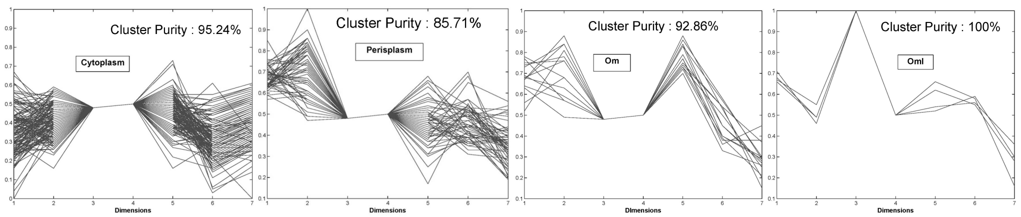


Figure 6.9: Clusters found by ORSC on the Ecoli data set ($\varepsilon = 0.25$)

type1: 59; type2: 71, type3: 48. Table 6.5 gives the clustering results of ORSC with parameter $\varepsilon = 0.6$. It detects 4 clusters and 8 instances are viewed as noise. Three main detected clusters match with corresponding wine types with high precision and recall. In detail, cluster 1 includes nearly all instances (58 out of 59 instances) of type 1 plus 3 instances of type 2. 53 instances in cluster 2 completely belong to type 2. Cluster 3 consists of all instances of type 3. In addition, the cluster 4 includes 4 instances of type 2. It is clear that ORSC can discover interesting patterns of the data set effectively. For 4C with parameter $Minpts = 6$, $\epsilon = 0.5$ and $\lambda = 13$, it discovers 2 clusters and 34 instances are regarded as noise. Many instances of the two clusters are wrongly clustered and it is difficult to distinguish the three wine types. The algorithm of ORCLUS can not obtain good clustering results although we manually specify the number of clusters with parameters $k = 3$ and $l = 13$. For Curler, it even can not find any correlation cluster for the data set with different parameters. All instances are regarded as a cluster. The evaluation of these clustering results is further indicated in Table 6.4.

Table 6.5: ORSC clustering results on Wine Data.

Cluster ID	Type1	Type2	Type3	Prec.	Rec.
1	58	3	0	95.2%	98.3%
2	0	53	0	100%	73.6%
3	0	5	48	90.6%	100%
4	0	4	0	100%	5.6%

6.4.2 Efficiency

Fig. 6.10 shows the results of runtime experiments for a 3-d synthetic data set varying numbers of objects in the range of 500 to 30,000. ORCLUS and ORSC scale nearly linear against the size of the data set while 4C scales quadratically. The runtime of Curler is rather high for large number of objects and thus only the runtime for maximal 10,000 objects is displayed. For comparison of the scalability in the dimensionality d, data sets consisting of 2000 objects of dimensionality ranging from 5 to 50 are generated. Fig. 6.11 displays the results of runtime against dimensionality. Since all algorithms are involved with PCA, they scale similar with the dimensionality. ORCLUS is the fastest approach but its effectiveness is not satisfying (cf. Section 6.4.1) and ORSC outperforms 4C and Curler. In summary, ORSC scales very well against the number of objects as well as dimensionality.

6.5 Conclusions

In this chapter, the algorithm ORSC is proposed for subspace clustering. Each dimension of object is considered as a phase oscillator and the dynamics

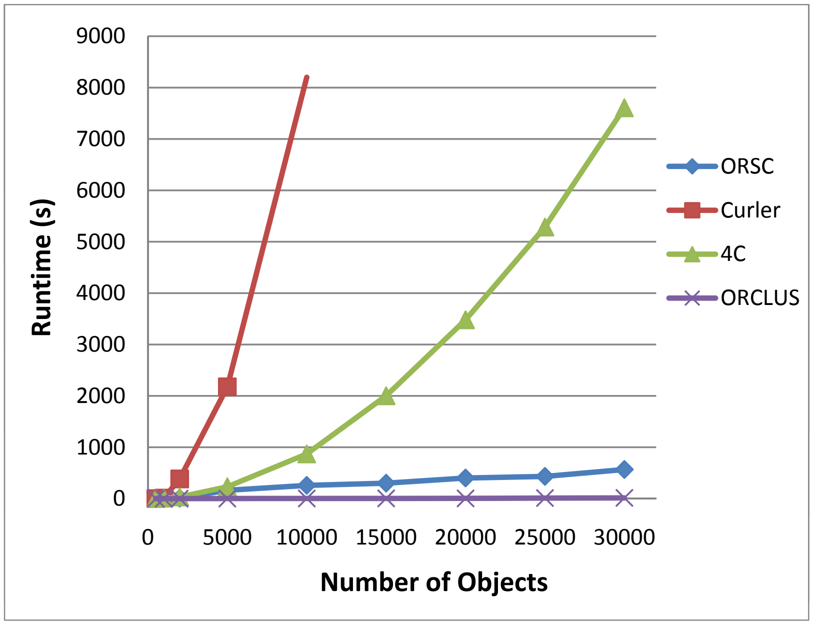


Figure 6.10: Scalability of ORSC against the size of the data set.

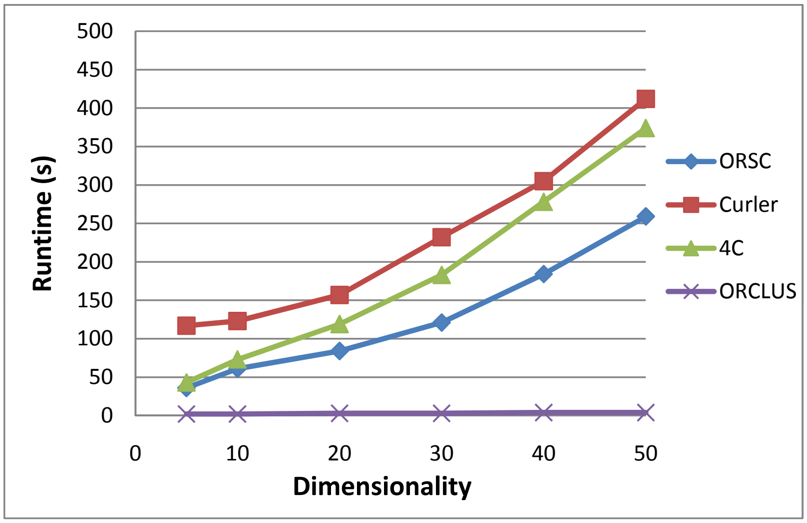


Figure 6.11: Scalability of ORSC against the dimensionality of the data set.

of each object are simulated according to the proposed weighted local interaction model. Through the weighted non-linear interaction among objects, all axis-parallel and arbitrarily oriented subspace clusters naturally synchronize together. To the best of our knowledge, ORSC is the first approach relating the problem of finding subspace clusters in high-dimensional data to synchronization. The extensive experiments demonstrate that ORSC algorithm has several desirable properties. (a) ORSC can naturally detect arbitrarily oriented subspace clusters driven by the natural topological structure of the data without assuming any data distribution; (b) ORSC is robust against noise points and outliers since they are difficult to synchronize with other objects. (c) ORSC is robust against parameter settings and easily to parameterize in comparison to existing approaches, requiring e.g. the number of clusters and subspace dimensionality (d) ORSC outperforms most comparison methods in terms of efficiency although the ORCLUS are much faster but cannot achieve the same level of quality.

Chapter 7

Robust Synchronization-based Graph Clustering

The last three chapters focus on the clustering of traditional feature vector data. However, complex graph data now arises in various fields like social networks, protein-protein interaction networks, World Wide Web, ecosystems, etc. In order to reveal the underlying patterns in graphs, an important task is to partition them into several meaningful clusters (also called communities). The question is: how to find the natural partitions which truly reflect the intrinsic patterns of a complex graph? In this chapter, *RSGC*, a novel approach to graph clustering, is introduced. The key philosophy of *RSGC* is to consider graph clustering as a dynamic process towards synchronization. Specifically, for each vertex, it is viewed as an oscillator and it interacts with other vertices (oscillators) according to the graph connection information, such as interaction weight and range. During the process towards synchronization, vertices with similar connectivity patterns will exhibit similar dynamics and thus naturally synchronize together to form a cluster. Compared with other existing graph clustering algorithms, *RSGC* has some major benefits: (a) it provides a novel and natural perspective for graph clustering based on the proposed interaction model, which captures the characteristics of real-world networks very well. (b) *RSGC* allows discovering graph clusters with arbitrary size and density. (c) *RSGC* is also robust against noise vertices or outliers. (d) *RSGC* does not involve any difficult or sensitive input parameters.

The remainder of this chapter is organized as follows: Introduction is given in Section 7.1. Section 7.2 briefly surveys the related work of graph clustering. Section 7.3 presents our algorithm in detail. Section 7.4 contains an extensive experimental evaluation and Section 7.5 concludes the chapter. Parts of the material presented in this chapter have been submitted in [148], where J.S. was mostly responsible for the development of the main concept; J.S. and X.H. implemented the algorithm; J.S., X.H. and Q.Y. performed the experiments; J.S., C.P., C.B., X.H. and Q.Y. contributed to the paper writing.

> "*Junming Shao, Xiao He, Claudia Plant, Qinli Yang and Christian Böhm. Robust Synchronization-based Graph Clustering. Submitted for publication.*"

7.1 Introduction

Advanced modern technologies produce huge amounts of data in various fields. As a data type, graph-structured data becomes increasingly popular in many applications, e.g. social network, protein-protein interaction network, World Wide Web, and ecosystems. Entities, often called nodes or vertices, are linked to each other by edges expressing relationships between entities. To reveal the underlying patterns from such complex graph data, graph clustering provides a promising way. It investigates the graph structure by identifying the meaningful clusters of graphs where vertices in a cluster are highly connected while the connections between clusters are sparse.

During the last decades, the study of graph clustering has attracted a huge volume of attention and a large number of algorithms are proposed, e.g. Metis [85], Cross-association [44], MCL [155], PaCCo [114] to mention a few. However, discovering the natural clusters in complex graphs is a non-trivial task. In real world, the connectivity between vertices in graph is often associated with different intensities. Therefore, for graph clustering, it should consider not only the information whether two vertices are connected or not, but the intensity of the connections (called edge weights) should also be well integrated. If the edge weights are neglected through binarization or thresholding, the true graph structure cannot be revealed but at most roughly approximated [114]. To illustrate a natural graph partitioning, Figure 7.1 displays the desired clustering result for a simple weighted graph. The vertices assigned to a cluster are drawn with the same color and are enclosed in a dashed circle. The objective of graph clustering is not only to minimize the number of edges between clusters but also consider the intensity of interactions (edge weights) effectively. In addition, another challenge of most established graph clustering algorithms is parametrization. Due to the lack of prior knowledge of given data, it is difficult for users to specify

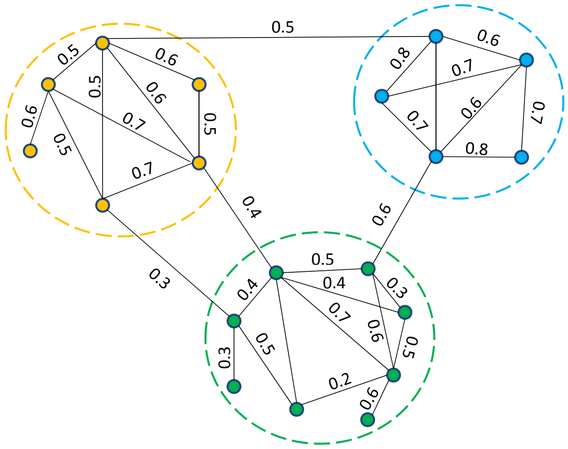


Figure 7.1: An example of clustering weighted graph. Here three clusters exist and are denoted by the dashed circles with different colors. Clustering a weighted graph maximizes the number of edges inside each cluster while minimizing edges between clusters also takes the edge weights into consideration.

some parameters, such as the number of clusters.

To deal with these problems, in this chapter, a new graph clustering based on the synchronization principle is proposed. To better illustrate the concept of synchronization, social networks can be taken as a good example. It is interesting to see that people in a social network with similar attributes, e.g. common interest, friends, or similar calling behaviors from phone companies tend to be a community. In such network, regarding an event, e.g. common opinion formation, it is easy to see synchronization phenomenon. Given a certain problem, in the beginning, each person has his/her own opinion. As time evolves, people will discuss with their known correspondents (connected vertices in the network) and change their opinions gradually. The closer relationship (higher weight) they have, the higher change they have. Through these weighted interactions, finally, people with similar attributes tend to achieve the same opinion. Unlike in Chapter 3, the interaction information of objects (connectivity information) is directly available in the graph data.

Therefore, for graph clustering, an intuitive idea is to consider it as a dynamic process towards synchronization. Each vertex is regarded as an oscillator and it interacts with other vertices (oscillators) relying on its intrinsic connection information. The graph clustering is thus transformed into investigating the dynamics of vertices during the process towards synchronization. A graph cluster is defined as a set of vertices which finally synchronize together.

Contributions

In this chapter, the algorithm *RSGC*: **R**obust **S**ynchronization-based **G**raph **C**lustering, is presented. The major benefits of *RSGC* can be summarized as follows:

1. *Natural Interaction Model.* Rooted in the concept of synchronization, *RSGC* proposes a natural interaction model to capture the properties of the real-world networks. The connection patterns among vertices, such as edge weights, interaction range, are naturally integrated into the interaction model for graph clustering.

2. *Intrinsic cluster structure discovery.* Based on the proposed interaction model, the hidden cluster structure of a graph can be effectively detected during the process of synchronization. *RSGC* allows discovering

graph clusters with arbitrary size and density.

3. *RSGC is robust against outliers.* Inherited from the properties of synchronization, *RSGC* can find the outlier vertices in graphs by investigating their different dynamic behaviors.

4. *Automatic.* *RSGC* is automatic and does not involve any difficult or sensitive input parameters.

7.2 Related Work

During the past several decades, many approaches have been proposed for graph clustering, such as Metis [85], MCL [155], Cross-association [44], Spectral [176], and PaCCo [114]. Here, only a very brief survey on graph clustering is provided. For detailed reviews, please refer to [140].

Graph Clustering: One of the most prevailing methods for graph clustering is spectral partitioning. This approach refers to a class of well-known techniques which rely on the Eigenvector decomposition of a similarity matrix to partition objects into disjoint clusters [165]. The algorithm proposed by [116] allows detecting arbitrarily shaped clusters in vector data by considering the clustering problem from a graph-theoretic perspective. A cluster is obtained by removing the weakest edges between highly connected subgraphs, which is formally defined by the normalized cut or similar objective functions. In [14], they propose a learning method to derive a new cost function based on a measure of error between a given partition and a solution of the spectral relaxation of a minimum normalized cut problem. Like k-Means [101], the problem of most spectral clustering approaches requires to specify the suitable number of k clusters. To overcome the difficulty of parameter setting, Zelnik-Manor and Perona [176] recently propose a new method to estimate the number of clusters by investigating the structure of the Eigenvectors.

Recently, a large number of approaches are proposed for graph clustering based on the information-theoretic principle. The key idea of these methods is to link the clustering problem to data compression. The more bits can be saved by compression, the better of the clusters are. One of fundamental techniques in this line is Cross-Association [44], which finds groups in unweighted graphs by loss-less compression with Mimimin Description Length (MDL). Similar to Cross-Association, the algorithm called PaCCo [114], is proposed for weighted graph, which combines the MDL principle with a bisecting k-Means strategy. The weights of a graph are further modeled by a predefined set of PDFs. In general, the MDL principle provides a good way to qualify the clustering results and thus avoids the parameter setting. However, it tends to fail if the weights of the graph do not correspond to the assumed model.

7.3 Synchronization-based Graph Clustering

Inherited from the concept of synchronization, the novel approach *RSGC* is introduced for graph clustering. As described in Section 7.1, the key idea is to consider the graph clustering as a dynamic process towards synchronization. For each vertex in a graph, it is regarded as a phase oscillator and interacts with other oscillators relying on its connectivity information. The dynamics of each vertex during the process towards synchronization is then investigated with an interaction model (cf. Section 7.3.2). Vertices with similar interaction patterns will finally synchronize together and form a cluster.

Thus, in order to model each vertex as an oscillator, the first step is to extract features from a graph to represent its phase. In the following, the vertex feature representation is first presented. After that, an interaction model for graph clustering is introduced. In Section 7.3.3 the algorithm *RSGC* is discussed in detail.

7.3.1 Vertex Feature Representation

Given an undirected graph G, in most cases, the only information can be gained is its connectivity patterns. Therefore, to extract features from graph to represent the phases of vertices, the dissimilarity among vertices based on their connectivity information is investigated. First, some necessary definitions are introduced.

Definition 7.1 (UNDIRECTED GRAPH) Let $G = (V, E, W)$ be an undirected graph, where V is the set of vertices and E is the set of edges. $e = \{u, v\} \in E$ indicates a connection between the vertices u and v. $W(e)$ represents the intensity of connection of edge e.

Definition 7.2 (NEIGHBORS OF VERTEX u) Given an undirected graph $G = (V, E, W)$, the neighborhood of a vertex $u \in V$ is the set $\Gamma(u)$ containing vertex u and its connected vertices.

$$\Gamma(u) = \{v \in V | \{u, v\} \in E\} \cup \{u\}$$

Currently, the most straightforward way of determining whether two vertices are similar is to study their structural similarity based on their common neighbors. The more common neighbors the two vertices have, the more similar they are. In order to quantify the similarities among vertices, a good choice is the Jaccard coefficient [77].

Given any two vertices u and v, the Jaccard coefficient is defined as:

$$\rho(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{7.1}$$

Based on the Jaccard coefficient, the Jaccard distance of any two vertices can be defined as follows.

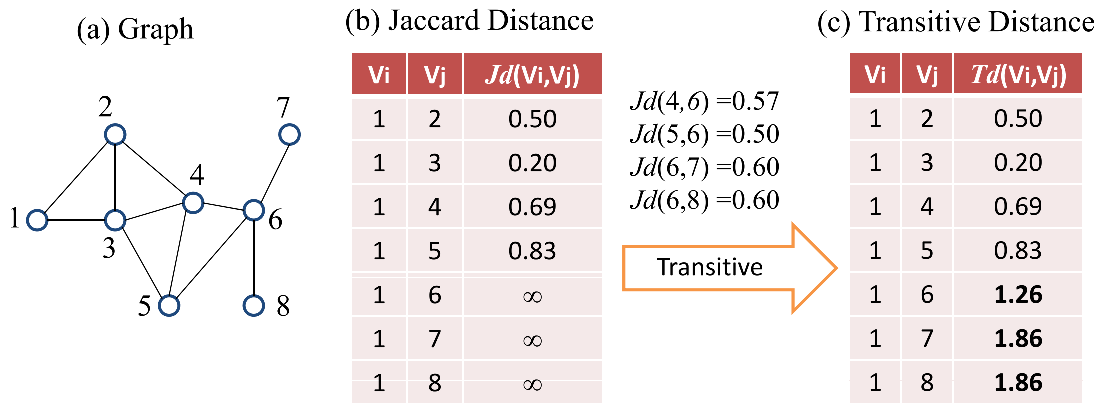
Definition 7.3 (Jaccard distance) Given any two vertices u and v in Graph G, their Jaccard distance $Jd(u,v)$ is written as:

$$Jd(u,v) = \begin{cases} 1 - \rho(u,v) & \rho(u,v) > 0 \\ \infty & else \end{cases} \tag{7.2}$$

The vertex similarity measure based on Jaccard coefficient normalizes the number of common neighbors by summing up the size of both neighborhoods. It thus effectively denotes the local connectivity density of any two adjacent vertices in an undirected graph. However, the Jaccard distance of the pairs of non-neighbor vertices is ∞, which is insufficient to represent the similarity of any two vertices effectively. Therefore, the Transitive distance is further defined.

Definition 7.4 (Transitive distance) If $(u, \cdots, v_i, w) \in S$ are the shortest paths between vertices u and w, v_i is the second end vertex of any shortest path, $\rho(u, v_i) \neq \infty$, $\rho(v_i, w) \neq \infty$, the Transitive distance between vertices u and w is defined as the minimal distance of these shortest paths according to Equation 7.2:

$$Td(u,w) = \begin{cases} Jd(u,w) & Jd(u,v) \neq \infty \\ \min\{Jd(u,v_i) + Jd(v_i,w)\} & else \end{cases} \tag{7.3}$$



Vi	Vj	*Jd*(Vi,Vj)
1	2	0.50
1	3	0.20
1	4	0.69
1	5	0.83
1	6	∞
1	7	∞
1	8	∞

Vi	Vj	*Td*(Vi,Vj)
1	2	0.50
1	3	0.20
1	4	0.69
1	5	0.83
1	6	**1.26**
1	7	**1.86**
1	8	**1.86**

Figure 7.2: Illustration of computing the dissimilarity among vertices. (a) The example graph. (b) The Jaccord distances from vertex 1 to other vertices. (c) The distance from vertex 1 to other vertices based on the Transitive distance.

Based on the Transitive distance, the dissimilarity matrix D for all pairs of vertices can be effectively captured, where each element $D_{uv} = Td(u,v)$ captures the dissimilarity between any two vertices u and v. Figure 7.2 illustrates the process of computing the dissimilarity from vertex 1 to all other vertices. E.g., for the distance between vertex 1 and vertex 6, the shortest paths from 1 to 6 contain {(1,2,4,6),(1,3,4,6),(1,3,5,6)}. The transitive distance between vertex 1 and vertex 6 is thus $Td(1,6) = min(Jd(1,4) + Jd(4,6), Jd(1,5) + Jd(5,6))$ =1.26.

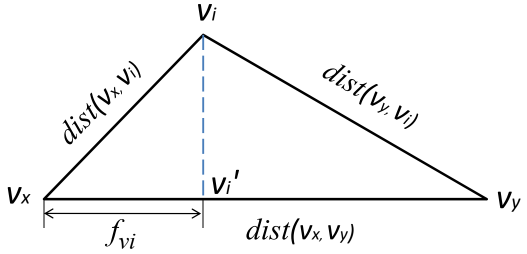


Figure 7.3: Illustration of the feature space transformation by projection in one dimension. V_x and V_y are two vertices with the maximal distance. Any vertex including the two vertices can be projected in the line (V_x-V_y). f_{vi} indicates the corresponding feature of vertex V_i.

To further obtain the features of each vertex, the dissimilarity matrix is mapped into 2-dimensional points in a feature vector space. Here, the well-known method, called $FastMap$ [56], is applied to transform the dissimilarity matrix in a metric space into a feature vector space. The key point is to project the vertices on a carefully selected 'line' and the feature of each vertex is then represented as the length of projection on the 'line'. For illustration, Figure 7.3 indicates the projection of the vertices in 1-dimensional space. Here, all vertices are projected into a 2-dimensional feature vector space and their corresponding coordinates are represented as their phases.

7.3.2 Interaction Model

The Kuramoto model (cf. Section 3.2) well describes the global synchronization behavior of all coupled phase oscillators. In real-world graphs or networks, the connectivity among vertices are often not full but partial, which indicates that only a local ensemble of vertices are connected. Therefore, it is necessary to extensively reformulate $Eq.(3.1)$. The natural way is to model the interaction as the graph structure itself.

Definition 7.5 (Interaction Range of vertex u) The interaction range of vertex u, namely R_u, is defined as the maximal distance among these neighbors, formally,

$$R_u = \max\{dist(u, v)\}|v \in \Gamma(u)\} \tag{7.4}$$

where $dist(\cdot)$ is the metric function. L_2 distance is used in this chapter.

After calculating the interaction range for each vertex, its dynamic behavior with other vertices can be simulated according to an interaction model. Formally, the interaction model for graph clustering is defined as follows.

Definition 7.6 (Interaction Model) Let u be a vertex in the graph $G(V, E, W)$. $\Gamma(u)$ are the neighbors of vertex u and u_i is the $i-th$ feature of vertex u. With the interaction rangeR_u, according to Eq. (3.1), the dynamics of $i-th$ phase u_i of the vertex u is governed by:

$$\frac{du_i}{dt} = \omega_i + \frac{K}{\sum_{v\in\Gamma(u)} W(v, u)} \sum_{v\in\Gamma(u)} W(v, u)\cdot \tag{7.5}$$

$$\Phi(u, v) \cdot \sin(v_i - u_i).$$

where W(v,u) is the edge weight between vertices u and v. The $\Phi(u, v)$ is used to check whether the vertex v is in the interaction range of u, which is defined as:

$$\Phi(u, v) = \begin{cases} 0 & dist(v, u) > R_u \\ 1 & else \end{cases}$$

Then, let $dt = \Delta t$, the Equation 7.5 can be further written as:

$$u_i(t + \Delta t) = u_i(t) + \Delta t \cdot \omega_i + \frac{\Delta t \cdot K}{\sum_{v\in\Gamma(u)} W(v, u)} \sum_{v\in\Gamma(u)} W(v, u)\cdot \tag{7.6}$$

$$\Phi(u(t), v(t)) \cdot \sin(v_i(t) - u_i(t)).$$

Here, as mentioned in last chapters, all vertices (oscillators) are assumed having the same frequency w. The term $\Delta t \cdot \omega_i$ is thus the same for each vertex and ignored. $\Delta t \cdot K = C$ is a constant. Finally the dynamics of $i-th$ phase u_i of the vertex u over time is provided by:

$$u_i(t + \Delta t) = u_i(t) + \frac{C}{\sum_{v \in \Gamma(u)} W(v, u)} \sum_{v \in \Gamma(u)} W(v, u) \cdot \tag{7.7}$$

$$\Phi(u(t), v(t)) \cdot \sin(v_i(t) - u_i(t)).$$

The vertex u at time step $t = 0$: $u(0)$ represents the initial phase of the vertex (the original feature of vertex u). The $u_i(t+\Delta t)$ describes the renewal phase value of i-th phase of vertex u at the $t = t + \Delta t$ time evolution.

To characterize the level of synchronization between oscillators during the synchronization process, a graph order parameter r_g is defined to measure the coherence of the local oscillator population in graphs.

Definition 7.7 (Graph Order Parameter) Let $u \in G(V, E, W)$, $|V| = N$, the graph order parameter r_g characterizing the degree of local synchronization in graphs is provided by:

$$r_g = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{\sum_{v \in \Gamma(u)} W(v, u)} \sum_{v \in \Gamma(u)} W(v, u) \cdot \Phi(u, v) \cdot ||v - u|| \right) \tag{7.8}$$

The r_g calculates the average interaction during the process towards synchronization among all vertices. Therefore, the more vertices synchronize together, the value of r_g becomes smaller. The dynamics of all vertices will terminate when the r_g converges, which indicates that the vertices in clusters synchronize together (in phase).

7.3.3 The RSGC Algorithm

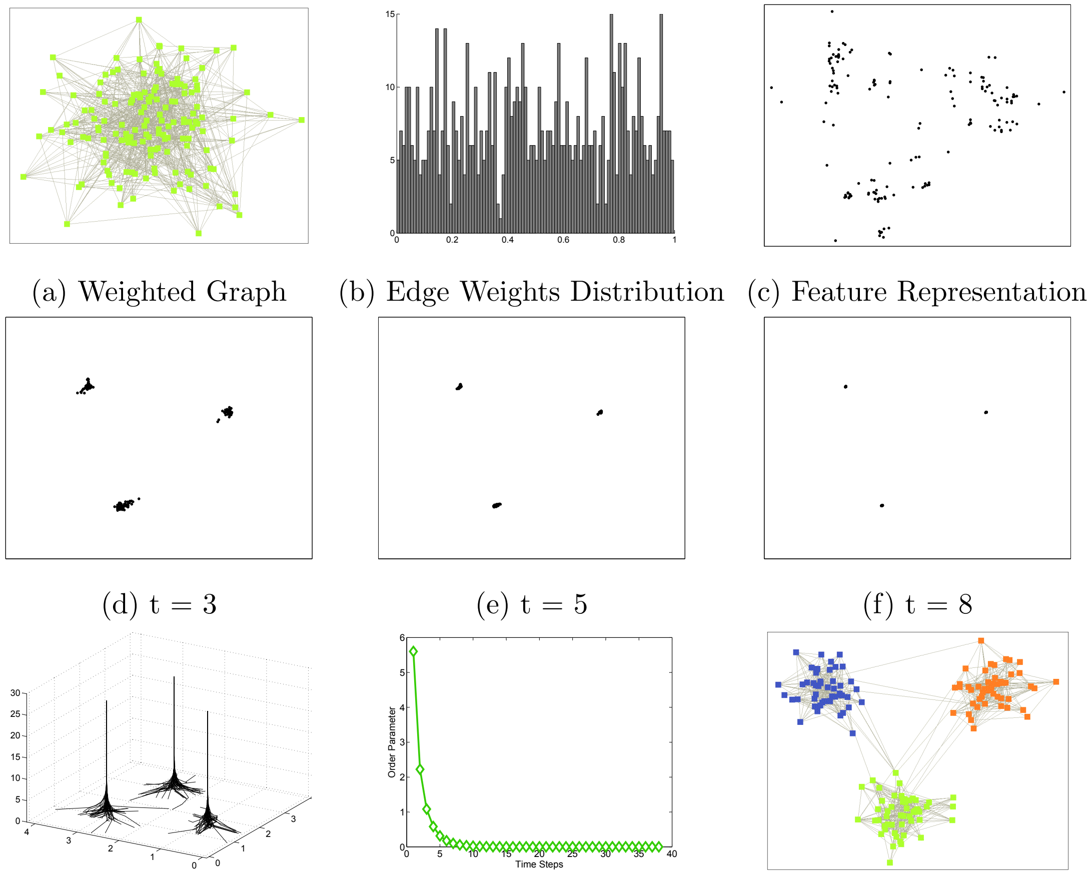
With the interaction model, the dynamics of vertices can be simulated. Generally, the process of the graph clustering based on synchronization involves the following steps:

1. **Vertices Feature Representation**: Given a graph G, the dissimilarity matrix D is first calculated according to Eq. (7.1) (7.2) (7.3). Then $FastMap$ is used to transform dissimilarity matrix D into feature space so that each vertex is represented as a 2-dimensional feature vector.

2. **Synchronization on Graph**: After feature representation for each vertex, its dynamics are investigated with the interaction model. In the beginning ($t = 0$), all vertices have their own phases and thus have N disconnected sets (N subgraphs). As time evolves, each vertex interacts with other vertices relying on its connection information according to Equation 7.7. The interaction strength of each pair of vertices is naturally fitted as the edge weight (for unweighted graph, all weights of edges are equal and set as 1). Through interactions, vertices with similar patterns will gradually synchronize together. The process of synchronization terminates when the graph order parameter converges (Definition 7.7).

3. **Clusters Discovery**: At the end of synchronization, all clusters are easily detected by exploring the final phases of vertices. A graph cluster is defined as a set of vertices with the same phase.

For illustration, a simple weighted graph is introduced in Figure 7.4(a). Figure 7.4(b) plots the edge distribution of the graph. The $RSGC$ starts



(a) Weighted Graph (b) Edge Weights Distribution (c) Feature Representation

(d) t = 3 (e) t = 5 (f) t = 8

(g) Dynamics of Vertices (h) Graph Order Parameter (i) Graph partitioning

Figure 7.4: The simulation of the dynamics of vertices during the process towards synchronization. (a) The original weighted graph without ordering and partition. (b) The underlying edge weights distribution. (c) The Feature Representation, where each point in the feature space represent a vertex. (d-f). The states of vertices at time steps $t = 3\Delta t$ and $t = 8\Delta t$. (g) The dynamics of vertices over time resolution. (h) Graph order parameters. (i) The result of graph partitioning.

with the vertices similarity computation and then projects it in feature space, which is indicated in Figure 7.4(c). After that, the dynamics of all vertices can be simulated according to *Eq.*(7.7). Figure 7.4(g) displays the dynamic movement of all vertices during the process towards synchronization. It is obvious that these vertices which are highly connected gradually move together and finally have the same phase. Figure 7.4(d)-(f) further depict the detailed states of vertices at time step $t = 3\Delta t, 5\Delta t, 8\Delta t$. When $t = 8\Delta t$, we can see the vertices in a cluster synchronize together. The process of synchronization will be terminated when the graph order parameter r_g converges, which indicates in Figure 7.4(h). Finally, the result of graph partitioning is showed in Figure 7.4(i). The pseudocode of *RSGC* algorithm is provided as follows.

7.4 Experiments

To extensively study the performance of *RSGC*, multiple experiments are conducted on several synthetic as well as real-world data sets. Furthermore, *RSGC* is compared with two other parameter-free weighted graph clustering paradigms: information-theoretic clustering *PaCCo* [114] and the spectral clustering approach proposed by Zelnik-Manor and Perona [176] (named *Spectral* in the following). *RSGC* is implemented in Java. The source codes of *PaCCo* and *Spectral* are obtained from the authors and are implemented in Java and C++ respectively. All experiments have been performed on a workstation with 2.0 GHz CPU and 8.0 GB RAM.

Algorithm 3 RSGC algorithm

```
Input: Graph $G(V, E, W)$

Compute matrix $A$ with Jaccard Coefficient ;
Obtain dissimilarity matrix $D$ with transitive distance & $A$;
Transform $D$ into features vectors $F$ using FastMap;

for (Each $u \in F$) do
  $u(0) = F_u(0)$
end for
while (loopFlag = True) do
  for (Each $u \in F$) do
    Obtain new phase $u(t+1)$ using $Eq.(7.7)$;
  end for
  Compute graph order parameter $r_g$;
  $F(t+1) = \sum_1^N \bigcup u(t+1)$;
  if $r_g$ converges then
    loopFlag = False;
    $C = synCluster(F(t+1))$;
  end if
end while

return $C$;
```

7.4.1 Evaluation Criteria

For objective comparison, a criterion called *modularity*[64] is applied to quantify the quality of a division of a network into modules or communities. The modularity value is higher if links inside a cluster are maximized while the links between clusters are minimized. Formally, modularity on a graph partitioning of an unweighted graph into k clusters is defined as

$$M = \sum_{c=1}^{k} \left(\frac{l_c}{L} - \left(\frac{d_c}{2L} \right)^2 \right)$$

with L being the number of all edges inside the graph, l_c being the number of links enclosed by the cluster c, and d_c being the sum of the degree of the nodes of cluster c.

Based on the above definition, we can thus reformulate modularity for a weighted graph, by considering the edge weights instead of their counts. We define weighted modularity as

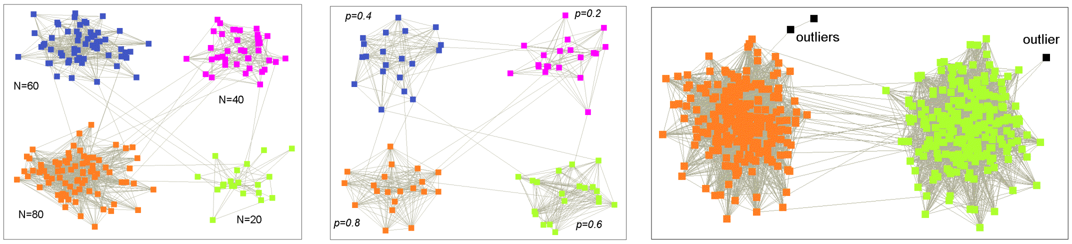
$$M = \sum_{c=1}^{k} \left(\frac{w_c}{W} - \left(\frac{d_{w_c}}{2W} \right)^2 \right)$$

where W indicates the sum of all edge weights and w_c is the sum of weighted edges inside the cluster c. d_{w_c} is the total weighted degrees inside a cluster.

7.4.2 Proof of Concept

The evaluation starts with some synthetic data sets to facilitate presentation and demonstrate the benefits of *RSGC* mentioned in Section 7.1.

Intrinsic Cluster Structure Discovery. First the performance of *RSGC* is evaluated to discover natural graph partitioning in difficult settings, starting with subgraphs of arbitrary size. The data set displayed in



(a) Different Sizes (b) Various Densities (c) Outliers Handling

Figure 7.5: Performance of *RSGC*.

Figure 7.5(a) consists of 4 clusters with different sizes, ranging from 20 to 80. The intra-connection is approximately 40% and the distribution of edge weights is Gaussian. *RSGC* successfully detects all clusters without any data or edge weights distribution assumptions. Moreover, we generate four clusters with different densities. For each cluster, it includes 20 vertices and the probabilities of intra-connection of each cluster vary from 20% to 80%, see Figure 7.5(b). *RSGC* allows detecting clusters with very different densities.

Outlier Handling. In real-world networks or graphs, there often exist some noise or outlier vertices, which behave with different roles or behaviors in graphs. Discovering such outlier vertices is very important for graph analysis. Inherited from the concept of synchronization, the *RSGC* algorithm allows detecting all these outliers effectively, where all these vertices in graphs are difficult to synchronize with other vertices and have different dynamics. As displayed in Figure 7.5(c), the outliers are naturally found by exploring these vertices which are not synchronized.

7.4.3 Real-world Data

In this section, the performance of *RSGC* is evaluated on several real-world data sets. Five author-collaboration networks are obtained from different communities: Network Theory (Netsci), PhD Student Network in Computer Science (CS-PhD), Computational Geometry (Geom), Arxiv General Relativity (CA-GrQc), Erdos Research (Erdos) [1]; three Protein-Protein Interaction networks from three species, which are *S. cerevisiae (Scere)*, *Escherichia coli (Ecoli)* and *C.elegans (Celeg)*[2] and an Autonomous Systems network of routers comprising the Internet (As) [3].

These 9 real-world data sets are selected to evaluate the performance of *RSGC* and compare with two other approaches. However, data preprocessing is first conducted. The reason is that some of these data sets (Netsci, CS-Phd, Geom, CA-GrQc, Scere, Ecoli and Celeg) only have one major subgraph. They contain many small isolated subgraphs, which have no connection to others at all. For instance, the author-collaboration network Netsci has 396 isolated subgraphs, among which there are one subgraph with 379 nodes and hundreds of subgraphs with only several vertices. The PPI network Celeg has 133 isolated subgraphs, among which there is one subgraph with 2880 nodes and the vertices that others contain are no more than 5. Obviously,

[1] CS-Phd, Geom, Erdos are Obtained from the website: http://vlado.fmf.uni-lj.si/pub/networks/data/default.htm.

Netsci: http://www-personal.umich.edu/ mejn/netdata.

CA-GrQc: http://snap.stanford.edu/data/

[2] Obtained from the Database of Interacting Proteins: http://dip.doe-mbi.ucla.edu/dip/Main.cgi

[3] As: http://snap.stanford.edu/data/

Table 7.1: The statistics of the real data sets.

Data Set	weight	$\lvert V\rvert$	$\lvert E\rvert$	Avg. degree	CC
Netsci	weighted	379	914	2.411	0.741
CS-Phd	unweighted	1025	1043	1.018	0.003
Geom	weighted	3621	9461	2.612	0.539
CA-GrQc	unweighted	4158	13422	3.228	0.557
Erdos	unweighted	6927	11850	1.711	0.124
As	unweighted	6474	12572	1.942	0.252
Scere	unweighted	5157	24879	4.824	0.098
Ecoli	unweighted	2595	11726	4.548	0.098
Celeg	unweighted	2880	4812	1.671	0.019

before clustering, it is necessary to preprocess such data sets. Here, for each graph, the subgraphs of the network are first obtained and then the biggest subgraph taken for comparison. The reason is that all these data sets only have one primary subgraph. If there is no preprocessing, it is difficult to define the similarity among vertices in our algorithm *RSGC* and *Spectral*. After preprocessing, the statistics of the data sets is further indicated in Table 7.1.

The experiments are performed on these real data sets and the performance of *RSGC* is compared to *PaCCo* and *Spectral*. Table 7.2 shows the clustering results in terms of *modularity score*. It is obvious that *RSGC* performs very well on all these data sets and obtains the best modularity scores for all experiments except the *Spectral* algorithm on *As* data set. For the algorithm of *PaCCo*, it can not yield good partition results for most data sets, especially for the unweighted graphs. The reason behind this is that *PaCCo* tends to fail if the weight distribution does not correspond to the cluster model. Therefore, for most data sets, *PaCCo* only partitions these graphs into two clusters. Like *PaCCo*, *Spectral* also obtains relatively few clusters except for the *CS-PhD* and *As* data sets.

Table 7.2: Performance of graph clustering algorithms on real-world data sets, including *RSGC*, *Spectral* and *PaCCo*.

Data Set	PaCCo		Spectral		RSGC 2	
	#clu.	Modu.	#clu.	Modu.	#clu.	Modu.
NetSci	11	0.542	4	0.696	19	**0.779**
CS-Phd	2	0.077	40	0.531	36	**0.715**
Geom	54	0.327	6	0.067	44	**0.465**
CA-GrQc	2	0.352	4	0.144	46	**0.627**
As	2	0.151	19	**0.439**	39	0.382
Erdos	2	0.049	2	0.001	24	**0.422**
Scere	2	0.085	3	0.053	32	**0.196**
Ecoli	2	0.052	2	0.002	25	**0.224**
Celeg	2	0.019	2	0.005	32	**0.309**

In total, it is noticeable that *RSGC* have obvious benefits over the two compared algorithms. First, without any weight or data distribution assumption, *RSGC* can automatically partition the graphs into reasonable clusters. Second, *RSGC* also allows detecting more meaningful clusters in the graphs (with high modularity scores).

Case Study: To further evaluate the performance of *RSGC*, a case study

Table 7.3: Biological significant clusters detected by different graph clustering algorithms.

Molecular Function Annotations		P-value
RSGC	structural constituent of ribosome	1.2e-17
	acetylcholine receptor activity	3.1e-6
	protein binding	0.002
PaCCo	structural constituent of ribosome	2.3e-9
Spectral	structural constituent of ribosome	1.3e-19

Biological Processing Annotations		P-value
RSGC	embryo development	2.1e-24
	reproduction	2.9e-11
	growth	0.004
	multicellular organismal reproductive pro.	0.006
	protein localization	0.007
	morphogenesis of an epithelium	0.007
	germ cell development	0.009
	translation	0.011
	inductive cell migration	0.045
PaCCo	reproduction	3.7e-20
	embryo development ending in birth	6.8e-18
Spectral	reproduction	3.1e-37

on a protein-protein interaction (PPI) network is illustrated. The objective of PPI network analysis is to investigate the structure of interaction networks of proteins and thus provide insight into their functions where proteins in a subgraph indicate the same biological process or similar molecular functions. Currently, it is an interesting and challenging problem since the functions of many proteins are unknown even for the most well-studied organisms such as yeast.

Here, the latest version of PPI network of C.elegans (Celeg) is used, which contains 2880 proteins and 4812 known interactions. This interaction network is analyzed with *RSGC* and the performance is compared to *PaCCo* and *Spectral*. *RSGC* discovers 32 clusters, while *PaCCo* and *Spectral* produce only 2 clusters, respectively. In the context of biology, the biological significance of obtained clusters can be evaluated with the help of the Gene Ontology database [13], which provides the ontology of defined terms representing gene product properties in three vocabularies of annotations: Molecular Function, Biological Process and Cellular Component. Researchers often apply the P-value to demonstrate the biological significance, which is defined as the probability to observe by chance at least x elements at the intersection between the query set and the reference set. Formally, let the total number of proteins be N with a total of M proteins sharing a particular annotation. Based on Hyper-geometric distribution, the P-value [31] of observing m or more proteins that share the same annotation in a cluster of n proteins is defined as:

$$P - value = \sum_{i=m}^{n} \frac{\binom{N}{n}\binom{N-M}{n-i}}{\binom{N}{n}} \tag{7.9}$$

Under the evaluation with Molecular Function annotations, *RSGC* finds 3 clusters which are enriched for three molecular functions. In contrast, *PaCCo* and *Spectral* only obtain one biological significance cluster for molecular functions. In addition, for all three approaches, they find a significant cluster enriched for the function *structural constituent of ribosome*, where the P-values are 1.2e-17, 2.3e-9 and 1.3e-19 for *RSGC*, *PaCCo* and *Spectral* respectively. In addition, *RSGC* finds another two clusters enriched for *protein binding* (P-value = 2.5e-03), *acetylcholine receptor activity* (P-value = 3.1e-06). Therefore, *RSGC* can detect more clusters which make sense biologically. Similarly, the clusters are also evaluated using biological processing annotations. Here, *RSGC* successfully obtains 9 significant clusters which are enriched for biological processing while *PaCCo* and *Spectral* have two and one significant clusters respectively. All of *RSGC*, *PaCCo* and *Spectral* methods discover one significant cluster which is enriched for the term *reproduction* with the P-values of 2.8e-11, 3.7e-20, 3.1e-37 respectively. Moreover, *RSGC* also reveals another 8 significant clusters enriched for different biological process, such as *embryo development*, *multicellular organismal reproductive process* , *morphogenesis of an epithelium*, etc. Please refer to Table 7.3 for details.

7.4.4 Runtime Complexity

For runtime comparisons, several synthetic data sets are generated, where the number of clusters k ranges from 10 to 50 and each cluster contains 100 vertices. Approximately 30 % of the intra cluster edges are connected and 5% inter cluster edges are linked. The edge distribution of all cluster are Gaussian. In order to obtain more accurate runtime results, for each method, each data set is processed 10 times and then the mean of the 10 rounds is found. Fig. 7.6 clearly shows that *RSGC* is faster than *Spectral* since *Spectral* has a time complexity of O(n^3) due to the eigenvalue decomposition

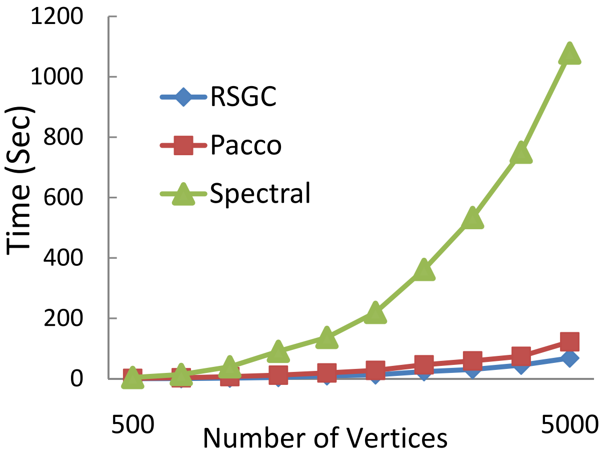


Figure 7.6: The runtime of the different graph clustering algorithms, including *RSGC*, *PaCCo* and *Spectral*.

and time complexity of *RSGC* is only super-linear. In addition, *RSGC* is also slightly better than *PaCCo* (approximately 2 times). In summary, *RSGC* scales very well against the number of vertices.

7.5 Conclusions

In this chapter, the algorithm *RSGC* is introduced for graph clustering. The extensive experiments demonstrate that *RSGC* algorithm has several desirable properties: (a) *RSGC* provides a natural way for graph clustering, where the proposed interaction model can capture the characteristics of the real-world networks very well, such as the interaction weights and range. (b) Relying on the proposed interaction model, *RSGC* allows discovering graph clusters with arbitrary size and density. Moreover, (c) *RSGC* is fully automatic and does not involve any difficult or sensitive input parameters.

Chapter 8

Outlier Detection: “Out of Synchronization”

The study of extraordinary observations is of great interest in a large variety of applications, such as criminal activities detection, athlete performance analysis, and rare events or exceptions identification. The question is: how can we naturally flag these outliers in a real complex data set? In this chapter, outlier detection is considered from a new perspective: “*Out of Synchronization*”. By simulating the dynamical behavior of each data object over time according to an extensive Kuramoto model, regular objects and outliers exhibit different interaction patterns during the process towards synchronization. Outlier objects are naturally detected by local synchronization factor (LSF).

The remainder of this chapter is organized as follows: In the following section, the introduction is given. Section 8.2 briefly surveys the related work of outlier detection. Section 8.3 presents our new concept and algorithm in detail. Section 8.4 contains an extensive experimental evaluation and Section 8.5 concludes the chapter. Parts of the material presented in this chapter have been published in [144], where J.S. was mostly responsible for the development of the main concept; J.S., Q.Y., C.P. and C.B. performed the experiments and wrote the paper.

> "*Junming Shao, Christian Böhm, Qinli Yang, Claudia Plant. Synchronization Based Outlier Detection. ECML/PKDD (3) 2010: 245-260.*"

8.1 Introduction

> "*An outlying observation, or outlier, is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.*" [76]

Such irregular observations often contain useful information on abnormal behavior of the system described by the data. The detection of these irregular data is thus equally or even more interesting and useful than finding regular patterns applicable to a considerable portion of objects in a data set [30]. For example, the identification of criminal activities, such as credit card fraud, is crucial in electronic commerce applications.

Currently, outlier detection has attracted increasing attention and many algorithms have been proposed (e.g. [30] [91] [92] [24]). However, they suffer from one or more of the following drawbacks: They explicitly or implicitly assume the data follows a given distribution model, such as Gaussian, uniform or Exponential Power Distribution. The results of many methods strongly depend on suitable parametrization and/or their results are difficult to interpret. In addition, most approaches are restricted to a flat data structure and do not support complex hierarchical data. A more detailed discussion on these studies will be given in Section 8.2.

In this chapter, outlier detection is considered from a novel perspective: *out of synchronization*. Inspired by natural synchronization phenomena, the outliers are considered as the elements which are difficult to synchronize with others.

8.2 Related Work

Currently, most existing approaches to outlier detection can be mainly classified into three categories: distribution-, distance-, and density-based outlier detection.

Distribution-based Outlier Detection. Methods in this category are mainly developed in the field of statistics. Generally, they assume a known distribution model (Gaussian, Poisson, Exponential Power Distribution, etc.) for the observations based on statistical analysis of a given data set. Outliers are defined as those objects that deviate considerably from the model assumptions [12, 76, 135]. In [12], numerous discordancy tests are discussed for different scenarios. In [174] and [173], authors propose SmartSifter (SS), which is an on-line real-time outlier detection algorithm. The basic principle of SS is to use a probabilistic model (a finite mixture model) to represent the underlying distribution of a given data set. Each time a datum is input and SS employs an on-line learning algorithm to adjust the probability model. An anomaly score is calculated for each datum based on the learned model. However, the method relies on histograms and requires preparing as many Gaussian mixture models as cells in the histogram density. Moreover, in real-world applications, it is not trivial to find an appropriate model to fit an arbitrary data distribution without prior knowledge. Recently, CoCo, an information-theoretic outlier detection approach has been proposed by Böhm, et al. [24]. Based on the MDL principle, outliers are flagged as those objects which need more coding cost than regular objects. Like most distribution-based methods, CoCo tends to fail if the estimated distribution does not fit the data model well (cf. Section 2.2) .

Distance-based Outlier Detection. The concept of distance-based outlier detection is proposed by E.M. Knorr and R.T. Ng [91] [92]. These techniques identify potential outliers from ordinary points based on the number of points in the specified neighborhood. They define a point in a data set T to be an outlier if at least p fraction of points in T have greater distance than d from it [91]. The basic notion is extended in [130] by computing the distances to the k nearest neighbors and then ranking the objects based on their proximity to their k-th nearest neighbors. Consequently top n outliers are obtained using a partition-based algorithm. However, it is difficult to accurately determine the parameters p and d for a arbitrary data set.

Density-based Outlier Detection. Breunig, et al. [30], introduce a notion of local outlier from a density-based perspective. An object is regarded as an outlier if its local density does not fit well into the density of its neighboring objects. The local outlier factor (LOF) is then proposed to capture the degree to which the object is an outlier. (cf. Section 2.2) In [157], a

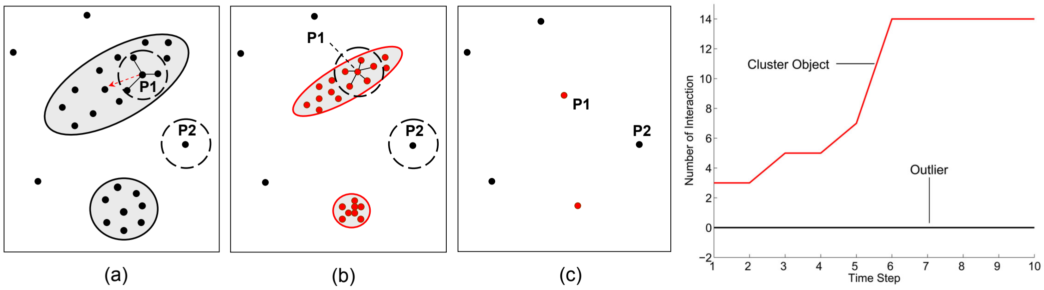


Figure 8.1: Illustration of synchronization-based outlier detection. (a)-(c): The dynamics of objects towards synchronization. (d): Interaction Plot.

connectivity-based outlier factor (COF) scheme is proposed to improve the effectiveness of LOF scheme when a pattern itself has a similar neighborhood density as an outlier. Although these approaches to outlier detection are useful, their performances are sensitive to the parameter $Minpts$ which can be very difficult to determine. The Local Outlier Integral (LOCI) [120] flags outliers based on the concept of a multi-granularity deviation factor (MDEF). Objects which deviate in their local object density more than three times of the standard deviation are regarded as outliers (cf. Section 2.2).

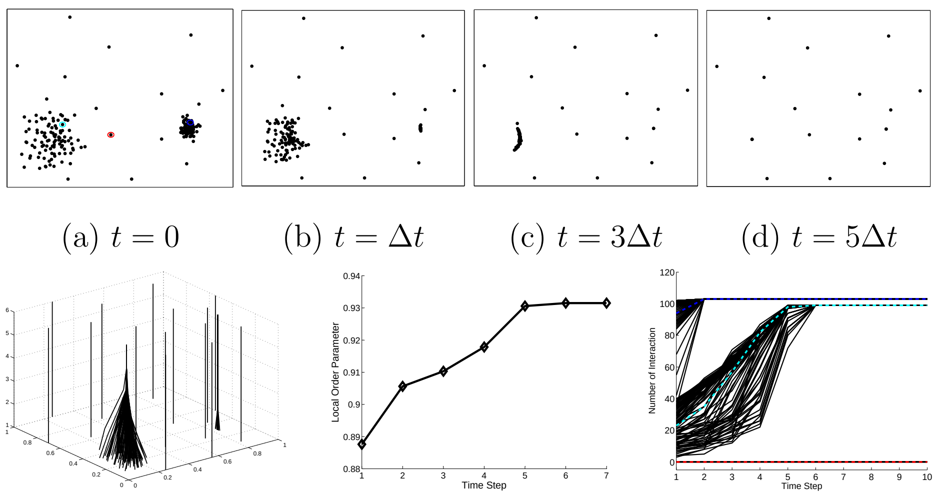
8.3 Synchronization-Based Outlier Detection

In this section, the algorithm SOD is introduced to detect outliers based on synchronization principle. The basic idea is first illustrated in Section 8.3.1. In Section 8.3.2 the algorithm SOD and its properties in detail are discussed.

8.3.1 Basic idea

The concept of synchronization provides a natural way to outlier detection. The basic idea is to flag outliers by distinguishing the object dynamical behaviors during the process towards synchronization. In this work, each data object is regarded as a phase oscillator and interacts dynamically with other objects according to the Extensive Kuramoto model (EKM), as introduced in Section 4.3.2. Finally, the local synchronization factor (LSF) is defined to distinguish outliers based on the different dynamics of objects during the process towards synchronization.

To gain some intuition into the synchronization-based outlier detection, consider a simple data set as illustrated in Figure 8.1. For an object within a cluster, such as $P1$, many similar objects are located around it and $P1$ starts interacting with these similar objects. Through non-linear dynamical interaction, the object changes its initial phase and moves towards the main direction of its interaction partners (Figure 8.1(a)). The situation is like the mutual influence of people in discussion. As time evolves, regular objects move gradually closer together through mutual interaction and thus more and more objects can interact with them. Figure 8.1(b) displays the new positions of the objects for comparison. The objects with similar attributes gradually synchronize together. The initial points (black color) are replaced by the red points after one time stamp. Then, in a sequential process, all these similar objects synchronize together, which finally have the same phase (Figure 8.1(c)). Outliers, such as $P2$, due to the significantly different attributes in comparison with regular objects, have difficulties to interact with other objects and tend to keep their own phases. Therefore, the dynamics of



(a) $t = 0$ (b) $t = \Delta t$ (c) $t = 3\Delta t$ (d) $t = 5\Delta t$

(e) Objects movement (f) Local Order Parameter (g) Interaction Plot

Figure 8.2: The dynamics of object toward to synchronization. (a)-(d) describe the detailed object movement during the time resolution. (e): Objects movement, (f): Local order parameter, (g). Interaction Plot for three objects circled in (a).

the objects show two different patterns during the process towards synchronization. For each regular object, as time evolves, it interacts with more and more objects and finally synchronizes with other objects. For outliers, there is none or only very minor interaction. Strong outliers keep their unique phase over the whole time during the process towards synchronization. The two different patterns can be easily visualized: Figure 8.1(d) shows the *Interaction Plot*, which displays for each object the number of interactions on the time scale.

8.3.2 The SOD Algorithm

In this section, we elaborate the *SOD* algorithm based on our extensive Kuramoto model.

First, without any interaction, all objects in a data set have their own phases. As time evolves, each object starts to interact with its ϵ-neighborhood, cf. Definition 4.1. The traces of all objects are in line with the main direction of their neighborhoods. Gradually, regular objects with similar attributes synchronize together following the intrinsic structure of a data set. In contrast, outliers are difficult to interact with other objects due to the large variance. Finally, the local regular objects with similar attributes synchronize together with same phase while outliers tend to keep their original phases. The objects synchronization process is terminated when the local order parameter converges.

To simply illustrate the objects dynamical movement, the Figure 8.2 (a)-(d) show the detailed dynamics of 2-dimensional points at time steps: $t = 0, \Delta t, 3\Delta t, 5\Delta t$. $t = 0$ indicates the original data set at the initial time. From that moment on, all objects with similar attributes start to synchronize together through the dynamical interaction according to Eq.(6.4) and finally, all objects in the data set synchronize at two different phases after 5 time steps. A more intuitive visualization of the objects movement is illustrated in Figure 8.2(e). Figure 8.2 (f) demonstrates the local order parameter of the data set with time evolution. During the process of synchronization, it is clear that regular objects and outliers behave different dynamics. Intuitively, for regular objects, since more and more objects interact with it and gradually group together, the pair-wise distances among their neighborhood become smaller and smaller. For outliers, they are isolated from other objects, the distances between objects to its neighborhood tend to much larger. Therefore, based on this observation, Local Synchronization Factor is proposed as follows, which is used to capture the degree of synchronization of each object during the process of synchronization.

Definition 8.1: (Local Synchronization Factor of an object x) The local synchronization factor $LSF(x)$ of object x is defined as:

$$LSF(x) = \frac{1}{T}\sum_{t=0}^{T}\left(\frac{1}{|Nb_{\epsilon}(x(t))|}\sum_{y(t)\in Nb_{\epsilon}(x(t))} cos(||y(t)-x(t)||)\right). \tag{8.1}$$

where T is the whole number of time steps for the process of synchronization. The synchronization factor captures the degree to which the object is an outlier. The smaller of the LSF value, the higher probability of being an outlier.

According to the definition, LSF shows major desirable properties:

1. *Intuitive.* Since the LSF value indicates each object synchronization factor, it provides an intuitive way to summarize its dynamical interaction behavior with other objects during the process towards synchronization. The easier an object synchronizes with other objects, the higher is its LSF value. Outliers are objects which are out of synchronization.

2. *Tight.* The range of LSF is restricted to [0 1). The lower bound of LSF value is 0, which means the object does not interact with any other object during the synchronization process. For cluster points which easily synchronize the LSF value is close to 1. The LSF value can thus

be easily interpreted as the degree of each object of being an outlier, e.g. Degree(p) $= 1 - LSF(p)$.

3. *Distinguishable.* Due to the different dynamics between regular objects and outliers during the synchronization process, the values of LSF are fairly distinguishable. For outliers, the LSF value is around 0 while the regular objects nearly to 1. It can easy to discern them.

In addition, in order to explore each object dynamically during the process towards synchronization, the *Interaction Plot* is defined to characterize the interaction behavior pattern between objects over time.

Definition 8.2: (*Interaction Plot*) For any object x, the plot of the number of objects involved in mutual interaction with x versus the time step, is called *Interaction Plot*.

As a valuable addition to LSF, the *Interaction Plot* provides a detailed visualization of the dynamic behavior of each object according to the Extensive Kuramoto model. With the same data set above, the interaction plot is illustrated in Figure 8.2(g). From this plot, two distinct interaction patterns become evident. For regular objects, during the synchronization process, more and more objects interact together over time. Outliers often fail to interact with other objects (maybe a few at the beginning). As time evolves, the number of interacting objects becomes constant. For example, two regular objects and one outlier are visualized with dashed color lines to illustrate the different interaction patterns in Figure 8.2 (a),(g).

Outliers Flagging.

After LSF is obtained for each object, all outliers usually exhibit a low value in comparison to the regular objects. The denser the local region of an object, the higher its value of LSF is. Therefore, selecting a suitable threshold for flagging outliers could be easily selected since the LSF value is distinct for outliers and regular objects. However, for automatically flagging, in this work, the K-Means algorithm is applied on the LSF values to split the data into two clusters: outliers and regular objects. Finally, the Pseudocode of the *SOD* is illustrated in Algorithm 4.

Runtime Complexity.

For SOD, to detect outliers based on synchronization, the runtime complexity with respect to the number of data objects is $O(T \cdot N^2)$, where N is the number of objects and T is the time evolution. In most cases, T is small with $5 \leq T \leq 20$. If there exists an efficient index, the complexity reduces to $O(T \cdot N \log N)$.

Parameter Setting.

In order to flag outliers based on synchronization principle, an interaction range (ϵ) needs to be specified for EKM. The question is: how to determine the ϵ value and how does the LSF value change when the ϵ value is adjusted?

Given a data set, theoretically, the ϵ value can be 0, which means there is no interaction at all among the objects. In order to generate object interaction, there should be a lower bound of ϵ. To generate a stable interaction for most objects, a heuristic way is to use the average value of the k-nearest neighbor distance determined by a sample from the data set for a small k.

Algorithm 4 $SOD(D, \epsilon)$

```
$LSF := \{\}$; // Synchronization Factors
while loopFlag=true do
  for each object $p \in D$ do
    Compute $N_\epsilon(p)$;
    Obtain new value of object $p$ using $Eq.$(4.5); // Update value
  end for
  Compute local order $r$ using $Eq.$(4.6);
  if $r$ converges then
    loopFlag=false;
    for each object $p \in D$ do
      Compute local synchronization factor $LSF(p)$ using $Eq.$(8.1);
    end for
  end if
end while

Flagging outliers by K-Means based on LSF values.
```

The ϵ should not be so large that encloses all data objects for interaction. In that way, the local object dynamical behaviors can be detected. However, the range of suitable values for ϵ is very large. In all experiments on different data sets, the ϵ is set as the average value of the k-nearest neighbor distances for k ranging from 10 to 50.

To asses the impact of ϵ value on LSF value and outlier detection, Figure 8.3 shows a simple data set which consists of 2 clusters with different size (C1:50, C2:200) and density. 17 outliers are added to the data. For each

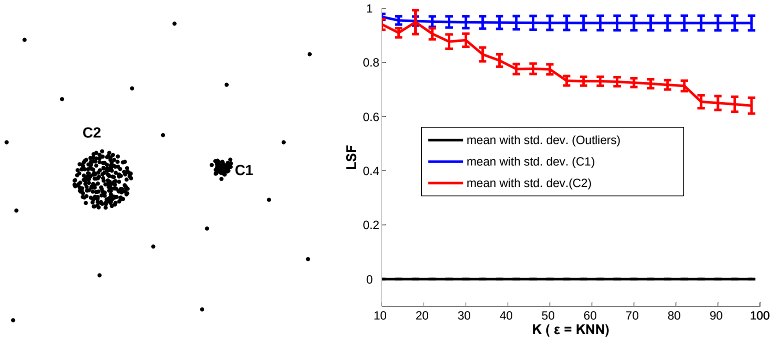
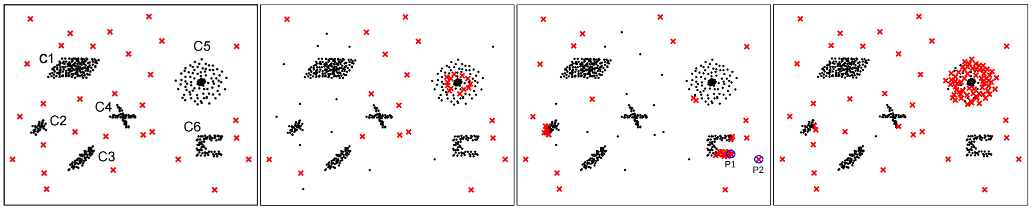


Figure 8.3: Influence of ϵ on LSF.

ϵ value, the mean and standard deviation of LSF values are calculated for each cluster as well as for the outliers. Figure 8.3 displays the LSF value with respect to ϵ. For all settings of ϵ, the mean LSF value for the points in cluster C1 and C2 are clearly much larger than 0, in most cases close to 1 while the mean and standard deviation of LSF value for outliers remain stable at 0. With the increase of ϵ, the LSF value of cluster objects begin to decrease. The reason behind it is that more and more objects are enclosed to interact with each other at each time step and thus more difficult to synchronize together. The situation is like two people finding it much easier to agree over some thing than a larger group of people. Moreover, since objects in denser cluster are easier to synchronize, the LSF values are much more closer to 1 (e.g. the mean LSF value of objects in C1 are larger than those in C2). For different parameters, outliers and regular objects show distinct LSF values and can be discerned easily. For further evaluation on the robustness of SOD w.r.t. parameter settings please refer to the experimental section.



(a) SOD (b) LOF (c) LOCI (d) CoCo

Figure 8.4: Outlier detection results with different methods. (a): SOD ($k = 10 - 40$), (b): LOF (MinPts = 20, selecting only the top 30 outliers), (c): LOCI (α= 1, $rmin$ = 10 and $rmax$ =50), (d): CoCo. Detected outliers are highlighted with red crosses.

8.4 Experimental Evaluation

In the following, the outlier detection *SOD* is evaluated in comparison to LOF [30], LOCI [120], CoCo [24] on synthetic data set as well as NBA data. SOD, CoCo, and LOF are implemented in Java and the implementation of LOCI and CoCo are obtained from the authors.

Synthetic Data.

The evaluation starts with 2-dimensional synthetic data set to facilitate presentation. The data set displayed in Figure 8.4 consists of six clusters (C1-C6): one Gaussian cluster (C2:39), two correlation clusters (C1:167 and C3:73), two arbitrarily shaped clusters (C4:60 and C6:67) and one a spherical hierarchical cluster (C5:131), including a nested cluster. 30 outliers are added to the data set. Figure 8.4 provides the results of outlier detection by using SOD, LOF, LOCI and CoCo for the same synthetic data set.

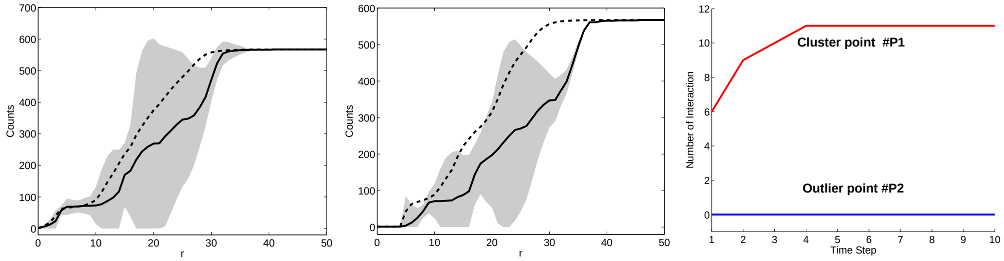


Figure 8.5: LOCI Plot of cluster point P1 (left) and outlier P2(middle); Interaction Plot of P1 and P2 (right).

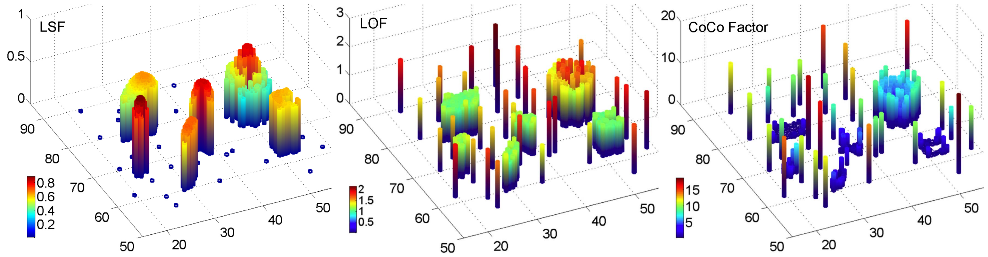


Figure 8.6: Visualization of the LSF, LOF and CoCo for the synthetic data set.

Without any prior knowledge, SOD successfully detects all 30 outlier points (Figure 8.4 (a)) from the complex data. The outliers are highlighted with red crosses and cluster objects are shown in black. Moreover, SOD obtains the same result with the parameter ϵ set to the average k-nearest neighbor distance for k ranging from 10 to 40.

For LOF, a wide range of different settings is tried for the parameters $MinPts$ from 10 to 50, which is suggested by authors. The top 30 outliers are obtained according to the LOF value. For different parameters, there are 17, 17, 9, 8 and 10 out of 30 correctly assigned with $MinPts$ =10,20,30,40,50, respectively. Most cluster objects, especially for the objects of hierarchical cluster are wrongly flagged as outliers. The best result is presented in Figure 8.4 (b) obtained with parameter $MinPts$ =20. Obviously, the result of LOF is very much influenced by the parameter $MinPts$. In addition, the priori information about the number of desired outliers is often unknown.

LOCI is applied to the synthetic data set with $\alpha = 1$, $rmin = 10$ and $rmax$ =50 (Figure 8.4 (c)). 46 outlier points are detected based on the suggested outlier flagging criteria and 16 true outliers are correctly detected. Many cluster points of C2, C4 and C6 are wrongly flagged. With the decrease of $rmin$ value, more true outlier points are found, but at the same time more cluster points are mislabeled as outliers. As the LOCI plot provides the information for each object, we thus have a closer look at the cluster point (Fig. 8.4 (c): $P1$) and outlier point ($P2$). It is noticeably that the two LOCI plots look very similar (Figure 8.5(a)-(b)). Using this plot, it is difficult to distinguish the cluster point $P1$ from outlier point $P2$. For comparison, *Interaction Plot* is displayed for the same two points (Figure 8.5(c)). It is very clear that the cluster point $P1$ and outlier point $P2$ have totally different interaction patterns, where the number of interactions for cluster point $P1$ gradually increase and finally has the same size while there is no any objects interact with it during the process of synchronization.

CoCo identifies all 95 outliers for the synthetic data and only one outlier is missing. However, 66 cluster points are wrongly flagged as outliers. CoCo relies on the assumption that the data follows an Exponential Power Distribution, which is not the case for our example and therefore CoCo yields many false-positives.

To visualize the degree of "outlierness" for each object, the Local Synchronization Factor (LSF) is further compared to the outlier factor of LOF and CoCo in Figure 8.6. For LSF, the local synchronization factor of all outlier points are nearly 0 and all cluster points are close to 1. Due to the desirable properties of LSF, such as tightness and distinguishable, cluster points and outliers are easily to differentiate. As to LOF and CoCo, the range of values is very wide and the gap between outliers and cluster points is not clear. For example, the range of LOF is from 0.85 to 2.15, which makes it difficult to determine a suitable threshold for outlier flagging.

NBA Performance Statistics.

After extensive evaluation of SOD on synthetic data set, the novel outlier detection method is further evaluated on the real data. The NBA data is used, which is available at the NBA website `http://www.nba.com`. In the Season 2008/09, the performance of 444 players are described with four attributes: the number of games played (GP), the number of points (PPG), the

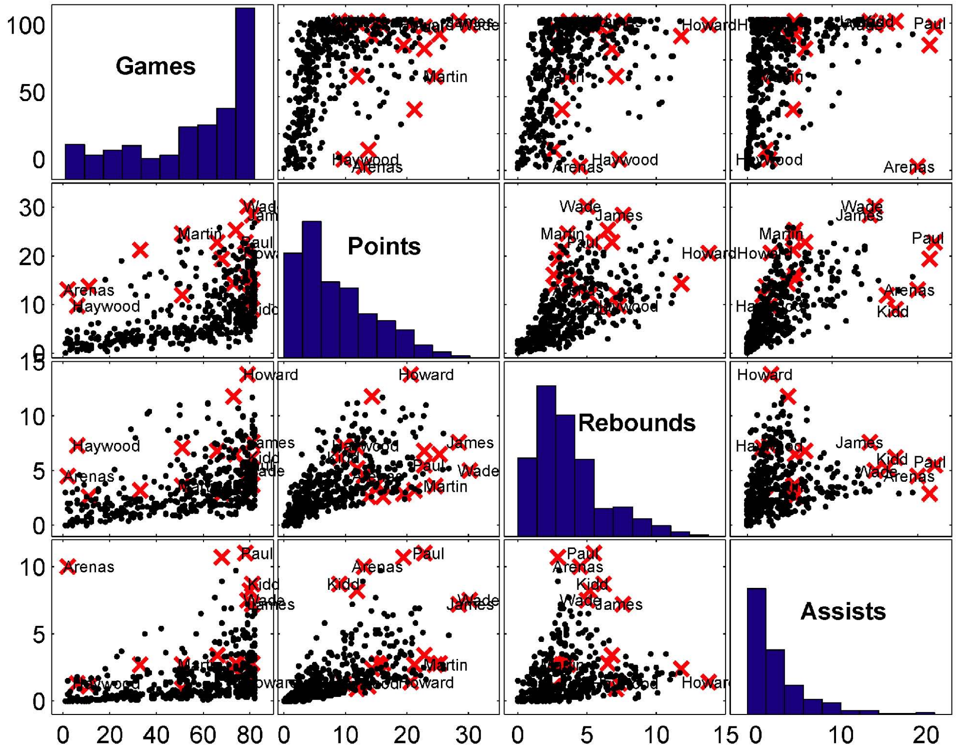


Figure 8.7: Outlier detection with SOD for the NBA data set. All 18 outliers are marked in red. The strongest outliers with LSF = 0 are also marked with the name.

rebounds (RPG), and assists (APG) per game. SOD is applied to this NBA data detecting 18 outliers with $k = 30$. Figure 8.7 displays scatter plots of the data with different attributes and the histogram of each attribute in the diagonal respectively. Obviously, the data distribution is non-Gaussian. All 18 outliers are highlighted in red. For the most outstanding players with LSF = 0 also the names are provided. For comparison with other techniques, Table 8.1 lists the top 10 outliers for various settings of k. For different parametrization (k=30,40,50) the same players are among the top 10 outliers of SOD. Eight of 10 players are strongest outliers with a LSF of 0 for all parameterizations. For comparison, Table 8.2 lists the top 10 outliers identified by LOF with MinPts =50. Only seven players are reproducibly detected as top 10 for MinPts =40. LOCI ($\alpha = 1$, $rmin = 10$ and $rmax$ =50) and CoCo detect the top 10 outliers listed in Table 8.3 and Table 8.4, respectively. For the comparison methods, the intersection with SOD is marked in bold. Gilbert Arenas with LSF = 0 is a strong outlier who is among the top 10 for all methods. Having played only 2 games, he has shown outstanding performance in terms of rebounds, points and especially in terms of assists. Most other players with an LSF of zero are also among the top 10 of at least one of the comparison methods, e. g. the well-known and truly outstanding players Brendan Haywood, Dwyane Wade and LeBron James. Interestingly, Chris Paul is not among the top 10 of any comparison methods although he is especially outstanding in the number of points and assists per game. In the 2006-07 season he has been ranked fourth in the overall NBA in assists (`http://www.nba.com/playerfile/chris_paul/bio.html`). Another strong outlier missed by the comparison methods is Dwight Howard who has been named the NBA Defensive Player of the Year in 2008-2009 season (`http://www.nba.com/playerfile/dwight_howard/bio.html`). Dwight Howard is especially characterized by an outstanding number of 13.8 rebounds per game.

LSF(k=30)	(k=40)	(k=50)	Name	GP	PPG	RPG	APG
0	0	0	Chris Paul (NOH)	78	22.8	5.5	11
0	0	0	Jason Kidd (DAL)	81	9	6.2	8.7
0	0	0	Dwyane Wade (MIA)	79	30.2	5	7.5
0	0	0	LeBron James (CLE)	81	28.4	7.6	7.2
0	0	0	Kevin Martin (SAC)	51	24.6	3.6	2.7
0	0	0	Dwight Howard (ORL)	79	20.6	13.8	1.4
0	0	0	Gilbert Arenas (WAS)	2	13	4.5	10
0	0	0	Brendan Haywood (WAS)	6	9.7	7.3	1.3
0	0.036	0	Cuttino Mobley (LAC)	11	13.7	2.6	1.1
0	0.260	0.035	Deron Williams (UTA)	68	19.4	2.9	10.7

Table 8.1: Top 10 outliers identified with SOD on NBA data.

LOF	Name	GP	PPG	RPG	APG
1.5058	**Gilbert Arenas (WAS)***	2	13	4.5	10
1.4938	**Brendan Haywood (WAS)***	6	9.7	7.3	1.3
1.4463	DJ White (OKC)*	7	8.9	4.6	0.9
1.4069	Michael Redd (MIL)*	33	21.2	3.2	2.7
1.4011	**Cuttino Mobley (LAC)**	11	13.7	2.6	1.1
1.3623	Carlos Boozer (UTA)*	37	16.2	10.4	2.1
1.3532	Monta Ellis (GSW)*	25	19	4.3	3.7
1.3492	Elton Brand (PHI)*	29	13.8	8.8	1.3
1.3365	Chris Kaman (LAC)	31	12	8	1.5
1.3294	Tracy McGrady (HOU)	35	15.6	4.4	5

Table 8.2: Top 10 outliers identified with LOF on NBA data.

Name	GP	PPG	RPG	APG
Dwyane Wade (MIA)	79	30.2	5	7.5
LeBron James (CLE)	81	28.4	7.6	7.2
Ben Wallace (CLE)	56	2.9	6.5	0.8
Monta Ellis (GSW)	25	19	4.3	3.7
Pops Mensah-Bonsu (TOR-SAS)	22	5	5.1	0.3
Corey Brewer (MIN)	15	6.2	3.3	1.7
Gilbert Arenas (WAS)	2	13	4.5	10
Kobe Bryant (LAL)	82	26.8	5.2	4.9
Jason Kidd (DAL)	81	9	6.2	8.7
Michael Redd (MIL)	33	21.2	3.2	2.7

Table 8.3: Top 10 outliers identified with LOCI on NBA data.

CoCo o.f.	Name	GP	PPG	RPG	APG
20.76	Michael Redd (MIL)	33	21.2	3.2	2.7
17.39	Andrew Bogut (MIL)	36	11.7	10.2	2
17.31	**Kevin Martin (SAC)**	51	24.6	3.6	2.7
16.55	Monta Ellis (GSW)	25	19	4.3	3.7
14.42	Elton Brand (PHI)	29	13.8	8.8	1.3
13.84	**Gilbert Arenas (WAS)**	2	13	4.5	10
13.80	Tracy McGrady (HOU)	35	15.6	4.4	5
12.12	Kobe Bryant (LAL)	57	17.5	3	5
11.85	Cuttino Mobley (LAC)	11	13.7	2.6	1.1
11.45	Tyson Chandler (NOH)	45	8.8	8.7	0.5

Table 8.4: Top 10 outliers identified with CoCo on NBA data.

8.5 Conclusions

In this chapter, the algorithm SOD is proposed for outlier detection based on the different interaction patterns of objects during the process towards synchronization. The major benefits of SOD can be summarized as follows:

1. *Natural outlier detection.* Based on the synchronization principle, outliers are naturally flagged from a data set due to their unique dynamical interaction pattern during the process towards synchronization.

2. *Intuitive to interpret.* Outliers are represented as those objects which hardly interact with other objects since they have been generated by a different mechanism. Outlier objects are the members of the system which are out of synchronization. The probability for each object of being an outlier can be characterized by a numerical value: The Local Synchronization Factor (LSF). The *Interaction Plot* provides a detailed view of the dynamical behavior pattern of each object.

3. *Without any data distribution assumption.* Without any data distribution assumption or any prior knowledge, outliers are easily flagged by investigating the different interact patterns, which are driven by the intrinsic data structure.

4. *Complex data handling.* The SOD allows detecting outliers from a complex data set including clusters with arbitrary number, shape, size and density as well as hierarchical data structures.

5. *Robustness to parametrization.* The outlier detection result is insensitive to parameter settings.

Part III

Brain Network Mining

Chapter 9

Background on Diffusion Tensor Imaging and Analysis

Diffusion Tensor imaging (DTI) is unique in its ability to explore the organization and integrity of human white matter tracts in vivo, using water diffusion properties as a probe [107]. It measures for every voxel the diffusivity of water molecules within the tissue, and thus gives valuable insight into the orientation of fiber tracts. Potential pathways of fiber tracts in the human brain can be reconstructed by deterministic or stochastic tractography based on the measured diffusion weighted magnetic resonance imaging (DW-MRI). Hence DTI modularity provides a promising way to study the structure, development, and diseases of brain. This chapter gives the background of DTI which also will be used in Chapter 10 and Chapter 11. Furthermore, a basic description of the diffusion process and the approaches used to measure the image diffusion with MRI is provided and the white matter tractography is briefly reviewed.

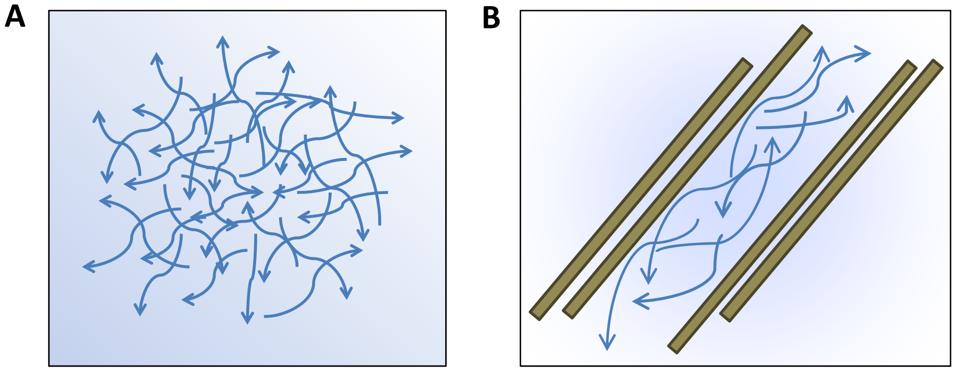


Figure 9.1: Water molecules diffusion in free medium and human brain. A: The random displacements of water molecules resulting from thermal agitation (Brownian motion). B: The diffusivity of water molecules in the human brain, which is hindered by obstacles, e.g. myelin, the axonal membrane and microtubule.

9.1 Water Diffusion in White Matter

The concept of diffusion refers to the random motion of molecules (also called Brownian motion) that results from their thermal energy [20]. This motion can be described as a random walk and the displacement of molecules with diffusion time can be well characterized by Einstein [53].

$$r = \sqrt{6Dt} \tag{9.1}$$

where r is the displacement, D is the diffusion constant and t is the diffusion time. Since diffusion is driven by the thermal energy of the molecules, the diffusion coefficient D depends only on the size of the molecules, the temperature and the nature of the medium [20].

In an isotropic medium, diffusivity is independent of the direction (See

Figure 9.1 A). However, what makes diffusion interesting is that the diffusivity of water measured in the brain highly relies on the direction. In the human brain, water molecules do not exist in such an isotropic environment. Instead, their diffusion is restricted due to the presence of structures such as myelin, the axonal membrane, microtubule (see Figure 9.1 B). The diffusivity perpendicular to the axon is thus smaller than that parallel to the axon. Thus, the non-invasive observation of the water molecular diffusion-driven movement provides important information of the microstructure of tissues.

9.2 Diffusion Weighted MRI

Diffusion Weighted MRI is deeply rooted in the concept of water anisotropic diffusion in whiter matter [20]. Briefly, a pair of dephasing and rephasing gradients is applied to investigate the molecular diffusion. With the different gradients, the MR signal intensity sensitized to the movement of molecules will result in the signal loss. This reduction in signal due to the amount of diffusion can be governed by the Stejskal-Tanner equation [15].

$$S = S_0 e^{-bD_g} \tag{9.2}$$

where S is the measured signal, S_0 is the signal in absence of a diffusion-sensitizing gradient. D_g is the diffusion constant in gradient g. The b-value is a measure of the diffusion weighting and depends on the magnitude and duration of the gradients. For the commonly used trapezoidal gradients, the b-value can be calculated as [168]:

$$b = \gamma^2 \delta^2 (\Delta - \frac{\delta}{3}) |g|^2 \tag{9.3}$$

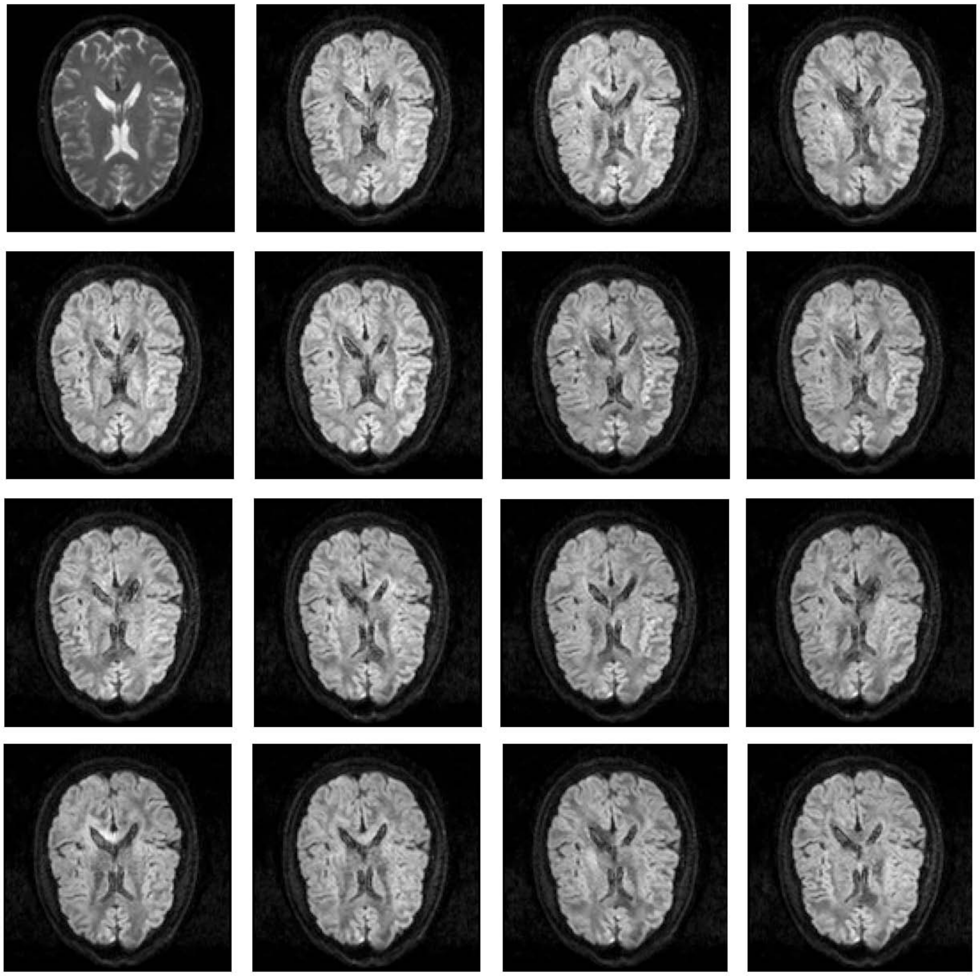


Figure 9.2: Sixteen diffusion-weighted images. The first is the B0 image and the remaining are diffusion-weighted images with 15 different magnetic field gradients which are employed to sensitize the image to diffusion in a particular direction. The direction is different for each image, resulting in a different pattern of signal loss (dark areas) due to anisotropic diffusion.

where γ is the gyro-magnetic ratio for the hydrogen nucleus (42 MHz/Tesla), δ is the duration of the diffusion weighting gradient, g is the magnitude of the diffusion sensitizing gradient, and Δ is the time between the gradient pulses.

For brain Diffusion Weighted MRI, the b-value usually ranges from 500 to1500 s/mm^2. By applying multiple magnetic field gradients, the diffusion weighted images can be obtained. Figure 9.2 displays an example of diffusion-weighted images, where the first is the B0 image without a diffusion-sensitizing gradient and the remaining are diffusion-weighted images with 15 different magnetic field gradients.

9.3 Diffusion Tensor MRI

In brain white matter, the diffusivity of water molecules is highly dependent on the environment (c.f. Section 9.1). To infer the microscopic nature of tissue from diffusion, a diffusion model is required to account for this anisotropic behavior. Currently, the most well-known model is diffusion tensor model [107], which represents the diffusion as a 3 × 3 symmetric tensor:

$$\overline{\overline{\mathbf{D}}} = \begin{pmatrix} D_{xx} & D_{xy} & D_{xz} \\ D_{yx} & D_{yy} & D_{yz} \\ D_{zx} & D_{zy} & D_{zz} \end{pmatrix} \tag{9.4}$$

To calculate the diffusion tensor, at least six gradient images and one non-diffused image are required. However, in practice, more than six gradient directions are measured to better fit the model. Since the calculated tensor

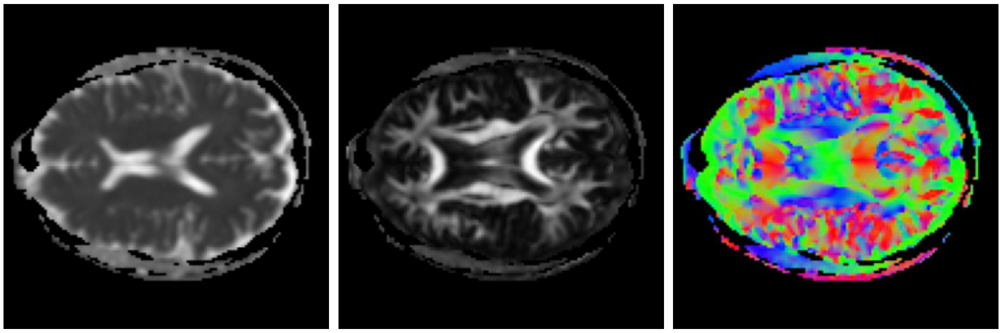


Figure 9.3: Diffusion Tensor Measures.

D is a positive definite, it can be further diagonalized as:

$$\overline{\overline{\mathbf{D}}} = E \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} E^{-1} \tag{9.5}$$

where λ_1, λ_2, and λ_3 are the eigenvalues of the tensor and E is the matrix whose columns are the corresponding eigenvectors, e_1, e_2, and e_3. Once the diffusion tensor has been calculated and decomposed, a range of quantities can be easily derived as follows.

9.3.1 Mean Diffusivity

Mean diffusivity (MD) is a measure of the average diffusivity of the water in a voxel.

$$Trace(\overline{\overline{\mathbf{D}}}) = D_{xx} + D_{yy} + D_{zz} \tag{9.6}$$

The mean diffusivity is defined as $MD = Trace(D)/3$ and can be rewritten as:

$$MD = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3} \tag{9.7}$$

9.3.2 Anisotropy Measures

Currently, the most widely used anisotropy measures are fractional anisotropy (FA), relative anisotropy (RA), and volume ratio (VR), which are defined as [17]:

$$FA = \sqrt{\frac{1}{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}}{\sqrt{(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}} \tag{9.8}$$

$$RA = \sqrt{\frac{1}{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}}{(\lambda_1 + \lambda_2 + \lambda_3)} \tag{9.9}$$

$$VR = \frac{\lambda_1 \lambda_2 \lambda_3}{\big((\lambda_1 + \lambda_2 + \lambda_3)/3\big)^3} \tag{9.10}$$

Recently, many experiments indicate that various diseases highly related to white matter atrophy, such as alzheimer's disease [42] or schizophrenia [121], tend to decrease the diffusion anisotropy. Therefore, the three anisotropy measures, especially FA, are usually used in clinical studies to diagnose diseases and assess the progress of the therapy.

9.3.3 Geometrical Anisotropy Measures

Assuming that the three eigenvalues of the tensor are $\lambda_1 \geq \lambda_2 \geq \lambda_3$, Westin et al. [168], [169] propose three intuitive measures C_l, C_p, and C_s to describe

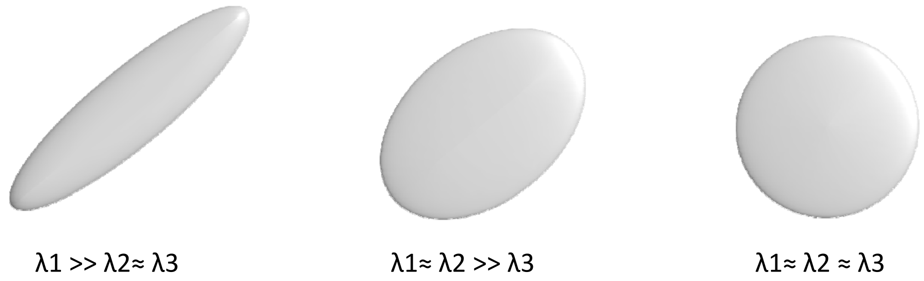


Figure 9.4: Ellipsoidal representation of a tensor with linear, planar, or spherical symmetry.

whether the shape of diffusion is like a cigar (linear), pancake (planar), or sphere (spherical).

$$C_l = \frac{\lambda_1 - \lambda_2}{\sqrt{(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}} \tag{9.11}$$

$$C_p = \frac{2(\lambda_2 - \lambda_3)}{\sqrt{(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}} \tag{9.12}$$

$$C_s = \frac{3\lambda_3}{\sqrt{(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}} \tag{9.13}$$

9.4 White Matter Tractography

As DTI characterizes underlying white matter structure by measuring the diffusivity anisotropy of water with a diffusion tensor model, the orientation of dominant axonal tracts thus can be reconstructed based on the diagonalized eigenvectors and eigenvalues. To date, various white matter tractography algorithms have been proposed, which can be generally classified as deterministic tractography and probabilistic tractography. The deterministic trac-

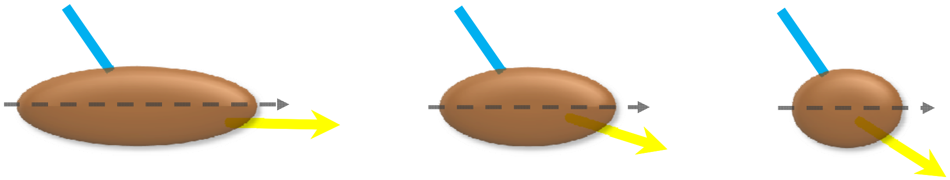
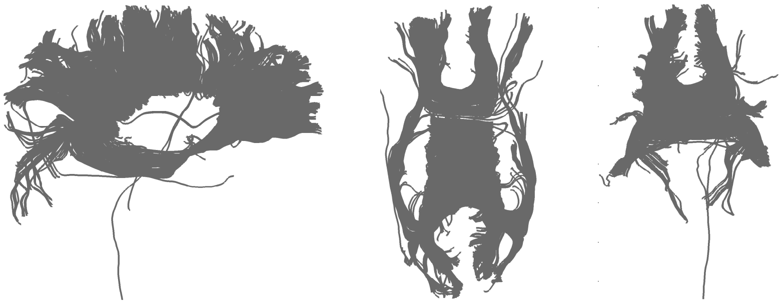


Figure 9.5: Illustration of tensor deflection (TEND) algorithm for different tensor shapes. For more anisotropic tensor (left) the incoming vector is deflected toward the major eigenvector. The amount of deflection decreases with decreasing anisotropy (from left to right) [96].

tography refers to a class of streamline propagation techniques that use local tensor information for each step of the propagation or global optimization techniques to estimate of the true fiber pathway using energy minimization algorithms. The main difference of probabilistic tractography is to consider the uncertainty of the connections among regions. This group of approaches usually provides connection probability maps derived using such frequency of connection methods.

9.4.1 Deterministic Tractography

Currently, the most intuitive and simplest way to reconstruct the fiber tracts is based on streamline propagation techniques [108, 16, 96]. The basic principle of these approaches is to associate the major eigenvector with the tangent to a fiber tract. Specifically, fiber tracking starts from any seed point with high FA value since these voxels are more likely to contain a high proportion of white matter. It propagates fiber paths by simply following the measured eigenvectors, using the major eigenvector of tensors [108, 16] or the entire



(a) Sagittal (b) Axial (c) Coronal

Figure 9.6: White matter tractography in corpus callosum with TEND algorithm.

diffusion tensor to deflect the estimated fiber trajectory in both directions [96]. Tracking stopped in voxels with low FA value or physiologically implausible curvature of the track. For illustration, Figure 9.6 gives an example of fiber tracking on Corpus Callosum with TEND algorithm. The main problem of streamline propagation techniques for white matter tractography is that the orientations of fiber tracts are estimated locally. Thus errors can be accumulated and the tracing can be confounded by regions of crossing fibers.

To deal with such a problem, another type of the deterministic tractography approaches is proposed based on global optimization techniques. Starting from a seed point, they attempt to explore the true fiber pathway using energy minimization techniques. For instance, Parker et al. [123] propose the fast marching technique to find the optimal connection paths. Similarly, Gossl et al. [65] track globally optimal paths relying on the smoothness criterion.

The common feature of the deterministic tractography approaches is that fiber orientations are determined in a deterministic way. However, this type of approach cannot handle the multiple fibers which cross or branch. One solution is to measure the local uncertainty of fiber orientation, which is called probabilistic tractography.

9.4.2 Probabilistic Tractography

Instead of using the major eigenvector, the full tensor or global optimization techniques, these probabilistic methods place a probability model on the fiber orientation at each voxel. Rather than producing one path from each seed point, a distribution of paths is produced by sampling. First, the uncertainty of the fiber orientation at each voxel is modeled as a probability density function (PDF) [32, 122, 61, 18]. After that, probabilistic tracking algorithms perform the streamline propagation procedure with different propagation directions relying on the probability density function. The main difference among existing methods is the diverse ways to model the uncertainty of fiber orientations. The drawback for probabilistic tracking methods is time consuming, which is not appropriate in practice.

Therefore, although many methods have been proposed, the most popular way in practice also relies on some basic methods such as the FACT [108], Euler [16] or TEND [96] methods.

9.5 Conclusion

In this chapter, the basic background of DTI is provided, including the principle of the water diffusion, diffusion tensor model and anisotropy measures. Finally, the methods of white matter tractography are brief reviewed. In next chapters, the data mining techniques will be applied in diffusion tensor imaging to explore and analyze the brain structural networks.

Chapter 10

Combining Time Series Similarity with Density-based Clustering to Identify Fiber Bundles in the Human Brain

It is well known that understanding the connectome of the human brain is a major challenge in neuroscience. Discovering the wiring and the major cables of the brain is essential for a better understanding of brain function. Diffusion Tensor imaging (DTI) provides the potential way of exploring the organization of white matter fiber tracts in human subjects in a non-invasive way, as introduced in Chapter 9. However, it is a long way from the approximately one million voxels of a raw DT image to utilizable knowledge. After preprocessing including registration and motion correction, fiber tracking approaches extract thousands of fibers from diffusion weighted images.

In this chapter, the question that how to identify meaningful groups of fiber tracks which represent the major cables of the brain is addressed. The ideas from time series mining are combined with density-based clustering to a novel framework for effective and efficient fiber clustering. First a novel fiber similarity measure based on dynamic time warping is introduced. This fiber warping measure successfully captures local similarity among fibers belonging to a common bundle but having different start and end points. A lower bound on this fiber warping measure speeds up computation. The result of fiber tracking often contains imperfect fibers and outliers. Therefore, fiber warping is combined with an outlier-robust density-based clustering algorithm.

The remainder of this chapter is organized as follows: An introduction is presented in the following Section 10.1. Section 10.2 gives a brief summary of the related work of fiber similarity and clustering. In Section 10.3, the fiber measure based on dynamic time warping is proposed. Afterwards, in Section 10.4 the framework of the fiber density-based clustering is explained in detail. Section 10.5 contains an extensive experimental evaluation on synthetic and real data sets. Finally, Section 10.6 gives a summary and concludes this chapter. Parts of the material presented in this chapter have been published in [145], where J.S., K.H, C.B and C.P designed the research; J.S. implemented algorithms; J.S. and K.H. performed the experiments; J.S., K.H., and Q.Y. analyzed the data; and J.S., K.H, Q.Y., C.B., A.M, N.M and C.P. wrote the paper.

> "*Junming Shao, Klaus Hahn, Qinli Yang, Christian Böhm, Afra M. Wohlschläger, Nicholas Myers, Claudia Plant. Combining*

Time Series Similarity with Density-Based Clustering to Identify Fiber Bundles in the Human Brain. ICDM Workshops 2010: 747-754."

10.1 Introduction

As introduced in Chapter 9, diffusion tensor imaging (DTI) provides a promising way to explore organization and integrity of white matter tracts in vivo, using water diffusion properties as a probe [107]. Through fiber tracking in the human brain, thousands of tracks can be obtained. The remaining problem in clinical research is how to classify this vast amount of fiber trajectories. To date, the most frequently used method is to select fibers based on expert knowledge, which is often referred to as *virtual dissection*. Experts first specify some regions of interest (ROIs) based on domain knowledge and then select all fibers that pass through these pre-defined ROIs [36]. This process tends to be inefficient especially since it is time consuming and limited by expert resources. Moreover, manual specification of ROIs in different patients may be biased. Therefore, a more promising approach is to cluster fiber tracks into fiber bundles which provide an overview of the structural organization of the fiber pathways. Grouping this large amount of data into more manageable clusters of tracks would be extremely useful for further applications, such as investigating connectivity in mental and neurological disorders. Another example is tract-based analysis. In comparison with traditional methods, such as ROI-based analysis or voxel-based morphometry (VBM), tract-based analysis computes the quantitative measures of interest along the white matter tracts. To accomplish this goal, algorithms are required to segment the trajectories into bundles. A further evident example is surgical planning, where it is important to consider the uncertainty associated with the estimated fiber bundles. Automatically clustering these fibers would not be biased in patients with brain lesions or distorted anatomy. Therefore, the development of effective and efficient algorithms for white matter fiber tract segmentation in diffusion MRI is of significant interest for neuroscience community.

Though several algorithms have been proposed for clustering trajectories into meaningful bundles, such as k-nearest neighbors [49], spectral clustering [35, 118] or hierarchical clustering [177], the following two problems are particularly vital but remain essentially unsolved:

1. Fiber Similarity Measure: An effective and efficient similarity measure for pair-wise fiber comparison is desired.

2. Outlier-robustness: Due to experimental limitations, like thermal noise or partial volume effects, the set of fibers produced by tractography contains also imperfect fibers or outliers. These fibers should not be clustered into any fiber bundle and need to be excluded.

In view of these issues, a novel fiber similarity measure based on adapted dynamic time warping is proposed to calculate the pair-wise similarity distance between fibers. Furthermore, a lower bounding technique for the similarity measure is proposed to save computation cost. Considering the noise in real data, density-based clustering is explored to group these fiber tracts. Imperfect fibers or outliers are eliminated during this clustering process.

10.2 Related Work

Due to the wide application of diffusion tensor imaging modality, white matter tract clustering has gained significant attention. In order to perform clustering, first a fiber similarity measure must be specified , which is then used as input to a clustering algorithm. Therefore, in this section, a brief survey on fiber similarity measures is first given and then the recently proposed fiber clustering algorithms are reviewed.

10.2.1 Fiber Similarity Measure

A fiber similarity measure is a function that computes the (dis)similarity between pairs of fibers. Two fibers are considered similar when they have comparable length, similar shape, and are separated by a small distance [49]. The early work by Brun et al. [35] assumes that two fiber tracts with similar end points should be considered as similar. Thus they define fibers as similar if the fibers start and end in the same area. Euclidean distance between the end points of fiber traces is then used to calculate the fiber similarity. However, the assumption is not reasonable in many cases since not all fiber bundles start and end in the same regions. It also ignores the information of most other points and the fiber pair-wise shape similarity. Ding et al. [49] propose a similarity measure by cutting each fiber into corresponding fiber segments and use the mean Euclidean distance between the segments to define piece-wise similarity. This similarity method is efficient but not effective since this measure also loses the point-by-point information. Several authors acknowledge that point-by-point correspondence of the trajectories should be used for accurate clustering and quantitative analysis. Zhang et al. [177] define the distance between two fibers as the average distance from any point on the shorter fiber to the closest point on the longer fiber, and only distances above a certain threshold contribute to this average. This similarity metric is thus not symmetrical and is affected by outliers or noise fibers. Corouge et al. [41, 63] form point pairs by mapping each point of one fiber to the closest point on the other fiber. The resulting point pairs are then used to define the distance between fiber pairs. They define three similarity distances: closest point distance, mean of closest point distance (MCP) and Hausdorff distance (HDD).

Currently, MCP [41, 106, 118] and HDD [41, 106, 63] are widely used fiber similarity measures. The formal definition of the two fiber similarity measures are as follows.

Suppose there are two fibers P and Q, MCP is defined as:

$$d_{MCP}(P,Q) = avg(d_m(P,Q), d_m(Q,P)). \tag{10.1}$$

with $d_m(P,Q) = avg_{p_i \in P}\{min_{q_j \in Q}||p_i - q_j||\}$.

While HDD is defined as:

$$d_{HDD}(P,Q) = max(d_h(P,Q), d_h(Q,P)) \tag{10.2}$$

with $d_h(P,Q) = \max_{p_i \in P}\{\min_{q_j \in Q}||p_i - q_j||\}$

The two measures are currently widely used for fiber similarity comparison. However, they have difficulty to capture the fiber shape characteristics effectively, such as local shift or distortion. In order to overcome these problems, a novel fiber similarity measure based on dynamic time warping (DTW) is proposed to better represent the fiber similarity.

10.2.2 Fiber Clustering

Instead of grouping fibers by hand with expert knowledge, fiber clustering approaches take advantage of the similarity of the fiber paths to cluster these fibers by algorithms. Fiber clustering methods analyze a collection of white matter tracts in 3D and separate them into meaningful bundles, or clusters, that contain paths with similar shape and spatial position. These bundles are expected to contain fiber paths with similar anatomy and function. To date, several fiber clustering methods have been proposed in the literature. Corouge et al. [41] use a k-nearest neighbors method to calculate the similarity metric between paired fiber tracts defined in terms of the length ratio and the Euclidean distance. It propagates cluster labels from a fiber to a neighboring fiber, which assigns each unlabeled fiber to the cluster of its closest neighbor if the closest neighbor is below a threshold. A partition of the data with a specific number of clusters can be acquired by setting a threshold on the maximal accepted distance. This is similar to the algorithm employed by Ding et al. [49], which establishes a corresponding segment to define the fiber similarity and then use $k-$nearest neighbors method to obtain fiber bundles. The k nearest-neighbors related method is very sensitive to the value of K and hard to obtain the proper threshold without prior knowledge. Another popular fiber clustering method is spectral clustering [35, 100], which refers to a class of techniques which rely on the eigenstructure of a similarity matrix. This type of method first uses a spectral embedding technique to map the fibers to a new feature space and then uses traditional clustering method, such as K-Means to group the fibers in this new space. Brun et al. [35] use a spectral embedding technique called Laplacian eigenmaps to map the fibers to a Euclidean feature space and then use a Gaussian kernel to compare the fibers in this new space. Donnell et al. [100] decompose the fiber similarity matrix into the eigenvalues and eigenvectors. The top eigenvectors are used to represent each fiber and then, using a specific clustering called k-way normal cut, they obtain fiber bundles. Spectral clustering can detect arbitrarily shaped clusters but needs much memory and thus is not convenient to cluster large fiber sets. Furthermore, spectral clustering is sensitive to outliers and the user should specify the number of clusters, which is hard to know in advance. Zhang and Laidlaw [177] use a hierarchical clustering algorithm for fiber clustering. The Hierarchical clustering algorithms decompose a data set into several levels of partitions, represented by a dendrogram. One of the most well-known hierarchical clustering approaches is Single-Link. Starting with the clustering obtained by placing every object in a unique cluster, in every step two closest clusters are merged until all objects are in a whole cluster. The main drawback of hierarchical clustering is that it is difficult to find the best partition of the dendrogram. In this chapter, the fiber clustering problem is considered from a density-based point of view, where fiber bundles are regarded as areas of high fiber density which are separated by areas of lower fiber density.

10.3 Fiber Warping

After deterministic tractography, a fiber is represented as an ordered set of points in space. The steps of the arc length, defined by two successive points of a fiber, are not necessarily identical, if e.g. numerical Runge Kutta methods with dynamical stepsizes are used for tracking. In addition, two fibers may have different lengths and consequently different numbers of points in space. To quantify similarity between two fibers, we adapt for this work the Dynamic Time Warping (DTW) method [81, 136, 139]. DTW is introduced for the comparison of time series which are out of phase and is applied in fields as diverse as speech recognition [136], bioinformatics [1] or data mining [88].

In the following section, DTW for time series is first briefly reviewed and then extended to a similarity measure for space curves. Finally, a convenient lower bounding technique is presented to save computational cost.

10.3.1 Dynamic Time Warping

Dynamic time warping (DTW) is a technique that looks for the optimal alignment of two time series. To achieve this goal, the time series are "warped" together non-linearly, by stretching or shrinking them along the time axes [137].

Suppose we have two time series X and Y, of lengths m and n respectively, where

$$X = (x_1, x_2, ..., x_i, ..., x_m) \tag{10.3}$$

$$Y = (y_1, y_2, ..., y_j, ..., y_n) \tag{10.4}$$

The objective is to optimize a warping path W:

$$W = (w_1, w_2, ..., w_k, ..., w_K) \tag{10.5}$$

where K is the length of W, with $max(m, n) < K < m + n - 1$. The k^{th} element of W is a pair of indices indicating a connection of time points in X and Y and is written as $w_k = (i, j)$, see Figure 10.1. A warping path follows the constraints [87] :

1. Boundary conditions: $w_1 = (1, 1)$ and $w_K = (m, n)$. This requires the warping path to start and finish in the first and last points of the series respectively;

2. Monotony: Given $w_k = (i, j)$, then $w_{k+1} = (i', j')$, with $i' - i >= 0$ and $j' - j >= 0$. This forces the points in W to be monotonically spaced in time.

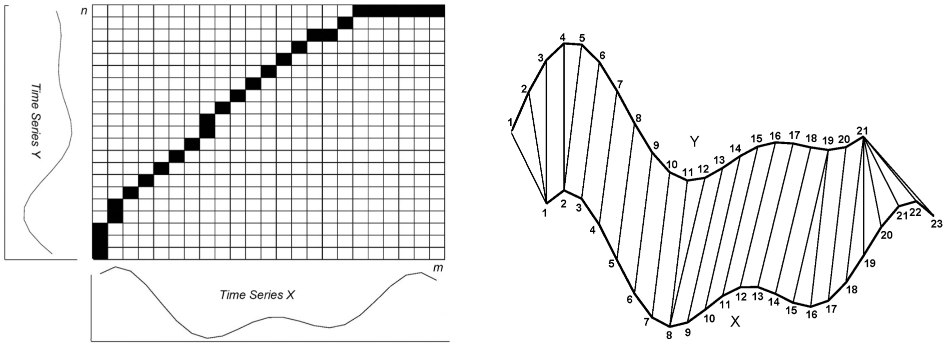
3. Continuity: Given $w_k = (i, j)$, then $w_{k+1} = (i', j')$, with $i' - i <= 1$ and $j' - j <= 1$. This restricts the admissible steps in the warping path to adjacent points of the series.

There are many warping paths satisfying the above conditions. In order to find a best match between two time series, the focus is looking for the path which minimizes the cumulative distance between them. The distance dtw for this optimum path is defined as:

$$dtw(X, Y) = \min(\sum_{k=1}^{K} d(w_k)) \tag{10.6}$$

where $d(\cdot)$ is a distance function. We define it as

$$d(w_k) \equiv d(i, j) \equiv |x_i - y_j| \tag{10.7}$$



(a) (b)

Figure 10.1: Illustration the dynamic time warping. (a) Cost Matrix. (b) Optimal warping path for time series

The optimum warping path for $dtw(X, Y)$ can be obtained through the dynamical programming approach [81]. It proceeds as follows: First, a m by n cost matrix D is constructed. A component $D(i, j)$ is defined recursively as sum of the distance $d(i, j)$ and the minimum of the cumulative distances in the adjacent elements :

$$D(i, j) = d(i, j) + min\{D(i-1, j-1), D(i-1, j), D(i, j-1)\} \tag{10.8}$$

After the entire cost matrix D is filled, starting from $D(1, 1)$, the minimum-distance warping path can be found in reverse order, starting from $D(m, n)$. For this purpose a greedy search is performed to evaluate cells to the left, down, and diagonally to the bottom-left. Whichever of these three adjacent cells has the minimum value is added to the beginning of the warping path found so far, and the search continues from that cell. The search stops if $D(1, 1)$ is reached. Figure 10.1(a) shows an example of two time series with their cost matrix and a minimum-distance warping path between them. The

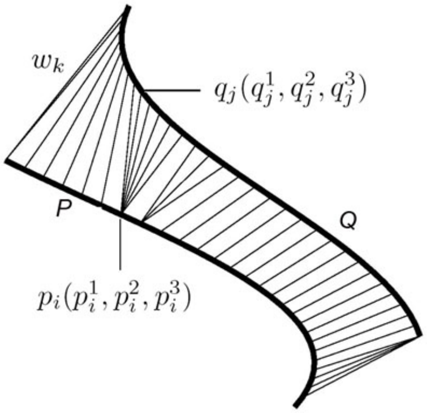
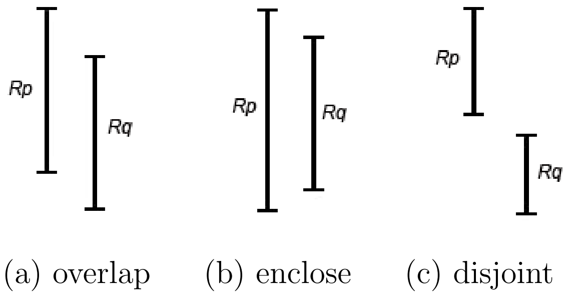


Figure 10.2: Illustration the optimal warping path of two 3D fibers.

warping path is $W = \{(1,1), (1,2), (1,3), (2,4), (2,5), (3,6), (4,7), (5,8), (6,9), (7,10), (8,11), (8,12), (9,13), (10,14), (11,15), (12,16), (13,17), (14,18), (15,19), (16,19), (17,20), (18,21), (19,21), (20,21), (21,21), (22,21), (23,21)\}$. If the warping path passes through a cell $D(i,j)$ in the cost matrix, the i^{th} point in time series X is warped to the j^{th} point in time series Y. Since a single point may map to multiple points in the other time series, dynamic time warping can handle time series with different lengths. An illustration of the optimum warping path between two time series can be found in Figure 10.1(b).

10.3.2 Fiber Similarity Measure with DTW

To calculate the similarity distance between fibers in space, the one dimensional concept of dynamic time warping is extended [159]. Suppose $p_i(p_i^1, p_i^2, p_i^3)$ and $q_j(q_j^1, q_j^2, q_j^3)$ are points of the fibers P and Q, indexed along the arc length, where the three coordinates are given within the brackets.



(a) overlap (b) enclose (c) disjoint

Figure 10.3: Illustration of possible arrangements of R_P and R_Q.

The distance between the two points is defined as:

$$d(p_i, q_j) = |p_i^1 - q_j^1| + |p_i^2 - q_j^2| + |p_i^3 - q_j^3| \tag{10.9}$$

Using this distance function, the spatial fiber problem is treated like a time series problem. The optimal warping path can be obtained through dynamical programming and is represented as $W = (w_1, ...w_k, ...w_K)$, where K is the length of the path and

$$d(w_k) \equiv d(p_i, q_j) \tag{10.10}$$

.

To reduce the effect of different lengths of fibers for similarity calculation, the similarity distance $DTW(P, Q)$ between the fibers P and Q is defined as the averaged distance for the optimal warping path, see Figure 10.2:

$$DTW(P, Q) = min(\frac{\sum_{k=1}^{K} d(w_k)}{K}) \tag{10.11}$$

10.3.3 Lower Bounding Distance

The time complexity of the fiber similarity measure DTW is $O(m \cdot n)$, which is demanding in terms of CPU time. To deal with this problem, an easily computed lower bounding distance LB in introduced. For two fibers, there is $LB \leq DTW$, where an efficient LB should be a tight lower bound to DTW.

For two given spatial fibers $P(P^1, P^2, P^3)$ and $Q(Q^1, Q^2, Q^3)$, they can be rewritten as three sequences of point pairs (P^1, Q^1), (P^2, Q^2) and (P^3, Q^3) respectively. First considering the pair-wise sequence (P^1, Q^1), $Max(P^1)$ and $max(Q^1)$ denote the maximum values in P^1 and Q^1, respectively. $Min(P^1)$ and $min(Q^1)$ define the minimum values. A pair $(min(P^1), max(P^1))$ defines the range R_P of P^1. Without loss of generality, it is assumed that $max(P^1) \geq max(Q^1)$. There are then three possible arrangements for the two ranges R_P and R_Q, see Figure 10.3.

The lower bounding distance between the two sequences is defined as follows [175]:

$$\mathsf{lb}(\mathsf{P}^1,\mathsf{Q}^1)=\begin{cases} \sum_{\mathsf{p}_i^1>\mathsf{max}(\mathsf{Q}^1)} |\mathsf{p}_i^1-\mathsf{max}(\mathsf{Q}^1)|+\sum_{\mathsf{q}_j^1<\mathsf{min}(\mathsf{P}^1)} |\mathsf{q}_j^1-\mathsf{min}(\mathsf{P}^1)| \\ \qquad\qquad\qquad\qquad \text{if } \mathsf{P}^1 \text{ and } \mathsf{Q}^1 \text{ overlap} \\ \sum_{\mathsf{p}_i^1>\mathsf{max}(\mathsf{Q}^1)} |\mathsf{p}_i^1-\mathsf{max}(\mathsf{Q}^1)|+\sum_{\mathsf{p}_i^1<\mathsf{min}(\mathsf{Q}^1)} |\mathsf{p}_i^1-\mathsf{min}(\mathsf{Q}^1)| \\ \qquad\qquad\qquad\qquad \text{if } \mathsf{P}^1 \text{ and } \mathsf{Q}^1 \text{ enclose} \\ \mathsf{max}(\sum_{i=1}^{|\mathsf{P}^1|} |\mathsf{p}_i^1-\mathsf{max}(\mathsf{Q}^1)|,\sum_{j=1}^{|\mathsf{Q}^1|} |\mathsf{q}_j^1-\mathsf{min}(\mathsf{P}^1)| \\ \qquad\qquad\qquad\qquad \text{if } \mathsf{P}^1 \text{ and } \mathsf{Q}^1 \text{ disjoint} \end{cases} \tag{10.12}$$

An analogous definition is used for the other pair-wise sequences (P^2, Q^2) and (P^3, Q^3). Finally, the lower bounding distance between fibers P and Q

```
algorithm rangeLBQuery $(P, \epsilon, \mathcal{D})$
    $Neigbors = \{\};$
    $For$ each object $Q \in \mathcal{D}$
        $LB\_dist = LB(Q, P);$
        $if(LB\_dist < \epsilon)$
            $DTW\_dist = DTW(Q, P);$
            $if(DTW\_dist < \epsilon)$
                $Neigbors.add(Q)$
            $EndIf$
        $EndIf$
    $EndFor$
Return $Neigbors;$
```

Figure 10.4: Pseudocode of ϵ-range query using lower bounding distance.

is defined as:

$$LB(P, Q) = \frac{lb(P^1, Q^1) + lb(P^2, Q^2) + lb(P^3, Q^3)}{m + n - 1} \tag{10.13}$$

The property that $LB(P, Q) \leq DTW(P, Q)$ for spatial fibers can be proven.

For two fibers $P = (p_1, ..., p_m)$ and $Q = (q_1, ..., q_n)$, for each dimension, there is $lb(p^d, q^d) \leq dtw(p^d, q^d), d = \{1, 2, 3\}$ [175]. wants to prove

$$DTW(P, Q) \geq LB(P, Q)$$

This is shown in the following.

$$
\begin{aligned}
DTW(P,Q) &= \frac{\sum_{k=1}^{K}(|p_i^1 - q_j^1| + |p_i^2 - q_j^2| + |p_i^3 - q_j^3|))}{K} \\
&\geq \frac{\sum_{k=1}^{K}(|p_i^1 - q_j^1| + |p_i^2 - q_j^2| + |p_i^3 - q_j^3|))}{m+n-1} \\
&= \frac{\sum_{k=1}^{K}(|p_i^1 - q_j^1|)}{m+n-1} + \frac{\sum_{k=1}^{K}(|p_i^2 - q_j^2|)}{m+n-1} + \\
&\quad \frac{\sum_{k=1}^{K}(|p_i^3 - q_j^3|))}{m+n-1} \\
&= \frac{dtw(p^1,q^1) + dtw(p^2,q^2) + dtw(p^3,q^3)}{m+n-1} \\
&\geq \frac{lb(p^1,q^1) + lb(p^2,q^2) + lb(p^3,q^3)}{m+n-1} \\
&= LB(P,Q)
\end{aligned}
$$

This concept is especially useful for density-based clustering, see the following Section 10.4. For clustering, it is necessary to perform a so-called range search. That means, it is necessary to query the data set for the most similar fibers with respect to P, i.e. for fibers Q with $DTW(P,Q)$ below a given threshold ϵ. First, $LB(P,Q)$ is computed for these fibers. $DTW(P,Q)$ must then only be computed for those fibers with $LB(P,Q) < \epsilon$. The Pseudocode of ϵ-range search with lower bounding distance is provided in Figure 10.4.

10.4 Fiber Clustering

10.4.1 Density-based clustering fiber tracts

After calculation of the fiber similarity, each fiber is regarded as a data object in metric space and a clustering approach can be applied to group the fibers. Since clustering is a common challenge in a large variety of applications, it has attracted much attention during the last decades, producing a vast number of research papers, books and surveys, eg. [47, 55, 8, 116, 82, 172] to mention a few. One very interesting branch of research considers the clustering problem from a density-based point of view: Clusters are regarded as areas of high object density which are separated by areas of lower object density. This cluster notion has several attractive benefits, four of which are of major importance for fiber clustering: (1) Unlike in many other methods, the user does not have to specify the number of clusters he wants to find. (2) Density-based clustering algorithms are able to detect clusters of arbitrary shape. (3) It is robust to noise and outliers. (4) Density-based clustering methods are not restricted to vector data but applicable on general metric spaces. Recently, some approaches to density-based clustering have been proposed, including DBSCAN [55], DENCLUE [78], and OPTICS [8]. However, to the best of knowledge none of them has been applied to the problem of fiber clustering so far. DBSCAN is the most wide spread algorithm for density-based clustering and its definitions are intuitive in our context. The density-based clustering notion is formalized in DBSCAN using two parameters: ϵ specifying a range and $MinPts$ specifying a number of objects. The central notion of DBSCAN is the *core object*. A fiber is a core object of a density-

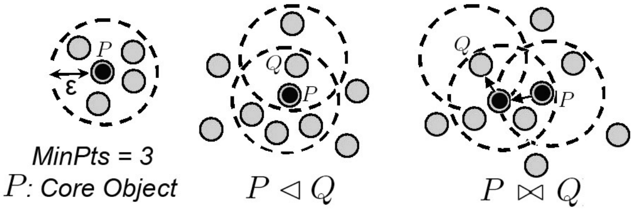


Figure 10.5: Illustration of Density-based Clustering with DBSCAN.

based cluster if at least $MinPts$ fibers are in its ϵ-neighborhood. Formally this is captured by the following definitions:

Definition 10.1 (Core Object) Let $\mathcal{D}$ be a set of n objects, $\epsilon \in \mathbb{R}^+$ and MinPts $\in \mathbb{N}^+$. An object $P \in \mathcal{D}$ is a core object, iff

$$|N_\epsilon(P)| \geq \text{MinPts, where } N_\epsilon(P) = \{Q \in \mathcal{D} : ||P - Q|| \leq \epsilon\}.$$

Two objects may be assigned to a common cluster. In density-based clustering this is formalized by the notions *direct density reachability, and density connectedness.*

Definition 10.2 (Direct Density Reachability) Let $P, Q \in \mathcal{D}$. Q is called directly density reachable from P (in symbols: $P \lhd Q$) iff

1. P is a core object in $\mathcal{D}$, and
2. $Q \in N_\epsilon(P)$.

If P and Q are both core objects, then $P \lhd Q$ is equivalent with $P \rhd Q$. The density connectedness is the transitive and symmetric closure of the direct density reachability:

Definition 10.3 (Density Connectedness) Two objects P and Q are called density connected (in symbols: $P \bowtie Q$) if there is a sequence of core objects $(P_1, ..., P_m)$ of arbitrary length m such that

$$P \rhd P_1 \rhd ... \lhd P_m \lhd Q.$$

For illustration, Figure 10.5 demonstrates the definitions of DBSCAN. Here, P is a core object, Q is directly density reachable and density connectedness from P in the middle and right of the figure, respectively.
In density-based clustering, a cluster is defined as a maximal set of density connected objects:

Definition 10.4 (Density-based Cluster) A subset $C \subseteq \mathcal{D}$ is called a cluster if the following two conditions hold:

1. Density connectedness: $\forall P, Q \in C : P \bowtie Q$.

2. Maximality: $\forall P \in C, \forall Q \in \mathcal{D} \setminus C : \neg P \bowtie Q$.

The algorithm DBSCAN [55] implements the cluster notion of Definition 10.4 using a data structure called *seed list* S containing a set of seed objects for cluster expansion. More precisely, the algorithm proceeds as indicated in Figure 10.6.

Since every object of the database is considered only once, the complexity is n times the complexity of $N_\epsilon(P)$. To illustrate the algorithmic paradigm, Figure 10.7 displays a snapshot of DBSCAN during cluster expansion. For simplicity, 2-dimensional vector data is displayed but note that all definitions of DBSCAN apply to general metric data as well. The light grey cluster on the left side has been processed already. The algorithm currently expands the dark grey cluster. The seed list S only contains the object P. P is a core object since there are more than $MinPts = 3$ objects in its ϵ-neighborhood.

```
algorithm Density-based Fiber Clustering
  Mark all objects as unprocessed.
  While($\mathcal{D}$ contains unprocessed object) Loop
     Consider arbitrary unprocessed object $P \in \mathcal{D}$
     $N_\epsilon(P)$ = rangeLBQuery($P$,$\epsilon$, $\mathcal{D}$);
     If $N_\epsilon(P).size < MinPts$
        assign $P$ Noise; Continue;
     Else
        assign new cluster-ID $C$
     EndIf
     For all elements $Q \in N_\epsilon(P)$
        If $Q$ is unprocessed
           mark element $Q$ with cluster-ID $C$
           insert object $Q$ into seed list $S$.
        EndIf
     EndFor
     While($S$ not $\emptyset$) Loop
        For all elements $P' \in S$
           $N_\epsilon(P')$ = rangeLBQuery($P'$,$\epsilon$,$\mathcal{D}$);
           If $N_\epsilon(P').size > MinPts$
              For all elements $Q' \in N_\epsilon(P')$
                 If $Q'$ is unprocessed or Noise
                    If $Q'$ is unprocessed
                       insert object $Q'$ into seed list $S$
                    EndIf
                    mark element $Q'$ with cluster-ID $C$
                 EndIf
              EndFor
           EndIf
        EndFor
     EndLoop
  EndLoop
```

Figure 10.6: Pseudocode of DBSCAN algorithm.

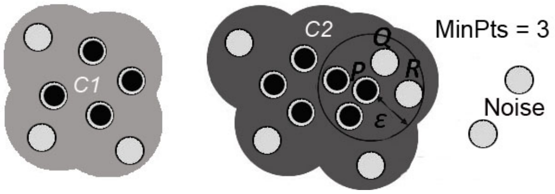


Figure 10.7: Illustration of Density-based Clustering with DBSCAN.

Two of these objects, Q and R have not been processed so far and are therefore inserted into S. This way, the cluster is iteratively expanded until S is empty. After that, the algorithm continues with an arbitrary unprocessed object until all objects have been processed.

Like other clustering approaches, DBSCAN requires user to specify two parameters: ϵ and $MinPts$, which can be roughly estimated by visualizing the first "valley" of a sorted $k - dist$ graph [55].

10.5 Experimental result and analysis

In this section, a series of numerical experiments are performed on synthetic and real data to explore the efficiency and effectiveness of the approach. All algorithms are implemented in Java and the fiber visualization functions are programmed in MatLab. The calculations have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM.

10.5.1 Cluster Validation Measure

To compare clustering results for different approaches with a ground truth, an information-theoretic external cluster-validity measure [50] is used. A short description of this measure is given: A clustering of a set of fibers can be described by a set of cluster labels, mapping every fiber to a cluster. The evaluation scheme measures how useful the calculated cluster labels are as predictors of the ground truth cluster labels. The clustering quality measure is defined as the entire encoding cost, which is the sum of the empirical conditional entropy and the code length for the number of clusters. For more information, please see Section 2.1.2.

10.5.2 Fiber Data

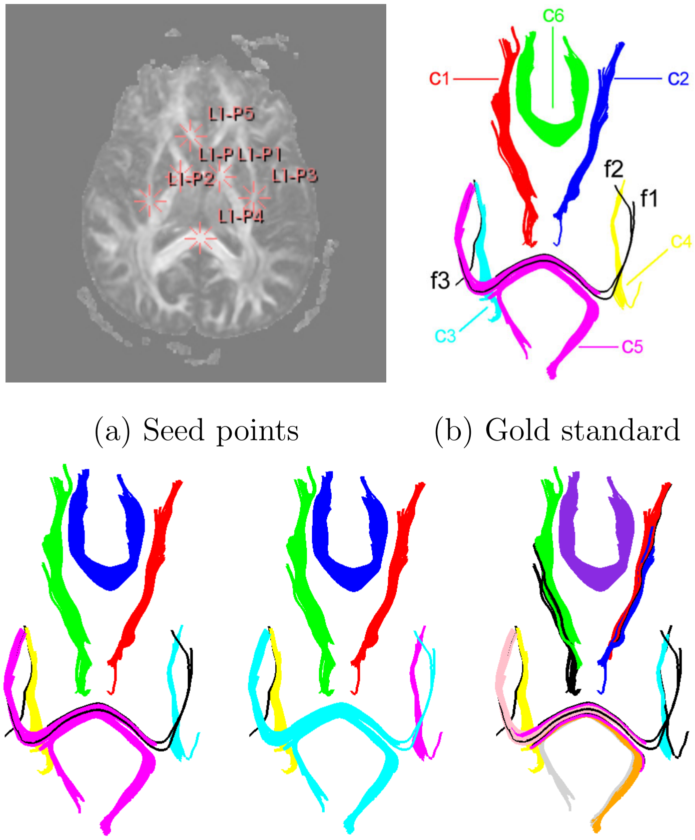
To evaluate the proposed approach, realistic human brain data is taken from the open source software “Slicer3-3.4”, folder “surgery case” (http://www.slicer.org/). This measurement comprises 55 non-collinear directions for the diffusion weighted images, with a $b - factor = 1000\ s/mm^2$, and 5 acquisitions for the reference, with a $b - factor = 0\ s/mm^2$. The brain volume contains $256 \times 256 \times 70$ voxels with size $1 \times 1 \times 2.6\ mm^3$. For fiber tracking the numerical Runge Kutta method (4-th order) is applied. The seed points are given in two ways : First, small fiducial seed regions are defined by specifying their central points within the map of Fractional Anisotropy (FA); second, ROI seeding is defined by all seed points within a larger constrained volume. The deterministic tracks for fiber clustering are started from the seed points, following the principal diffusion in both directions. They are stopped for too low FA and for too high curvature of the tracks.

10.5.3 Experiments on Fiber Similarity Measure

To explore different similarity measures, a realistic set of fibers is obtained by specifying 6 fiducial seed regions manually (see Figure 10.8(a) for the seed regions). They are located in the internal and external Capsules and in the Corpus Callosum. After fiber tractography, gold standard fiber clusters (ground truth) are created with the help of an experienced physician (See Figure 10.8(b)). This gold standard includes 372 fibers and shows 6 anatomically meaningful fiber bundles assigned by C1-C6 with different colors and one group of noise or outliers (3 fibers, black color). These fibers $f1, f2, f3$ differ from their neighbours by shape and length.

Based on these data, the similarity measure (DTW) is compared with two frequently used measures: Mean of closest point distance (MCP) and Hausdorff distance (HDD). For all three measures, the density-based clustering method is applied to achieve the result which is closest to the ground truth. It is clear that DTW and MCP separate the data into the 6 gold standard clusters of the ground truth, whereas HDD already segments the data for a global threshold into 10 clusters. For further comparison between DTW and MCP, the focus is the noise fibers indicated in Figure 10.8(b) in black color. Whereas MCP does not separate $f1, f2$ from the neighbors, DTW matches the ground truth. As MCP averages the minimum distances of point pairs from one fiber to another, it seems to "smooth out" the information in the data more than DTW.

Apart from the evaluation of effectiveness of a fiber similarity measure, the efficiency (computation cost) should also be considered. For this purpose experiments are performed on a range search (search the $\epsilon-$neighborhood



(a) Seed points (b) Gold standard

(c) DTW (d) MCP (e) HDD

Figure 10.8: Experimental results of different fiber similarity measures. (a): Seed points, (b): Gold standard, (c-e): Fiber clustering results with different measures, where the same fiber bundles in identical coloring.

fibers for one fixed fiber), which is the most time consuming step in fiber clustering. The involved similarity measures include: DTW with lower-bounding distance LB, DTW, MCP and HDD. The number of test fibers in the data set range from 1000 to 5000. Figure 10.9 presents the computational cost of range search for increasing numbers of fibers.

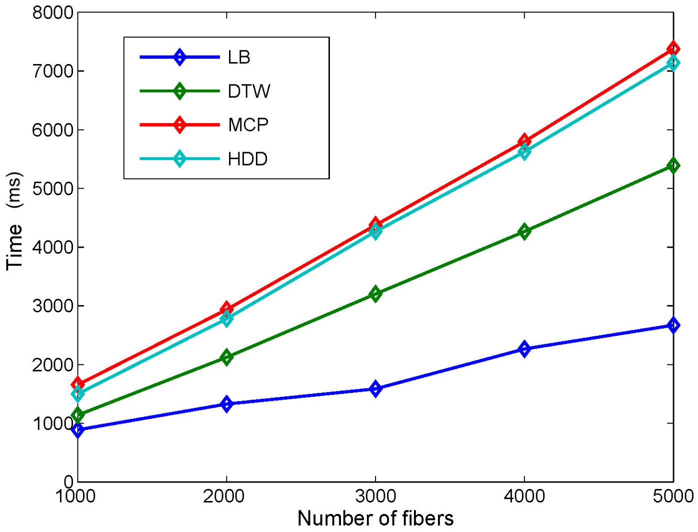


Figure 10.9: Comparison of efficiency with different fiber similarity measures.

Figure 10.9 shows that the efficiency of DTW outperforms that of MCP and HDD. LB further improves the efficiency of DTW. The larger the size of the data, the more the advantage of the LB technique becomes apparent. For example, LB costs about 60% and 35% of the computation time needed for MCP, when they are applied to 1000 and 5000 fibers respectively. Compared with DTW, LB improves efficiency by about 50% for 5000 fibers.

10.5.4 Experiments on Fiber Clustering

In this section, experiments are performed on synthetic data as well as many real brain data to test the density-based fiber clustering scheme using the fiber similarity measure with lower-bounding technique. Other two fiber clustering approaches: Spectral clustering [116]and Hierarchical clustering (Single Link)[177] are also implemented in Java to compare with our fiber clustering approach.

Synthetic Data

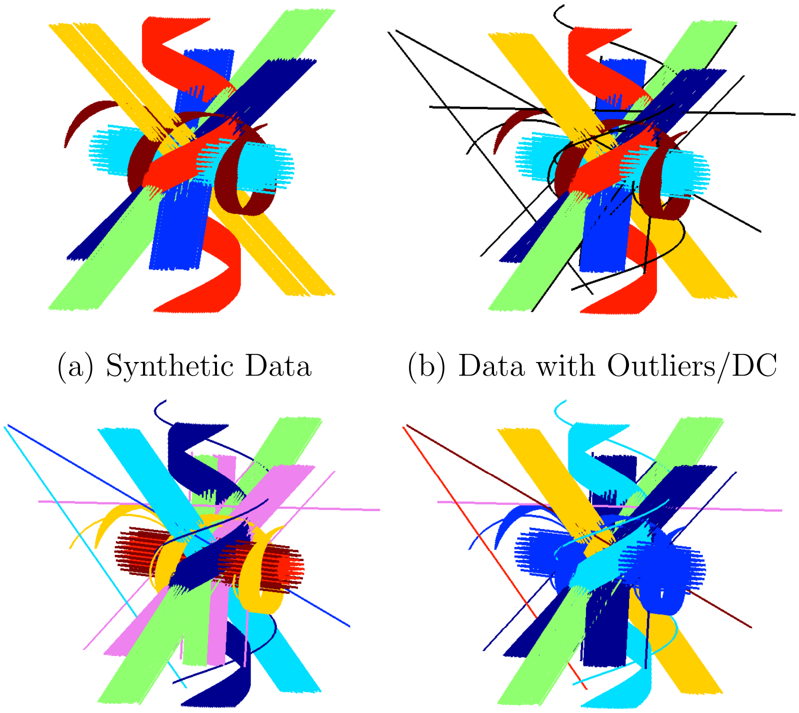
The ground truth of our synthetic data in three dimensions includes five clusters of straight lines and two clusters of helices (See Figure 11.4(a)), these clusters are composed of 410 individual fibers. Due to limitations of the experiment and of the tracking method, outliers may appear in fiber data. To evaluate the three clustering methods with respect to such outliers, 8 outlier lines and 2 outlier helices are added to the synthetic data (Figure 11.4(b)), creating a total of 420 synthetic fibers.

For all three clustering methods the ground truth presented in Figure 11.4(a) is reproduced perfectly. The situation is different for Figure 11.4(b). Only the density-based fiber clustering can reproduce the ground truth, if the outliers are included, see (Figure 11.4(b-d)). As the knowledge of the synthetic data (the true class label for each object) is known, like the comparison of similarity measures, the information-theoretic external cluster-validity measure is applied to qualify the clustering results with different clustering methods. The Table 10.1 presents the quality of clustering results of data without and with outliers, which further indicates the effectiveness of the density-based fiber clustering.

Real Data

In this section, the approach is applied on three real data sets with different clustering structures, to demonstrate the method's effectiveness.

Data 1: For Figure 10.11, 825 fibers are seeded from 4 different compact



(a) Synthetic Data (b) Data with Outliers/DC

(c) Spectral Clustering (d) Single Link

Figure 10.10: Different fiber clustering results based on synthetic data.

seed regions. From the mid-sagittal FA slice, one seed region is taken for occipital and temporal tracks and another one for prefrontal tracks. Two additional seed regions are taken in a symmetric way from a coronal FA slice to track projection fibers ending within the brain in the corona radiata [110]. Using the estimating parameters ($\epsilon = 10, MinPts = 6$) for clustering, Figure 10.11 (a-c), shows the 4 clusters are in agreement with the seed regions. Three fibers are viewed as outliers with dark blue color (Figure 10.11 (d)).

Data 2: For the data set, 973 fibers are seeded from one seed region from the mid-sagittal FA slice. For clustering, six fiber bundles and few noisy fibers are detected, which are illustrated in Figure 10.12.

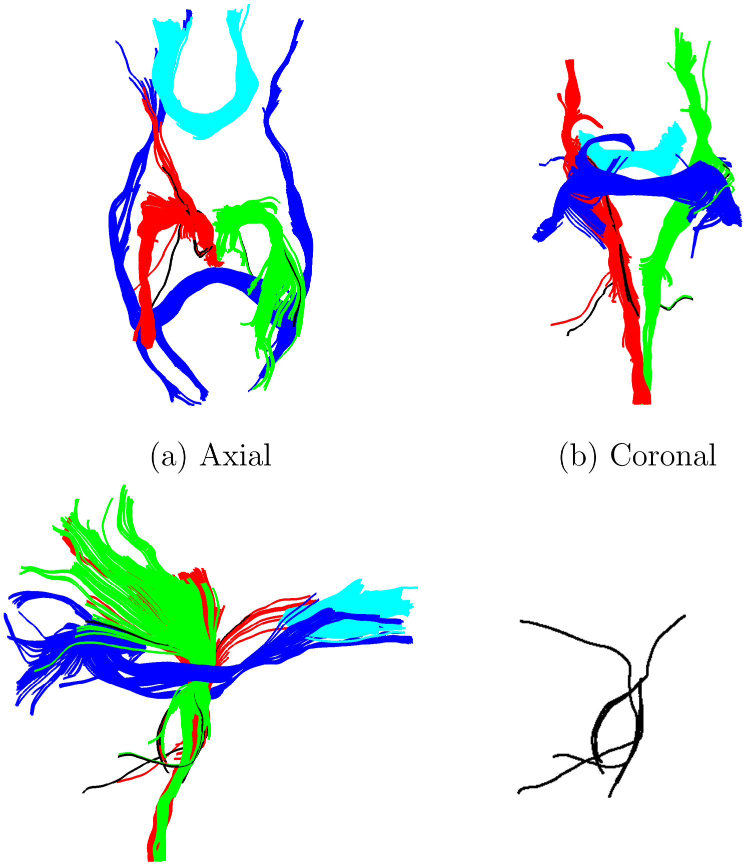
Table 10.1: Comparison of different fiber clustering methods without and with outliers

Measures	Performance without and with Outliers		
	Conditional-entropy	*Code-length*	*Encoding-cost*
DC	0.0|0.0	0.304|0.358	**0.304|0.358**
SL	0.0|0.515	0.304|0.311	**0.304|0.826**
SC	0.0|0.775	0.304|0.232	**0.304|1.001**

Corpus Callosum: The corpus callosum is a white matter structure located just ventral to the cortex that connects the left and right cerebral hemispheres to allow communication between the two halves of the brain. From the mid-sagittal slice, 1100 seed points are taken from the corpus callosum via a FA map. Main part of the fibers connects cortical areas in approximate mirror-image sites. A smaller set is built of commissural trajectories to the temporal lobes [110]. With parameters ($\epsilon = 10, MinPts = 6$), three main fiber bundles and noisy fibers are successfully identified (Figure 10.13).

10.6 Conclusion

In this chapter, a novel framework is presented for clustering white matter tracts. The technique combines a novel fiber similarity measure using adapted dynamic time warping and an outlier robust density-based clustering. The extensive experiments on synthetic as well as real data demonstrate the effectiveness and efficiency of our approach. In summary, the proposed



(a) Axial

(b) Coronal

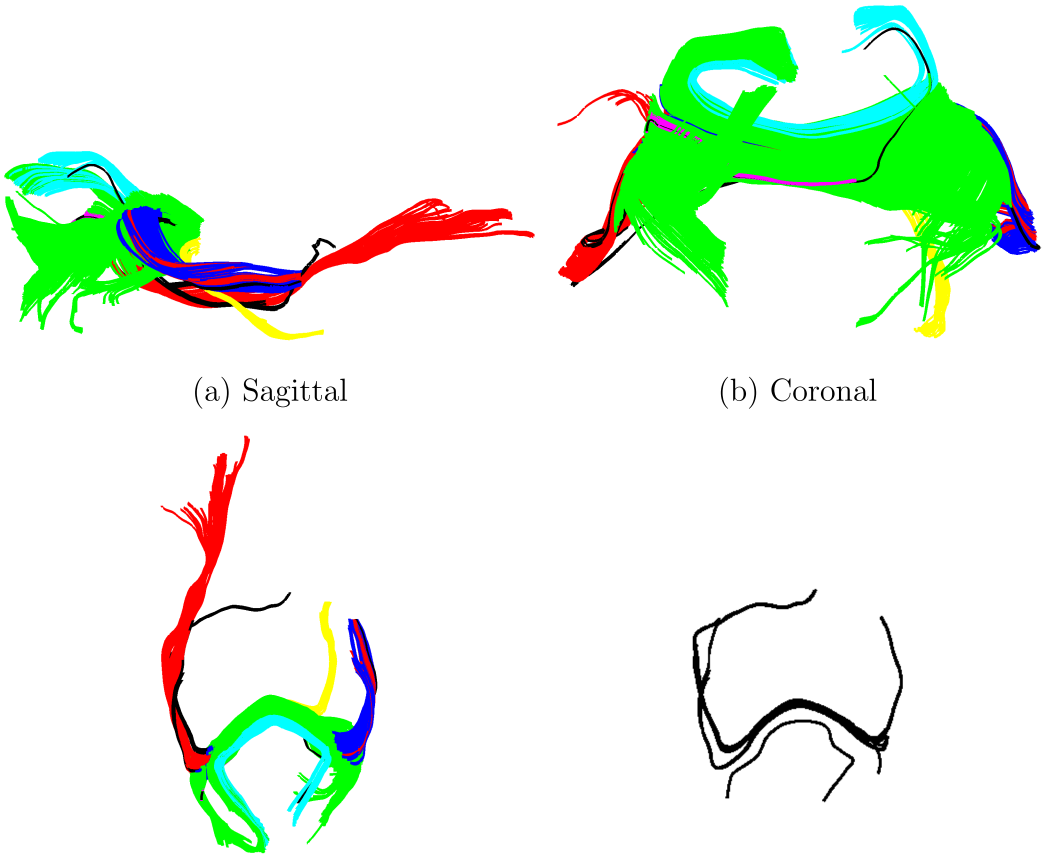
(c) Sagittal

(d) Sagittal (Noise)

Figure 10.11: The automatic fiber clustering result for Real Data 1 with parameters $MinPts = 6, \epsilon = 10$.

fiber clustering method shows several desirable properties:

1. The adapted dynamic time warping is proposed to define fiber similarity, which allows to capture local similarity among fibers belonging to a common bundle but having different start and end points. It is more effective and efficient than MCP or HDD.



(a) Sagittal

(b) Coronal

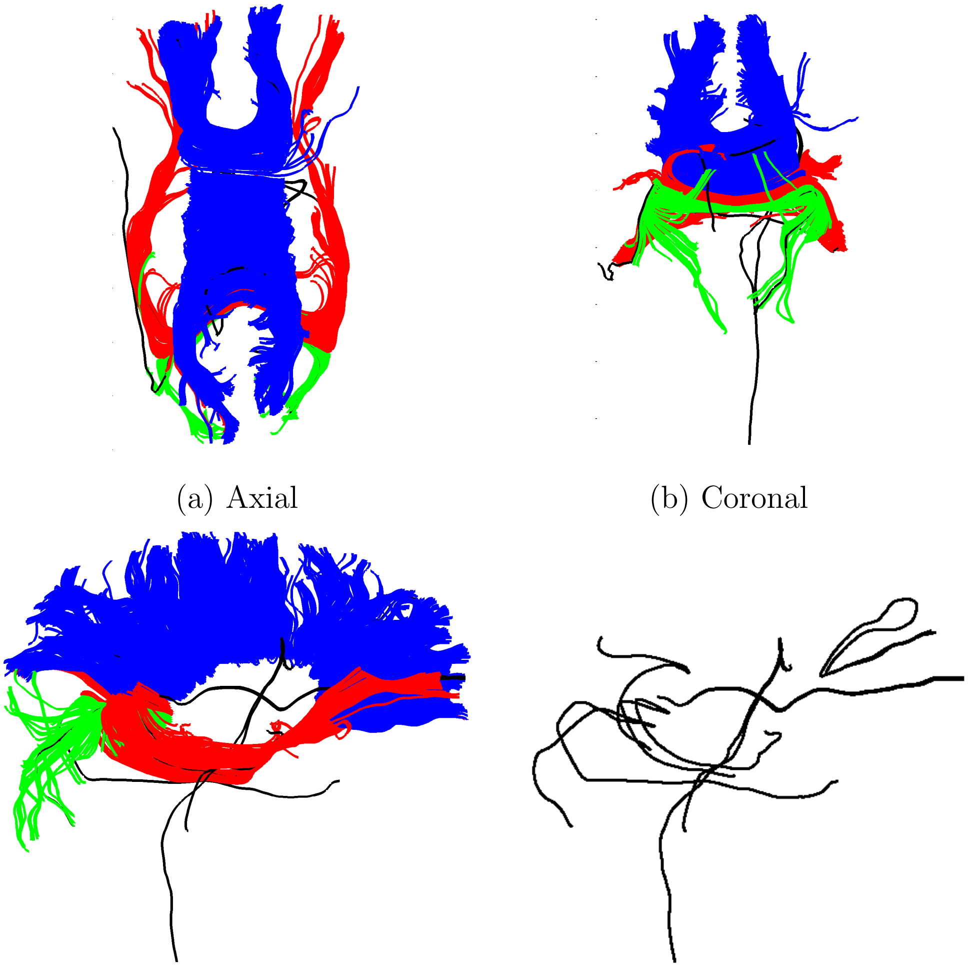
(c) Axial

(d) Axial (Noise)

Figure 10.12: The automatic fiber clustering result for Real Data 2 with parameters $MinPts = 6, \epsilon = 5$.

2. In order to deal with the time complexity of fiber similarity, a lower-bounding technique is proposed for 3D fibers to speed up computational cost.

3. To explore meaningful fiber bundles in the human brain, the fiber clustering problem is considered from a density-based point of view. The



(a) Axial

(b) Coronal

(c) Sagittal

(d) Sagittal (Noise)

Figure 10.13: The automatic fiber clustering result for Corpus Callosum with parameters $MinPts = 6, \epsilon = 10$.

number of clusters is not needed a priori and the DBSCAN parameters can be heuristically estimated by a $k - dist$ graph.

4. The density-based clustering approach can deal with noisy fibers which may be caused by limitations of imaging or by the fiber tracking tech-

nique. These noisy fibers are easy to be excluded from any cluster, which is of significant importance for further processing such as group tract-based analysis, etc.

Chapter 11

Hierarchical Density-based Clustering of White Matter Tracts in the Human Brain

In Chapter 10, the density-based clustering algorithm DBSCAN is applied for automated white matter tract clustering. However, the approach limited the data clustering in only one scale. In the real fiber data sets, many of them are represented in a hierarchical structure. It is more reasonable to analyze them in a hierarchical way. Thus, in this chapter, the algorithm OPTICS is applied, to sort the data into a reachability plot, visualizing the clustering structure of the data. Afterwards, interactive and automatic clustering algorithms are then introduced to obtain the hierarchical clusters. The hierarchical density-based clustering method enables to group fiber tracts into meaningful bundles on multiple scales as well as eliminating noisy fibers.

The remainder of the chapter is organized as follows: A brief introduction is given in Section 11.1. Section 11.2 presents the hierarchical density-based fiber clustering in detail. A series of clustering experiments on synthetic and real data and the relevant results are described in Section 11.3. A discussion of the results follows in Section 11.4. Finally, this chapter is concluded in Section 11.5. Parts of the material presented in this chapter have been published in [146], where J.S., K.H., C.B. and C.P. designed the research; J.S. implemented algorithms; J.S. and K.H. performed the experiments; J.S., K.H., and Q.Y. analyzed the data; and J.S., K.H., Q.Y., C.B., A.M., N.M. and C.P. wrote the paper.

> "*Junming Shao, Klaus Hahn, Qinli Yang, Afra M. Wohlschläger, Christian Böhm, Nicholas Myers, Claudia Plant. Hierarchical Density-Based Clustering of White Matter Tracts in the Human Brain. IJKDB 1(4): 1-25 (2010).*"

11.1 Introduction

During the last decade, some hierarchical fiber clustering algorithms have been proposed. Zhang and Laidlaw [178] use an agglomerative hierarchical clustering algorithm. This type of algorithm decomposes a fiber set into several levels of partitions, represented by a dendrogram. A drawback of hierarchical clustering is that it is hard to find the best partition of the dendrogram. Maddah et al. [103], [102] incorporate anatomical expert knowledge into clustering. They use an atlas of fiber tracts, labeled by the number of their bundle. To build such an atlas, they start with a set of labeled ROIs specified by an expert and calculate the corresponding fiber bundles seeded from those ROIs. Affine registration is then used to map extracted fibers of some subject to the atlas. To make the comparison between fibers of the subject and of the atlas efficient, B-spline representations of the tracts are used. Similar subject and atlas fibers are labeled identically. Donnell et al. [119] also present an atlas based segmentation of fiber tracts. Their segmentation is performed in two steps. First, an atlas, labeled by experts, is learned from a population of subjects using spectral clustering. Based on this model, fibers of a novel subject are clustered by an extension of the spectral clustering solution stored in the atlas, using the Nystrom method. This segmentation across subjects enables testing of neuroscientific hypotheses with respect to group differences.

Though appreciable progress has been achieved in fiber clustering, the following problems are still challenging and may justify a novel approach:

1. Flexibility: The fiber clustering approach should be flexible enough to cluster fibers on multiple scales. Hierarchical cluster structures are prevalent in nature and can be found e.g. for the fibers of the Corpus Callosum.

2. Transparency: A frequently used visualization of a hierarchical clustering result is the dendrogram. This graph lacks transparency for large data sets and is hard to analyze. More informative visual presentations of a clustering structure would be useful.

3. Outlier-detection and -robustness: Due to experimental limitations, like thermal noise or partial volume effects, the set of fibers produced by tractography contains also imperfect fibers or outliers. It is benefi-

cial for further tract-based analysis to separate out those fibers during clustering to obtain compact clusters. Another aspect is the impact of such outliers on the results of some clustering algorithms. Algorithms which are robust to outliers are necessary.

In view of these issues, in this chapter, the hierarchical density-based clustering is applied to group fiber tracts. The resulting clustering structure of the fiber data can be visualized in a transparent manner by a so-called reachability plot of OPTICS. From this plot, the clusters are subsequently calculated by interactive or automatic fiber clustering algorithms on multiple scales. Imperfect fibers or outliers are eliminated during the clustering process. The approach is tested on several synthetic as well as real fiber data sets.

11.2 Hierarchical Density-based Fiber Clustering

After calculation of the fiber similarities based on dynamic time warping (cf. Section 10.3), a clustering approach can be applied to group fibers. Here, hierarchical density-based clustering is used.

11.2.1 Hierarchical Density-Based Clustering

The idea behind density-based clustering is, that the object density in neighborhood of each cluster object P is sufficiently high. The basic algorithm for density-based clustering is DBSCAN [55], also see Section 10.4. In DB-

SCAN a flat clustering of the data is computed. OPTICS [8] is a hierarchical density-based clustering algorithm, which emerges from DBSCAN. In contrast to DBSCAN, OPTICS does not assign cluster membership, but orders objects of a dataset D into a hierarchical cluster structure. This structure contains clustering information, which could also be achieved by all possible DBSCAN clusterings with respect to distances ϵ' that are smaller than the generating distance ϵ. To describe the hierarchical cluster structure, each object is characterized by two measures, its core-distance and its reachability-distance. Formally this is captured by the following definitions:

Definition 11.1 (Core-distance of an object P) Let $P \in \mathcal{D}$, $MinPts \in \mathcal{N}$, $\epsilon \in \mathcal{R}$, $N_\epsilon(P)$ be the ϵ-neighborhood of P, $MinPts_{dist}(P)$ be the distance from P to its $MinPts$-nearest neighbor. The core-distance of object P is defined as:

$$\mathrm{CoDist}(P) = \begin{cases} \text{UNDEFINED}, |N_\epsilon(P)| < MinPts \\ \\ MinPts_{dist}(P), otherwise. \end{cases}$$

The core-distance of an object P measures its local density. It is defined as the $MinPts$-nearest neighbor distance of P if P is a core object. Otherwise it is UNDEFINED.

Definition 11.2 (Reachability-distance of O with respect to object P) Let $P, O \in \mathcal{D}$, $MinPts \in \mathcal{N}$, $\epsilon \in \mathcal{R}$, $N_\epsilon(P)$ be the ϵ-neighborhood of P,the reachability distance of object O w.r.t P is defined as:

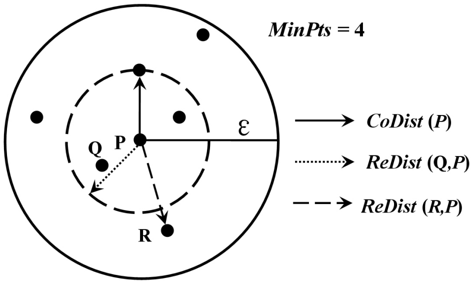


Figure 11.1: Illustration of core-distance and reachability-distance. The radius of the solid circle indicates the generating distance ϵ while the radius of the dashed circle is the $MinPts$-nearest neighbor distance of P (core-distance).

$$\text{ReDist}(O, P) = \begin{cases} \text{UNDEFINED}, |N_\epsilon(P)| < MinPts \\ \\ max(\text{CoDist}(P), dist(O, P)), otherwise \end{cases}$$

Figure 11.1 illustrates the notions of core-distance and reachability-distance. The reachability-distance of Q w.r.t. P is the core-distance of P while the reachability-distance of R w.r.t. P is the distance between P and R.

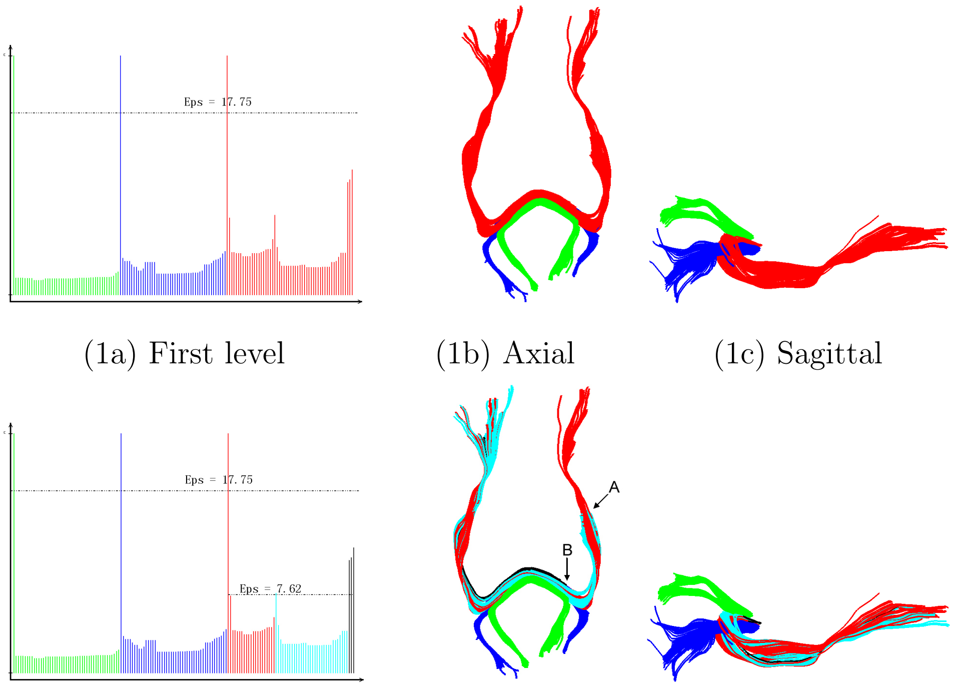
The result of the algorithm OPTICS is an ordering of the data set reflecting the hierarchical cluster structure. More precisely, during the run of the algorithm, OPTICS computes for each object the core distance and a suitable reachability distance and writes this information to an output file. OPTICS starts with an arbitrary unprocessed object O, sets its reachability distance to UNDEFINED and determines its core distance. The object O is now processed and written with its core and reachability distance to the output. If O is a core object, all objects in the ϵ-neighborhood of O are retrieved and inserted into a data structure called *seed list*. The seed list is a priority queue and the objects in the seed list are sorted according to the minimum reachability distance with respect to any of the objects processed before. As next object, OPTICS always selects the top object of the seed list, or in case the seed list is empty, an arbitrary unprocessed object from the data set. Consider the case that the start object O is a core object and object P is the top object in the seed list which has minimum reachability distance w.r.t. O. Object P is processed now, which implies that P is written with its core distance and reachability distance to the output. If P is a core object, all objects in the ϵ neighborhood are added to the seed list. If necessary, the seed list is updated: Some objects already in the seed list may have a smaller reachability distance from P than from O. The algorithm terminates as soon as all objects have been processed. In this ordering, for every object the smallest reachability distance w.r.t. its preceding object is determined.

Finally, the structure of the data ordering can be visualized in a reachability plot. This reachability plot is a 2D plot showing the objects' position index on the x-axis and the objects' reachability-distance on the y-axis. The reachability plot provides a visual representation of the cluster structure of the data from the density-based point of view and enables an intuitive way to find clusters. Valleys in this plot indicate clusters, the deeper the valley, the denser the cluster. Clusters can be obtained by cutting the reachability plot with a horizontal line. An example of a reachability plot for a 2D data set is presented in Figure 2.2. The reachablity plot provides information about the general number of clusters, the densities of different clusters, and the hierarchical structure of the data.

Like DBSCAN, OPTICS requires two parameters: $MinPts$ and ϵ. For DBSCAN it is difficult to find the optimal parameters for clustering [55]. OPTICS, however, is rather insensitive to the input parameters [8]. For all experiments in this chapter, $MinPts = 10$ and $\epsilon = 30$ have been used. See Section 11.4 for a discussion of parameter selection.

11.2.2 Fiber Cluster Extraction

The simplest way to obtain clusters in a reachability plot is to select a suitable parameter $0 < \epsilon_i < \epsilon$ and to cut the reachability plot with the corresponding horizontal line. Intuitively, the valleys below the line indicate the clusters. Also their left borders are part of the clusters, if they are not noise. Noise or outliers of clusters are defined by the condition that their core- and reachability distances are larger than ϵ_i. See for an illustration Figure 2.2. With parameter ϵ_1, four clusters A, B, C, and D are obtained. If the line is moved down to ϵ_2, four new clusters B, C, D1, and D2 are obtained, and some data which are interpreted as outliers (e.g. the cluster A and the group on the right side of D1). This demonstrates that a global threshold may not be sufficient, to extract all meaningful clusters (e.g. A with ϵ_2), since clusters can have varying densities. Therefore, two improved methods are proposed for fiber clustering.



(1a) First level (1b) Axial (1c) Sagittal

(2a) Second level (2b) Axial (2c) Sagittal

Figure 11.2: Illustration of the interactive clustering approach on a real data set. (1a): interactive fiber cluster extraction at the first level based on the reachability plot. (1b-1c): The corresponding segmented fiber bundles in Axial and Sagittal perspectives. Different colors indicate different fiber bundles. (2a): Fiber cluster extraction at the second level. (2b-2c): The segmented fiber bundles at the second level, where the fiber bundles with red color are further split into two new fiber bundles: one with cyan color ending at A and another with the red color. Three fibers with black color ending at B are interpreted as noise.

Interactive Fiber Cluster Extraction

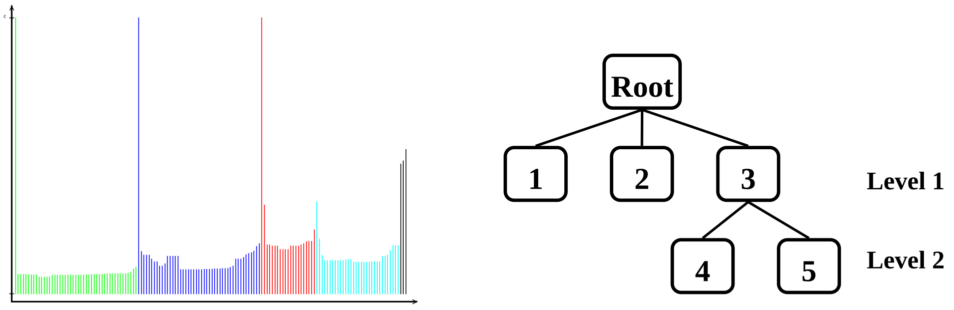
To extract fiber bundles with different densities, it is desirable to use a series of localized parameters. A novel software is implemented to enable such interactive fiber clustering. The parameters can be selected from a screengraph of the reachability plot in several steps.

1. For a first segmentation of the fibers, a global ϵ^1 is selected (first level). This may create several large clusters $(C_1^1, ..., C_i^1, ..., C_n^1)$ and one cluster O for noise, where the core- and reachability distances are above ϵ_1. Quantification of noise follows the definition proposed for DBSCAN.

2. For each cluster C_i^l found at level l, there may exist several valleys inside. To obtain those smaller subclusters, it goes to next level $l = l+1$, selecting the parameter ϵ_i^l only within C_i^{l-1} and proceed like in step 1). This creates new subclusters and in some cases noise.

3. To find clusters at higher levels, step 2) is repeated until the fibers are appropriately segmented.

To select the parameter values ϵ on the screen-graph, users only need to double-click one position (x,y) in the reachability plot. The y-coordinate of the position indicates the ϵ, the x-coordinate is used to indicate its range. The first range is automatically global, while further ranges are localized to the cluster chosen. Figure 11.2 illustrates this approach on a real data set. Here, 250 fibers are seeded in the Splenium of Corpus Callosum (see Section 10.5.2 for details of the data and of the tractography). For the first segmentation into three clusters $\epsilon^1 = 17.75$ is chosen (see Figure 11.2 /(1a-1c) for the segmented reachability plot and for the fiber clusters in identical coloring). At the second level, we use $\epsilon^2 = 7.62$, to separate the two obvious valleys within the red cluster. See Figure 11.2/(2a-2c) for the resulting clusters and for the three noise fibers. The arrows in Fig. 11.2/2b indicate the endpoints



(a) Reachablity Plot (b) Fiber Cluster Structure

Figure 11.3: Illustration of automatic fiber clustering. (a): Automatic fiber clustering based on the reachability plot with $ratio = 0.7, minSize = 10$. (b): The cluster structure of the data set.

of the cluster colored in cyan (A) and of the noise cluster (B), colored black. This fast and easy to use software provides users with a flexible method to quantify a visually meaningful clustering of a reachability plot.

Automatic Fiber Cluster Extraction

The interactive method is useful to get a segmentation of data sets with a simple cluster structure, which can be inspected visually. For more complex data sets, however, an automatic segmentation is needed. To achieve this goal, the Tree Clustering algorithm [138] is adapted, which extracts a hierarchical clustering from a reachability plot and creates a cluster tree. The basic idea in this algorithm is to identify significant local maxima separating the valleys in the reachability plot and to use these points for cluster extraction. A local maximum is regarded as significant if its height is well above the level of the valleys to its right and left side.

To determine the clusters between the local maxima, two parameters are introduced: $MinSize$, specifying the minimum number of fibers in a cluster or the minimum scale of segmentation, and *Ratio*, which determines if the local maximum is sufficiently above the neighbouring valleys to use it as a split point between two clusters.

The adaption of the Tree Clustering algorithm is briefly described, which can now also segment highly irregular reachability plots. First, all points P are collected, whose reachability distances (RD) are local maxima of the right and left neighbourhoods that enclose at least $MinSize$ points. This set of points P is then sorted in descending order according to their RD $P.r$. Clusters are determined from this list by recursively removing the point P with largest RD, possibly separating clusters, until the list is empty. P is regarded as split point SP of two neighbouring clusters, if the median values of the RDs in both valleys are significantly lower than $SP.r$ ($median/SP.r < Ratio < 1$). Noise points N within a cluster are detected by the condition $N.r > SP.r$ and $CoDist(N) > SP.r$. Clusters without noise are listed in a tree graph, and noise is collected separately. There are two places in the tree where new nodes can be added: they can become children of the current node, or children of the parent node, replacing the current node. If the RDs of the current and parent split points are close, the newly created nodes are attached to the parent node instead of the current node itself.

Figure 11.3 illustrates the application of this method to the same data set which was used for interactive clustering. The reachability plot (Fig. 11.3(a)) and the cluster tree (Fig. 11.3(b)) indicate the same fiber clustering and noise, which is presented in Fig. 11.2. $MinSize = 10$ and $Ratio = 0.7$ are applied. See section 11.4 for a discussion of parameter setting.

11.3 Experimental Result and Analysis

In this section, a series of numerical experiments are performed on synthetic and on real data to explore the efficiency and effectiveness of the fiber clustering approach. All algorithms are implemented in Java and the fiber visualization functions are programmed in Matlab. The calculations have been performed on a workstation with 2.4 GHz CPU and 2.0 GB RAM. The different clustering algorithms are first evaluated with respect to outliers. Finally, an application of hierarchical density-based fiber clustering to special real fiber sets is presented.

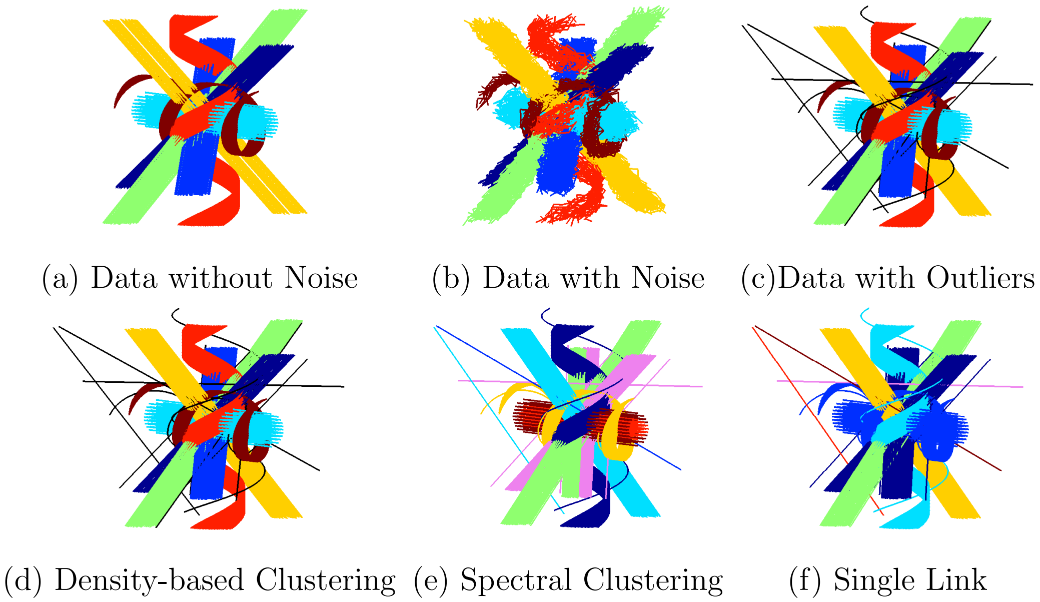
11.3.1 Experiments on Fiber Clustering

In this section, experiments are performed on synthetic data as well as on real brain data using DTW with LB. The clustering method OPTICS is first compared to two different approaches: Spectral clustering (SC) [116] and Hierarchical clustering (Single Link, SL) [177].

Synthetic Data

To evaluate the performance of the different clustering algorithms, experiments are performed on the same data set in Section 10.5.4. Moreover, to test the sensitivity to variations within the fibers, the gaussian noise is added to the fibers. For every point $(x(t), y(t), z(t))$ of a synthetic fiber, the coordinates by $(x(t) + \varepsilon(t),\ y(t) + \varepsilon(t),\ z(t) + \varepsilon(t))$ is modified, where $\varepsilon(t)$ is Gaussian noise (Figure 11.4(b)).

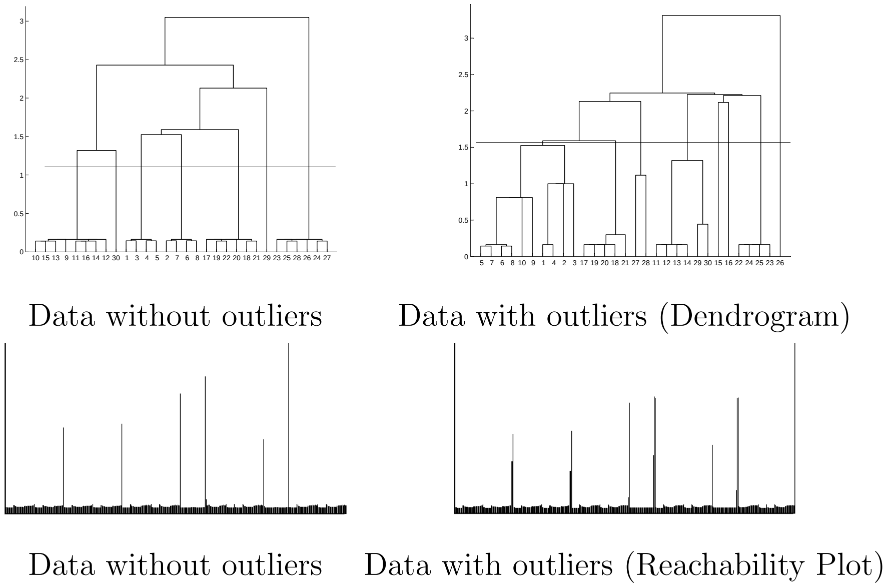
For all three clustering methods the ground truth presented in Figure



(a) Data without Noise (b) Data with Noise (c)Data with Outliers

(d) Density-based Clustering (e) Spectral Clustering (f) Single Link

Figure 11.4: Different fiber clustering results based on the synthetic data.



Data without outliers Data with outliers (Dendrogram)

Data without outliers Data with outliers (Reachability Plot)

Figure 11.5: Dendrograms and reachability Plots for the synthetic data.

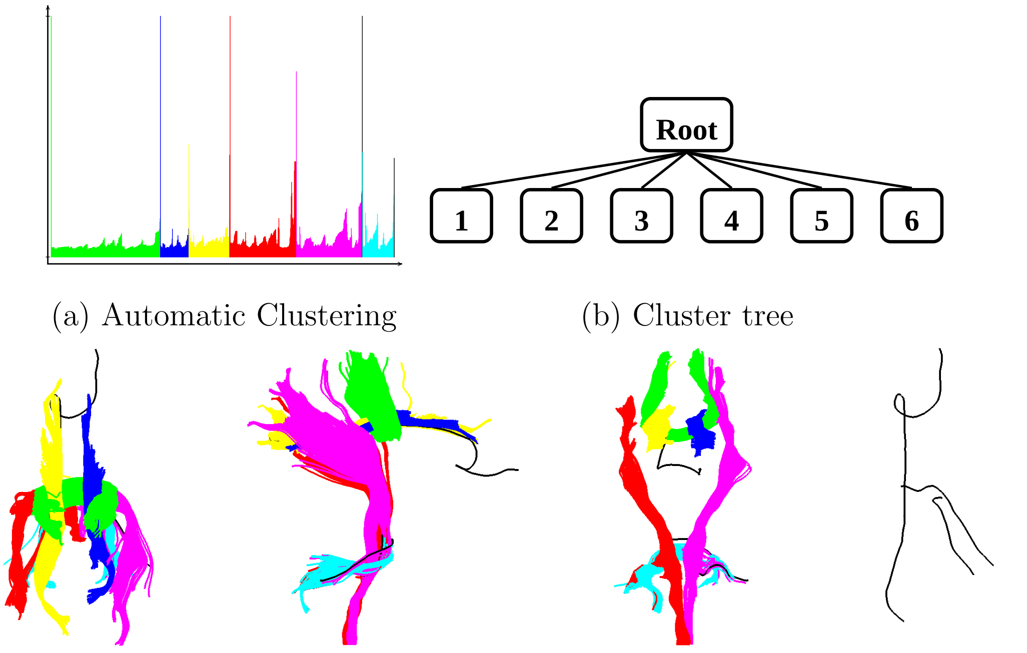
11.4(a-b) is reproduced perfectly. The situation is different for Figure 11.4(c). Only OPTICS can reproduce the ground truth, if the outliers are included, see (Figure 11.4(d-f)). For SC the user has to specify the desired number of clusters (8 clusters are applied in (Figure 11.4(e)). For SL either the perfect split of the dendrogram has to be chosen or, like in SC, the number of clusters must be given (8 clusters are chosen). For the perfect split some information theoretic methods, like Minimal Description Length [68], can be used.

Figure 11.5 shows the dendrograms and reachability plots for the synthetic fibers without (410 fibers) and with the 10 additional noisy fibers. Only the top 30 nodes in the dendrogram are visualized: the total number of layers is 419. It can be seen that the dendrogram is appreciably complicated by adding a few outliers. The reachability plots in both cases, however, are robust to the outliers and already indicate visually the 7 clusters and the outliers.

Applications of Hierarchical Density-based Clustering to Real Data

In this section, OPTICS and automatic clustering are applied on two real data sets with very different clustering structure, to demonstrate the flexibility of the method. For the two real data sets, they are generated according to Section 10.5.2.

Real Data 1: For Figure 11.6, 1768 fibers are seeded from 5 fiducial seed regions, chosen according to [110]. Six big fiber groups are thus created, comprising two branches of the Corticospinal Tracts (red, magenta), part of the middle Cerebellar Peduncle (cyan), two parts of the Cingulum (yellow, blue) and a smaller section of the Corpus Callosum (green). Figure 11.6 (a)



(a) Automatic Clustering (b) Cluster tree

(1a) Axial (1b) Sagittal (1c) Coronal (1d) Axial (Noise)

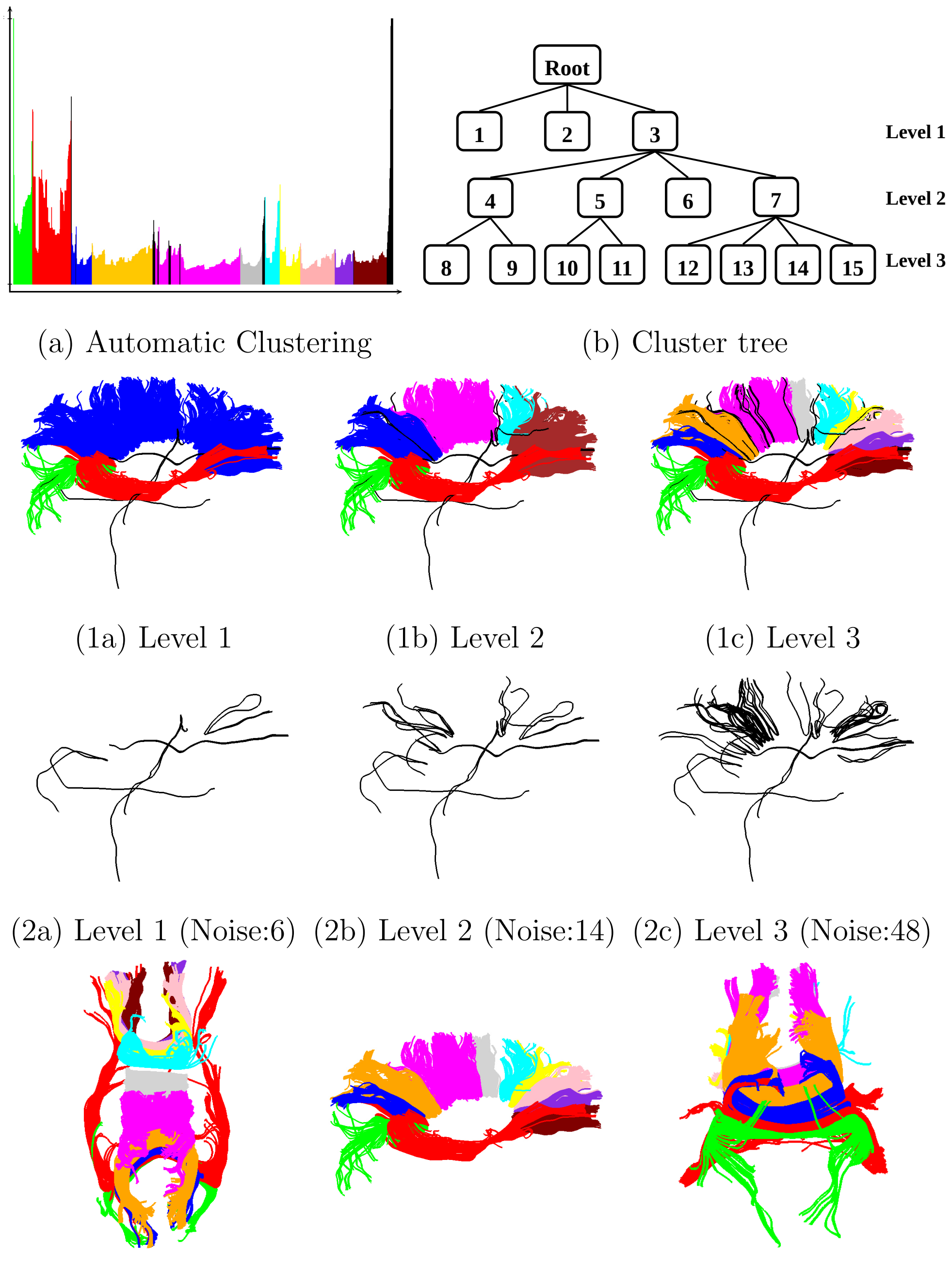
Figure 11.6: Automatic fiber clustering on the real data 1. (a): Automatic clustering of the fibers with parameter $ratio = 0.2, MinSize = 30$ based on the reachablity plot. (b): The fiber cluster structure of the data, where the data are split into 6 fiber bundles; (1a-1c): Fiber clustering results, the different color indicates various fiber bundles and black color means outliers. (1d): The outliers in axial perspective.

shows the reachability plot and its colored segmentation, which corresponds to the cluster structure in Figure 11.6 (b). A low $Ratio = 0.2$ and $MinSize = 30$ are used to achieve a coarse clustering. Finer details of the structures within the clusters should not be detected. Figures 11.6 (1a-1c) present the result in different perspectives. The six clusters are well detected, but some noise is found as well. Noise is presented in Figure 11.6 (1d).

Real Data 2: Figure 11.7 presents a segmentation of the Corpus Callosum. Starting from the mid-sagittal slice of an FA-map within the Corpus Callosum, 1100 fiber tracts are calculated by ROI seeding of the slicer data. The main part of the fibers connects cortical areas in approximate mirror-image sites. A smaller subset is built of commissural trajectories to the Temporal Lobes [110]. Automatic fiber clustering is applied to the reachability plot of these data with $Ratio = 0.7$ and $MinSize = 30$ to detect also subclusters within any hierarchy. The final reachability plot with colored segmentation and the cluster tree are given in Figure 11.7 (a-b). Automatic Tree Clustering creates three levels, as illustrated in Figure 11.7 (1a-1c). Noise fibers are presented in black color. At the first level, only three big clusters are detected. The second level splits up cluster number three (blue color) into four clusters. At the third level, a finer segmentation into nine clusters of similar fibers is performed. Noise is given separately in Figures 11.7 (2a-2c). The final compact clustering without noise is given in Figures 11.7 (3a-3c) in three perspectives.

11.4 Discussion

In this chapter a novel framework is presented for clustering of white matter tracts. Based on the similarity measure between two fibers is introduced in Chapter 11, the robust hierarchical density-based clustering method OPTICS is applied. To the best of knowledge, this is the first application of OPTICS to the fiber clustering problem. The outcome is a reachabiltiy plot, which



(a) Automatic Clustering (b) Cluster tree

(1a) Level 1 (1b) Level 2 (1c) Level 3

(2a) Level 1 (Noise:6) (2b) Level 2 (Noise:14) (2c) Level 3 (Noise:48)

(3a) Axial (3b) Sagittal (3c) Coronal

Figure 11.7: The automatic fiber clustering result for Corpus Callosum. (a):Automatic clustering of fibers with parameters $ratio = 0.7, MinSize = 30$. (b): The Cluster tree of Corpus Callosum. (1a-1c): Clustering results at three levels respectively. (2a-2c): The noise fibers at different levels. (3a-3c): The different views of clustering results at level 3 (without noise fibers)

gives a transparent visualization of the clustering structure of the fibers. The clusters or valleys in the reachability plot are then determined by a novel interactive method and by a new modification of an automatic method, called Tree Clustering. Noise or outlier fibers are detected in a local fashion.

For the evaluation, OPTICS is compared with the clustering method Spectral Clustering and the hierarchical method Single Link . Synthetic examples show no advantage of OPTICS for regular data, but more robustness if outliers are included.

The result of OPTICS is a reachability plot visualizing the clustering structure of the data. This plot must be further analyzed to quantify the location of the clusters. For this purpose we introduce an easy to use interactive software and demonstrate its utility on a realistic fiber set seeded in the Splenium of Corpus Callosum. The main bundles are reasonably detected, also noise fibers are found. The same results are achieved with an automatic method. This method is an adaption of Tree Clustering to the irregular reachability plots based on DTW. To make Tree Clustering more robust, the median is used to calculate the significance of the split points and eliminate noise fibers at every recursive clustering step. Noise or outliers are identified by a criterion defined in density based clustering [8]. The proposed method detects noise in a local fashion, as at every level of the cluster hierarchy noise can be detected. A fiber in a cluster is defined as noise, if its core and reachability distances are above the reachability distance of the corresponding split point. Other concepts of local outliers or noise in density based clustering can be found in [30]. A comparative exploration of the different noise concepts is beyond the scope of this work.

Finally two very different realistic data sets are segmented, to demonstrate the flexibility of our method. For data set 1, including partly the Cingulum (left, right branch), the Corpus Callsoum, the Corticospinal Tracts (left, right branch) and the Cerebellar Peduncle, a coarse clustering is performed. This is achieved by special parameters in Tree Clustering. For $Ratio = 0.2$ the finer details of any hierarchy in the clusters are suppressed and $MinSize = 30$ prevented too small clusters. The cluster tree shows just one level of a flat clustering. Ignoring any hierarchy, the fiber clusters fit to a segmentation into the 6 big groups of the ground truth. For data set 2 the Corpus Callosum is segmented, a hierarchy should be detected. For this purpose the parameters $Ratio = 0.7$ and $MinSize = 30$ are applied and produced three levels in the hierarchical cluster tree. The clustering fits well to the anatomical expectation, looking for groups with similar fibers. At the end 11 clusters are detected.

Also, at every level, noise is detected in data set 2, which is comprised of 1100 fibers. The first level clustering detects 6 noise fibers, this adds up to 14 at the second level. Finally we detect a total of 48 noise fibers or outliers. These numbers reflect, on the one hand, the degree of irregularity in the fiber data set, indicating erroneous fiber tracking due to wrong seed points, experimental noise, or partial volume effects. On the other hand, as mentioned, these numbers may also be influenced by the special definition of noise applied.

The proposed approach of automatic fiber clustering needs 4 parameters to be specified. For the algorithm OPTICS $MinPts$ and eps must be given. For Tree Clustering, $Ratio$ and $MinSize$ are needed.

All calculations in this chapter with OPTICS are performed with the same parametrization: $MinPts = 10$ and $eps = 30$. This insensitivity of OPTICS to its parameters is discussed by Ankerst et al. [8], and greatly simplifies the application of OPTICS. Ankers et al. [8] show that there is a broad range of possible parameters which can be seen in the same clustering structure of the data in the reachability plots. What is important is that *eps* is sufficiently large. The smaller the *eps* is defined, the more objects have an UNDEFINED reachability distance. Consequently, clusters with low density may disappear.

For Tree Clustering, the two parameters are more versatile and their selection should be guided by the user's intention: $MinSize$ constrains the resolution or scale of clustering, preventing very small clusters. *Ratio* determines whether a flat or more hierarchical clustering should be detected. The larger *Ratio*, the more valleys in the reachability plot are accepted as clusters, enabling the detection of the hierarchy within the reachability plot.

11.5 Conclusion

In this chapter, for the first time, the application of hierarchical density-based clustering to the fiber clustering problem is presented. The algorithm OPTICS is applied to calculate reachability plots. These plots present the cluster structure of the fiber data in a transparent manner. A novel interactive and an automatic cluster extraction method are introduced. Experiments on several synthetic and real data sets demonstrate that the proposed methods compare well with existing techniques and show advantages with respect to transparency, flexibility, effectivity, efficiency and outlier robustness.

Chapter 12

Automated Prediction of Very Early Alzheimer's Disease Based on Individual Structural Connectivity Networks

Alzheimer's disease (AD) degrades progressively the grey and white matter of brain. White matter changes reflect changes of brain's structural connectivity pattern. Here, whether individual structural connectivity networks (ISCNs) enable to distinguish patients with prodromal or mild AD from healthy controls (HC) in individual scans? Patients with prodromal AD are characterized by mild cognitive impairment and conversion to AD-related dementia within three years. Diffusion-weighted MRI, diffusion tractography and 96 predefined cortical regions are used to map cortico-cortical ISCNs of 61 individual subjects in a fully automated procedure. For each between-region

connection of ISCN, three attributes (fiber density, fractional anisotropy and mean diffusivity) are extracted to represent its pattern. Afterwards, relying on feature selection criteria, most distinctive connections for the discrimination among HC, prodromal and mild AD are identified. Finally, three typical classifiers including Support Vector Machine, Naïve Bayes, and k-Nearest Neighbor are applied to predict HC, prodromal and mild AD. Patients with mild AD are separated from HC with accuracy above 95% for each attribute, patients with prodromal AD from HC with accuracy more than 90% for each attribute, and patients with mild AD from these with prodromal AD with accuracy of about 86% for fiber density and mean diffusivity. The finding provides evidence that individual cortico-cortical structural connectivity networks are changed in earliest forms of AD. Cortico-cortical ISCNs may be useful as a white matter-based imaging biomarker to distinguish healthy aging from AD.

The remainder of this chapter is organized as follows: A brief introduction is given in Section 12.1. Section 12.2 presents the material and methodology in detail. The results of the experiments are demonstrated in Section 12.3. Discussion of the results and the conclusion of this chapter are presented in Section 12.4. Finally, the supplemental information is provided in Section 12.5. Parts of the material presented in this chapter have been submitted in [149], where J.S., N.M and C.S designed the research; J.S., N.M., C.S., Q.Y., H.F., A.K., C.Z. and A.W. performed the research; J.S., N.H., and Q.Y. analyzed the data and J.S., N.M., C.S., Q.Y., C.M., V.R., and A.W. wrote the paper.

"*Junming Shao, Nick Myers, Qinli Yang, Jing Feng, Claudia*

Plant, Christian Böhm, Hans Förstl, Alexander Kurz, Claus Zimmer, Chun Meng, Valentin Riedl, Afra Wohlschläger, Christian Sorg. Prediction of Alzheimer's disease using individual structural connectivity networks. Submitted for publication."

12.1 Introduction

Alzheimer's disease (AD), the most common cause of age-related dementia, is a neurodegenerative disease characterized by increasing cognitive and behavioral deficits [21]. AD is neuropathologically characterized by amyloid plaques, neurofibrillary tangles and the loss of neurons, with changes starting regionally and spreading out gradually across brain's grey matter [29, 160]. Complementary, post-mortem histological and in-vivo imaging studies demonstrate widespread alterations of patients' white matter [34, 54, 134, 28, 39, 154]. Since brain functions depend critically on the integration of regionally remote neural processes by structural white matter connections, the biological hypothesis of AD as disconnection syndrome suggests that increasing biochemical and micro-structural changes at the level of synapses and cell compartments result in progressive cell death and white matter degeneration with both accompanied by progressive cognitive and behavioral deficits [33, 142, 46, 152, 99]. Based on these findings, it is hypothesized that subject-specific patterns of changed white matter connectivity reflect AD's impact on patients' health status.

Diffusion weighted magnetic resonance imaging (DWI) is a technique that can be used to explore white matter microstructure in-vivo [83]. The

DWI signal is sensitive to the diffusion of water molecules, which is direction dependent in the brain e.g. in a coherent fiber bundle water diffusion is less restricted along the bundle than across it. During DWI, multiple brain images are acquired each sensitive for a distinct direction. At each voxel measured data is fitted to a mathematical diffusion tensor model describing diffusion as an ellipsoid or tensor. Both local properties of water diffusion (such as fractional anisotropy (FA) or mean diffusivity (MD)) and diffusion-based tractography (direction estimates used for the reconstruction of fiber paths) can be derived from the voxel-wise diffusion tensor [109, 96]. DWI-based imaging studies in AD demonstrate aberrant FA and MD values in patients' white matter, involving frontal, occipital, and temporal lobes [28, 59, 153, 143] and selected tracts such as the corpus callosum or cingulum [58, 153, 98, 45, 39, 154, 143]. DWI-based studies in mild cognitive impairment (MCI), which is a high-risk state for AD, demonstrate similarly distributed but less distinctive FA/MD changes in patients [153, 59, 58, 39]. Recently Lo and colleagues [99] use DWI-based tractography to explore explicitly individual fiber-based structural connectivity changes in patients with AD, i.e. this study focuses on subject-specific tracts in contrast to previous studies focusing on regions that are spatially consistent across subjects; they find that the topological organization of structural connectivity networks is altered in AD and these alterations correspond with cognitive deficits of patients.

The objective of the this study is to use DWI-based diffusion tractography to explore subject-specific patterns of individual structural connectivity networks (ISCN) in the spectrum of prodromal and mild AD. Patients with prodromal AD are defined as patients with MCI, which convert to AD-related dementia within 3 years. It addresses two questions: (i) Are ISCNs of patients with prodromal AD changed in comparison to that of healthy controls? (ii) Do ISCNs separate patients with prodromal or mild AD from healthy controls at the individual subject level? For each individual subject, the cortico-cortical ISCN is constructed based on the DWI-based diffusion tractography. ISCN patterns are then compared across subjects using machine-learning-based pattern recognition techniques [127, 90]. Machine-learning-based pattern classifiers enable disease state estimations of individuals by their characteristic training-test procedure; i.e. specific algorithms representing the pattern recognition machine are trained for well-characterized data and then used to categorize new input patterns as members of separate (clinical) groups.

12.2 Materials and Methods

12.2.1 Subjects

17 patients with mild AD (age 68.9 years +/- 8.1, 7 females), 23 patients with prodromal AD (age 67.6 +/- 5.4, 11 females) and 21 healthy controls (HC, age 66.3 +/-7.4, 13 females) participated in this study (see Table 12.1). All participants provided informed consent in accordance with the Human Research Committee guidelines of the Klinikum Rechts der Isar, Technische Universität, Mnchen. Patients were recruited from the Memory Clinic of the Department of Psychiatry, healthy controls by word-of-mouth advertising. Examination of every participant included medical history, neurological ex-

amination, neuropsychological assessment (Consortium to establish a registry for AD, CERAD [112]), structural MRI, and (for patients only) informant interview (Clinical Dementia Rating, CDR [111]) as well as blood tests. Patients with mild AD fulfilled criteria for dementia (CDR global score=1) and the NINCDS-ADRDA criteria for AD [104]. Patients with prodromal AD met criteria for amnestic MCI and converted to mild AD within 3 years. Amnestic MCI criteria included reported and neuropsychologically assessed memory impairments, largely intact activities of daily living, and excluded dementia (CDR=0.5) [62]. Conversion to AD was assessed during annual follow-up clinical assessments after baseline including medical history, neurological examination, informant interview (CDR), and neuropsychological assessment (CERAD). 4 patients converted to AD after one year, 7 after 2, and 12 after 3 years. Exclusion criteria for entry into the study were other neurological, psychiatric, or systemic diseases (e.g., stroke, depression, alcoholism) or clinically remarkable MRI (e.g., stroke lesions) potentially related to cognitive impairment. 21/10 patients/controls were treated for hypertension (Beta-blockers, ACE-inhibitors, and Calcium channel blockers), 14/9 for hypercholesterolemia (statins). 4 patients had diabetes mellitus, 5 patients received antidepressive medication (Mirtazapine, Escitalopram), and all patients with mild AD received cholinesterase inhibitors.

12.2.2 Data Acquisition

A 3T-MRI scanner (Achieva, Philips) together with a pulsed gradient spin-echo echo planar imaging sequence with a parallel imaging (SENSE) factor of 2.5, TE = 60 ms, and TR = 6516 ms was used for DWI. Images were

Table 12.1: Demographical and neuropsychological scores.

Group	HC (n=21)	AD (n=17)	Prodromal AD (n=23)	P
Gender				
Female	13	7	11	0.42
Male	8	10	12	
Age	66.4±7.5	68.9±8.1	67.6±5.4	0.53
MMSE Score	29.4±0.8	22.1±4.3	26.8±2.0	¡0.01
Delayed Recall	6.5±2.1	0.9±1.8	3.1±2.0	¡0.01

Note: AD: Alzheimer's disease; HC: healthy controls; pAD: prodromal AD; MMSE: Mini-Mental-State -Examination; CERAD: Consortium to Establish Registry for AD; p: p-value; for statistical evaluation of group differences χ^2 (gender) and F-Tests (age, MMSE, delayed recall) are used.

acquired for 112 × 112 matrix size of slice and subsequently reconstructed for a 128 × 128 matrix size, with a resolution of 1.75 mm in plane and a slice thickness of 2 mm. A total of 60 contiguous slices were acquired to give complete brain coverage containing 128 × 128 × 60 voxels with size $1.75 \times 1.75 \times 2\ mm^3$. Diffusion gradients were applied in 15 non-collinear directions with $b = 800\ s/mm^2$. B0 image without diffusion weighting, b=0 s/mm^2, was additionally acquired.

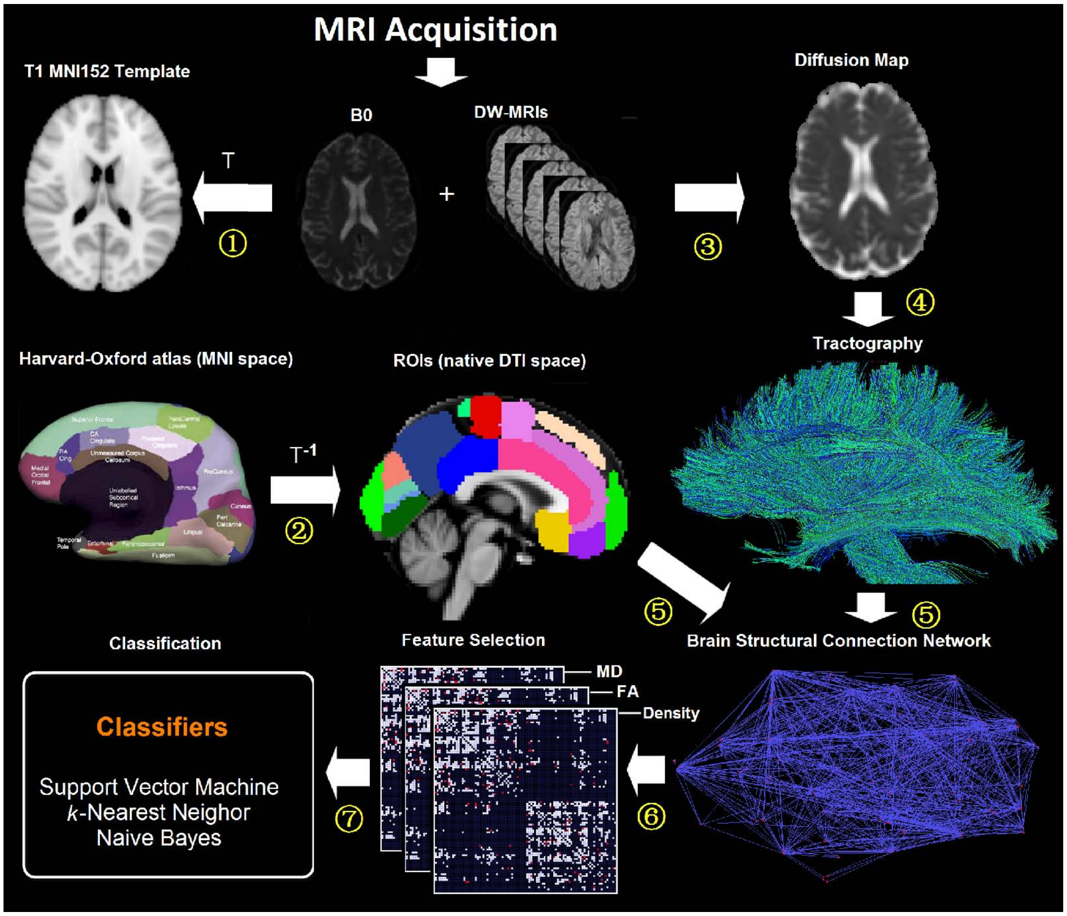


Figure 12.1: The framework of automated prediction of Alzheimer's disease.

12.2.3 Construction of Individual Structural Connectivity Networks (ISCNs)

To construct the structural connectivity network for each participant, the procedure involved the following steps (Figure 12.1 gives the proposed framework of automated prediction of Alzheimer's disease).

Cortical parcellation. This step was to partition the cerebral cortex of each participant into 96 cortical regions of the Harvard-Oxford brain atlas (http://www.mrib.ox.ac.uk/fsl/). Individual's non-diffusion weighted (B0) image was first affine-registered to the ICBM 152 template of Montreal Neurological Institute space (MNI, http://www.bic.mni.mcgill.ca) to obtain the transformation matrix (T). The inverse transformation matrix (T-1) was then applied to the Harvard-Oxford atlas and (B0) image to generate corresponding cortical regions in each individual's diffusion-weighted image native space.

Diffusion tractography. To determine the connectivity between pair-wise cortical regions, diffusion tractography was used to reconstruct the white matter axonal bundle trajectories in the human brain based on fiber tracking algorithms. Distortion induced by motion for weighted MRI images was first corrected by aligning all diffusion-weighted images to the non-diffusion (B0) image using a 2-D linear registration algorithm (AIR, Automated Image Registration) [171]. Maps of the diffusion tensor elements, mean diffusivity (MD), and fractional anisotropy (FA) were calculated for each voxel by using in-house software. Then fiber tracking algorithm TEND [96] was applied to investigate the white matter in the whole brain. Since voxels with high FA were more likely to contain a high proportion of white matter, voxels with an FA value above 0.3 were taken as seed points of fiber tracking. Tracking started from these seed points, using the entire diffusion tensor to deflect the estimated fiber trajectory in both directions. Tracking stopped in voxels with low FA value (<0.2) or physiologically implausible curvature of the track (>60 degrees).

ISCN Construction. Then the output of both cortical parcellation and diffusion tractography was combined to construct individual structural connectivity networks ISCNs for each subject. Each cortical region was regarded as a network node. Connectivity of each pair of nodes was measured by the attribute of fiber across two regions. If there existed at least one fiber with end-points in one pair of regions (e.g. region i and region j), the two cortical regions were assumed to be connected [71, 70]. In order to investigate abnormal ISCNs in AD, three attributes were used to capture the properties of each connection: (a) fiber density was proposed to quantifying the density of white matter in brain regions [71, 70, 132]; it was defined as proportion of all fibers connecting the two cortical regions (n_{ij}) over the total number of fibers of the subject (n_{all}), namely $cd_{ij} = n_{ij}/n_{all}$. (b) FA reflected fractional anisotropy for all voxels located in the connected fibers between two cortical regions. (c) MD reflected mean diffusivity for all voxels located in the connected fibers between two cortical regions. After calculating the three attributes for each connection, each attribute reflected the weighted edge of the network. I.e. finally three ISCNs characterizing different attribute patterns were obtained for each subject.

12.2.4 Pattern Classification of ISCNs

After construction of ISCNs, most distinctive connections among groups were identified by a feature selection criterion. Information Gain criterion [129, 73] was used to rate the information-based interestingness of each connection and attribute for prediction and classification (c.f. refsec:SupplementalInformation). After that, pattern classification algorithms were applied to classify subjects based on selected connections for each ISCN type. In order to evaluate whether potential changes between ISCNs depend on used pattern classifier, three representative classification approaches with very different algorithmic paradigms were selected: Support Vector Machine (SVM), k-Nearest Neighbor (k-NN), and Naïve Bayes (NB).

The basic idea of SVM procedures [163] is to construct a separating hyperplane between the training instances of both classes. Among all possible hyperplanes, that one with the maximum margin between classes was selected [37]. Given training vectors $x_k \in R^n (k = 1, , m)$ in two classes, and a vector of labels $y \in R^m$ such that $y_k \in \{1, -1\}$, then SVM solved a quadratic optimization problem.

$$\min_{\omega,b,\varepsilon} \frac{1}{2}\omega^T\omega + C\sum_{k=1}^{m}\varepsilon_k \tag{12.1}$$

with $y_k(\omega^T\phi(x_k) + b) \geq 1 - \varepsilon_k, \varepsilon_k \geq 0, k = 1, \cdots, m$

where ω is a normal vector, b is a scalar and ε_k are non-negative variables, C is a penalty parameter on the training error, y_k is the class label, and $\phi(\cdot)$ is a map function to transfer the training data into a higher dimensional space. For any testing instance x, the critical decision function (predictor) has then the form.

$$f(x) = sgn(\omega^T\phi(x) + b) \tag{12.2}$$

where $f(x)$ is the prediction function, $sgn(\cdot)$ is a symbol function, ω^T is the permutation of normal vector ω.

The k-NN [7] is a method for classifying objects based on closest training examples in the feature space. It is a typical instance-based learning algorithm where an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common amongst its k nearest neighbours. Given an instance, its k closest neighbours are found in terms of the Pearson correlation coefficient, and then its label value is determined by these k neighbours using the majority vote manner principle. In this study, $k = 6$ was specified for all experiments.

The NB-classifier [51] is a simple probabilistic classifier based on applying Bayes' theorem with strong (i.e. naïve) independence assumptions. A Naïve Bayes classifier assumes that the presence/absence of a particular feature of a class is unrelated to the presence/absence of any other feature. The fundamental idea of Bayesian classification is to classify instances x based on a probability model, which is defined as:

$$NB(x) = \operatorname*{argmin}_{c \in C} p(c|x_1, \cdots, x_n) \tag{12.3}$$

with classes C and feature variables$x_1, \cdots, x_n$ constituting the components of x. Using Bayes' theorem, it can be re-written as:

$$NB(x) = \operatorname*{argmin}_{c \in C} \frac{p(c)p(x_1, \cdots, x_n|c)}{p(x_1, \cdots, x_n)} = \operatorname*{argmin}_{c \in C} p(c)p(x_1, \cdots, x_n|c) \tag{12.4}$$

Since Naïve Bayes assumes that the conditional probabilities of the independent variables are statistically independent, the prediction function of Naïve Bayes is finally defined as:

$$NB(x) = \operatorname*{argmin}_{c \in C} \prod_i p(x_i|c) \tag{12.5}$$

12.3 Results

For each type of ISCN, information gain for distinctive connections among groups was calculated for all possible 4560 inter-regional connections and all possible group comparisons. E.g. regarding fiber-density-based ISCNs and group comparison between healthy controls and patients with mild AD, 53 connections of were selected by information gain above 0 (ranging from 0.13 to 0.41). Figure 12.2 (a)-(c) shows the distinguishing connections (with red points) between the healthy control (HC) group and mild AD group based on the attributes of fiber density, FA and MD, respectively (analog Figure 12.3 and 12.4 for group comparisons HC vs. prodromal AD and prodromal AD vs. mild AD, respectively).

The classification framework was applied with 10-folds cross-validation on the data set to predict the health status of each subject based on the selected distinct structural connections among groups. Table 12.2 summarizes the classification results for all group comparisons.

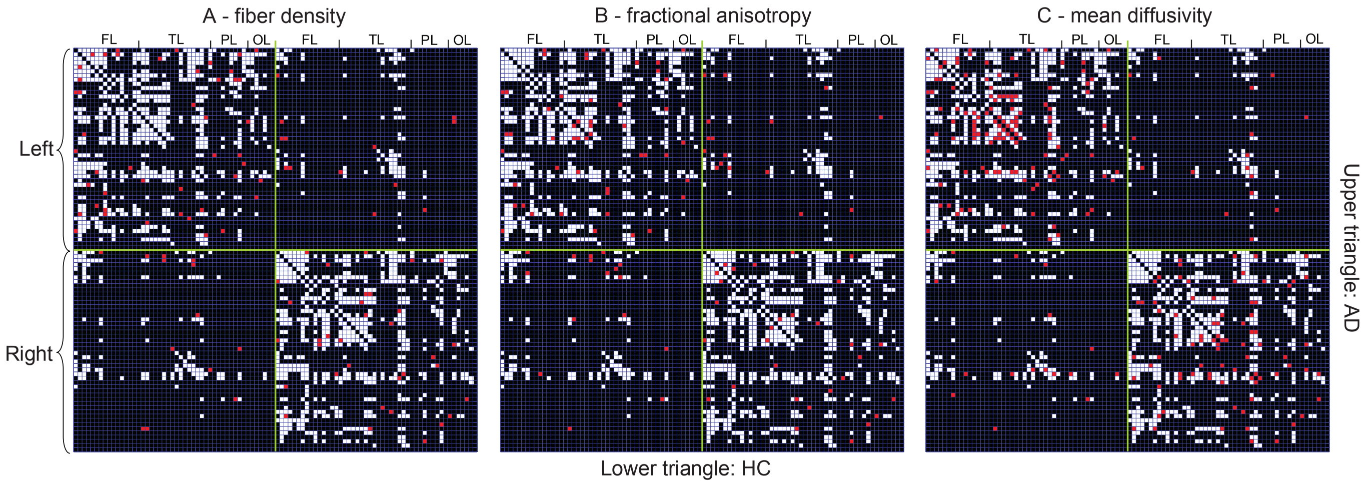


Figure 12.2: Averaged individual structural connectivity networks ISCNs for patients with mild AD and healthy controls. After cortical parcellation and white matter tractography, individual structural connection networks ISCNs are calculated for each individual. ISCNs for fiber density (A), fractional anisotropy (B) and mean diffusivity (C) are then averaged for patients with mild AD (upper triangle) and healthy controls (lower triangle) and represented as structural connectivity matrices: each element of the matrix represents the connection between two cortical regions, white points indicate that there exists at least one connection in more than 50% of group members; red points indicate distinctive connections among groups based on feature selection using Information Gain criterion.

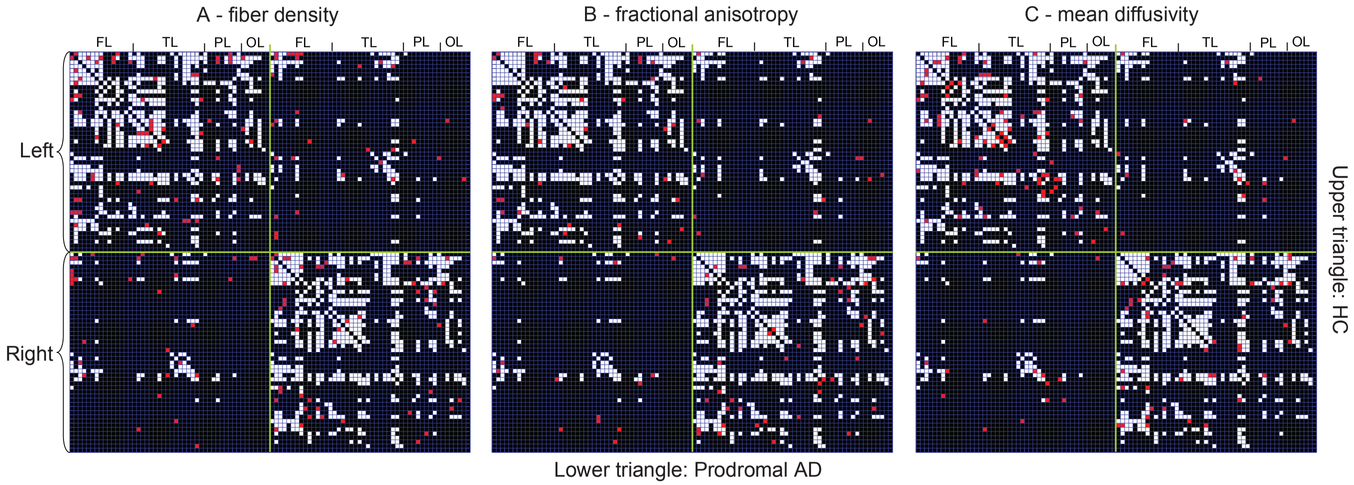


Figure 12.3: Averaged individual structural connectivity networks ISCNs for patients with prodromal AD and healthy controls. ISCNs for fiber density (A), fractional anisotropy (B) and mean diffusivity (C) are averaged for healthy controls (HC, upper triangle) and patients with propromal AD (lower triangle): each element of the connectivity matrix represents the connection between two cortical regions, white points indicate that there exists at least one connection in more than 50% of group members; red points indicate distinctive connections among groups based on feature selection using Information Gain criterion.

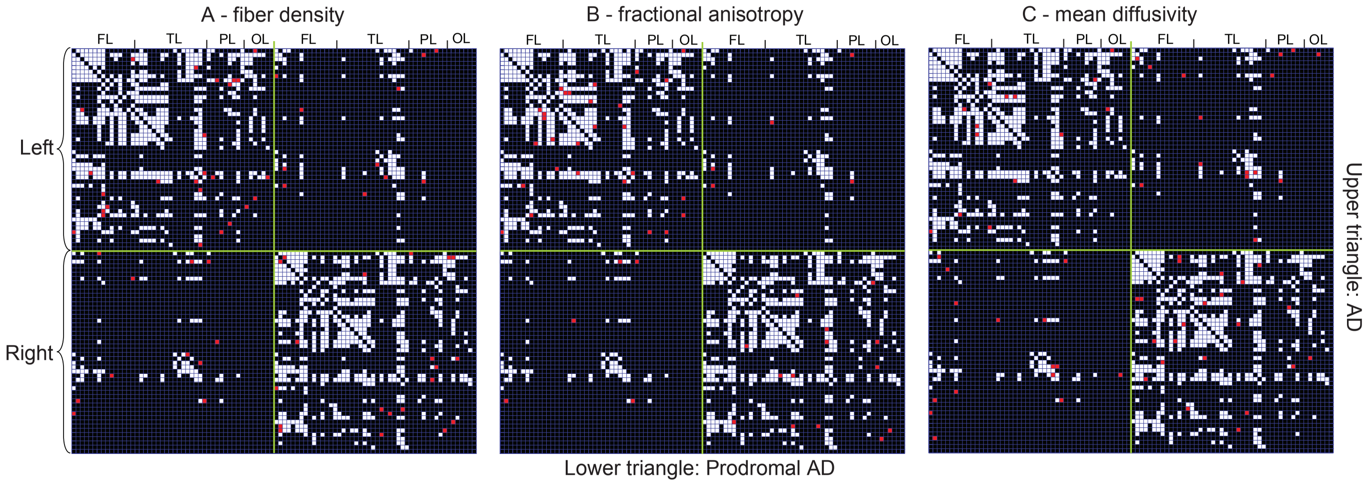


Figure 12.4: Averaged individual structural connectivity networks ISCNs for patients with mild AD and patients with prodromal AD. ISCNs for fiber density (A), fractional anisotropy (B) and mean diffusivity (C) are averaged for patients with mild AD (upper triangle) and patients with prodromal AD (lower triangle): each element of the connectivity matrix represents the connection between two cortical regions, white points indicate that there exists at least one connection in more than 50% of group members; red points indicate distinctive connections among groups based on feature selection using Information Gain criterion.

(i) Mild AD vs. HC: All three classifiers SVM, k-NN and NB distinguished mild AD and HC subjects effectively. Particularly, SVM obtained fully correct predictions of 100% based on the attributes fiber density and MD; regarding FA, SVM obtained classification accuracy of 92.11%. For k-NN classifier, it yielded the same classification accuracy (94.74%) with respect to each attribute. As with SVM, the predictions of NaiveNaïve Bayes were completely correct (100%) with respect to the attributes fiber density and FA.

(ii) Prodromal AD vs. HC: Regarding the prediction of prodromal AD and HC, SVM gained classification accuracy of 97.73% with respect to fiber density. For FA and MD, classification accuracies of SVM were 84.09% and 93.18%, respectively. The NB classifier achieved the classification accuracy of 95.45%, 97.73% and 100% with respect to fiber density, FA and MD. The k-NN classifier yielded accuracies of 81.82%, 88.64% and 86.36% for fiber density, FA and MD.

(iii) Prodromal AD vs. mild AD: Between the patient groups of prodromal AD and mild AD, SVM achieved classification accuracies of 85%, 82.5%, 85% for fiber density, FA and MD respectively. Similarly, k-NN received classification accuracies of 85%, 75%, 82.5% corresponding to fiber density, FA and MD. The NB classifier gained the highest classification accuracies of 95%, 85% and 90%, respectively, for fiber density, FA and MD.

Table 12.2: Accuracy of pattern classification based on ISCNs.

	SVM	k-NN	Naïve Bayes
AD vs. HC			
Density	100.0%	94.7%	100.0%
FA	92.1%	94.7%	100.0%
MD	100.0%	94.7%	89.5%
AD vs. pAD			
Density	85.0%	85.0%	95.0%
FA	82.5%	75.0%	85.0%
MD	85.0%	82.5%	90.0%
pAD vs. HC			
Density	97.7%	81.8%	95.5%
FA	84.1%	88.6%	97.7%
MD	93.2%	86.4%	100.0%

Note: ISCN: individual structural connectivity network; SVM: support vector Machine; k-NN: k-nearest neighbor; AD: Alzheimer's disease; HC: healthy controls; pAD: prodromal AD; FA: fractional anisotropy; MD: mean diffusivity

12.4 Discussion and Conclusion

In current study, diffusion tractography-based individual structural connectivity networks (ISCNs) and machine-learning-based pattern recognition have been used to study white matter changes in prodromal and mild AD. Compared with healthy controls, ISCNs of patients critically changed for fiber density and fiber integrity characterized by FA and MD. ISCNs enabled the predictions of prodromal and mild AD, respectively, with accuracies of about 90% and 95%.

First, the critical methodological issues of the approach were discussed and then the finding of disrupted ISCNs in AD particularly in prodromal AD was focused, finally it gave the discussion on whether ISCNs have the potential to apply as an imaging-based biomarker to separate AD from healthy aging.

Previous DWI-based studies in AD and MCI demonstrated widely distributed changes in the white matter of patients (for review [39, 154, 143]). Sexton and colleagues [143] performed a quantitative meta-analysis of 41 DWI studies in AD/MCI and found that patients with AD were characterized by decreased FA and increased MD in the white matter of all brain lobes (except that of the parietal lobe for FA changes) and in regionally selected white matter tracts. Similarly distributed but less distinctive changes were observed in patients with MCI [143]. These findings suggested progressive, early starting (in pre-dementia stages), widely distributed changes of structural fiber connectivity in AD. However, most of these DWI-based studies did not explicitly account for the individual connectivity network of subject-specific tracts (i.e. usually individual FA/MD images were normalized into a standard space without explicitly considering subject-specific tracts; subsequently across-subject-comparisons of these normalized images were performed for all voxels and then statistically evaluated (e.g. [105], for selected regions see e.g. [153], for restricted tract skeletons see e.g. [42]).

The objective of the current study was to explore AD's increasing effects on white matter structure based on subject-specific, tract-based patterns of white matter connectivity. Since in the proposed approach subject-specific tract-based pattern scores were fully derivable in an automated way (i.e.

without any white matter focused normalization procedure), this approach fulfills basic formal criteria for a biomarker.

(i) Subject-specific diffusion tractography: Diffusion tractography and a predefined cortical atlas were used to realize a completely automated characterization of each individual's ISCN. This approach was based on individual's native space, on specific properties of subject-specific tracts, and it accounted for all white matter relevant for cortico-cortical connections. Lo et al. applied a very similar approach as in the current study to characterize systemic properties of ISCNs in AD [99]. In order to compare patients with controls, these authors derived graph-theory-based topological scores from ISCNs and compared them across groups by the use of univariate statistical methods. Instead of focusing on scalar-based scores reflecting the systemic organization of an ISCN, here it focused on individual ISCN patterns based on high-dimensional feature vectors.

(ii) Pattern-focused approach: Patterns of structural connectivity networks based on DWI strongly varied across healthy individuals (e.g. [70, 48]). This might be partly due to used imaging/tractography methods but likely more due to individual variability [83]. The impact of neurodegenerative diseases such as AD may further increased this variability of individual connectivity. One way to handle this problem of pattern variability across subjects (particularly when being interested in a classification-focused evaluation of "individual" ISCN patterns), was the use of pattern recognition techniques. These techniques evaluated the connectivity pattern as a whole instead of element by element. Here we used canonical machine-learning-based pattern recognition procedures (SVM, k-NN, and Naïve Bayes), which have been applied previously in MRI data analysis [127, 90]. From a group-point-of-view, findings of high accuracy for group separation resulting from machine-learning-based classification indicated that the pattern-of-interest was different across groups. In classical terms this was addressed by the issue of brain reading, i.e. the separation of two brain-states with respect to behavioral differences [113]. This means for this study that ISCNs of healthy controls, patients with prodromal or mild AD were typically different (discussed in detail in the next paragraph). From an individual-point-of-view, machine learning-based pattern classifiers fulfill some requirements of diagnostic procedures [90]; i.e. specific algorithms underlying the classifier were trained on well-characterized data in order to test subsequently new input data against trained sets and then classified them as members of a clinical group (below discussed in detail).

Concerning group differences between healthy controls, prodromal and mild AD, all classification results for different ISCN-pattern types (fiber density, FA, MD) were largely independent from selected classifiers, indicating that results reflected real group differences in ISCN patterns. Furthermore different ISCN features of fiber density, FA and MD produced largely comparable accuracy scores for each possible group comparison, being in line with previous voxel-/ROI-based results of comparable FA and MD changes in AD/MCI and therefore suggesting that AD does not selectively impact on diffusion tensor properties.

(i) Mild AD: Patients with mild AD were separated from healthy controls with an accuracy of more than 95%. Studies using similar pattern recognition methods but instead of white matter grey matter features (such as regional volume) observed comparable separation results [127, 90, 158, 43]. E.g. Klöppel and his colleagues used T1-weighted grey matter images and SVM to separate pathologically verify AD patients from healthy controls with accuracy of 96% [90]. The congruence of this almost perfect separation of AD from healthy controls based once on grey once on white matter features was in line with the disconnection hypothesis of AD. I.e. AD induces progressive neuron degeneration, which is associated with progressive, functionally relevant white matter degeneration [142, 46]. The study of Lo et al., which was to our knowledge the only study using also tractography-based ISCNs in AD, observed changed topological ISCN features in patients [99]. The results of this study complement these findings from a more pattern-focused point of view.

(ii) Prodromal AD: Patients with prodromal AD were separated from healthy controls with accuracy of more than 90%. The result is in line with previous findings of widely distributed FA decreases and MD increases in MCI (see [143] for review). Few studies of MCI patients using similar pattern recognition methods and grey matter features such as tissue density found comparable separation results [158, 43]. E.g. Davatzikos et al. used grey matter density as well as a combination of both high-dimensional image analysis and pattern classification to separate MCI patients from healthy controls with 90% accuracy [43]. Assessed MCI patients of this study converted to AD-related dementia within 3 years, i.e. patients fulfilled criteria of prodromal AD for the time point of scanning. This means that AD alters ISCNs substantially in already pre-dementia stages. Furthermore it is found that groups of prodromal and mild AD were separated by an accuracy of about 86%, suggesting increasing ISCN changes along the course of AD. Future studies have to explore whether the selected tracts of ISCNs are preferentially affected along the course of AD and whether ISCN changes are accompanied by increasing changes of topological ISCN characteristics.

The findings of the current study demonstrated several reasons why ISCNs - at least in principle - may have the potential to realize an imaging- and white matter-based biomarker to distinguish healthy aging from AD. (i) White matter scores of our study, which were used to compare subjects, were directly derived from measurements of the individual subject. They were calculated within the native space, i.e. no normalization procedure was used, and derived in a fully automated way by tractography. The algorithm underlying tractography realized a deterministic streamlining approach allowing for maximal control of single steps of data analysis. (ii) White matter scores of across-subject comparisons were tract-based pattern scores. Fibers and tracts organized brain's white matter. Patterns of individual fiber connectivity were characterized by a large inter-subject variability, which was likely further increased by AD. Therefore scores based on tract-based patterns seemed to be appropriate to reflect AD's impact on white matter. It will be a topic of future studies to decide whether the use of the whole brain connectivity pattern or parts of it (as used in the current study) or combination of both is most suitable to separate patients from healthy controls. (iii) As already mentioned above, machine learning-based pattern classifiers have an appropriate training-test structure, which fulfills criteria of a diagnostic tool. Classification methods used have the potential to separate AD from normal aging already in pre-dementia stages of AD. However, further studies are necessary on whether results of group separation with accuracies of more than 90% are robust for enlarged training samples, different imaging protocols, and measurements at different scanners. (iv) The imaging protocol of the current study was chosen economically, i.e. time saving, with 15 gradients for one measurement and the use of the B0 image for cortical parcellation. Once a training data set is prepared, representing the most time consuming step, tractography, feature network definition and classification are performed within several minutes without any user interaction, i.e. time for the whole procedure is not a practical limiting factor for using ISCNs as biomarkers.

Conclusively, fully automated derivable ISCNs separate healthy aging from prodromal and mild AD with accuracy of 90% and 95%, respectively, indicating ISCN's potential as an imaging-based white matter-focused biomarker for AD.

12.5 Supplemental Information

12.5.1 Anatomical Atlas

For cortical parcellation, the whole cerebral cortex of each participant is segmented into 96 cortical regions via Harvard-Oxford cortical structural atlas (http://www.fmrib.ox.ac.uk/fsl/data/atlas-descriptions.html). This anatomical atlas is a probabilistic population-based atlas; i.e. cortical regions are thresholded in a way that only voxels, which are estimated above 35% probability of being in that structure, are included in the mask. Table S1 illustrates the names of these cortical regions and their corresponding consequential

Table 12.3: Decoding anatomical labels.

ID	Cortical Regions	ID	Cortical Regions
1	Frontal Pole	25	Parahippocampal Gyrus (anterior)
2	Insular Cortex	26	Parahippocampal Gyrus (posterior)
3	Superior Frontal Gyrus	27	Temporal Fusiform Cortex (anterior)
4	Middle Frontal Gyrus	28	Temporal Fusiform Cortex (posterior)
5	Inferior Frontal Gyrus, pars triangularis	29	Temporal Occipital Fusiform Cortex
6	Inferior Frontal Gyrus, pars opercularis	30	Planum Polare
7	Precentral Gyrus	31	Heschl's Gyrus (includes H1 and H2)
8	Frontal Medial Cortex	32	Planum Temporale
9	Juxtapositional Lobule Cortex	33	Postcentral Gyrus
10	Subcallosal Cortex	34	Superior Parietal Lobule
11	Paracingulate Gyrus	35	Supramarginal Gyrus (anterior)
12	Cingulate Gyrus (anterior)	36	Supramarginal Gyrus (posterior)
13	Frontal Orbital Cortex	37	Angular Gyrus
14	Frontal Operculum Cortex	38	Cingulate Gyrus (posterior)
15	Central Opercular Cortex	39	Precuneous Cortex
16	Temporal Pole	40	Cuneal Cortex
17	Superior Temporal Gyrus (anterior)	41	Parietal Operculum Cortex
18	Superior Temporal Gyrus (posterior)	42	Lateral Occipital Cortex (superoir)
19	Middle Temporal Gyrus (anterior)	43	Lateral Occipital Cortex (inferior)
20	Middle Temporal Gyrus (posterior)	44	Intracalcarine Cortex
21	Middle Temporal Gyrus (temporooccipital)	45	Lingual Gyrus
22	Inferior Temporal Gyrus (anterior)	46	Occipital Fusiform Gyrus
23	Inferior Temporal Gyrus (posterior)	47	Supracalcarine Cortex
24	Inferior Temporal Gyrus (temporooccipital)	48	Occipital Pole

Note: Here only cortical regions of left hemisphere are displayed, where IDs (1-15) indicate the Frontal Lobe (FL), IDs (16-32) the Temporal Lobe (TL), IDs (33-41) the Parietal Lobe (PL) and IDs (42-48) the Occipital Lobe (OL).

numbers.

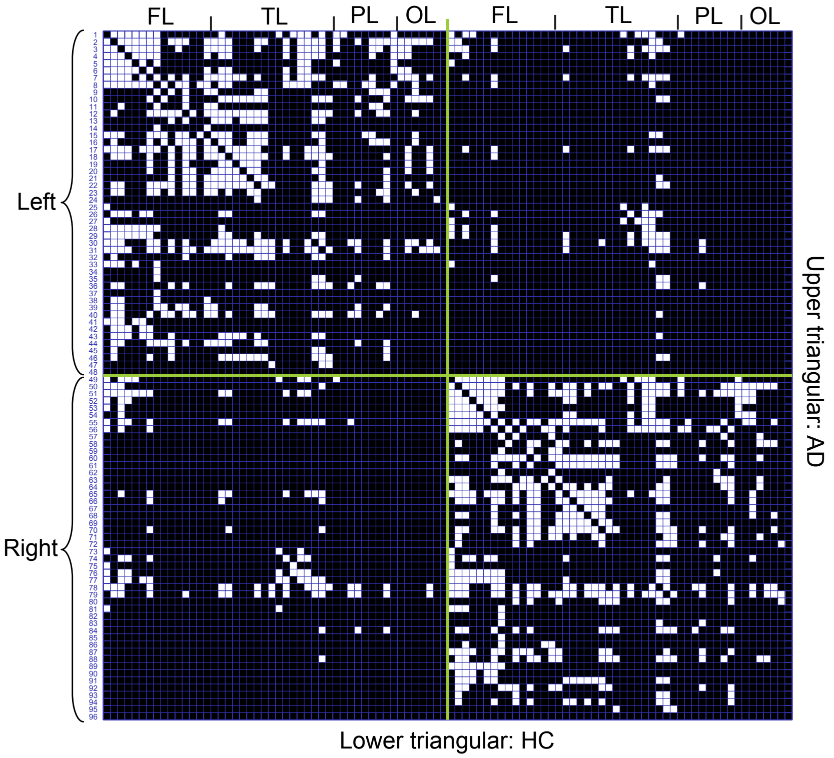


Figure 12.5: Illustration of structural brain connectivity matrix. The consequential numbers on the left of the matrix correspond to the cortical regions in Table S1 respectively. The upper triangular matrix represents the structural connectivity of patients with mild AD while the lower triangular matrix represents structural connectivity of healthy controls. Each element of the matrix represents the connection between two cortical regions. White points indicate there exists at least one connection in more than 50% of members of the group of interest. FL: frontal lobe, TL: temporal lobe, PL: parietal lobe, OL: occipital lobe, HC: healthy controls.

12.5.2 Structural Connectivity Matrices Reflect Across Group Members Averaged ISCNs

After cortical parcellation and white matter tractography, the individual structural connection network ISCN for each individual is obtained. These ISCNs are then averaged for each group and represented as structural connectivity matrices, see Figure 12.5 and Figure 12.3, 12.3, 12.4. Here each element of the matrix represents the connection between two cortical regions. White points indicate there exists at least one connection in more than 50% of members of the group of interest; black points indicate that this is not the case.

12.5.3 Feature Selection

After construction of ISCNs, the most distinctive connections among groups are identified by feature selection. Information Gain criterion [129, 73] is used to rate the information-based interestingness of each connection for prediction and classification. For a given attribute A with respect to the class attribute C, the Information Gain is the reduction in uncertainty about the value of C when A is known. Uncertainty about class C is measured by its entropy $H(C)$, which is defined as:

$$H(C) = \sum_{c \in C} p(c) \log_2 p(c) \tag{12.6}$$

where $p(c)$ denotes the probability of class c. The entropy of the class C

after known the attribute A is given as:

$$H(C|A) = -\sum_{a \in A} \sum_{c \in C} p(c|a) \log_2 p(c|a) \tag{12.7}$$

where $p(c|a)$ is the conditional probability of c for a given a. Since the reduction amount of the entropy of the class after knowing the attribute A reflects the additional information about the class provided by the attribute, Information Gain captures the importance of attribute for the class concept. Formally, the Information Gain $IG(A)$ is defined as:

$$IG(A) = H(C) - H(C|A) \tag{12.8}$$

In order to find most distinguishing connections among different groups, therefore, the power of each connection for group comparison is measured by the information gain. Since three attributes (fiber density, FA and MD) are used to characterize each connection, the distinguishing connections selection with each attribute are considered separately (indicated by red dots of Figure 12.3, 12.3, 12.4).

Part IV

Conclusion

Chapter 13

Summary and Outlook

In this thesis, the preliminaries of data mining are introduced in part I and novel clustering and outlier detection methods are then proposed from the perspective of synchronization in part II. The applications of data mining algorithms in neuroscience databases are further investigated in Part III. In this chapter, the major contributions of this thesis are summarized (cf. Section 13.1) and possible directions for future research are pointed out (cf. Section 13.2).

13.1 Summary

The large amounts of data arising in various fields increases the need for efficient and effective data mining tools to uncover the information in the data. This thesis mainly proposes innovative data mining algorithms (cf. Part II) and apply data mining algorithms in neuroscience databases (cf. Part III). The summary of these contributions is given as follows.

Preliminaries (Part I)

The part of preliminaries provides some background information of this thesis. After a very general introduction to the process of Knowledge Discovery in Databases and Data Mining in Chapter 1, a brief introduction to clustering and outlier detection is illustrated in Chapter 2.

Synchronization-based Data Mining (Part II)

Synchronization is a prevalent phenomenon in nature that a group of events spontaneously come into co-occurrence with a common rhythm through mutual interactions. The mechanism of synchronization allows controlling of complex processes by simple operations based on interactions between objects. Based on this powerful concept, innovative algorithms for data mining are proposed and Junming Shao (J.S.) made the main contribution to all these works consisting of proposing the idea; implementing the algorithms; performing the experiments and writing the papers.

In Chapter 4, the algorithm *Sync* is proposed to clustering inspired by the concept of synchronization. The key idea is to regard each data object as a phase oscillator and interacts dynamically with similar objects by a reformulated Kuramoto model. Through interactions, similar objects tend to synchronize together and form as clusters. Inherited from prosperities of synchronization, the clusters revealed by dynamic synchronization truly reflect the intrinsic structure of the data set. *Sync* does not rely on any distribution assumption and allows detecting clusters of arbitrary number, shape and size. As additional value add, *Sync* allows discovering the clusters without requiring sensitive input parameters in combination with the Minimum Description Length principle (MDL). A preliminary version of this chapter has been published in [27], where J.S. made the main contribution to this work.

In Chapter 5, *hSync* is proposed by extending the *Sync* to hierarchical data analysis. The algorithm *hSync* explores the hierarchical cluster structure from micro-scale to macro-scale by simulating the way to synchronization over different levels of locality. Compared with existing hierarchical clustering algorithms, *hSync* robustly reveals the natural cluster hierarchy regardless of size and data distribution of the clusters. The MDL principle guides *hSync* to generate an interpretable cluster tree only consisting of meaningful levels, each representing a clustering of high quality. Besides the cluster tree, the output of *hSync* includes the locality-quality diagram, a visualization which allows the user to comprehensively assess the quality of the cluster hierarchy over all levels. Part of this chapter has been submitted in [147].

To cure the *curse of dimensionality* in clustering, ORSC (Arbitrarily ORiented Synchronized Clusters), a novel effective and efficient method to subspace clustering is introduced in Chapter 6. ORSC works on the proposed local weighted interaction model, which ensures all objects in a common cluster, which accommodate in arbitrarily oriented subspace, naturally move together. The subspace search is easily transformed to find all synchronized phases in arbitrarily oriented subspaces. In contrast to previous methods, ORSC naturally detects correlation clusters in arbitrarily oriented subspaces, including arbitrarily-shaped non-linear correlation clusters. ORSC is robust against noise points and outliers. ORSC is easy to parameterize, since there is no need to specify the subspace dimensionality, and all interesting subspace clusters can be detected. A preliminary version of this chapter has been published in [150].

In order to reveal the underlying patterns in graphs, a graph partitioning approach *RSGC* is presented in Chapter 7. The key philosophy of *RSGC* is to consider graph clustering as a dynamic process towards synchronization. For each vertex, it is viewed as an oscillator and interacts with other vertices (oscillators) according to the graph connection information. During the process towards synchronization, vertices with similar connectivity patterns will exhibit similar dynamics and thus naturally synchronize together to form a cluster. Compared with other existing graph clustering algorithms, *RSGC* has some major benefits: it provides a novel and natural perspective for graph clustering based on the proposed interaction model, which can capture the characteristics of the real-world networks very well; *RSGC* allows discovering graph clusters with arbitrary size and density; *RSGC* is also robust against noise vertices or outliers; *RSGC* is fully automatic. Part of this chapter has been submitted in [148].

Local Synchronization Factor (LSF) is introduced in Chapter 8. This definition is designed to naturally flagged outlier objects relying on the different dynamic behaviors of objects during the process towards synchronization. The outliers are considered as the perspective: "*Out of Synchronization*". By simulating the dynamical behavior of each data object over time according to an extensive Kuramoto model, regular objects and outliers exhibit different interaction patterns. Interaction plot further provides a detailed view of the dynamical behavior pattern of each object. In contrast to outlier detection algorithms, LSF is intuitive, tight, and distinguishable. A preliminary version of this chapter has been published in [144].

Brain Network Mining (Part III)

It is well known that understanding the connectome of the human brain is a major challenge in neuroscience. In this thesis, the brain structural networks are analyzed involving with the techniques of clustering, feature selection and classification.

Discovering the wiring and the major cables of the brain is essential for a better understanding of brain function. In Chapter 10, a framework to identify meaningful groups of fiber tracks which represent the major cables of the brain is developed. The proposed approach combines ideas from time series mining with density-based clustering to an effective and efficient fiber clustering. A novel fiber similarity measure based on dynamic time warping is proposed, which can captures local similarity among fibers effectively. A lower bound on this fiber warping measure is introduced to speed up computation. Finally, the meaningful fiber bundles in the Human Brain are identified by density-based clustering. Parts of the material presented in this chapter have been published in [145], where J.S., K.H, C.B and C.P designed the research; J.S. implemented algorithms; J.S. and K.H. performed the experiments; J.S., K.H., and Q.Y. analyzed the data; and J.S., K.H, Q.Y. C.B. A.M, N.M and C.P. wrote the paper.

In Chapter 11, the enhancement and variation of this approach is discussed allowing for a more robust, efficient, or effective way to find hierarchies of fiber bundles. With the reachability plots, the novel interactive and an automatic cluster extraction method are introduced. The proposed methods compare well with existing techniques and show advantages with respect to transparency, flexibility, effectiveness, efficiency and outlier robustness. The framework is evaluated successfully on several synthetic and real data sets. Parts of the material presented in this chapter have been published in [146], where J.S., K.H, C.B and C.P designed the research; J.S. implemented algorithms; J.S. and K.H. performed the experiments; J.S., K.H., and Q.Y. analyzed the data; and J.S., K.H, Q.Y. C.B. A.M, N.M and C.P. wrote the paper.

As Alzheimer's disease (AD) degrades progressively the white matter of brain , Chapter 12 explores whether individual structural connectivity networks (ISCNs) enable to distinguish patients with prodromal or mild AD from healthy controls (HC) in individual scans. Therefore, Diffusion-weighted MRI, diffusion tractography and 96 predefined cortical regions are used to map cortico-cortical ISCNs of 61 individual subjects in a fully automated procedure. For each between-region connection of ISCN, three attributes (fiber density, fractional anisotropy and mean diffusivity) are extracted to represent its pattern. Afterwards, relying on feature selection criteria, most distinctive connections for the discrimination among HC, prodromal and mild AD are identified. Finally, three typical classifiers including Support Vector Machine, Naive Bayes, and k-Nearest Neighbor are applied to predict HC, prodromal and mild AD. The experiments indicate that individual cortico-cortical structural connectivity networks are changed in earliest forms of AD. Cortico-cortical ISCNs may be useful as a white matter-based imaging biomarker to distinguish healthy aging from AD. A preliminary version of this chapter have been submitted in [149], where J.S., N.M and C.S designed the research; J.S., N.M, C.S., Q.Y., H.F., A.K., C.Z. and A.W. performed the research; J.S., N.H., and Q.Y. analyzed the data and J.S., N.M, C.S., Q.Y., C.M., V.R., and A.W. wrote the paper.

13.2 Outlook

The last section of this thesis points out some major directions for future research. In particular, for synchronization-based on data mining, future research could be explored in the following directions:

General model: More general model to understand the basic mechanism of synchronization should be investigated. In this thesis, Kuramoto model is analyzed and it is reformulated in different ways to the field of data mining. However, a more general, robust model is needed. For instance, the coupling function should not be restricted to sine function. Many other functions should be further considered.

Synchronization in stream data: This thesis focuses on synchronization-based data mining on static data sets. How to use the synchronization principle to stream data is an interesting direction.

Visualization: During the process towards synchronization, objects exhibit different dynamic behaviors. It is very interested in visualizing the clustering process and find clusters or patterns in a interactive way.

Application: It is well known that synchronization is rooted in human life from the metabolic processes in our cells to the highest cognitive tasks we perform as a group of individuals. For instance, synchronization seems to be a central mechanism for neuronal information processing. Therefore, using the synchronization principle to explore the brain networks is very promising.

Bibliography

[1] J. Aach and G. Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17:495–508, 2001.

[2] J. A. Acebron, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler. The kuramoto model: A simple paradigm for synchronization phenomena. *Rev. of Modern Physics*, 77(2):137–185, Jan. 2005.

[3] D. Aeyels and F. D. Smet. A mathematical model for the dynamics of clustering. *Physica D: Nonlinear Phenomena*, 273(19):2517–2530, 2008.

[4] C. C. Aggarwal, C. M. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park. Fast algorithms for projected clustering. In *SIGMOD Conference*, pages 61–72, 1999.

[5] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In *SIGMOD Conference*, pages 70–81, 2000.

[6] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD Conf.*, pages 94–105, 1998.

[7] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Mach. Learning*, 6:37–66, 1991.

[8] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *SIGMOD Conf*, pages 49–60, 1999.

[9] E. V. Appleton. The automatic synchronization of triode oscillator. In *Proc. Cambridge Phil. Soc. (Math. and Phys. Sci.)*, pages 231–248, 1922.

[10] A. Arenas, A. Diaz-Guilera, J. Kurths, Y. Moreno, and C. Zhou. Synchronization in complex networks. *Phys. Rep.*, 469:93–153, 2008.

[11] A. Arenas, A. Diaz-Guilera, and C. J. Perez-Vicente. Plasticity and learning in a network of coupled phase oscillators. *Phys. Rev. Lett.*, 96:7, 2006.

[12] V. Arnett and T. Lewis. *Outliers in Statistical Data.* John Wiley, 1994.

[13] M. Ashburner, C. A. Ball, J. A. Blake, and et al. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat. Genet.*, 25:25–29, 2000.

[14] F. Bach and M. Jordan. Learning spectral clustering. In *Proc. of NIPS.* MIT Press, 2004.

[15] P. Basser. Inferring microstructural features and the physiological state of tissues from diffusion-weighted images. *NMR in Biomedicine*, 8, 1995.

[16] P. Basser, S. Pajevic, C. Pierpaoli, J. Duda, and A. Aldroubi. In vivo fiber tractography using dt-mri data. *Magn. Reson. Med.*, 44:625–632, 2000.

[17] P. Basser and C. Pierpaoli. Microstructural and physiological features of tissues elucidated by quantitative-diffusion-tensor mri. *J. Magn. Reson. Ser. B*, 111:209–219, 1996.

[18] T. Behrens, H. Berg, S. Jbabdi, M. Rushworth, and M. Woolrich. Probabilistic diffusion tractography with multiple fibre orientations: what can we gain? *NeuroImage*, 34:144–155, 2007.

[19] I. Ben-Gal. *Outlier detection. Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers.* Kluwer Academic Publishers, 2005.

[20] D. L. Bihan. Looking into the functional architecture of the brain with diffusion mri. *Nature Reviews Neuroscience*, 4(6):469–480, 2003.

[21] K. Blennow, M. de Leon, and H. Zetterberg. Alzheimer's disease. *Lancet*, 368:387–403, 2006.

[22] C. Böhm, C. Faloutsos, J.-Y. Pan, and C. Plant. Robust information-theoretic clustering. In *KDD Conference*, pages 65–75, 2006.

[23] C. Böhm, C. Faloutsos, and C. Plant. Outlier-robust clustering using independent components. In *SIGMOD Conf.*, pages 185–198, 2008.

[24] C. Böhm, K. Haegler, N. S. Müller, and C. Plant. Coco: coding cost for parameter-free outlier detection. In *KDD*, pages 149–158, 2009.

[25] C. Böhm, K. Kailing, P. Kröger, and A. Zimek. Computing clusters of correlation connected objects. In *SIGMOD Conference*, pages 455–466, 2004.

[26] C. Böhm and C. Plant. Hissclu: a hierarchical density-based method for semi-supervised clustering. In *EDBT*, pages 440–451, 2008.

[27] C. Böhm, C. Plant, J. Shao, and Q. Yang. Clustering by synchronization. In *KDD*, pages 583–592, 2010.

[28] M. Bozzali, A. Falini, M. Franceschi, M. Cercignani, M. Zuffi, and G. Scotti. White matter damage in alzheimer's disease assessed in vivo using diffusion tensor magnetic resonance imaging. *Journal of Neurology, Neurosurgery and Psychiatry*, 72:742–746, 2002.

[29] H. Braak and E. Braak. Neuropathological stageing of alzheimer-related changes. *Acta Neuropathol (Berl)*, 82:239–59, 1991.

[30] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *SIGMOD Conference*, pages 93–104, 2000.

[31] S. Brohee, K. Faust, G. Lima-Mendez, G. Vanderstocken, and J. van Helden. Network analysis tools: from biological networks to clusters and pathways. *Nat. Protoc.*, 3:1616–1629, 2008.

[32] A. Brun, M. Bjornemo, R. Kikinis, and C. Westin. White matter tractography using sequential importance sampling. In *In Proc. ISMRM Annual Meeting*, page 1131, 2002.

[33] A. Brun and E. Englund. Brain changes in dementia of alzheimer's type relevant to new imaging diagnostic methods. *Prog Neuropsychopharmacol Biol Psychiatry*, 10:297–308, 1986.

[34] A. Brun and E. Englund. A white matter disorder in dementia of the alzheimer type: A patho-anatomical study. *Ann. Neurol*, 19:253–262, 1986.

[35] A. Brun, H. Park, H. J. Knutsson, and C. F. Westin. Coloring of dt-mri fiber traces using laplacian eigenmaps. In *proc. of EUROCAST 2003*, pages 564–572, 2003.

[36] M. Catani, R. J. Howard, S. Pajevic, and D. K. Jones. Virtual in vivo interactive dissection of white matter fasciculi in the human brainvirtual in vivo interactive dissection of white matter fasciculi in the human brain. *NeuroImage*, 17:77–94, 2002.

[37] Y. W. Chen and C. J. Lin. Combining svms with various feature selection strategies. *In: Guyon I, Gunn S, Nikravesh M, Zadeh L, editors. Feature extraction, foundations and applications, Springer*, 15:315–324, 2006.

[38] C. H. Cheng, A. W.-C. Fu, and Y. Zhang. Entropy-based subspace clustering for mining numerical data. In *KDD*, pages 84–93, 1999.

[39] T. Chua, W. Wen, M. Slavin, and P. Sachdev. Diffusion tensor imaging in mild cognitive impairment and alzheimer's disease: a review. *Current Opinion in Neurology*, 21:83–92, 2008.

[40] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Patt. Analy. Mach. Intell*, 24(5):603–619, 2002.

[41] I. Corouge, S. Gouttard, and G. Gerig. Towards a shape model of white matter fiber bundles using diffusion tensor mri. *IEEE Int. Symp. on Biomedical Imaging*, pages 344–347, 2004.

[42] J. S. Damoiseaux, S. M. Smith, M. P. Witter, E. J. Sanz-Arigita, F. Barkhof, P. Scheltens, C. J. Stam, M. Zarei, and S. A. Rombouts. White matter tract integrity in aging and alzheimer's disease. *Human Brain Mapping*, 30:1051–1059, 2009.

[43] C. Davatzikos, Y. Fan, X. Wu, D. Shen, and S. M. Resnick. Detection of prodromal alzheimer's disease via pattern classification of magnetic resonance imaging. *Neurobiol Aging*, 29:514–523, 2008.

[44] C. Deepayan, P. Spiros, M. D. S., and C. Faloutsos. Fully automatic cross-associations. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 79–88, New York, NY, USA, 2004.

[45] L. Delano-Wood, N. Abeles, J. M. Sacco, C. E. Wierenga, N. R. Horne, and A. Bozoki. Regional white matter pathology in mild cognitive impairment: differential influence of lesion type on neuropsychological functioning. *Stroke*, 39:794–799, 2008.

[46] X. Delbeuck, M. V. der Linden, and F. Collette. Alzheimer's disease as adisconnection syndrome? *Neuropsychol Rev*, 13:79–92, 2003.

[47] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39:1–31, 1977.

[48] M. P. V. den Heuvel, R. C. W. Mandl, R. S. Kahn, and H. E. H. Pol. Functionally linked resting state networks reflect underlying structural connectivity architecture of the human brain. *Human Brain Mapping*, 30:3127–3141, 2009.

[49] Z. Ding, J. Gore, and A. Anderson. Classification and quantification of neuronal fiber pathways using diffusion tensor mri. *Magnetic Resonance in Medicine*, 49:716–721, 2003.

[50] B. E. Dom. An information-theoretic external cluster-validity measure. *Technical Report RJ10219, IBM*, 2001.

[51] P. Domingos and M. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.

[52] W. H. Eccles and J. H. Vincent. *British Patent Spec. clxiii p.462*. 1920.

[53] A. Einstein. *Investigations on the Theory of Brownian Movement*. Dover, New York, 1956.

[54] E. Englund. Neuropathology of white matter changes in alzheimer's disease and vascular dementia. *Dementia and Geriatrric Cognitive Disorders*, 9:6–12, 1998.

[55] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD Conf.*, pages 226–231, 1996.

[56] C. Faloutsos and K. Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *SIGMOD Conf.*, pages 163–174, 1995.

[57] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining: Towards a unifying framework. In *KDD*, pages 82–88, 1996.

[58] A. Fellgiebel, M. J. Müller, P. Wille, A. Scheurich, L. G. Schmidt, and P. Stoeter. Colour-coded diffusion-tensor-imaging of cingulate white matter in mild cognitive impairment. *Neurobiol Aging*, 26:1193–1198, 2005.

[59] A. Fellgiebel, P. Wille, M. J. Müller, A. Scheurich, G. Vucurevic, L. G. Schmidt, and P. Stoeter. Ultrastructural and hippocampal white matter lesions in patients with mild cognitive impairment: a diffusion tensor imaging study. *Dement Geriatr Cogn Disord*, 18:101–108, 2004.

[60] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[61] O. Friman, G. Farneback, and C. Westin. A bayesian approach for stochastic white matter tractography. *IEEE Trans. Med. Imag.*, 25(8):965–978, 2006.

[62] S. Gauthier, B. Reisberg, M. Zaudig, R. C. Petersen, K. Ritchie, K. Broich, and S. Belleville. Mild cognitive impairment. *Lancet*, 367:1262–1270, 2006.

[63] G. Gerig, S. Gouttard, and I. Corouge. Analysis of brain white matter via fiber tract modeling. In *Proc. IEEE Int. Conf. EMBS*, pages 4421–4424, 2004.

[64] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–7826, 2002.

[65] C. Gossl, L. Fahrmeir, B. Putz, L. Auer, and D. Auer. Fiber tracking from dti using linear state space models: detectability of the pyramidal tract. *NeuroImage*, 16:378–388, 2002.

[66] C. M. Gray and W. Singer. Stimulus specific neuronal oscillations in the cat visual cortex: A cortical functional unit. 1987.

[67] F. E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11:1–21, 1969.

[68] P. Gruenwald. A tutorial introduction to the minimum description length principle. In *Advances in Minimum Description Length: Theory and Applications*, pages 3–81, 2005.

[69] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. In *SIGMOD Conference*, pages 73–84, 1998.

[70] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J. Wedeen, and O. Sporns. Mapping the structural core of human cerebral cortex. *PLoS Biology*, 6:e159, 2008.

[71] P. Hagmann, M. Kurant, X. Gigandet, P. Thiran, V. J. Wedeen, R. Meuli, and J. P. Thiran. Mapping human whole-brain structural networks with diffusion mri. *PLoS ONE*, 2:e597, 2007.

[72] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J. Intell. Inf. Syst.*, 17(2):107–145, 2001.

[73] M. Hall and J. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Engineering*, 15:1437–1447, 2003.

[74] G. Hamerly and C. Elkan. Learning the k in k-means. In *Proceedings of NIPS*, 2003.

[75] J. Han and M. Kamber. *Data Mining: Concepts and Techniques.* Academic Press, 2001.

[76] D. Hawkins. *Identification of Outliers.* Chapman and Hall, London, 1980.

[77] C. Hennig and B. Hausdorf. Design of dissimilarity measures: A new dissimilarity measure between species distribution ranges. In *Data Science and Classification, Studies in Classification, Data Analysis, and Knowledge Organization, SpringerVerlag GmbH, Berlin, Germany*, pages 29–38, 2006.

[78] A. Hinneburg and D. A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Proc. 4th Int. Conf. on Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[79] A. Hinneburg and D. A. Keim. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *VLDB*, pages 506–517, 1999.

[80] C. Huygens. *Horologium Oscillatorium.* (Apud F. Muguet Parisiis, France; English translation: 1986, Iowa State University Press, Ames), 1673.

[81] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Trans. Acoustics, Speech, and Signal Proc.*, 23:52–72, 1975.

[82] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice-Hall, 1988.

[83] H. Johansen-Berg and M. F. Rushworth. Using diffusion imaging to study human connectional anatomy. *Annu Rev Neurosci*, 32:75–94, 2009.

[84] M. C. Jones, J. S. Marron, and S. J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of American Statistical Association*, 91(433):401–407, March 1996.

[85] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, 1998.

[86] L. Kaufmann and P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 1990.

[87] E. Keogh. Exact indexing of dynamic time warping. In *Proc. of the 28th Int. Conf. on Very Large Data Bases*, pages 406–417, 2002.

[88] E. Keogh and M. Pazzani. Derivative dynamic time warping. In *Proc. of the First SIAM Int. Conf. on Data Mining*, 2001.

[89] C. S. Kim, C. S. Bae, and H. J. Tcha. A phase synchronization clustering algorithm for identifying interesting groups of genes from cell cycle expression data. *BMC Bioinformatics*, 9(56), 2008.

[90] S. Klöppel, C. M. Stonnington, C. Chu, B. Draganski, R. I. Scahill, and J. D. Rohrer. Automatic classification of mr scans in alzheimer's disease. *Brain*, 131:681–689, 2008.

[91] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB*, pages 392–403, 1998.

[92] E. M. Knorr and R. T. Ng. Finding intensional knowledge of distance-based outliers. In *VLDB*, pages 211–222, 1999.

[93] P. Kröger, H.-P. Kriegel, and K. Kailing. Density-connected subspace clustering for high-dimensional data. In *SDM*, pages 246–257, 2004.

[94] Y. Kuramoto. In *Proceedings of the International Symposium on Mathematical Problems in Theoretical Physics,Lecture Notes in Physics*, pages 420–422. edited by H. Araki (Springer, New York,USA), 1975.

[95] Y. Kuramoto. *Chemical oscillations, waves, and turbulence*. Springer-Verlag, New York,USA, 1984.

[96] M. Lazar, D. Weinstein, J. Tsuruda, K. Hasan, K. Arfanakis, E. Meyerand, B. Badie, H. Rowley, V. Haughton, A. Field, B. Witwer, and A. Alexander. White matter tractography using tensor deflection. *Ann. Neurol.*, 18:306–321, 2003.

[97] J. Liu and W. Wang. Op-cluster: Clustering by tendency in high dimensional space. In *ICDM*, pages 187–194, 2003.

[98] Y. Liu, G. Spulber, K. K. Lehtimäki, M. Könönen, I. Hallikainen, and H. Gröhn. Diffusion tensor imaging and tract-based spatial statistics in alzheimer's disease and mild cognitive impairment. *Neurobiol Aging*, 32(9):1558–1571, 2011.

[99] C. Y. Lo, P. N. Wang, K. H. Chou, J. Wang, Y. He, and C. P. Lin. Diffusion tensor tractography reveals abnormal topological organization in structural cortical networks in alzheimer's disease. *J Neurosci*, 30:16876–16885, 2010.

[100] L.ODonnell, M. Kubicki, M. E. Shenton, W. E. L. Grimson, and C.-F. Westin. A method for clustering white matter fiber tracts. *American Journal of Neuroradiology*, 27(5):1032–1036, 2006.

[101] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of 5-th Berkeley Symposium on Math. Stat. and Prob.*, pages 81–297. University of California Press, 1967.

[102] M. Maddah, W. Grimson, and S. Warfield. Statistical modeling and em clustering of white matter fiber tracts. In *IEEE Int. Symp. on Biomedical Imaging*, pages 53–56, 2006.

[103] M. Maddah, A. Mewes, S. Haker, W. E. L. Grimson, and S. Warfield. Automated atlas-based clustering of white matter fiber tracts from dtmri. In *Int. Conf. on Medical Image Computing and Computer Assisted Intervention*, pages 188–195, 2005.

[104] G. McKhann, D. Drachman, M. Folstein, R. Katzman, D. Price, and E. M. Stadlan. Clinical diagnosis of alzheimer's disease: report of the nincds-adrda work group under the auspices of department of health and human services task force on alzheimer's disease. *Neurology*, 34:939–944, 1984.

[105] D. Medina, L. DeToledo-Morrell, F. Urresta, J. D. Gabrieli, M. Moseley, and D. Fleischman. White matter changes in mild cognitive impairment and ad: A diffusion tensor imaging study. *Neurobiol Aging*, 27:663–672, 2006.

[106] B. Moberts, A. Vilanova, and J. J. van Wijk. Evaluation of fiber clustering methods for diffusion tensor imaging. In *Proc. IEEE Visualization*, pages 65–72, 2005.

[107] S. Mori. *Introduction to Diffusion Tensor Imaging*. Elsevier, New York, 2007.

[108] S. Mori, B. Crain, V. Chacko, and P. van Zijl. Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging. *Ann. Neurol.*, pages 265–269, 1999.

[109] S. Mori, B. J. Crain, V. P. Chacko, and P. van Zijl. Three-dimensional tracking of axonal projections in the brain by magnetic resonance imaging. *Ann. Neurol.*, 45:265–269, 1999.

[110] S. Mori, S. Wakana, L. M. Nagae-Poetscher, and P. C. M. Zijl. *MRI Atlas of Human White Matter*. Elsevier, 2005.

[111] J. C. Morris. The clinical dementia rating (cdr): current version and scoring rules. *Neurology*, 43:2412–2414, 1993.

[112] J. C. Morris, A. Heyman, R. C. Mohs, J. P. Hughes, G. van Belle, and G. Fillenbaum. The consortium to establish a registry for alzheimer's disease (cerad). part i. clinical and neuropsychological assessment of alzheimer's disease. *Neurology*, 39:1159–1165, 1989.

[113] J. Mourao-Miranda, A. L. Bokde, C. Born, H. Hampel, and M. Stetter. Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional mri data. *Neuroimage*, 28:980–995, 2005.

[114] N. Mueller, K. Haegler, J. Shao, C. Plant, and C. Böhm. Weighted graph compression for parameter-free clustering with pacco. In *2011 SIAM International Conference on Data Mining*, pages 932–943, 2011.

[115] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *Comput. J.*, 26(4):354–359, 1983.

[116] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856, 2001.

[117] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *Proceedings of VLDB Conf.*, pages 144–155. Santiago, Chile, Sept. 1994.

[118] L. ODonnell, M. Kubicki, M. E. Shenton, W. E. L. Grimson, and C.-F. Westin. A method for clustering white matter fiber tracts. *American Journal of Neuroradiology*, 27(5):1032–1036, 2006.

[119] L. J. ODonnell and C. F. Westin. Automatic tractography segmentation using a highdimensional white matter atlas. *IEEE Trans. Medical Imaging*, 26(11):1562–1575, 2007.

[120] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *ICDE*, pages 315–340, 2003.

[121] H. J. Park, C. F. Westin, M. Kubicki, S. E. Maier, M. Niznikiewicz, A. Baer, M. Frumin, R. Kikinis, F. A. Jolesz, R. W. McCarley, and M. E. Shenton. White matter hemisphere asymmetries in healthy subjects and in schizophrenia: A diffusion tensor mri study. *NeuroImage*, 24:213–223, 2004.

[122] G. Parker and D. Alexander. Probabilistic monte carlo based mapping of cerebral connections utilising whole-brain crossing fibre information.

In *In: Proc.Information Processing in Medical Imaging*, pages 684–695, 2003.

[123] G. Parker, C. Wheeler-Kingshott, and G. Barker. Estimating distributed anatomical connectivity using fast marching methods and diffusion tensor imaging. *IEEE Trans. Med. Imag.*, 21(5):505–512, 2002.

[124] M. PaulWand and M. C. Jones. *Kernel Smoothing.* Chapman and Hall/CRC, London, 1995.

[125] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the 17th ICML*, pages 727–734, 2000.

[126] A. Pikovsky, M. Rosenblum, and J. Kurths. *Synchronization. A Universal Concept in Nonlinear Sciences.* Cambridge Univ. Press, Cambridge, 2001.

[127] C. Plant, S. J. Teipel, A. Oswald, C. Böhm, T. Meindl, and J. Mourao-Miranda. Automated detection of brain atrophy patterns based on mri for the prediction of alzheimer's disease. *Neuroimage*, 50:162–174, 2010.

[128] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *SIGMOD Conference*, pages 418–427, 2002.

[129] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[130] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *SIGMOD Conference*, pages 427–438, 2000.

[131] J. Rayleigh. *The Theory of Sound.* Dover Publishers, New York, 1945.

[132] T. P. Roberts, F. Liu, A. Kassner, S. Mori, and A. Guha. Fiber density index correlates with reduced fractional anisotropy in white matter of patients with glioblastoma. *Am J Neuroradiol*, 26:2183–2186, 2005.

[133] P. R. Roelfsema, A. K. Engel, P. Koenig, and W. Singer. Synchronization between areas of the visual, parietal and motor cortex of the awake behaving cat. *Nature*, 385:157–161, 1997.

[134] S. E. Rose, F. Chen, J. B. Chalk, F. O. Zelaya, W. E. Strugnell, and M. Benson. Loss of connectivity in alzheimer's disease: an evaluation of white matter tract integrity with colour coded mr diffusion tensor imaging. *Journal of Neurology, Neurosurgery and Psychiatry*, 69:528–530, 2000.

[135] P. Rousseeuw and A. Leroy. *Robust Regression and Outlier Detection.* John Wiley and Sons, 1987.

[136] H. Sakoe and S. Chiba. Dynamic programming optimization for spoken word recognition. *IEEE Trans. on Acoustics,Speech and Signal Proc.*, 26:623–625, 1978.

[137] S. Salvador and P. Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. In *The 3rd Wkshp. on Mining Temporal and Sequential Data, ACM KDD*, pages 22–25, 2004.

[138] J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky. Automatic extraction of clusters from hierarchical clustering representations. In *Proc. 7th Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, 2003.

[139] D. Sankoff and J. B. Kruskal. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison.* Addison-Wesley, Reading, MA, 1983.

[140] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27 – 64, 2007.

[141] P. Seliger, S. C. Young, and L. S. Tsimring. Plasticity and learning in a network of coupled phase oscillators. *Phys. Rev. E*, 65:137–185, Jan. 2002.

[142] D. J. Selkoe. Alzheimer's disease is a synaptic failure. *Science*, 298:789–791, 2002.

[143] C. Sexton, U. Kalu, N. Filippini, C. Mackay, and K. Ebmeier. A meta-analysis of diffusion tensor imaging in mild cognitive impairment and alzheimer's disease. *Neurobiology of Aging. In press.*

[144] J. Shao, C. Böhm, Q. Yang, and C. Plant. Synchronization based outlier detection. In *ECML/PKDD*, pages 245–260, 2010.

[145] J. Shao, K. Hahn, Q. Yang, C. Böhm, A. M. Wohlschläger, N. Myers, and C. Plant. Combining time series similarity with density-based clustering to identify fiber bundles in the human brain. In *ICDM Workshops*, pages 747–754, 2010.

[146] J. Shao, K. Hahn, Q. Yang, A. M. Wohlschläger, C. Böhm, N. Myers, and C. Plant. Hierarchical density-based clustering of white matter tracts in the human brain. *IJKDB*, 1(4):1–25, 2010.

[147] J. Shao, X. He, C. Böhm, Q. Yang, and C. Plant. Synchronization-inspired partitioning and hierarchical clustering. In *Submitted for publication.*

[148] J. Shao, X. He, C. Plant, Q. Yang, and C. Böhm. Robust synchronization-based graph clustering. In *Submitted for publication.*

[149] J. Shao, N. Myers, Q. Yang, J. Feng, C. Plant, C. Böhm, H. Förstl, A. Kurz, C. Zimmer, C. Meng, V. Riedl, A. Wohlschläger, and C. Sorg. Prediction of alzheimer's disease using individual structural connectivity networks. *Submitted for publication.*

[150] J. Shao, Q. Yang, C. Böhm, and C. Plant. Detection of arbitrarily oriented synchronized clusters in high-dimensional data. In *ICDM 2011, in press.*

[151] B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London, 1986.

[152] C. Sorg, R. P. V. Riedl, A. Kurz, and A. Wohlschlager. Impact of alzheimer's disease on the functional connectivity of spontaneous brain activity. *Curr Alzheimer Res*, 6:541–553, 2009.

[153] R. Stahl, O. Dietrich, S. J. Teipel, H. Hampel, M. F. Reiser, and S. O. Schoenberg. White matter damage in alzheimer disease and mild cog-

nitive impairment: assessment with diffusion-tensor mr imaging and parallel imaging techniques. *Radiology*, 243:483–492, 2007.

[154] G. Stebbins and C. Murphy. Diffusion tensor imaging in alzheimer's disease and mild cognitive impairment. *Behavioural Neurology*, 21:39–49, 2009.

[155] D. Stijn. A cluster algorithm for graphs. Technical report, CWI (Centre for Mathematics and Computer Science), Amsterdam, The Netherlands, The Netherlands, 2000.

[156] A. Strehl and J. Ghosh. Cluster ensembles–a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[157] J. Tang, Z. Chen, A. W.-C. Fu, and D. W.-L. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *PAKDD*, pages 535–548, 2002.

[158] S. J. Teipel, C. Born, M. Ewers, A. L. Bokde, M. F. Reiser, H. J. Möller, and H. Hampel. Multivariate deformation-based analysis of brain atrophy to predict alzheimer's disease in mild cognitive impairment. *Neuroimage*, 38:13–24, 2007.

[159] G. ten Holt, M. Reinders, and E. Hendriks. Multi-dimensional dynamic time warping for gesture recognition. In *Thirteenth annual conference of the Advanced School for Computing and Imaging*, 2007.

[160] D. Thal, U. Rueb, M. Orantes, and H. Braak. Phases of ab-deposition in the human brain and its relevance for the development of ad. *Neurology*, 58:1791–1800, 2002.

[161] A. K. H. Tung, X. Xu, and B. C. Ooi. Curler: Finding and visualizing nonlinear correlated clusters. In *SIGMOD Conference*, pages 467–478, 2005.

[162] B. van der Pol. Forced oscillations in a circuit with non-linear resistance. *Phil. Mag.*, 3:64–80, 1927.

[163] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[164] N. X. Vinh and J. Epps. Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In *ICML Conf.*, pages 1073–1080, 2009.

[165] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, Dec 2007.

[166] H. Wang, W. Wang, J. Yang, and P. S. Yu. Clustering by pattern similarity in large data sets. In *SIGMOD Conference*, pages 394–405, 2002.

[167] X. Wang, A. Wirth, and L. Wang. tructure-based statistical features and multivariate time series clustering. In *ICDM*, pages 351–360, 2007.

[168] C.-F. Westin, S. E. Maier, H. Mamata, A. Nabavi, F. A. Jolesz, and R. Kikinis. Processing and visualization of diffusion tensor mri. *Medical Image Analysis*, 6(2):93–108, 2002.

[169] C.-F. Westin, S. Peled, H. Gudbjartsson, R. Kikinis, and F. A. Jolesz. Geometrical diffusion measures for mri from tensor basis analysis. In *In ISMRM 97*, page 1742, 1997.

[170] A. T. Winfree. Biological rhythms and the behavior of populations of coupled oscillators. *Journal of Theoretical Biology*, 16:15–42, 1967.

[171] R. P. Woods, S. T. Grafton, J. D. G. Watson, N. L. Sicotte, and J. C. Mazziotta. Automated image registration: Ii. intersubject validation of linear and nonlinear models. *Journal of Computer Assisted Tomography*, 22:153–165, 1998.

[172] X. Xu, M. Ester, H.-P. Kriegel, and J. Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *Proc. of the 14th Int. Conf. on Data Engineering*, pages 324–331, 1998.

[173] K. Yamanishi and J. ichi Takeuchi. Discovering outlier filtering rules from unlabeled data: combining a supervised learner with an unsupervised learner. In *KDD*, pages 389–394, 2001.

[174] K. Yamanishi, J. ichi Takeuchi, G. J. Williams, and P. Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In *KDD*, pages 320–324, 2000.

[175] B. Yi, H. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proc. of the 14th Int. Conf. on Data Engineering*, pages 201–208, 1998.

[176] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, volume 17, pages 1601–1608, 2004.

[177] S. Zhang, C. Demiralp, and D. H. Laidlaw. Visualizing diffusion tensor mr images using streamtubes and streamsurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):454–462, 2003.

[178] S. Zhang and D. H. Laidlaw. Dti fiber clustering and cross-subject cluster analysis. In *Proc. Int. Society for Magnetic Resonance in Medicine*, page 1, May 2005.

[179] T. Zhang, R. Ramakrishnan, and M. Livny. An efficient data clustering method for very large databases. In *SIGMOD Conf.*, pages 103–114, May 1996.