Fuzzy Clustering of Incomplete Data

Inaugural-Dissertation

zur

Erlangung des Doktorgrades der

Mathematisch-Naturwissenschaftlichen Fakultät

der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Ludmila Himmelspach

aus Petropawlowsk

November 2015

Aus dem Institut für Informatik
der Heinrich-Heine Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Referent: Prof. Dr. Stefan Conrad
Koreferent: Prof. Dr. Martin Lercher
Tag der mündlichen Prüfung: 10.12.2015

Dedicated to

Carolina

Acknowledgements

First, I would like to thank my advisor and first referee *Prof. Dr. Stefan Conrad* for giving me the opportunity to write this thesis and for supporting me in my research. I am very thankful for his constructive comments and feedback, and for providing a motivating and comfortable working atmosphere. It was a great pleasure for me to work under his supervision. I also thank the second reviewer of this thesis, *Prof. Dr. Martin Lercher*, for his interest in my work and willingness to be the second referee.

I would like to thank my coauthors *João Paulo Carvalho* and *Daniel Hommers* for a good collaboration and inspiring discussions. I also thank *Prof. Dr. Wilhelm Singhof* for his advise in mathematical questions.

My special compliments go to my former colleagues at the database group *Johanna Vompras*, *Hieu Quang Le*, *Katrin Zaiß*, *Sadet Alcic*, *Tim Schlüter*, *Thomas Scholz*, *Jiwu Zhao* and to my new colleagues *Magdalena Rischka*, *Daniel Braun*, *Robin Küppers*, *Thi Thuy Anh Nguyen*, *Michael Singhof*, *Janine Golov*, and *Matthias Liebeck* for an amicable atmosphere and enlightening discussions. I thank *Johanna* for her support at the beginning of my work. I thank *Magdalena* for her patient listening and the conversations about the research, teaching and everything. I am also grateful to *Magdalena*, *Daniel*, *Michael*, and *Matthias* for proofreading parts of this thesis.

Furthermore, I want to thank *Guido Königstein*, *Sabine Freese*, and *Marga Potthoff* for their excellent technical and administrative support.

I want to express my deepest and warmest thanks to my friends and my family, especially to my parents for their love, encouragement, and support. I am also grateful to my brother *Alexander* for proofreading some of my conference papers short before the submission deadlines.

Finally, I deeply want to thank and dedicate this thesis to my beloved daughter *Carolina*, my little researcher, for bringing joy to my life, giving me motivation to accomplish this thesis, being patient with me, cheering me up, showing how research really works, and her untiring interest in my work. I will always associate my work on this thesis with your cute, smiling, interesting, sleepy baby face.

Düsseldorf, Germany
December, 2015 *Ludmila Himmelspach*

Abstract

Clustering is one of the important and primarily used techniques for the automatic knowledge extraction from large amounts of data. Its task is identifying groups, so-called clusters, of similar objects within a data set. Clustering methods are used in many areas, including database marketing, web analysis, information retrieval, bioinformatics, and many others. However, if clustering methods are applied on real data sets, a problem that often comes up is that missing values occur in the data sets. Since traditional clustering methods were developed to analyze complete data, there is a need for data clustering methods handling incomplete data. Approaches proposed in the literature for adapting the clustering algorithms to incomplete data work well on data sets with equally scattered clusters. In this thesis we present a new approach for adapting the fuzzy c-means clustering algorithm to incomplete data that takes the scatters of clusters into account. In the experiments on artificial and real data sets with differently scattered clusters we show that our approach outperforms the other clustering methods for incomplete data.

Since the quality of the partitioning of data produced by the clustering algorithms strongly depends on the assumed number of clusters, in the second part of the thesis we address the problem of finding the optimal number of clusters in incomplete data using cluster validity functions. We describe different cluster validity functions and adapt them to incomplete data according to the "available-case" approach. We analyze the original and the adapted cluster validity functions using the partitioning results of several artificial and real data sets produced by different fuzzy clustering algorithms for incomplete data. Since both the clustering algorithms and the cluster validity functions are adapted to incomplete data, our aim is finding the factors that are crucial for determining the optimal number of clusters on incomplete data: the adaption of the clustering algorithms, the adaption of the cluster validity functions, or the loss of information in the data itself.

Discovering clusters of varying shapes, sizes and densities in a data set is more useful for some applications than just partitioning the complete data set. As a result, density-based clustering methods become more important. Recently presented approaches either require the input parameters involving the information about the structure of the data set, or are restricted to two-dimensional data. In the last part of the thesis, we present a novel density-based clustering algorithm, which uses the fuzzy proximity relations between the data objects for discovering differently dense clusters without any a-priori knowledge of a data set. In experiments, we show that our approach is able to correctly detect the clusters closely located to each other and clusters with wide density variations.

Zusammenfassung

Clustering ist eine der wichtigen und primär benutzten Techniken für die automatische Wissensextraktion auf großen Datenmengen. Seine Aufgabe ist es Gruppen, so genannte Cluster, von ähnlichen Objekten auf Datenmengen zu identifizieren. Die Methoden der Clusteranalyse finden in vielen Bereichen ihre Anwendung, einschließlich Database Marketing, Web-Analyse, Information Retrieval, Bioinformatik, und vielen anderen. Wenn Clusteringmethoden jedoch auf realen Daten angewendet werden, entsteht oft das Problem, dass fehlende Werte in Datenmengen vorkommen. Da die klassischen Clusteringmethoden entwickelt wurden, um auf vollständigen Daten Analysen durchzuführen, werden Clusteringmethoden benötigt, die mit unvollständigen Daten umgehen können. Die in der Literatur vorgeschlagenen Verfahren zum Anpassen der Clusteringmethoden auf unvollständige Daten funktionieren gut auf Datenmengen mit gleichgroßen Clustern. In dieser Dissertation stellen wir ein neues Verfahren zum Anpassen des Fuzzy C-Means Algorithmus an unvollständige Daten vor, das die Streuung der Cluster berücksichtigt. In Experimenten auf künstlichen und realen Datensätzen mit unterschiedlich großen Clustern zeigen wir, dass die Leistung unseres Verfahrens andere Clusteringmethoden für unvollständige Daten übertrifft.

Da die Qualität der Partitionierung von Daten, die von den Clusteringalgorithmen erzeugt wird, stark von der angenommenen Clusteranzahl abhängt, befassen wir uns im zweiten Teil der Doktorarbeit mit dem Problem der Bestimmung der optimalen Clusteranzahl auf unvollständigen Daten mittels Indizes zur Clustervalidierung. Wir beschreiben unterschiedliche Gütekriterien zur Clustervalidierung und passen sie entsprechend der „available-case“-Methode auf unvollständige Daten an. Wir analysieren die originalen und die angepassten Indizes zur Clustervalidierung unter der Benutzung der Partitionierungsergebnisse von mehreren künstlichen und realen Datensätzen, die von unterschiedlichen Fuzzy Clusteringalgorithmen für unvollständige Daten erzeugt wurden. Da sowohl die Clusteringalgorithmen als auch die Bewertungsfunktionen auf unvollständige Daten angepasst wurden, ist es unser Ziel die Faktoren zu bestimmen, die für die Bestimmung der optimalen Clusteranzahl auf unvollständigen Daten ausschlaggebend sind: das Anpassen von Clusteringalgorithmen, das Anpassen von Funktionen zur Clustervalidierung oder der Informationsverlust in Daten.

Für einige Anwendungen ist die Bestimmung von Clustern unterschiedlicher Form, Größe und Dichte in Datenmengen nützlicher als die bloße Partitionierung des kompleten Datensatzes. Infolgedessen gewinnen die dichtebasierten Clusteringmethoden zunehmend an Bedeutung. Die jüngst vorgestellten Verfahren erfordern entweder Eingabeparameter, die Information über die Datensatzstruktur erfordern, oder sind auf zweidimensionale Daten beschränkt. Im letzten Teil der Doktorarbeit stellen wir einen neuen dichtebasierten Clusteringalgorithmus vor, der sich Fuzzy Proximity Relationen zwischen den Datenobjekten zu Nutze macht, um Cluster unterschiedlicher Dichte oh-

ne jedes a-priori Wissen über den Datensatz aufzufinden. Wir zeigen in Experimenten, dass unser Verfahren fähig ist, die dicht beieinanderliegenden Cluster und Cluster stark variierender Dichte korrekt zu bestimmen.

Contents

1

Introduction

1.1 The KDD Process and Data Mining

The development of possibilities for collecting and storing large volumes of data has lead to a data overload across a wide range of areas. The huge amounts of data are collected for many reasons. Some data are collected and stored to be used for control or archival purposes. For example, the sales data gathered at the scanner cash registers at the supermarkets are primary stored for understanding and evaluation of sales. Another reason for the digital storage of data is to make them available anytime and any place. An example here could be the data from the health care sector. In case of emergency hospitalization the immediate access to the information about the patient's medical history and the drug intolerances can be lifesaving. A large amount of data is gathered for analysis purposes concerning particular questions. Independent from the reasons for the data collection these volumes of data can be used for the extraction of new useful knowledge that is involved in the data. For example, the sales data from the supermarkets can be used for revealing products that are bought together, or for determining products that are sold well at supermarkets in particular areas. The patients' medical data can be used for recognizing the risk groups or for finding the successful treatment for particular diseases. Before we obtain the *interesting knowledge*, our data passes through a long process. This process is also called the process of *Knowledge Discovery in Databases (KDD)*. Fayyad, Piatetsky-Shapiro, and Smyth defined KDD in [FPS96b] as follows:

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

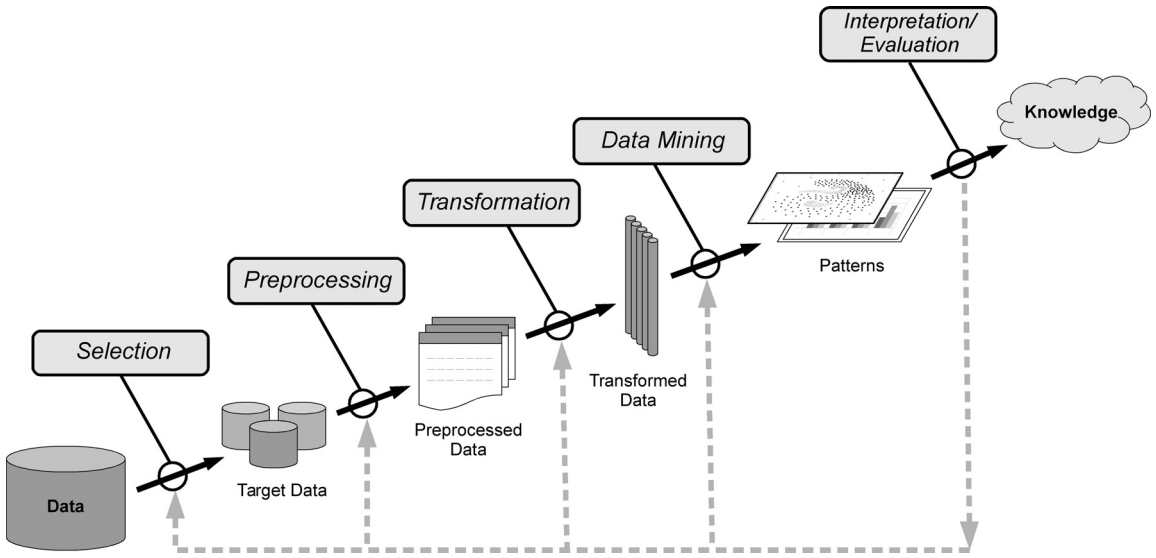


Figure 1.1: The main steps of the KDD process (adapted from [FPS96b]).

The aim of the knowledge extraction from the low-level data is finding *patterns* that can be either a model fitting to a set of the data or structures describing subsets of the data set. These patterns should be *valid* for new data, unknown and unexpected to the user or the application, *potentially useful* for the initial question. Moreover, the patterns should be *understandable* (at least after a post-processing step) to be suitable for deriving *knowledge* from them. As we already stated, before we gain the desired knowledge, the raw data pass through the *KDD process*. The KDD process is an interactive and iterative process, meaning that many decisions have to be made by the user and moving back to the previous steps of the process may be required to achieve the best results. In the following, we briefly describe the basic steps of the KDD process, the whole process is outlined in Figure 1.1.

Basic steps of the KKD process:

1. **Understanding the application domain and identifying the goal of the KDD process** from the viewpoint of the application. This step also involves acquiring the relevant prior application knowledge.

2. **Selecting and creating a target data set:** finding out which data are required, selecting data, obtaining the additional relevant data.

3. **Preprocessing and cleaning:** integration of data from different data sources, resolving inconsistencies, removing noise and outliers, handling missing and uncertain values.

4. **Data transformation:** selecting useful features. Using dimensionality reduction or feature transformation methods in order to identify relevant variables or to

obtain an invariant representation of the data.

5. **Data Mining:** choosing and applying the appropriate data mining algorithms in order to extract patterns from data.

6. **Evaluation/Interpretation** of the found patterns: validation of mined patterns, if necessary repeating the process from any of the previous steps. This step also includes the visualization of found patterns and documentation of discovered knowledge.

The fifth step, *Data Mining*, is often used as a synonym for the entire KDD process. The reason is that the data mining methods are applied for extracting patterns from data. Although the quality of the found patterns depends on the previous steps, data mining is considered as the core of the KDD process. Data mining involves methods at the intersection of different research areas including statistics, databases, pattern recognition, machine learning, artificial intelligence, and high performance computing [FPS96c, FPS96b]. Data mining techniques are applied in several research fields that became the new distinct fields in data mining, for example, *Text and Web Mining* [SW08], *Multimedia Data Mining* [Thu01], *Spatial Data Mining* [KAH96, SC03], *Temporal Data Mining* [AO01, LOW02], and others. In our brief overview we focus on the main tasks of data mining.

- **Cluster Analysis:** *Clustering* is an important data mining task for identifying groups or clusters of similar data objects within a data set [EC02, Bez81, HK00].

- **Classification and Prediction:** The task of *Classification* and *Prediction* is learning a function that predicts the class of new data items based on a training set of data objects with known class membership [DHS00, HK00].

- **Association Rule Mining:** *Association Rule Mining* is a method for finding interesting relations between the data items in large data sets and identifying rules using different criteria for interestingness [AIS93, HGN00].

- **Summarization:** The goal of *Summarization* is creating a compact description of a data set so that the short version retains the principal characteristic of the original data [Mie05, FPS96a].

The data mining tasks are not used as stand-alone techniques in the KDD process. Depending on the application they are commonly combined in the data mining step so that some methods use the results produced by the other data mining techniques. To discover classes within a data set, the clustering methods are often performed on the data before classifying new data items. But there are also clustering approaches that cluster classified data to identify subclasses within the large classes [BTK00].

Some classification methods draw on the concepts of association rule mining [HLZS99, ZLSE08]. The clustering methods are often used for summarizing large data sets [BH67, MV13]. Beyond those there are many other interlacings between the data mining methods. In this thesis we focus on different methods for data clustering.

1.2 Cluster Analysis

Clustering is an important and one of the primary used techniques for the automatic knowledge extraction from large amounts of data. Its task is exploring the distribution of objects in a data set. Generally, clustering is defined as a technique for partitioning a data set into groups (clusters) so that

- the data items in the same cluster are as similar as possible, and
- the data items assigned to different clusters are as dissimilar as possible.

Data clustering is used in many fields, for instance, in database marketing for customer segmentation, in image processing for object extraction, or in bioinformatics for microarray data analysis. The clustering techniques can be used in the same area for different tasks. In text mining, for example, the clustering methods are often used for the text summarization, but the same methods are also used for generation of author profiles. In information retrieval, clustering is used for document grouping but also for query expansion and visualization of search results. Depending on the application task the discovered clusters in the data can have different characteristics. Therefore, different clustering methods were proposed in the literature over the years. We give a short overview of the most important techniques summarized according to the clustering strategy:

- **partition-based clustering methods** partition the data set into a predefined number of clusters which are usually represented by their cluster prototypes. Some popular algorithms are k-Means [Mac67, Llo82], Fuzzy-C-Means [Bez81], PAM [KR90], CLARANS [NH94], and the EM algorithm [DLR77].
- **density-based clustering methods** regard clusters as dense regions of data points in the feature space separated by regions of lower density. Two established density-based clustering algorithms are DBSCAN [EHPJX96] and DENCLUE [HK98].
- **hierarchical clustering methods** create a hierarchical representation of the data set in a dendrogram either by merging clusters (agglomerative clustering [DE84]) or by dividing them (divisive clustering [KR90, GHJ91]).

- **subspace clustering methods** identify clusters of data items that are similar in some subspaces of the feature space. Depending on the search sequence of the subspaces the subspace clustering algorithms are divided in *top-down* (PROCLUST [AWY+99], ORCLUS [AY00], and COSA [FM04]) and *bottom-up* (CLIQUE [AGGR98], CLTree [LXY00], and SUBCLUE [KKK04]) approaches.

Furthermore, there are also clustering methods that combine characteristics of the aforementioned approaches. BIRCH [ZRL96] and CURE [GRS98], for instance, employ a combination of partition-based and hierarchical clustering, and OPTICS [ABKS99] is a density-based hierarchical clustering algorithm.

1.3 Fuzzy Logic

In the previous section we gave a definition of clustering as a technique for "partitioning a data set into groups" or "identifying clusters as groups of similar data items". Based on this notion the clusters appear as closed sets of data objects with clear boundaries. In real world applications the boundaries of clusters are usually difficult to define because it is nearly impossible to find reasonable criteria that include some data objects into a cluster but exclude others. This problem can be explained considering the example of customer segmentation that usually uses demographic information of customers like age, household income, gender, education, children, debts, property, savings, etc. A clustering algorithm partitions the data set into clusters that can be described based on the feature values of objects within the clusters. Let us consider the feature *age*. Presumably the clustering algorithm will discover a group of "young people" using, for example, a numerical threshold $20 \leq age \leq 30$ (compare Figure 1.2). Independently from the choice of the threshold values, the question arises: Why are the 31-year-olds excluded from the group of "young people" although the difference between 31 and 30 is smaller than the difference between 30 and 20? Unlike babies who tend to "grow overnight" [VP08], people usually do not significantly change or develop in such a short period of time at that age. The problem is the artificial thresholds that we set to distinguish groups or clusters although the transitions between the states are *vague*.

This problem was addressed by Lotfi Zadeh who is the founder of *Fuzzy Logic*. In his seminal work *"Fuzzy Sets"* from 1965, he proposed to soften the concept of the membership to a set [Zad65]. Instead of using the Boolean values *true* and *false*, the basic idea of fuzzy logic is to introduce the membership degrees that range between 0 and 1 to express the membership of a data object to a set. The closer the membership value to 1 is, the higher the grade of membership to the *fuzzy set* is. Formally a *fuzzy set* is defined as follows [Zad65]:

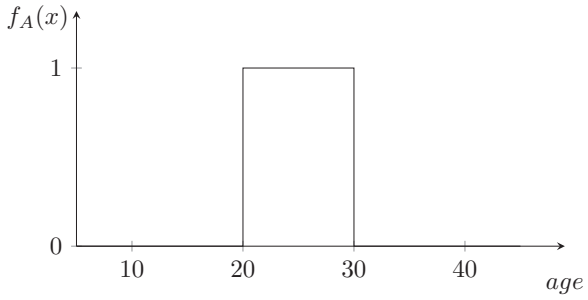


Figure 1.2: Boolean membership function for the set "young people".

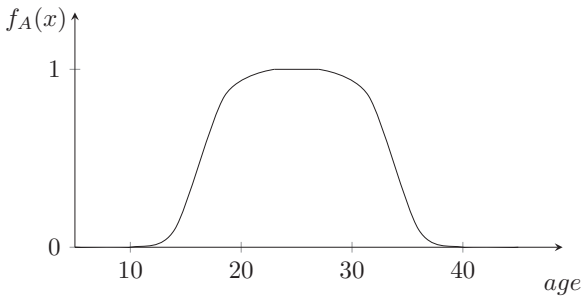


Figure 1.3: Fuzzy membership function for the set "young people".

Definition 1. *Let X be a set of objects denoted generically by x. A* ***fuzzy set*** *A in X is characterized by a* ***membership function*** *$f_A(x) : X \to [0, 1]$ which associates with each object in X a real number in the interval $[0, 1]$ with the value of $f_A(x)$ at x representing the "grade of membership" of x in A.*

Going back to the example of the description for the group of "young people", instead of using a bivalent set we can model soft transition between the groups using a fuzzy set to characterize this subset. Figure 1.3 shows an example for a fuzzy membership function for the subset "young people". As the boolean membership function depicted in Figure 1.2 this modelling makes possible to express that a 40-year-old does not belong to the group of "young people" ($f_A(40) = 0$) but a 25-year-old is definitely a part of it ($f_A(25) = 1$). Unlike the boolean membership function the fuzzy membership function is able to express that a 31-year-old still belongs to the subset of "young people" but with a smaller degree than a 30-year-old or a 25-year-old ($f_A(25) > f_A(30) > f_A(31)$).

Previously we focused only on the group of "young people" but a (customer) data set may contain several groups along the feature *age*. The additional groups could be "middle-aged people" and "elderly people". Since these subsets can not be precisely defined either, they can also be defined as fuzzy sets. Modelling all groups as fuzzy sets makes possible not only to describe the typical representatives for each group, but also enables us to represent data items that belong to several groups with different membership degrees. This preliminary example shows the basic idea of *Fuzzy Clustering* [Rus69, Bez81, BEF84, HKK96] which is one of the important applications of fuzzy logic in the data mining area.

In the previous example we used fuzzy sets to characterize vague subsets of a data set. The fuzzy sets can also be used to describe imprecisely defined relations between objects [Zad65, Zad75]. This kind of relations is denoted as *fuzzy relations* which play an important role in the fuzzy logic theory. Formally a *fuzzy relation* is defined as follows [Zad65]:

Definition 2. *Let X be a set of objects denoted generically by x. An* ***n-ary fuzzy relation*** *in X is defined as a fuzzy set A in the Cartesian product space $X \times X \times ... \times X$. The membership function for the n-ary fuzzy relation is of the form $f_A(x_1, ..., x_n)$, where $x_i \in X$ for $i = 1, ..., n$.*

A simple example of a fuzzy relation is the fuzzy equality ("nearly equal") that can be characterized by a fuzzy set A with the following representative values of the membership function: $f_A(1, 100) = 0$, $f_A(1, 1) = 1$, $f_A(10, 15) = 0.85$, etc. Another example of a fuzzy relation describes the relational concept "is close to". Representative values of the corresponding membership function might be: $f_A(\text{Paris}, \text{Beijing}) = 0$, $f_A(\text{Paris}, \text{Berlin}) = 0.5$, $f_A(\text{Madrid}, \text{Lisbon}) = 0.8$, $f_A(\text{Berlin}, \text{Berlin}) = 1$, etc.

In this introductory chapter we only outlined the basic concepts of fuzzy logic that have been employed in this thesis. Since the introduction of *fuzzy sets* in 1965, the fuzzy logic has experienced a huge development. The concepts of fuzzy logic have found applications in many areas including control theory, image processing, artificial intelligence, data analysis, data mining, web engineering, information retrieval, signal processing, natural language processing and many others [HKK96, MWZ94, DP99, KNB99, Jan07, BKKP06, LM91, Zim12, Nov92, YZ12, CBC12].

1.4 Contributions

Fuzzy Clustering is one of the important application areas of *fuzzy logic* in the data mining field. This thesis addresses the problems and contributes to the fields of fuzzy clustering and fuzzy cluster validity on incomplete data. Moreover, we propose a new density-based clustering algorithm that uses fuzzy proximity relations. Below we briefly summarize the main contributions of this thesis.

- The fuzzy clustering methods for incomplete data proposed in the literature perform poorly on incomplete data sets with differently scattered clusters [Him08, HC10a]. For this reason, we propose an enhanced fuzzy clustering approach for incomplete data which uses a new membership degree for missing value estimation taking the cluster dispersion into account [HC10b]. Since the evaluation of the clustering algorithms adapted to incomplete data requires both the incomplete test data and the corresponding complete data, we developed a missing data generator that makes a specified percentage of values in a data set absent according to different missing data mechanisms.

- Another contribution of this work addresses the problem of finding the optimal number of clusters on incomplete data using the cluster validity functions. We adapted different cluster validity functions to incomplete data and evaluated them

in extensive studies on incomplete test data sets with different data distributions [HHC11, HCC12]. We reveal the factors that are crucial for the cluster validity for fuzzy clustering of incomplete data and indicate the factors of cluster validity functions that make them resistant against incomplete data.

- A further contribution of this thesis is the presentation of a new density-based algorithm DENCFURE (Density-Based Clustering using Fuzzy Proximity Relations) [HC11] for discovering differently dense clusters in a data set in presence of noise. Using the fuzzy proximity relations between data objects this algorithm is able to detect clusters closely located to each other and clusters with wide density variations without any a-priori knowledge of the data set.

1.5 Outline of this Work

The thesis is organized as follows. **Chapter 2** provides the basic background knowledge required to follow the approaches presented in this thesis. First we introduce the basic concepts of fuzzy clustering and present some important partitioning fuzzy clustering algorithms. Then we give an introduction to the analysis of incomplete data describing some important missing-data patterns and the different types of missing-data mechanisms. In the same chapter we give an overview over the methods for adapting fuzzy clustering algorithms to incomplete data proposed in the literature.

In **chapter 3**, we introduce our enhanced fuzzy clustering approach for incomplete data. Our method uses a new membership degree for estimation of missing values taking the cluster dispersion into account. In experiments on incomplete data sets with differently shaped and sized clusters we demonstrate the capabilities of our approach and compare it with the existing fuzzy clustering algorithms for incomplete data.

Chapter 4 approaches the problem of finding the optimal number of clusters on incomplete data. We give an overview over the different cluster validity functions proposed in the literature and adapt them to incomplete data according to the "available-case" approach. In **chapter 5** we present the evaluation results of the adapted cluster validity indexes in an extensive study on incomplete test data sets with different data distributions. Furthermore, we reveal the factors of cluster validity functions that make them resistant against incomplete data.

We introduce a new density-based algorithm DENCFURE (Density-Based Clustering using Fuzzy Proximity Relations) in **chapter 6**. Using the fuzzy proximity relations between data objects this algorithm discovers differently dense clusters in a data set in presence of noise. In experiments on several test data sets we show that our algorithm is able to detect clusters closely located to each other and clusters with wide density variations without any a-priori knowledge of the data set.

Finally, **chapter 7** concludes this thesis with a short summary and a discussion of future research.

2

Background

Clustering is usually used as the primary technique for the automatic knowledge extraction from large amounts of data. Uncertain, erroneous and missing values in data is a challenging problem for the traditional clustering methods because they were developed to analyze complete data sets. In this thesis we address the problem of adapting the fuzzy clustering algorithms and the validity functions to incomplete data. This chapter provides the relevant background knowledge required to follow the approaches presented in the thesis. First we give an introduction to the basic concepts of fuzzy clustering and present some important partitioning fuzzy clustering algorithms. Then we provide insight into the incomplete data analysis describing some important missing-data patterns and the different missing-data mechanisms. Finally, we give an overview over the methods for adapting fuzzy clustering algorithms to incomplete data proposed in the literature.

2.1 Fuzzy Clustering

Clustering is an important data mining technique for the automatic exploration of the distribution of data objects in a data set. Its aim is to partition a given data set into groups, so called *clusters*, so that the data items within a cluster are as similar as possible and the data items from different clusters are as dissimilar as possible. As we already mentioned in the introductory chapter, over the years, different clustering methods were proposed in the literature. The oldest and the most widely used clustering techniques are the *partition-based clustering* methods. They partition a data set into a predefined number of clusters which are usually represented by their cluster prototypes. The traditional clustering techniques assign each data item to exactly one

cluster which makes their results less informative. They draw no distinction between the memberships of a typical representative of a cluster and the boundary data items. The information about the clustering structure, like if there are overlapping clusters in the data set, gets lost. Instead, the overlapping clusters are artificially divided by "crisply" assigning the data points located in the overlap of clusters.

To overcome these drawbacks, in *fuzzy clustering*, the clusters are modelled by *fuzzy sets* assigning each data item of the data set to each cluster with a membership degree that ranges between 0 and 1. A membership degree of 1 indicates a certain assignment of the data item to the cluster. A membership degree of 0 indicates that the data item does not belong to the cluster. The membership degrees of data items located in the overlap of clusters should be approximately equal to the overlapping clusters. In this way, the fuzzy clustering methods are able to model the soft transitions between clusters which conforms to human perception more than crisp partitioning of a data set into clusters. Moreover, the information about the clustering structure and the overlaps between clusters can be derived from the partitioning results produced by the fuzzy clustering methods. Therefore, the fuzzy clustering results are of great use for both: as the output of the data mining process and as the interim result for the further processing.

In the following, we explain the working principle of fuzzy clustering methods on the example of the fuzzy c-means algorithm. Besides, we present two relevant fuzzy clustering methods: the Gustafson-Kessel algorithm and the fuzzy maximum likelihood estimation algorithm.

2.1.1 Fuzzy C-Means Algorithm (FCM)

The objective of cluster analysis involves two requirements for an adequate partitioning of a data set: the data items in the same cluster have to be similar and the data items from different clusters have to be dissimilar. However, both criteria can not be always satisfied at the same time. For example, in the case of overlapping clusters the data points located in the overlap of clusters have approximately the same membership degrees to the overlapping clusters. Therefore, the most clustering approaches aim to fulfil only the requirement of the homogeneity of data items within clusters. In the partition-based clustering approaches, each cluster is described by its representative, so called *cluster prototype*. The idea to fulfil the homogeneity criterion within clusters is to partition the data set into clusters so that the data items within clusters are as similar as possible to the cluster prototypes. Since the similarity or rather the dissimilarity between the data items is usually expressed by the distances between them, the basic idea of partition-based clustering methods is to partition a data set into clusters so that the overall distances between the data items and the cluster prototypes

are minimal. Both fuzzy and hard partition-based clustering approaches describe the clustering problem using an objective function that they optimize in an iterative process satisfying constraints on its variables.

The best known and the most widely used fuzzy clustering algorithm is the *fuzzy c-means clustering algorithm (FCM)* [Bez81]. Fuzzy c-means is a partitioning clustering algorithm that can be considered as a fuzzy generalization of the hard k-means algorithm [Mac67]. FCM partitions a given data set $X = \{x_1, ..., x_n\}$ in a d-dimensional metric data space into c clusters that are represented by their cluster prototypes $V = \{v_1, ..., v_c\}$. Unlike the k-means algorithm, which assigns each data object to exactly one cluster, fuzzy c-means algorithm assigns data items to clusters with membership degrees [Bez81, BEF84, HKK96]. The membership degree $u_{ik} \in [0, 1]$ expresses the relative degree to which the data point x_k with $1 \leq k \leq n$ belongs to the cluster C_i, $1 \leq i \leq c$. Fuzzy c-means algorithm is a *probabilistic* clustering algorithm, which means that the sum of the membership degrees for each data item equals 1 (see Condition (2.1)), and there are no empty clusters in the partitioning (see Condition (2.2)).

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \forall k \in \{1, ..., n\}, \tag{2.1}$$

$$\sum_{k=1}^{n} u_{ik} > 0 \quad \forall i \in \{1, ..., c\}. \tag{2.2}$$

Condition (2.1) also ensures that all data items have equal weights and, therefore, they are equally included into the partitioning of the data set. Both constraints together prevent assigning all data items to one single cluster in the partitioning.

The objective function of the fuzzy c-means algorithm is defined as follows:

$$J_m(X, U, V) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \cdot d^2(v_i, x_k) \ . \tag{2.3}$$

The similarity between the data items and the cluster prototypes is expressed by the Squared Euclidean function. The parameter m, $m > 1$, is the *fuzzification parameter*, also called *fuzzifier*. The fuzzification parameter determines the vagueness of the resulting partitioning. In the case the value of m is chosen close to unity, the assignment of data items into clusters becomes clearer or "harder" in the partitioning. The larger the value of m is, the softer the boundaries between the clusters are. Usually $m = 2$ is chosen.

The objective function of the fuzzy c-means algorithm has to be minimized to obtain a good partitioning of the data set. Since the objective function cannot be optimized directly, it is minimized using an *alternating optimization (AO)* scheme [Bez81]. The

objective function is alternately optimized over the membership degrees and the cluster prototypes in an iterative process. Although the algorithm may get stuck in a local optimum of the objective function, the alternating optimization is used in the partitioning clustering algorithms due to its practicability. The objective function achieves a local minimum when its partial derivatives in respect to its parameters, the membership degrees and the cluster prototypes, equal to zero satisfying Constraints (2.1) and (2.2). In the fuzzy c-means algorithm, the membership degrees are updated according to Formula (2.4).

$$u_{ik} = \begin{cases} \dfrac{(d^2(v_i, x_k))^{\frac{1}{1-m}}}{\sum_{j=1}^{c} (d^2(v_j, x_k))^{\frac{1}{1-m}}} & \text{if } I_{x_k} = \emptyset, \\ \lambda, \lambda \in [0,1] \text{ with } \sum_{v_i \in I_{x_k}} u_{ik} = 1 & \text{if } I_{x_k} \neq \emptyset, v_i \in I_{x_k}, \\ 0 & \text{if } I_{x_k} \neq \emptyset, v_i \notin I_{x_k}, \end{cases} \tag{2.4}$$

where $I_{x_k} = \{v_i \mid d^2(v_i, x_k) = 0\}$. The partial derivatives of the objective function in respect to the cluster prototypes result in the following update formula for the cluster prototypes:

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, \qquad 1 \leq i \leq c \ . \tag{2.5}$$

The computation of the cluster prototype as a weighted mean of all data items depending on their membership degrees to the cluster conforms the intuition of a cluster representative and provides the name of the fuzzy c-means algorithm. Here, we skip the details of the derivations of the formulae for calculation of the membership degrees and the cluster prototypes. Instead, we refer to the relevant literature [Bez81, HKK96].

The general form of the fuzzy c-means algorithm is represented in Algorithm 1. The algorithm begins with the initialization of the cluster prototypes v_i' which can be either the first c data items of the data set or c randomly chosen data items or c randomly chosen points in the data space. Alternatively, the membership degrees can be initialized. In the first iteration step the membership degrees of each data item to each cluster are calculated according to Formula (2.4). In the second iteration step the new cluster prototypes are calculated based on all data items depending on their membership degrees to the cluster (see Formula (2.5)). The iterative process continues as long as the cluster prototypes change up to a value ϵ. Basically, the algorithm can also be stopped when the number of iterations exceeds some predefined number of maximal iterations. Although the fuzzy c-means algorithm is known as a stable and robust clustering algorithm that does not often get stuck in a local optimum [KDL07, KKW15], it is sensible to evaluate the algorithm for different initializations to achieve the optimal partitioning results.

Algorithm 1 FCM(X, c, m, ϵ)

Require: X is a d-dimensional data set with n data items, $2 \leq c \leq n$ is a number of clusters, $m > 1$ is a fuzzification parameter, $\epsilon > 0$ is a termination accuracy

1: Initialize the set of data centers $v' = \{v'_1, ..., v'_c\}$
2: $v = \{\}$
3: **repeat**
4: $v = v'$
5: Calculate the membership degrees u_{ik} of each data item x_k to each cluster C_i according to Formula (2.4) // *Step 1*
6: Calculate the set of new cluster prototypes $v' = \{v'_1, ..., v'_c\}$ according to Formula (2.5) // *Step 2*
7: **until** $\|v - v'\| < \epsilon$
8: **return** v'

As we already mentioned above, in the probabilistic fuzzy clustering algorithms all data items are equally included into the partitioning of the data set (see Condition (2.1)). The partitioning results of such algorithms can be misleading in the case there are *outliers* (data items that are distant from other data items) in the data set because the membership degrees of outliers can be similar to the membership degrees of data items located in the overlaps of clusters. One of the solutions for the problems caused by the normalization of the membership degrees is detecting and eliminating the outliers before clustering. Another solution is omitting Condition (2.1) and handling the membership degrees as possibility degrees of data items to be assigned to clusters. Fuzzy clustering algorithms that use possibilistic membership degrees are referred to as *possibilistic fuzzy clustering* algorithms [KK93, KK96, TBDK04, ZL04, PPKB05, HCJJ11]. Since we focus on the probabilistic fuzzy clustering algorithms in this thesis, we do not expand on the possibilistic fuzzy clustering.

2.1.2 Gustafson-Kessel Algorithm (GK)

Using the Euclidean distance function as a similarity criterion the basic fuzzy c-means algorithm partitions a data set assuming that all clusters are spherical. In [GK78] Gustafson and Kessel proposed to use a cluster specific similarity measure that takes the shapes of clusters into account. In the *Gustafson-Kessel algorithm (GK)* each cluster C_i is described by the cluster center v_i that specifies the spatial location of the cluster in the data space, and the *fuzzy covariance matrix* Cov_i that captures the extent of the cluster in the different dimensions. The cluster centers v_i are calculated according to Formula (2.5), the fuzzy covariance matrices Cov_i are calculated according

to Formula (2.6).

$$Cov_i = \frac{\sum_{k=1}^{n} (u_{ik})^m (x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^{n} (u_{ik})^m} \quad \text{for} \quad 1 \leq i \leq c. \tag{2.6}$$

The fuzzy covariance matrices are directly integrated in the calculation of the distances between the data items and the cluster prototypes. The Gustafson-Kessel clustering algorithm uses a cluster specific *Mahalonobis distance function* [Mah36]. Hence, the distance between the data item x_k and the cluster center v_i is calculated as follows:

$$d^2(x_k, v_i) = \sqrt[d]{\det(Cov_i)}(x_k - v_i)^T Cov_i^{-1}(x_k - v_i), \tag{2.7}$$

where $1 \leq k \leq n$, $1 \leq i \leq c$, and d is the number of dimensions of the data set X. Using the distance function from Formula (2.7) the GK algorithm uses a separate similarity measure for each cluster depending on its shape. Hence, the resulting partitioning of a data set with nonspherical clusters better corresponds to the intuition than the partitioning produced by the basic FCM algorithm. However, to avoid the minimization of the objective function J_m by decreasing the covariance matrices, the condition $\det(Cov_i) = 1$ has to be satisfied. Consequently, the Gustafson-Kessel algorithm enables the cluster shapes to be variable but the cluster sizes are fixed.

The computational costs of the Gustafson-Kessel algorithm are higher than of the basic FCM algorithm because the covariance matrices of all clusters have to be inverted in each iteration step. In order to decrease the computational costs, the Gustafson-Kessel algorithm is usually initialized with the cluster prototypes produced by the basic fuzzy c-means algorithm after few iterations.

2.1.3 Fuzzy Maximum Likelihood Estimation Algorithm (FMLE)

The *Fuzzy Maximum Likelihood Estimation algorithm (FMLE)* proposed by Gath and Geva in [GG89] creates a partitioning of a data set taking the cluster shapes, the cluster densities, and the cluster sizes (here, the number of data items in each cluster) into account. The algorithm partitions a data set assuming that the data set was created as realizations of d-dimensional normally distributed random variables. Thus, in the FMLE algorithm each cluster C_i is represented by its cluster center v_i, the cluster specific covariance matrix Cov_i, and the a priori probability p_i for selecting this cluster. The cluster prototypes are calculated according to Formulae (2.5), (2.6), and (2.8) for

the a priori probability p_i.

$$p_i = \frac{\sum_{k=1}^{n}(u_{ik})^m}{\sum_{k=1}^{n}\sum_{j=1}^{c}(u_{jk})^m}, \qquad 1 \leq i \leq c\ . \tag{2.8}$$

Similar to the Gustafson-Kessel algorithm the algorithm proposed by Gath and Geva uses a cluster specific similarity measure based on maximum likelihood estimation. The distance between the data item x_k and the cluster center v_i is inversely proportional to the a posteriori probability (likelihood) that the data item x_k was generated by the normal distribution N_i with the expected value v_i and the covariance matrix Cov_i [Höp99]. The a posteriori probability is given in Formula (2.9).

$$\frac{p_i}{\sqrt{\det(Cov_i)(2\pi)^d}}\exp\left(-\frac{1}{2}(x_k - v_i)^T Cov_i^{-1}(x_k - v_i)\right). \tag{2.9}$$

Thus, in the fuzzy maximum likelihood estimation algorithm, the membership degrees are updated using the distances $d(x_k, v_i)^2$ that are calculated as follows:

$$d(x_k, v_i)^2 = \frac{1}{p_i}\sqrt{\det(Cov_i)}e^{\frac{1}{2}(x_k - v_i)^T Cov_i^{-1}(x_k - v_i)}, \tag{2.10}$$

where $1 \leq k \leq n$ and $1 \leq i \leq c$. Using the cluster specific distance measure given in Formula (2.10), the FMLE algorithm takes the variability in shapes, densities, and sizes of clusters into account. On the other hand, using this distance function the FMLE algorithm produces a "harder" assignment of data items into clusters [Höp99]. Therefore, it tends to get stuck in a local optimum more than the basic fuzzy c-means or the Gustafson-Kessel algorithms. For that reason, the FMLE algorithm is usually initialized with the cluster prototypes produced by the basic FCM or the Gustafson-Kessel algorithms after few iterations.

2.2 Analysis of Incomplete Data

The quality of the data is one of the most important factors that might affect the results of the overall KDD process. Problems during the data collection and data preprocessing often lead to uncertain, erroneous or missing values in the data sets. Since the completion or correction of data is often expensive or even impossible through the repetition of the data collection or the data preprocessing steps, there is a need of data analysis methods that can deal with such imperfect data. In this thesis, we focus on the fuzzy clustering methods for dealing with incomplete data. According to [LR02], we define a value as *missing* if there should be a meaningful value but it is unknown. Since the values in data can be missing for different reason, it is useful to

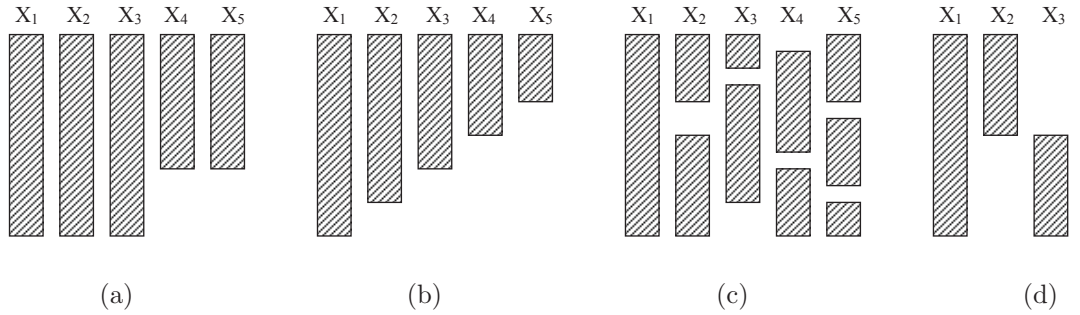


Figure 2.1: Missing-data patterns: (a) multivariate, (b) monotone, (c) general, and (d) file-matching [LR02].

consider the modelling of missing data for dealing with them. In this section we give an overview of some important missing-data patterns and then we briefly describe the different types of missing-data mechanisms. Afterwards, we describe the commonly used approaches for handling missing values in the data analysis.

2.2.1 Missing-data Patterns

There are different reasons for values to be missing in data. The missing values can be caused by technical problems like device failures during experiments. In questionnaires, some questions can be unanswered due to understanding problems or lack of time or motivation. Missing values can also occur in the data as a result of the data cleaning process or failures during the data transfer or data coding. Although the reasons for missingness of values are diverse, there are only few missing-data patterns that result due to the missing values in the data sets. The *missing-data patterns* describe which values are observed and which values are missing [LR02]. The missing values can be differently arranged in the data matrix. Some clustering algorithms adapted to incomplete data apply quite similarly to any of missing-data patterns, whereas other algorithms are restricted to a special pattern. In general, there are four common missing-data patterns: multivariate, monotone, general, and file-matching missing-data patterns. Figure 2.1 (a) shows the *multivariate* missing-data pattern in which the missing values occur in a group of attributes that are either completely observed or missing. The multivariate missing-data pattern occurs, for example, in surveys where one group of respondents gets a complete questionnaire and another groups gets a short version of it. If missing values only occur in one attribute, the missing-data pattern is denoted as *univariate*. The *monotone* missing values usually occur as a result of longitudinal studies and have a stair-like arrangement of the values in the data matrix (compare Figure 2.1 (b)). When data are joined together from several data sources, it often happens, that the data sets from different sources have both common attributes and the source-specific attributes. Consequently, the combined data set has completely

observed attributes and the attributes that are never observed together. Figure 2.1 (d) shows the missing-data pattern for the *file-matching* problem. Often the missing values are arbitrary arranged in the data matrix as shown in Figure 2.1 (c). This pattern results in the data matrix due to ignoring or overlooking of questions in the questionnaires, or cleaning of erroneous values, or loss of data during the transfer. If the missing-data pattern in the data matrix does not fit either to the multivariate or to the monotone, or to the file-matching missing-data patterns, the missing-data pattern is denoted as *general*.

2.2.2 Missing-data Mechanisms

While the missing-data patterns describe which values are missing in the data matrix, the *missing-data mechanisms* give information about the reasons for occurrence of missing values in data. The missing-data mechanisms refer the relationship between the missingness and the attribute values in the data matrix. While the missing-data patterns indicate which data values can be used for the data analysis, the missing-data mechanisms provide an indication how the available values should be treated during the data analysis to achieve the best results.

Formally, Rubin and Little defined the missing-data mechanism as a probability that a value is available or missing in the data set [LR02]. Let $X = (x_{ij})_{\substack{1\leq i\leq n\\ 1\leq j\leq d}}$ be the data matrix. We divide X in the set X_{obs} of observed or available values and the set X_{mis} of missing values ($X = X_{obs} \cup X_{mis}$). Define the *missing-data indicator matrix* $M = (m_{ij})_{\substack{1\leq i\leq n\\ 1\leq j\leq d}}$ that indicates if the data value $x_{ij} \in X$ is missing ($x_{ij} \in X_{mis}$) or observed ($x_{ij} \in X_{obs}$). Then the missing-data mechanism is defined as the conditional probability of M given X, $p(M \mid X, \phi)$, where ϕ denotes unknown parameters. Generally, there are three different kinds of missing-data mechanisms: MCAR, MAR and NMAR [LR02]. The missing values are called *missing completely at random (MCAR)*, if the missingness does not depend on the data values in the data set independently whether they are missing or observed. That is

$$p(M \mid X, \phi) = p(M \mid \phi) \qquad for\ all\ x_{ij} \in X\ and\ \phi. \tag{2.11}$$

The missing values are denoted as *missing at random (MAR)*, if the missingness depends only on the observed values in the data matrix, and not on the components that are missing:

$$p(M \mid X, \phi) = p(M \mid X_{obs}, \phi) \qquad for\ all\ x_{ij} \in X_{mis}\ and\ \phi. \tag{2.12}$$

The missing values are MAR when, for example, in a questionnaires especially young people remain the income question unanswered. Since the missingness of values in the

variable *income* does not depend on the amount of income itself but on the values of the variable *age*, the condition MAR is fulfilled.

If the probability for a data value to be missing in the data set depends on the missing value itself, then the missing-data mechanism is called *not missing at random (NMAR)*. That is the case when, for example, people with an over average income refuse to reveal their income in the questionnaires. Since the missingness of values in the variable *income* depends on the amount of income itself, the condition NMAR is fulfilled.

Usually, in practice, the mechanisms that lead to missing data are not known in advance. Nevertheless, the missing-data mechanisms can be ascertained or at least excluded via suitable statistical test procedures such as one-sample test for missing-data mechanism NMAR, two-sample test for missing-data mechanism MAR or the Little's MCAR test to verify the missing-data mechanism MCAR [Lit88]. Unlike the tests for the missing data mechanisms MAR and NMAR the test for the missing data mechanism NMAR requires the knowledge about the value distribution in the complete data set, though. For more details see also [LR02, FPP98].

2.2.3 Methods for Handling Missing Values in Data

The best method for handling incomplete data during the data analysis is avoiding the missing values in data through a better study design or the repetition of the data collection or the data preprocessing steps. However, usually the investigation and the repetition of working steps where the missing values occurred are too expensive or impossible. Therefore, there is a need of methods for handling incomplete data. Generally, there are three common approaches for dealing with missing values in data [LR02, Wag04]:

- **Elimination:** As long as the amount of missing values in the data set is relatively small, the common approach is ignoring the data items or the attributes containing missing values and performing the data analysis on the available data. In the literature, this approach is denoted as the "complete-case analysis" or the "complete-variable analysis" [LR02, Ban95]. This method is uncritical if the missing data are missing completely at random. If the missing data are NMAR, ignoring the attributes or the data items where the missing values occur may cause the distortion of data and the clustering results.

- **Imputation:** A common technique for dealing with incomplete data is replacing the missing values with estimated values that are usually derived from the available data. This approach is denoted as the "missing value imputation" in the literature. The imputation techniques range from simple ones like replacing the mis-

sing values with the minimum, maximum or means, to more sophisticated approaches like the regression based approaches, the expectation maximization (EM) and the maximum likelihood (ML) approaches [LR02, Sch97, Rub04, MFK+14]. The major advantage of the imputation approach is that once the missing values have been filled in, the standard data analysis methods can be used. However, the drawback of the imputation is that the quality of results of the data analysis is significantly affected by the used imputation methods because the imputed values are treated as the observed values.

- **Adaption of data analysis methods to incomplete data:** An elegant approach for dealing with incomplete data is adapting the data analysis methods so that they can handle data sets with missing values. This includes methods that estimate missing values during the data analysis but distinguish between the observed and imputed values. The major advantage of the adaption approach is that all observed data can be used for the data analysis avoiding the drawbacks of the missing value imputation.

In this thesis we address the problem of adapting the fuzzy clustering methods and the indexes for fuzzy clustering validation so that they can handle incomplete data.

2.3 Fuzzy Clustering Methods for Incomplete Data

The fuzzy clustering algorithms presented in this chapter cannot be directly applied to incomplete data because they require all feature values of each data item to be present for calculation of the cluster prototypes and the distance measures. In the literature, several approaches for adapting fuzzy clustering algorithms to incomplete data have been proposed. In this section we describe five main strategies for adapting the basic FCM algorithm to incomplete data. These approaches are not specified to any missing-data patterns and they usually assume the missing-data mechanism MCAR. Besides them, there are some proposals in the literature for adapting the GK and the FMLE algorithms to incomplete data using the same strategies [TK98, TDK02, Tim02]. There are also approaches for the adaption of the fuzzy clustering algorithms to incomplete data with specific missing-data mechanisms like missing values with class specific probabilities [TDK03, TDK04].

2.3.1 Whole Data Strategy FCM (WDSFCM)

A simple way for clustering incomplete data is first omitting the incomplete data items and applying the FCM algorithm on the complete data items [HB01]. Afterwards, each incomplete data item can be assigned to the cluster to which center it has the

minimal partial distance (compare Formula (2.13)). This approach is denoted by the *whole data strategy (WDS)* and the modified version of the FCM algorithm is referred to as WDSFCM. Of course, this method can be counted as a "complete-case analysis" because incomplete data items are not involved in the calculation of the cluster prototypes. On the other hand, the incomplete data items are not totally excluded from the data analysis. Their hard or fuzzy cluster memberships are estimated in the end of the clustering process and can be used in the post-processing steps.

The whole data strategy can be applied for clustering of incomplete data as long as the percentage of incomplete data items is relatively low. In [HB01], the authors propose the limit of not less than 75% of complete data items in the data set. Since the cluster prototypes are calculated only on the basis of the complete data items, this strategy is recommendable only if the complete data items are representative for the entire data set.

2.3.2 Partial Distance Strategy FCM (PDSFCM)

One of the strategies for the data analysis on incomplete data proposed by Dixon is estimating the distances between two data items using the *partial distance function* [Dix79]. The partial distance function calculates the sum of the squared Euclidean distances between all available feature values of the data items and scales it by the reciprocal of the proportion of values used during the calculation. If two data items are completely available, the partial distance function calculates the squared Euclidean distance between them. In [HB01, TDK02] the authors adapted the fuzzy c-means algorithm according to the *partial distance strategy (PDS)*. The resulting algorithm is denoted by PDSFCM. In the first iteration step of the FCM algorithm the membership degrees are updated based on the partial distances between the incomplete data items and the cluster centers. The partial distances are calculated according to the following Formula:

$$D_{part}(x_k, v_i) = \frac{d}{\sum_{j=1}^{d} i_{kj}} \sum_{j=1}^{d} (x_{kj} - v_{ij})^2 i_{kj}, \tag{2.13}$$

where

$$i_{kj} = \begin{cases} 1, & \text{if } x_{kj} \text{ is available} \\ 0 & \text{else} \end{cases} \qquad \text{for } 1 \leq j \leq d,\ 1 \leq k \leq n.$$

In the second iteration step of the FCM algorithm the cluster prototypes are updated only on the basis of the available feature values of data items. Thus, the calculation of the cluster prototypes in Formula (2.5) is replaced with

$$v_{ij} = \frac{\sum_{k=1}^{n}(u_{ik})^m i_{kj} x_{kj}}{\sum_{k=1}^{n}(u_{ik})^m i_{kj}} \qquad \text{for } 1 \leq i \leq c,\ 1 \leq j \leq d. \tag{2.14}$$

Since the scaling factor $\frac{d}{\sum_{j=1}^{d} i_{kj}}$ in $D_{part}(x_k, v_i)$ has no effect on the calculation of the cluster prototypes and the membership degrees are based on the relation of the distances between the data items and the cluster prototypes, the scaling factor is completely irrelevant and can be omitted in Formula (2.13).

In contrast to the whole data strategy the advantage of applying the partial distance strategy to the FCM algorithm is that the resulting algorithm can be used on data sets with a large number of missing values and even if missing values occur in all data items and all features.

2.3.3 Optimal Completion Strategy FCM (OCSFCM)

The idea of the *optimal completion strategy (OCS)* is to iteratively compute the missing values as the additional variables over which the objective function is minimized [HB01, TDK02]. The fuzzy c-means algorithm is modified by adding an additional iteration step where the missing values are updated. We obtain Formula (2.15) for estimation of missing values by setting the partial derivatives of the objective function with respect to the missing values to zero.

$$x_{kj} = \frac{\sum_{i=1}^{c}(u_{ik})^m v_{ij}}{\sum_{i=1}^{c}(u_{ik})^m}, \qquad 1 \leq k \leq n \text{ and } 1 \leq j \leq d\ . \tag{2.15}$$

In this way, the missing values are imputed by the weighted means of all cluster centers in each iteration step.

The algorithm begins with the initialization of cluster prototypes. Additionally, the missing values are initialized by random values. The calculation of the membership degrees and the cluster prototypes in the first two iteration steps works in the same way as in the basic FCM. The available and the imputed values in the data set are not distinguished. In the third iteration step, missing values are imputed according to Formula (2.15). The cluster prototypes to which the incomplete data item has higher membership degree have more influence during the estimation of missing values of the data item. The FCM algorithm changed in this way is referred to as OCSFCM.

The advantage of this approach is that missing values are imputed during the clustering process. However, the drawback of the OCSFCM is that the cluster prototypes are computed using the estimated and the available feature values. At the same time the missing values are imputed on the basis of these biased cluster prototypes. In this way, the calculation of the cluster prototypes and the imputation of the missing values influence each other. In [Tim02] the author proposed to diminish the influence of the

imputed values to the calculation of cluster prototypes by reducing the membership degrees of incomplete data items depending on the number of missing values. The resulting algorithm loses the property of a probabilistic fuzzy clustering algorithm, though.

2.3.4 Nearest Prototype Strategy FCM (NPSFCM)

The *nearest prototype strategy (NPS)* is a simple modification of OCSFCM. The idea is to completely substitute the missing values of an incomplete data item by the corresponding feature values of the cluster prototype to which the data item has the smallest partial distance [HB01, TDK02]. The resulting algorithm is denoted by NPSFCM. The algorithm results from the OCSFCM by changing the third iteration step. The missing values of an incomplete data item are calculated as follows:

$$x_{kj} = v_{ij} \text{ with } D_{part}(x_k, v_i) = \min\{D_{part}(x_k, v_1), D_{part}(x_k, v_2), ..., D_{part}(x_k, v_c)\} \quad (2.16)$$

for $1 \leq k \leq n$ and $1 \leq j \leq d$.

The drawbacks of the OCSFCM even intensify in the NPSFCM. The imputation of missing values is influenced by their own imputation in the previous iteration. Substituting missing values by the feature values of cluster prototypes the incomplete data items illegitimately get higher membership degrees to the clusters. In this way, they have a greater impact on the computation of the cluster prototypes that for their part provide the basis for the missing value imputation.

2.3.5 Distance Estimation Strategy FCM (DESFCM)

The approach proposed in [SL01] benefits from the fact that not the data items themselves but the distances between them and the cluster prototypes are important for the calculation of the membership degrees. Therefore, the authors proposed not to estimate the missing values but the distances between the incomplete data items and the cluster prototypes. The algorithm is designed for the missing values missing at random (MAR). We refer to this method here as the *distance estimation strategy (DES)*. The DES version of the FCM algorithm is denoted by DESFCM.

The data set X is divided into the set of incomplete data items Z and the set of completely available data items Y, $X = Z \cup Y$. Additionally, Z is divided into the set of observed data values Z_{obs} and the set of missing data values Z_{mis}, $Z = Z_{obs} \cup Z_{mis}$. The DESFCM uses the version of the FCM algorithm that initializes the membership degrees at the beginning. In the first iteration step the cluster prototypes are computed based only on the completely available data items:

$$v_i = \frac{\sum_{k=1}^{|Y|}(u_{ik})^m y_k}{\sum_{k=1}^{|Y|}(u_{ik})^m}, \qquad 1 \leq i \leq c \ , y_k \in Y. \tag{2.17}$$

In the second iteration step the membership degrees for all $y_k \in Y$ are calculated according to Formula (2.4). For all incomplete data items $z_k \in Z$ the membership degrees to the clusters are computed according to the same formula but replacing the distance function $d^2(z_k, v_i)$ by the distance function that is defined as follows:

$$d^2(z_k, v_i) = (z_{k1} - v_{i1})^2 + (z_{k2} - v_{i2})^2 + ... + (z_{kd} - v_{id})^2 \ \text{ for } 1 \leq k \leq |Z| \text{ and } 1 \leq i \leq c \tag{2.18}$$

with

$$(z_{kj} - v_{ij})^2 = \frac{\sum_{k=1}^{|Y|} u_{ik}(y_{kj} - v_{ij})^2}{\sum_{k=1}^{|Y|} u_{ik}} \quad \text{for all } z_{kj} \in Z_{mis} \ .$$

In the new distance function the distances between the missing values of the incomplete data items and the corresponding feature values of the cluster prototypes $(z_{kj} - v_{ij})^2$ are imputed by the weighted mean distances between the feature values of completely available data items and the corresponding feature values of the cluster prototypes. To take the distance of an incomplete data item to the cluster prototype into account, the authors assume that the weighted distance $(z_{kj} - v_{ij})^2/\sigma_j^2$ is linearly dependent on the weighted distance between z_{kl}, $\forall l \neq j$, and v_{ij}. Thus, they estimate the weighted distance $(z_{kj} - v_{ij})^2/\sigma_j^2$ as follows:

$$\begin{aligned}(z_{kj} - v_{ij})^2/\sigma_{ij}^2 = {} & w_1(z_{k1} - v_{i1})^2/\sigma_{i1}^2 + ... + w_{j-1}(z_{k(j-1)} - v_{i(j-1)})^2/\sigma_{i(j-1)}^2 \\ & + w_j \frac{\sum_{k=1}^{|Y|} u_{ik}(y_{kj} - v_{ij})^2}{\sum_{k=1}^{|Y|} u_{ik}}/\sigma_{ij}^2 + w_{j+1}(z_{k(j+1)} - v_{i(j+1)})^2/\sigma_{i(j+1)}^2 \\ & + ... + w_d(z_{kd} - v_{id})^2/\sigma_{id}^2\end{aligned} \tag{2.19}$$

where

$$\sigma_{ij}^2 = \frac{\sum_{k=1}^{|Y|} u_{ik}(y_{kj} - v_{ij})^2}{\sum_{k=1}^{|Y|} u_{ik}} \quad \text{for } 1 \leq i \leq c, 1 \leq j \leq d \tag{2.20}$$

is the scatter of the cluster C_i in the jth dimension and w_j, $1 \leq j \leq d$, is the weighting parameter that indicates the importance of the jth feature. If there is no knowledge about the importance of the features, all features are considered to be equally important, i.e. $w_j = 1/d \ \forall j \in \{1, ..., d\}$ is chosen.

Unlike the other estimation strategies presented above, the DESFCM takes the scatter of the clusters during the estimation of the distances into account because the estimation formula of the distances between the missing values of the incomplete

data items and the corresponding feature values of the cluster prototypes $(z_{kj} - v_{ij})^2$ equals to the Formula (2.20) for the calculation of the scatter of clusters. Besides the limitations of the DESFCM mentioned above, this approach can be applied as long as the set of completely available data items is representative for the entire data set because the calculation of the cluster prototypes and the estimation of distances between the incomplete data items and the cluster prototypes are performed on the basis of the completely available data items.

2.3.6 Summary

In this section we described different strategies for adapting the fuzzy c-means clustering algorithm to incomplete data. We discussed the advantages and the limitations of the presented approaches. In the next chapter we present a new approach for adapting the basic FCM algorithm to incomplete data. This method can be regarded as the extension of the OCSFCM and the NPSFCM that takes the scatter of clusters during the estimation of missing values into account but avoids some drawbacks of the DESFCM.

3

Fuzzy Clustering of Incomplete Data Based on Cluster Dispersion

Clustering algorithms are used to identify groups of similar data objects within large data sets. Since traditional clustering methods were developed to analyze complete data sets, they cannot be applied to many practical problems because missing values often occur in the data sets. Approaches proposed for adapting clustering algorithms to incomplete data work well on uniformly distributed data sets. However, in real world applications clusters are generally differently scattered. In this chapter [1] we present a new approach for adapting the FCM algorithm to incomplete data. It can be regarded as an extension of the *optimal completion strategy* that uses the information about the dispersion of clusters. In the experiments on artificial and real data sets we show that our approach outperforms the other clustering methods for incomplete data.

3.1 Introduction

Clustering is an important technique for automatic knowledge extraction from large amounts of data. Its task is to identify groups or clusters of similar objects within a data set [HK00]. Data clustering is used in many areas, including database marketing, web analysis, information retrieval, bioinformatics, and others. However, if clustering methods are applied on real data sets, a problem that often comes up is that missing values occur in the data sets. Missing values could be caused for example by problems or failures during the data collection, data transfer, data cleaning or as a result of the data fusion from various data sources. Depending on the cause of missingness, missing

[1] This chapter is a revised and updated version of [HC10b].

values can be missing completely at random or depending on the values of variables in the data set.

Traditional clustering methods were developed to analyze complete data. In cases where the completion of data sets by repeated data collection is undesirable or impossible, e.g. for financial or time reasons, there is a need for data analysis methods handling incomplete data. In the previous chapter we presented different existing fuzzy clustering algorithms adapted to incomplete data. However, the results of the experiments conducted in [Him08, HC10a] have shown that these methods work well as long as the clusters are similarly scattered. But in real world applications clusters generally have different dispersions. In this chapter we present a new approach for adapting the FCM algorithm to incomplete data. Our method can be regarded as an extension of the *optimal completion strategy* that takes the dispersion of clusters during the estimation of missing values into account [HC10b]. In experiments on artificial and real data sets, we demonstrate the capabilities of our approach and show the benefit over the basic form of the OCSFCM on incomplete data sets with differently scattered clusters. We give a particular attention to the analysis of the performance of the methods depending on the different missing-data mechanisms and the percentage of missing values in the data set.

The remainder of the chapter is organized as follows. In the next section we describe our idea for missing data imputation using the information about the cluster scatters and present the modified FCM algorithm. The evaluation results of our method and the comparison with the OCSFCM are presented in Section 3.3. In Section 3.4 we close this chapter with a short summary and the discussion of future research.

3.2 Fuzzy Clustering of Incomplete Data Based on Cluster Dispersion

In the previous chapter we described the optimal completion and the nearest prototype strategies to adapt the fuzzy c-means clustering algorithm to incomplete data. The OCSFCM and the NPSFCM impute the missing values of an incomplete data item either by the weighted mean of all cluster centers or by the corresponding feature values of the nearest cluster prototype. As we can see in Formulae (2.15) and (2.16) the missing values are estimated only depending on the distances between the incomplete data items and the cluster centers. In the OCSFCM the influence of the cluster centers is expressed by the membership degrees but they are based on the distances between the data items and the cluster prototypes. Hence, the OCSFCM and the NPSFCM completely disregard the information about the scatters of clusters during the imputation of missing values. However, real world data sets usually contain differently scattered

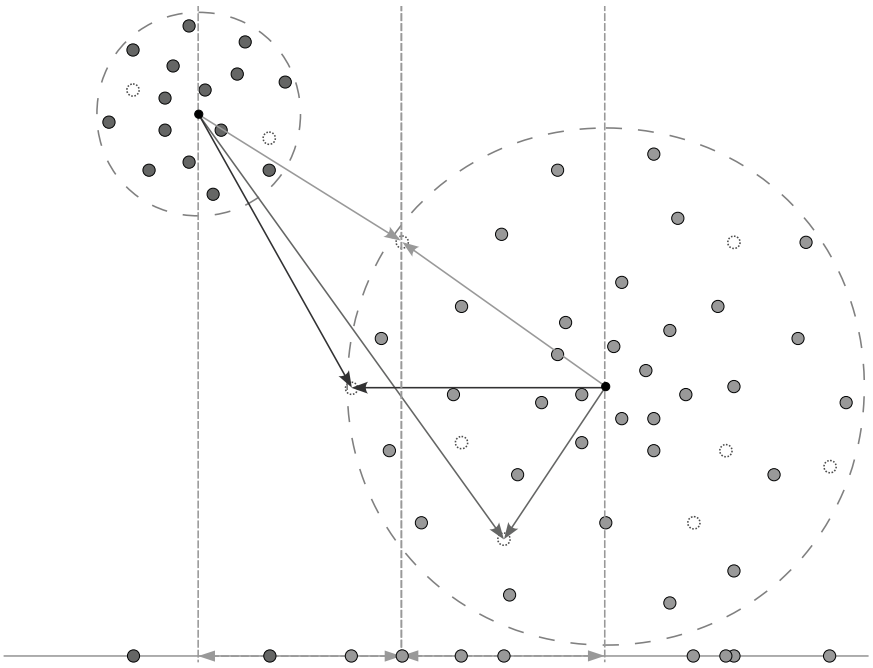


Figure 3.1: Two differently scattered clusters.

clusters where the data items located on the boundaries of large clusters have larger distances to the cluster centers than the marginal data items of smaller clusters.

Figure 3.1 illustrates the problem outlined above. It shows a data set where some of the data items contain missing values in one dimension. The data items are distributed in two differently scattered clusters. Relying on the distances between the data items and the cluster centers the incomplete data item with grey arrows will be correctly assigned to the large cluster. Since the distances between the incomplete data item with light grey arrows and the both cluster centers are equal, the missing value in this data item will be imputed by the weighted mean of both cluster centers and it will be correctly assigned to the large cluster. Although the incomplete data item with dark grey arrows visually belongs to the large cluster, it will be incorrectly assigned by the OCSFCM and the NPSFCM to the small cluster because the distance between this data item and the center of the small cluster is smaller than the distance between the data item and the center of the large cluster.

If missing values of incomplete data items are estimated only on the basis of the distances between the data items and the cluster centers, it is highly possible that the marginal data items of a large cluster are falsely assigned to the nearest small cluster. As we already mentioned in the previous chapter, the estimation of missing values and the calculation of the cluster centers influence each other. Therefore, ignoring the cluster dispersions during the imputation of missing values leads to the distorted computation of the cluster centers and consequently to inaccurate clustering results. The results of the experiments conducted in [Him08], [HC10a] reflect this fact. The

OCSFCM and the NPSFCM produced more accurate results on uniformly distributed data sets than on the data sets with differently scattered clusters. In order to improve the estimation of missing values in OCSFCM and NPSFCM, in [HC10b] we proposed a new membership degree u_{ik}^* for the imputation of missing values. Our membership degree determines the influence of the cluster centers taking the cluster dispersions into account.

3.2.1 A New Membership Degree using Cluster Dispersion

We divide the data set X into X_{com}, the set of completely available data items, and X_{inc}, the set of data items with missing values. Furthermore, we divide the feature set F into F_{com}, the set of completely available features, and F_{inc}, the set of features where missing values occur. For each cluster C_i, $1 \leq i \leq c$, we calculate the squared dispersion s_i^2 as a squared averaged distance of data items to their cluster centers according to Formula (3.1).

$$s_i^2 = \frac{1}{\mid C_i \cap X_{com} \mid -1} \sum_{x_k \in C_i \cap X_{com}} \sum_{f \in F_{com}} (x_k.f - v_i.f)^2 \,, \tag{3.1}$$

where $x_k \in C_i \Leftrightarrow u_{ik} = max\{u_{1k}, ..., u_{ck}\}$ and $\mid C_i \cap X_{com} \mid \geq 2$. Instead of using the fuzzy membership degrees for the calculation of the cluster dispersion we use the crisp membership degrees computing the cluster dispersion similar to the sample variance. In this way we try to avoid reducing the influence of distances between the cluster center and the data items located between clusters.

In Formula (3.1) we calculated the cluster dispersion only on the basis of completely available data items assuming that they are representative for the entire data set. Since the values in completely available features are available for incomplete data items as well, we can also include the data items with missing values during the calculation of cluster dispersion. If missing values occur in a large number of data items but only in few attributes, this alternative enables to include more available values during the calculation of cluster dispersion than in Formula (3.1) described. Furthermore, in this way we avoid the restriction that each cluster must consist at least of two completely available data items. Then the cluster dispersion is computed as follows:

$$s_i^{*^2} = \frac{1}{\mid C_i \mid -1} \sum_{x_k \in C_i} \sum_{f \in F_{com}} (x_k.f - v_i.f)^2 \quad \text{for } 1 \leq i \leq c \,, \tag{3.2}$$

where $x_k \in C_i \Leftrightarrow u_{ik} = max\{u_{1k}, ..., u_{ck}\}$ and $\mid C_i \mid \geq 2$.

So far we calculated the cluster dispersion on the basis of completely available features. In this way we tried to avoid the distortion of the cluster dispersion caused by missing values not missing at random (NMAR). On the other hand this restriction

makes our approach inapplicable to incomplete data sets where missing values occur in many attributes. Assuming that missing values in the data set are not NMAR, we can compute the cluster dispersion using all available data values according to the partial distance strategy. In this case the cluster dispersion is calculated as follows:

$$s_i^{**^2} = \frac{1}{\mid C_i \mid -1} \sum_{x_k \in C_i} D_{part}(x_k, v_i) \quad \text{for } 1 \leq i \leq c \ , \tag{3.3}$$

where $x_k \in C_i \Leftrightarrow u_{ik} = max\{u_{1k}, ..., u_{ck}\}$ and $\mid C_i \mid \geq 2$. The partial distances $D_{part}(x_k, v_i)$ are computed according to Formula (2.13).

In our approach we integrate the cluster dispersion in the membership degree formula for the estimation of missing values as follows:

$$u_{ik}^* = \frac{s_i^2 d^2(v_i, x_k)^{1/(1-m)}}{\sum_{j=1}^{c} s_j^2 d^2(v_j, x_k)^{1/(1-m)}}, \qquad 1 \leq k \leq n \text{ and } 1 \leq i \leq c \ . \tag{3.4}$$

The larger the dispersion of the cluster and the smaller the distance between the data item and the cluster center, the higher the new membership degree is. If all clusters are uniformly distributed, then the membership degree u_{ik}^* depends only on the distances between the data item and cluster centers.

The new membership degree works with all formulae for the calculation of cluster dispersion outlined above. Depending on the arrangement of missing values in the data set, the cluster dispersion in Formula (3.4) can be calculated in an appropriate way according to the Formulae (3.1) - (3.3). Since the membership degree u_{ik}^* is normalized by the dispersions of all clusters, the sum of the new membership degrees for each data item equals 1. Thus, Conditions (2.1) and (2.2) are fulfilled and the resulting algorithm maintains the property of a probabilistic clustering algorithm.

3.2.2 FCM for Incomplete Data based on Cluster Dispersion

We integrate the new membership degree for the estimation of missing values in the OCSFCM algorithm. We refer the resulting algorithm to as *Fuzzy C-Means Algorithm for Incomplete Data based on Cluster Dispersion (FCMCD)*. The working principle of FCMCD is depicted in Algorithm 2. The cluster prototypes and the missing values are initialized at the beginning of the algorithm. The membership degrees and the cluster prototypes are updated in the first two iteration steps in the same way as in the OCSFCM. In the third iteration step the missing values are estimated depending on the cluster prototypes and the dispersion of clusters according to Formula (3.5).

$$x_{kj} = (\sum_{i=1}^{c} (u_{ik}^*)^m v'_{ij}) / (\sum_{i=1}^{c} (u_{ik}^*)^m), \qquad 1 \leq k \leq n \text{ and } 1 \leq j \leq d \ . \tag{3.5}$$

Algorithm 2 FCMCD(X, c, m, ϵ)

Require: X is a d-dimensional incomplete data set with n data items, $2 \leq c \leq n$ is a number of clusters, $m > 1$ is a fuzzification parameter, $\epsilon > 0$ is a termination accuracy

1: Initialize the set of data centers $v' = \{v'_1, ..., v'_c\}$
2: Initialize all missing values x_{kj} in X with random values
3: $v = \{\}$
4: **repeat**
5: $v = v'$
6: Calculate the membership degrees u_{ik} of each data item x_k to each cluster C_i according to Formula (2.4) // *Step 1*
7: Calculate the set of new cluster prototypes $v' = \{v'_1, ..., v'_c\}$ according to Formula (2.5) // *Step 2*
8: Impute the missing values x_{kj} according to Formula (3.5) // *Step 3*
9: **until** $\|v - v'\| < \epsilon$
10: **return** v'

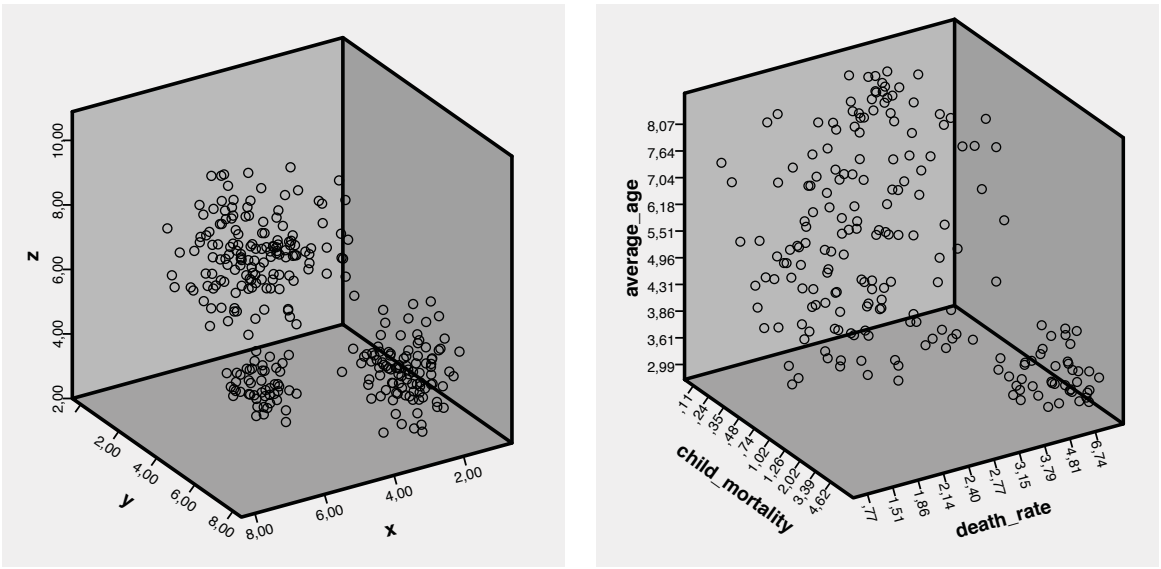
The new membership degrees for the estimation of missing values are updated in each iteration. As in the OCSFCM and the NPSFCM the imputation of missing values and the calculation of cluster prototypes influence each other because the estimation of missing values is based on the assignment of data items to clusters and the cluster centers that are computed using the estimated and the available feature values.

The NPSFCM can also be modified using the new membership degree in a straightforward way. The missing values of incomplete data items are substituted by the corresponding feature values of the cluster prototypes to which the data item has the highest membership degree u^*_{ik}. The comparison of the numerators of the membership degrees u^*_{ik} is here sufficient.

3.3 Data Experiments

3.3.1 Test Data

We have performed several experiments on artificial and real data sets to demonstrate the capabilities of our approach. The artificial data set was generated by a composition of three 3-dimensional Gaussian distributions. It consists of 300 data points which are unequally distributed on three differently sized and scattered clusters with 52, 101 and 147 data items. The real world data set contains the demographic information about 203 countries. For our experiments we only used the attributes average age, death rate and child mortality. We tested the FCM algorithm for different numbers of clusters and evaluated partitioning results using the cluster validity indexes NPC [Rou78, Dav96] and the Xie-Beni index [XB91]. We achieved the best results for two



(a) (b)

Figure 3.2: Three-dimensional representation of (a) artificial and (b) real data sets.

clusters with 46 and 157 data items. For our experiments we normalized the feature values in both data sets to the range $[0, 10]$. Since dependent features do not provide additional information for the clustering, we ensured that values of different features are uncorrelated in both data sets. The data sets are depicted in Figure 3.2.

To generate the incomplete data sets, both data sets were modified by successively removing values in two of three features with different probabilities according to a multivariate missing-data pattern [LR02]. The percentage of missing values was calculated in relation to all feature values in the data sets. Since missing values can cause a random or a conditional reduction of a data set, we deleted the values from the test data sets according to the common missing-data mechanisms MCAR, MAR and NMAR using a missing data generator outlined below. In this way we wanted to test whether the performance of the clustering algorithms depends on different missing-data mechanisms.

3.3.2 Missing Data Generator

We implemented a missing data generator to create incomplete data sets from complete data. All incomplete data sets used in experiments described in this thesis were generated using this missing data generator. Our missing data generator removes values from complete data sets with given probability according to the chosen missing-data mechanism. The working principle of our generator is very similar to the missing data generator DataZapper described in [WKN10] that was published after our generator was already implemented and used for our first experiments.

The missing data generator supports common data formats that are supported by

the Weka machine learning platform [HFH+09] which is a standard toolkit in the machine learning environment. After loading the complete data set all relevant metadata like attributes, number of data items, number of missing values are extracted. Since for some experiments the correlation between the feature values is important, the Pearson correlation coefficient between the values of different features is computed. After specifying the percentage of missing values, the user has a choice to select the attributes where missing values should occur. Since the percentage of missing values relates to all feature values in the data sets, the generator internally computes the percentage of missing values per attribute evenly distributing the missing values to all selected attributes.

The user has a choice from three supported missing-data mechanisms: MCAR, MAR and NMAR. The default missing-data mechanism is MCAR. In the case MAR or NMAR mechanisms are chosen, the user is requested to specify the attribute on which values the missingness of values in the same or other attributes should be dependent. If the missing-data mechanism MCAR is chosen, the values are removed from the entire value range of the attributes. In the case the values should be removed according to the missing-data mechanisms MAR or NMAR, the generator internally sorts the data items by the attribute on which values the missingness of values in the same or the other attributes should be dependent. Then it removes values from the upper or low value range of the specified attributes. Theoretically, if the average feature values are missing or the missingness of values in an attribute depends on the average values in the other attribute, the missing values are NMAR or MAR. Unfortunately, in this case the missing-data mechanisms MAR and NMAR cannot be verified using the statistical test procedures without any knowledge about the complete data set. Since our aim is to generate incomplete data sets where the missing-data mechanisms can be verified, the missing data generator removes values from the upper or low value range of the attributes. The range in which the feature values are removed is computed from the percentage of missing values per attribute plus the so called gap of 10% to avoid an artificial generation of incomplete data sets. After all feature values are removed, the missing data generator verifies the predefined missing-data mechanism via suitable statistical test procedures. In the case, the missing-data mechanism cannot be verified, the missing data generator repeats the whole procedure successively reducing the gap by 1% in each iteration until it manages to create an incomplete data set according to the given requirements. If the missing data generator does not manage to create the incomplete data set after zeroing the gap, it stops with a message that an incomplete data set cannot be generated from the given complete data set satisfying the specified requirements. Usually it is true for the MAR or NMAR missing-data mechanisms. In this case the user can specify another attribute on which values the missingness of values should depend, and start another attempt to generate an incomplete data set.

If the missing data generator could not create an incomplete data set with missing values MCAR, it is highly possible that the missing-data mechanism MCAR could not be verified for the data set with randomly removed feature values. In this case it is advisable to start another attempt to generate an incomplete data set without any changes.

3.3.3 Experimental Setup

In our experiments we proceeded as follows: first we clustered the complete data sets with the basic FCM algorithm to find out the actual distribution of data items into clusters. We used these clusterings as a baseline for the comparison. Then we clustered the incomplete data sets with the basic versions of the OCSFCM, the NPSFCM, and the FCMCD using Formulae (3.1) and (3.2) for the cluster dispersion. To create the test conditions as real as possible, we initialized the cluster prototypes with random values at the beginning of the algorithms. For the stopping criterion $\|v - v'\| < \epsilon$ we used the Frobenius norm distance defined in Formula (3.6).

$$\|V - V'\|_F = \sqrt{\sum_{i=1}^{c}\sum_{j=1}^{d} |v_{ij} - v'_{ij}|^2}, \tag{3.6}$$

where V and V' are the sets of old and updated cluster centers. In all our experiments we set the value ϵ to 0.0001.

3.3.4 Experimental Results

In this subsection we present the results of our experiments organized according to the missing-data mechanisms. Since the experimental results for the modified versions of the OCSFCM and the NPSFCM were very similar, below we present the results on the example of the modified versions of the OCSFCM. We refer the modified OCSFCM algorithm using Formula (3.1) to as FCMCD. The modified OCSFCM algorithm using Formula (3.2) is referred to as FCMCD*.

3.3.4.1 Test Results for Data with Missing Values MCAR

Figure 3.3 presents the performance results for OCSFCM, FCMCD and FCMCD* on the artificial and the real data sets with missing values MCAR. To evaluate the performance of algorithms, we compared the averaged accuracy (the percentage of correctly classified data items) obtained over 30 trials in relation to the percentage of missing values in the data sets. For 0% missing values, all approaches reduce to the basic FCM algorithm and, therefore, produced the same partitionings of the data sets as the FCM

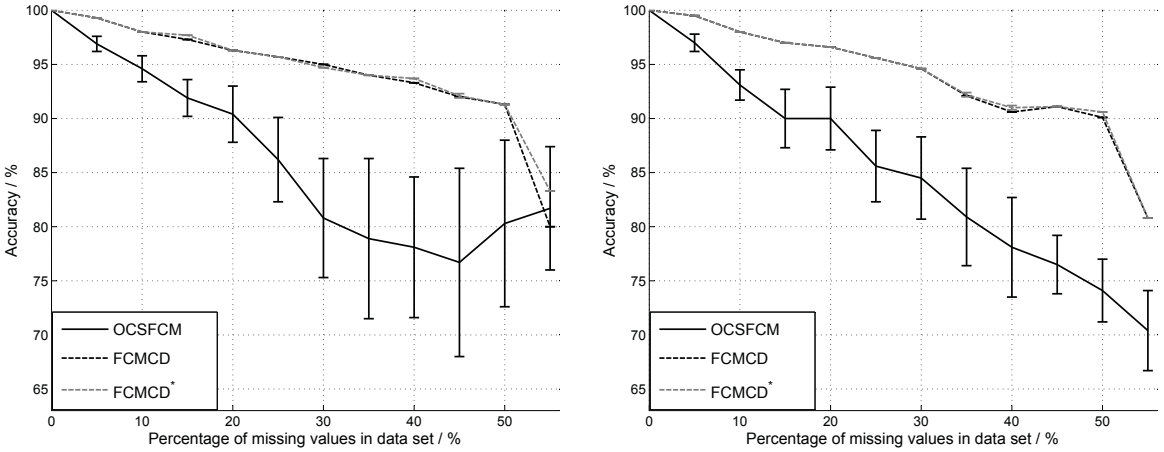


Figure 3.3: Averaged results of 30 trials for the accuracy on (a) artificial and (b) real data sets with missing values MCAR (bars indicate +/- on standard deviation).

algorithm. For 5% or more missing values in the data sets, the performance results of FCMCD and FCMCD* were quite similar. The FCMCD* algorithm performed only slightly better than the FCMCD algorithm for a large percentage of missing values in the data sets. Since the missing values were MCAR, the complete data items represented the entire data set well so that the cluster dispersion could be well estimated on the basis of the complete data items. Both algorithms produced a lower number of misclassification errors than the basic version of the OCSFCM. The averaged accuracy of these two algorithms exceeded 90% when the amount of missing values was not larger than 50%. Moreover, FCMCD and FCMCD* were considerably more stable than OCSFCM. With a few exceptions these algorithms produced the same partitionings of data items independent of the initial partitioning. In contrast, OCSFCM produced from trial to trial different partitionings of data items into clusters. Consequently, different numbers of misclassification errors were obtained by OCSFCM in each trial. We captured the performance variations of OCSFCM with standard deviation (bars in figures). Furthermore, the standard deviation for OCSFCM significantly increased with the increasing number of missing values in the data sets.

3.3.4.2 Test Results for Data with Missing Values MAR

The performance results for OCSFCM, FCMCD and FCMCD* on both data sets with missing values MAR are shown in Figure 3.4. All algorithms performed in a quite similar way as long as the percentage of missing values was relatively low. We observed the significant differences in the performance of the algorithms when the amount of missing values in the data sets exceeded 15%. In comparison to the results on the data sets with missing values MCAR, the algorithms performed somewhat worse on the data

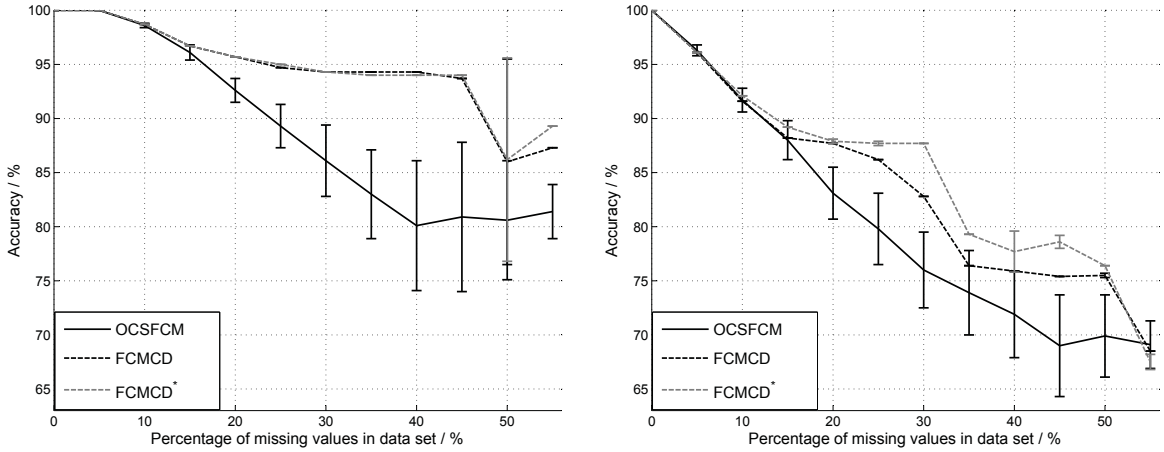


Figure 3.4: Averaged results of 30 trials for the accuracy on (a) artificial and (b) real data sets with missing values MAR (bars indicate +/- on standard deviation).

with missing values MAR, especially on the real data set. This is due to the fact that missing values MAR occurred in the data items depending on the values of available features and thus, they occurred in the data items with particular properties. In this way, the completely available data items did not represent the entire data set anymore. Therefore, the computation of the cluster scatters was distorted by the complete data items and the missing values MAR could not be estimated as well as the missing values MCAR. Additionally, the computation of the cluster prototypes was affected by the inaccurate imputation of missing values. All that led to more misclassifications with the increasing number of missing values in the data sets. Since in FCMCD* the cluster dispersions was computed on the basis of feature values of all data items, this algorithm performed slightly better than the FCMCD using Formula (3.1) for the calculation of the cluster dispersions.

3.3.4.3 Test Results for Data with Missing Values NMAR

Figure 3.5 shows the experimental results for OCSFCM, FCMCD and FCMCD* on the data sets with missing values NMAR. As in the case of missing values MAR, the performance results of the algorithms are worse than on the data sets with missing values MCAR. Since missing values NMAR occur in the data items with particular properties, some clusters may be affected by the absence of feature values more than the others. In this way, these clusters may not be identified as such by the clustering algorithms. Consequently, the clustering results produced by the algorithms on the data sets with missing values NMAR may be different from the actual partitioning.

In our experiments, the OCSFCM split clusters with a low number of incomplete data items in several clusters and distributed data items of clusters with a high number

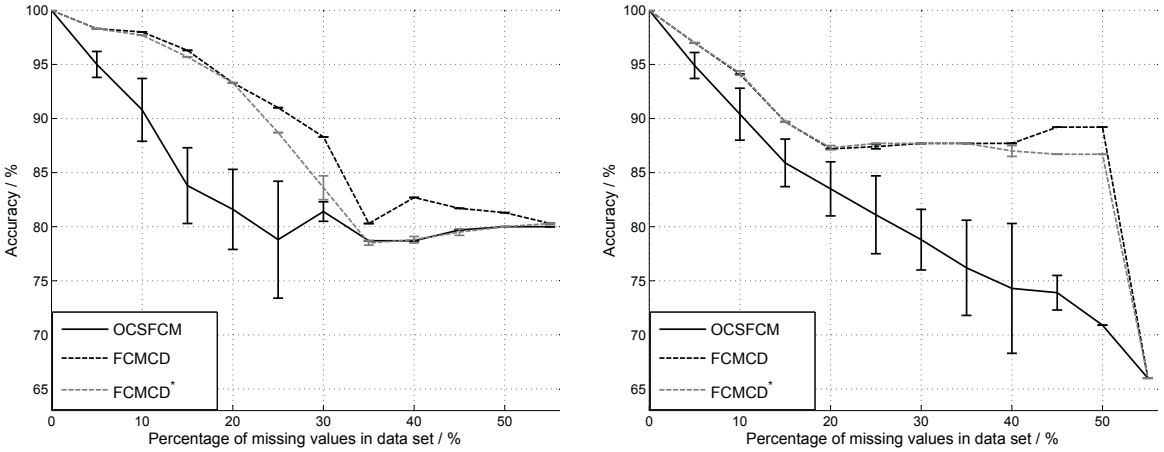


Figure 3.5: Averaged results of 30 trials for the accuracy on (a) artificial and (b) real data sets with missing values NMAR (bars indicate +/- on standard deviation).

of incomplete data items to other clusters. In contrast, FCMCD and FCMCD* strived to maintain the clustering structure by imputing the missing values with regard to the cluster scatters. As Figure 3.5 shows, FCMCD and FCMCD* achieved better performance results than the basic version of the OCSFCM on the data sets with missing values NMAR. The performance of FCMCD and FCMCD* declined and converged to the performance of OCSFCM only for a large number of missing values in the data sets.

3.3.5 Prototype Error and Runtime

In our experiments we also compared the runtime (here: the mean number of iterations to termination) of the algorithms. Table 3.1 gives the average number of iterations required to terminate for the clustering algorithms OCSFCM, FCMCD, and FCMCD* obtained over 30 trials on the real data set with missing values MCAR. Like the basic FCM all algorithms required about 8-12 iterations to termination on the complete data sets. With the increasing number of missing values in the data set, the mean number of iterations strongly increased. From 35% of missing values in the data set, OCSFCM required almost the double number of iterations to termination than the other two algorithms. There were no significant differences in the runtime between FCMCD and FCMCD*.

For some applications, the information about the location of clusters is as much important as the information about the partitioning of the data items into clusters. Therefore, we analyzed the algorithms regarding the determination of the cluster prototypes in presence of missing values in the data sets. Table 3.2 gives the average Frobenius norm distance between the terminal cluster prototypes obtained by the ba-

Table 3.1: The average number of iterations to termination.

% missing	mean number of iterations		
	OCSFCM	FCMCD	FCMCD*
5	22.2	**17.7**	17.8
15	27.8	21.4	**19.8**
25	35.4	**22.7**	**22.7**
35	46.6	**26.3**	26.5
45	85.5	**48.0**	49.8
55	143.5	**89.6**	102.4

sic FCM algorithm on the complete real data set and the corresponding terminal cluster prototypes computed by the three algorithms on the real data set with missing values MCAR. When the percentage of missing values was not greater than 40% in the data set, the terminal cluster prototypes obtained by FCMCD and FCMCD* were considerably more accurate than the terminal prototypes obtained by OCSFCM. For 45% and more of missing values in the data set, OCSFCM produced more accurate terminal cluster prototypes than its extended versions. It bears mentioning that in this range the accuracy obtained for FCMCD and FCMCD* was still about 10% higher than for OCSFCM (compare Figure 3.3 (b)). This is due to the fact that OCSFCM imputes the missing values by values, which are very close to the corresponding feature values of the nearest cluster prototype. In this way the cluster prototypes are better maintained, but the clustering structure may get lost. In order to maintain the clustering structure, FCMCD takes the cluster dispersions into account during the calculation of the membership degrees and, consequently, the cluster prototypes. In this way FCMCD produced a lower number of misclassification errors than OCSFCM, but the terminal cluster prototypes obtained by FCMCD were less accurate in the case of high percentage of missing values in data set.

3.4 Conclusions and Future Work

The already existing fuzzy c-means algorithms for incomplete data that impute missing values leave the cluster dispersions out of consideration during the estimation of missing values. For this reason, they fail to work on the incomplete data sets with differently scattered clusters. Our approach uses a new membership degree for the missing values imputation based on the cluster dispersion. In experiments on the artificial and the real data sets with differently scattered clusters, we have shown that our approach

Table 3.2: The average prototype error.

% missing	mean prototype error		
	OCSFCM	FCMCD	FCMCD*
5	0.1756	0.1065	**0.1043**
15	0.4237	**0.1504**	**0.1504**
25	0.5467	**0.1585**	**0.1585**
35	0.7468	**0.3265**	0.3283
45	**0.8791**	1.0844	1.1387
55	**1.1558**	2.2040	2.1811

outperforms the basic versions of the OCSFCM and the NPSFCM. It produced less misclassification errors, it is more stable, it required less iterations to termination, and it produced more accurate terminal cluster prototypes in the cases, where the percentage of missing values in the data set was not greater than 40%.

Although the data experiments have shown the promising results for our method, we believe that we can improve them by reducing the influence of the imputed values during the computation of cluster prototypes. In [Tim02] the authors proposed to reduce the weight of incomplete data items while the computation of cluster centers achieving a slightly improvement in the performance of their method. Our idea is to exclude the missing values from the computation of cluster prototypes calculating them only on the basis of available feature values as in the PDSFCM. In this way, the imputation of missing values will be influenced by the cluster prototypes but the computation of cluster prototypes will not be influenced by the imputed values. Furthermore, our experiments showed that all clustering methods performed worse on the data sets with missing values MAR and NMAR than on the data with missing values MCAR. In order to improve the performance of our approach on the incomplete data sets with a conditional absence of values, we also plan to combine our approach with an approach presented in [TDK04] that uses class specific probabilities for missing values.

Another idea to improve the accuracy of clustering results on incomplete data is imputing the missing values with a sophisticated imputation technique in a preprocessing step and clustering the completed data sets with clustering algorithms for uncertain data. In [Him09] we imputed missing values with the EM algorithm and clustered the completed data with the basic FCM algorithm. The performance of our approach on data sets with equally scattered clusters was similar to the performance of PDSFCM, OCSFCM and NPSFCM. The computational costs were, of course, much higher than for the other approaches. Anyway, we expect a better estimation of missing values

MAR and NMAR using the regression based imputation approaches because they consider the relationships between the feature values. On the other hand, the clustering approaches for uncertain data treat the available and uncertain (in our case imputed) values in different ways. Combining these both features we expect the more accurate clustering results for this approach on data with conditional absence of values than for the clustering algorithms adapted to incomplete data.

Furthermore, in all our experiments we assumed the real number of clusters to be known because we computed it on complete data sets using the cluster validity indexes NPC [Rou78, Dav96] and the Xie-Beni index [XB91]. However, in the real world applications the number of clusters is generally unknown in advance. Therefore, in the next chapter of this thesis we analyze in an extensive study to what extend the partitioning results produced by the clustering methods for incomplete data reflect the distribution structure of the data items and whether the optimal number of clusters can be determined using the cluster validity indexes.

4

Cluster Validity for Fuzzy Clustering of Incomplete Data

The quality of the resulting partitioning of the data produced by clustering algorithms strongly depends on the assumed number of clusters. In this chapter, we address the problem of finding the optimal number of clusters on incomplete data using cluster validity functions. We describe different cluster validity functions and adapt them to incomplete data according to the available case approach.

4.1 Introduction and Related Work

Generally, cluster analysis is defined as an unsupervised learning technique for detecting subgroups or clusters of similar objects. Since clustering is an unsupervised process, no *a priori* knowledge about the resulting distribution of data objects into clusters can be assumed. In order to achieve an optimal partitioning of a data set, cluster analysis is usually performed as a multi-level process that involves determining the optimal number of clusters, partitioning data objects into clusters and validation of found clusters. In the literature, a great number of well-performing algorithms have been proposed that assign data objects into a pre-defined number of hard or fuzzy partitions. However, the resulting partitioning of data produced by these methods satisfactorily reflects the real distribution of data objects in a data set if the number of clusters used by the clustering algorithm corresponds to the real number of clusters in the data set.

Determining the optimal number of clusters turned out to be a difficult problem and much effort has been done in different directions. Below we briefly describe different

approaches for finding the optimal number of clusters.

- One of the best known and most commonly used methods for determining the optimal number of clusters in a data set is carrying out the clustering algorithm for different numbers of clusters and to assess the partitioning results using a cluster validity function after each trial. The data partitioning which is rated with the best value for the *Cluster Validity Index (CVI)* is regarded as the optimal partitioning of the data set. Since the first fuzzy clustering algorithm was developed, a variety of post-clustering measures has been proposed. Different cluster validity functions consider different aspects of an optimal partitioning like the clarity of the assignment, separation between clusters, compactness within clusters, etc. and therefore they yield different results for different data distributions. Most cluster validation indexes provide the optimal number of clusters on data with a simple distribution structure, but they fail if some clusters partly overlap or if clusters are hierarchically ordered building distant groups of clusters. To overcome those problems and to determine the optimal number of clusters in data with a complicated distribution structure, more and more complicated cluster validation measures have been proposed.

- Another method is estimating the optimal number of clusters from the distance matrix between data items in a pre-processing step. The idea of this approach is to reorder the rows and columns of the dissimilarity matrix between data items and to represent it in a grey-scale image, where the clusters are highlighted as dark blocks along the diagonal. The main challenge of this method is the way of processing the dissimilarity matrix so that the final image clearly highlights the cluster tendency. The idea of representing the data structure in an image is not new, the first attempts have been made in the seventies of the last century [Lin73]. Determining the optimal number of clusters from the reordered dissimilarity images (RDI) found its renaissance during the last decade after Bezdek and Hathaway proposed their algorithm *Visual Assessment of (Cluster) Tendency (VAT)* [BH02]. Since then different and more sophisticated improvements of the VAT method have been proposed for different types of data [HBH05, HBH06, BHH07, SHB08, WGB$^+$10].

- Stability based cluster validity methods represent another approach for determining the optimal number of clusters in a data set. They are based on the assumption that the most stable partitioning of a data set under perturbation of data contains the optimal number of clusters [Bre89, BK06]. In general, there are two strategies for the evaluation of clustering stability: supervised and unsupervised. In the supervised scheme the data set is divided into equally sized and

non-overlapping training and test sets several times. Each training set is used for training a classifier that predicts the assignment of data items into clusters in the corresponding test set. The average stability measure is calculated by comparing the predicted and the real partitionings of the test sets [LRBB04, DF02]. In the unsupervised scheme the data set is repeatedly divided into several data subsamples with or without overlaps. Each subsample is clustered independently and the partitioning results are cross-compared in order to obtain the average stability measure [BHEG01, LJ03, Bor07, FGP$^+$10]. Similar to the first method the partitioning results of a data set are compared with each other for different numbers of clusters. Instead of using CVIs that assess each partitioning separately, stability based methods calculate a stability parameter comparing partitionings of different subsamples for each number of clusters.

In our overview we only focused on methods that have been proposed recently or are still widely used and discussed in the literature. Besides them there are several other approaches for estimating the optimal number of clusters in the data set. Worth to mention are methods that perform the clustering algorithms with a large number of clusters and separately assess each cluster in the partitioning after each run. Then, either similar and adjoining clusters are merged as in *Compatible Cluster Merging (CCM)* [KF92] and *Similar Cluster Merging (SCM)* [Stu98] strategies or small clusters are removed as in *Competitive Agglomeration Clustering (CAC)* strategy [FK97]. After that the clustering algorithm is performed with a smaller number of clusters. Unlike the approach using CVIs in these methods the cluster prototypes are adopted from the partitioning results of the previous run. On the one hand this strategy yields benefits by reducing the run time of clustering algorithms before the convergence in each run. On the other hand, since partitioning results of clustering algorithms might differ depending on the initialization in different trials, an inappropriate partitioning of the data set might distort the results of all following runs and the entire cluster validity process.

All aforementioned approaches for determining the optimal number of clusters were developed for complete data. In this chapter we address the question to what extent the optimal number of clusters can be found on incomplete data. Except the approach that determines the optimal number of clusters from the reordered dissimilarity images in a pre-processing step, all other methods presented in this section use partitioning results produced by the clustering algorithms. In our study, we use clustering algorithms described in the last chapter for partitioning incomplete data. We exclude approaches that determine the optimal number of clusters from the reordered dissimilarity images (RDI) from our consideration for two reasons. First, since those approaches are based on the distance matrix between data items, there is no straight way of adapting them

to incomplete data. In [BH02] authors proposed to estimate the distances between incomplete data items by partial distances [Dix79]. We consider this suggestion as limited because the calculation of the partial distance is based on the feature values that are available in both data items. Since data sets might contain incomplete data items that do not have values in the same dimensions, the partial distance function cannot be applied. Second, the experiments conducted in [Höf11] have shown that VAT like methods are not able to determine the optimal number of clusters if some clusters are hierarchically ordered building distant groups of clusters even if clusters within those groups are clearly separated from each other.

We also exclude stability based cluster validity methods from the consideration. On one side, the computational costs of those methods are very high because for each number of clusters the data set has to be repeatedly divided into several subsamples each of which has to be clustered. Although partitioning of subsamples is less computing intensive than partitioning of the entire data set, the number of subsamples to be partitioned suspends the benefit of computational costs. Moreover, to ensure the reliability of resulting partitionings of data subsamples the clustering algorithm has to be performed several times for different initializations of cluster prototypes. Additionally, the partitionings of subsamples have to be compared with each other for calculation of stability measure. On the other side, empirically stability based cluster validity methods often tend to underestimate the optimal number of clusters [Efi12, Bor07, FGP$^+$10] because clustering algorithms produce more stable partitionings of data for a small number of clusters. For the same reason stability based methods detect only rough data structures ignoring single clusters located close to each other.

We focus our consideration on the analysis and the adaption of different cluster validity indexes. In our opinion this approach conforms to the natural perception of clustering because the number of clusters results from the best partitioning of the data set in a learning process. Cluster validity functions assess the structure quality of the data partitionings and choose the one that conforms the definition of the clustering as best. Concerning partitioning of incomplete data, some cluster validity functions use only the information provided by the clustering methods adapted to incomplete data. Therefore, such CVIs can be used on incomplete data without any changes. We adapt the cluster validity indexes to incomplete data that use some additional information like the data set for their calculation. Since both the clustering algorithms and the cluster validity indexes are adapted to incomplete data, later in experiments we analyze on several data sets to what extend the partitioning results produced by the clustering methods for incomplete data reflect the distribution structure of the data items and whether the optimal number of clusters can be determined using the original and the adapted cluster validity indexes.

4.2 Cluster Validity Indexes for Incomplete Data

Since cluster validity indexes are computed using partitioning results obtained by clustering algorithms, the determined optimal number of clusters depends on both clustering methods and cluster validity indexes. While the cluster validity problem on complete data has been extensively examined and different cluster validity indexes were tested and compared with each other in the literature (see [WZ07] for example), there are only few works that have analyzed this problem on incomplete data [HHC11, HCC12]. In this section we present an overview of different cluster validity indexes. We differentiate them into different categories according to the type of information they use and the aspects of an optimal partitioning they consider. Some of the cluster validity indexes combine the properties of more than one category. We adapt the cluster validity indexes to incomplete data pursuing the available case approach. The advantage of adapting the CVIs in this way is that they automatically reduce to the original versions of the corresponding CVIs without additional computational costs in the case of the complete data sets. So they can be used on any data set regardless of whether it is complete or incomplete. Although some clustering algorithms adapted to incomplete data estimate missing values in the data set, involving only available feature values while calculation of cluster validity indexes turned out to provide better validity results than involving estimated values [HHC11].

4.2.1 Cluster Validity using Membership Degrees

In this section we give an overview of cluster validity functions that use only membership degrees of data items to clusters for the calculation. While the *Partition Coefficient* and the *Partition Entropy* aggregate the membership degrees of data items to clusters, the *KKLL* and the *Overlap and Separation Indexes* consider the geometrical properties of a partitioning like overlap and separation between clusters. The cluster validity indexes of this category have the advantage that they can be used on incomplete data without any changes because they only use information that is provided by all clustering methods adapted to incomplete data.

4.2.1.1 Partition Coefficient

The *Partition Coefficient (PC)* [Bez74] rates a partitioning of a data set as optimal if data items are clearly assigned into clusters. This means that membership degrees should be close to 1 or close to 0. The index is defined as

$$V_{PC}(U, c) = \frac{1}{n} \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2. \tag{4.1}$$

The partition coefficient also evaluates the compactness within clusters implicitly because the membership degrees express the relative distance of data points to cluster centers. The compacter the clusters are, the clearer is the assignment of data points to clusters. In the optimal partitioning where all data points are clearly assigned to clusters, the partition coefficient achieves a value of 1. In the worst case when all data points are ambiguously assigned to clusters, i.e. the membership degrees to different clusters are equal, the partition coefficient achieves a value of $\frac{1}{c}$. Thus, the range of the partition coefficient is $[\frac{1}{c}, 1]$, where a high value indicates a good partitioning. Since the lower bound of the partition coefficient depends on the parameter c, a clustering with a large number of clusters can be assessed as poorer than a clustering with a small number of clusters.

Normalized Partition Coefficient Since the partition coefficient is not normalized to the number of clusters, the lower bound for a small c is higher than for a large c. In this way PC has a bias towards a smaller number of clusters. To overcome this drawback, the *Normalized Partition Coefficient (NPC)* was proposed in [Bac78, Rou78, Dav96] that is defined as follows:

$$V_{NPC}(U,c) = 1 - \frac{c}{c-1}\left(1 - V_{PC}(U,c)\right) = 1 - \frac{c}{c-1}\left(1 - \frac{1}{n}\sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^2\right). \tag{4.2}$$

Unlike the partition coefficient the range of its normalized version is $[0, 1]$. The normalized partition coefficient maintains the property of PC that in the case of an optimal partitioning its value achieves 1:

$$1 - \frac{c}{c-1}\left(1 - V_{PC}(U,c)\right) = 1 - \frac{c}{c-1}\left(1 - 1\right) = 1 - 0 = 1.$$

The minimal value of NPC is 0. We get it when membership degrees of all data points to all clusters are $\frac{1}{c}$. While PC achieves a value of $\frac{1}{c}$ in this case, NPC has a value of 0 that does not depend on the parameter c anymore:

$$1 - \frac{c}{c-1}\left(1 - V_{PC}(U,c)\right) = 1 - \frac{c}{c-1}\left(1 - \frac{1}{c}\right) = 1 - \frac{c-1}{c-1} = 0.$$

4.2.1.2 Partition Entropy

The *Partition Entropy (PE)* is based on the idea of Shannon's entropy [Sha48]. In information theory, entropy is a measure of uncertainty, unpredictability or "randomness". Based on the equivalent notation of entropy as a measure of the average information content of a unit of data, Shannon defined the entropy H of a discrete random variable X with m possible values $\{X_1, ..., X_m\}$ whose probabilities of occurrence are

$p(X_1), ..., p(X_m)$ as the expected value of the information content of X:

$$H(X) = E(I(X)) = \sum_{i=1}^{m} p(X_i)I(X_i) = \sum_{i=1}^{m} p(X_i) \log \frac{1}{p(X_i)} = -\sum_{i=1}^{m} p(X_i) \log p(X_i) \tag{4.3}$$

with $\lim_{p \to 0+} p \log p = 0$.

The meaning of entropy can be explained by taking the example of a single coin toss experiment. In the case of a fair coin the probability of heads and tails are equal: $p(head) = p(tail) = \frac{1}{2}$. So the result of each coin toss is completely unpredictable and the entropy is as high as possible and is equal 1.

Using the concept of entropy, the partition entropy (PE) measures the amount of uncertainty of a clustering [Bez75, Bez81]. It interprets the probabilities $p(X_1), ..., p(X_m)$ as the membership degrees of data points to clusters and rates a partitioning of a data set as optimal if the clusters are clearly separated:

$$V_{PE}(U, c) = -\frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik} \log u_{ik}. \tag{4.4}$$

As in the example above, the partition entropy achieves its highest value of $\log c$ when all data points are ambiguously assigned into clusters, i.e. the membership degrees of all data points to all clusters are $\frac{1}{c}$. In the case of an optimal partitioning, i.e. all data points are clearly assigned into clusters and the amount of uncertainty is minimal, the partition entropy obtains a value of 0. So the value range of the partition entropy is $[0, \log c]$, where a small value indicates a good partitioning.

Normalized Partition Entropy Like in the case of the partition coefficient the upper bound of the partition entropy depends on the parameter c which means that clusterings with a large number of clusters can be automatically underestimated. To overcome this drawback several normalizations were proposed in the literature [Dun77, Bez75, BWE80]:

$$V_{NPE}(U) = \frac{V_{PE}(U)}{\log c}. \tag{4.5}$$

$$V_{MPE}(U) = \frac{nV_{PE}(U)}{n - c}, \tag{4.6}$$

While Bezdek's $V_{NPE}(U)$ is obviously normalized to the interval $[0, 1]$ for all c, Dunn's $V_{MPE}(U)$ is very similar to the basic version of the partition entropy for $c \ll n$.

4.2.1.3 Kim-Kim-Lee-Lee Index

While the partition entropy (PE) measures the amount of uncertainty of a partitioning as the average entropy of data items, the idea of the *Kim-Kim-Lee-Lee index (KKLL)* is

to calculate the average overlap between pairs of clusters as a relative degree of sharing of data points in the overlap [KKLL04]. In this way the KKLL index considers the geometrical structure of clusterings using only the membership matrix and rates the partitioning with the smallest degree of overlap between clusters as an optimal one.

Kim et al. define the relative degree of sharing between two fuzzy clusters C_i and C_j at the data item x_k as follows:

$$S_{rel}(x_k : C_i, C_j) = \frac{u_{ik} \wedge u_{jk}}{\frac{1}{c}\sum_{l=1}^{c} u_{lk}}. \tag{4.7}$$

For the calculation of the relative degree of sharing between two fuzzy clusters the authors use a fuzzy AND operator that is defined as $u_{ik} \wedge u_{jk} = min\{u_{ik}, u_{jk}\}$ [Zad65]. The higher the membership degrees of a data item to both clusters are, the higher is the relative degree of sharing between two clusters at that data point which means a large overlap between clusters. In Formula (4.7) the relative degree of sharing $u_{ik} \wedge u_{jk}$ is normalized by the average membership degree of the data point x_k over all c clusters. Since $\sum_{l=1}^{c} u_{lk} = 1$ holds for all data points in probabilistic FCM, this term can be neglected.

According to [KKLL04] the relative degree of sharing between two fuzzy clusters C_i and C_j is defined as weighted sum of the relative degrees of sharing at each data item x_k between two clusters of the pair.

$$S_{rel}(C_i, C_j) = \sum_{k=1}^{n} c \cdot [u_{ik} \wedge u_{jk}]h(x_k) \quad \text{with } h(x_k) = -\sum_{i=1}^{c} u_{ik}\log_a u_{ik}. \tag{4.8}$$

To reinforce the impact of vague data items in the calculation of the relative degree of sharing between pairs of clusters, Kim et al. use the entropy of data items as a weighting parameter. In this way the relative degree of sharing between two clusters is the higher the more vague data items are shared by both clusters. Then the Kim-Kim-Lee-Lee index is defined as the average relative degree of sharing of all possible pairs of clusters:

$$V_{KKLL}(U) = \frac{2}{c(c-1)} \sum_{i=1}^{c-1} \sum_{j=i+1}^{c} \sum_{k=1}^{n} [c \cdot [u_{ik} \wedge u_{jk}]h(x_k)]. \tag{4.9}$$

As mentioned above the KKLL cluster validity index as a separation index differs from partition entropy (PE) because it measures not only the ambiguity of a partitioning but also the overlap between clusters which is a geometrical property. On the other hand, it differs from many other CVIs because it provides information about the separation of clusters without being biased by the distances between cluster centers.

4.2.1.4 Overlap and Separation Index

The cluster validity index proposed by Le Capitaine and Frélicot in [CF11] combines overlap and separation criteria of a partitioning and only uses the membership degree matrix for the calculation. The idea of the *Overlap and Separation Index (OSI)* is to measure the ratio of overlap and separation at each data item and to rate the clustering of a data set regarding the average ambiguity between fuzzy clusters in the partitioning. According to [CF11] the overlap-separation measure at data point $x_k \in X$ is defined as follows:

$$OS_{\perp}(u_k(x_k), c) = \frac{O_{\perp}(u_k(x_k), c)}{S_{\perp}(u_k(x_k), c)}. \tag{4.10}$$

The calculation of overlap and separation measures at the data point x_k is based on the membership degrees of this data point to all clusters in the partitioning. The overlap measure for the data item x_k evaluates the degree of overlap between different numbers of clusters at the data point x_k. Unlike other cluster validity indexes, e.g. KKLL index, OSI calculates and compares the degrees of overlap not only between each pair but also between each triplet up to c-tuple of clusters. Then the overlap degree of the combination of clusters with the "highest" overlap value determines the overall degree of overlap $O_{\perp}(u_k(x_k), c)$ for the data point x_k.

$$O_{\perp}(u_k(x_k), c) = \underset{l=2,c}{\overset{1}{\perp}} \left(\underset{i=1,c}{\overset{l}{\perp}} (u_{ik}) \right). \tag{4.11}$$

Le Capitaine and Frélicot use the *l-order fuzzy OR* operator (fOR-l for short) defined in [MBF08] for the calculation of the overlap measure $O_{\perp}(u_k(x_k), c)$. The l-order fuzzy OR operator concatenates triangular norms (short: t-norms) and conorms (short: t-conorms) which combine membership degrees in order to measure the k-order ambiguity, i.e. the ambiguity between k fuzzy sets. Mascarilla et al. generalized the l-order fuzzy OR operator to all triangular norms and conorms. Some basic t-norms and t-conorms are summarized in Table 4.1. According to [MBF08] the l-order fuzzy OR operator is defined as follows:

$$\underset{i=1,c}{\overset{l}{\perp}} u_{ik} = \underset{A \in \mathcal{P}_{l-1}}{\top} \left(\underset{j \in C \setminus A}{\perp} u_{jk} \right), \tag{4.12}$$

where $\mathcal{P}$ is the power set of $C = \{1, 2, ..., c\}$ and $\mathcal{P}_l = \{A \in \mathcal{P} \mid |A| = l\}$. $|A|$ is the cardinality of the subset A.

The effect of the l-order fuzzy OR operator can be explained by taking the example of the standard t-norm. In this case the l-order fuzzy OR operator $\overset{l}{\perp}(u_k)$ calculates the l-th largest element of u_k with $u_k = \{u_{1k}, ..., u_{ck}\}$ (see the proof in [MBF08]). If we take a closer look at the overlap measure defined in Formula (4.11), we see that

the l-order fuzzy OR operator is used to determine the maximum overlap value among all l-order overlap values for $l = 2, ..., c$ each of which corresponds to the maximum overlap value among all possible combinations of l clusters at the data point x_k. In the end, the overlap measure for the data point x_k using the standard t-norm is defined by the second largest element of u_k with $u_k = \{u_{1k}, ..., u_{ck}\}$.

Table 4.1: Examples of T-norm ($\top$) and T-conorm ($\perp$) couples.

Name	$a \top b$	$a \perp b$
Standard (S)	$\min(a, b)$	$\max(a, b)$
Algebraic (A)	ab	$a + b - ab$
Łukasiewicz (L)	$\max(a + b - 1, 0)$	$\min(a + b, 1)$
Hamacher (H_γ)	$\frac{ab}{\gamma+(1-\gamma)(a+b-ab)}$	$\frac{a+b-ab-(1-\gamma)ab}{1-(1-\gamma)ab}$
Dombi (D_γ)	$\frac{1}{1+\left(\left(\frac{1-a}{a}\right)^\gamma+\left(\frac{1-b}{b}\right)^\gamma\right)^{1/\gamma}}$	$1 - \frac{1}{1+\left(\left(\frac{a}{1-a}\right)^\gamma+\left(\frac{b}{1-b}\right)^\gamma\right)^{1/\gamma}}$

In contrast to the overlap measure the separation measure calculates the degree of separation between clusters at the data item x_k. In case of the standard triangular norm it measures how well the data item x_k is assigned to the cluster to which it has the largest membership degree. According to [CF11] the separation measure at the data point x_k is defined as follows:

$$S_\perp(u_k(x_k), c) = \overset{1}{\perp}\left(\underbrace{\underset{i=1,c}{\overset{1}{\perp}} u_{ik}, \ldots, \underset{i=1,c}{\overset{1}{\perp}} u_{ik}}_{c-1 \text{ times}}\right). \tag{4.13}$$

Formula (4.13) can be simplified using the property $\overset{1}{\underset{i=1,c}{\perp}} u_{ik} = \underset{j \in C}{\perp} u_{jk}$ [MBF08]. While the separation measure can be simplified to $\max\{u_{1k}, \ldots, u_{ck}\}$ for the standard t-norm, the calculation of the separation measure for other triangular norms is more complicated and must be normalized in accordance with the overlap measure. Since the overlap measure in Formula (4.11) calculates the fuzzy disjunction of $|l| = c-1$ overlap degrees of combinations of clusters, the authors use the fuzzy disjunction of $c-1$ single $\overset{1}{\perp}(u_k)$ measures in the separation measure. In this way the overlap-separation measure $OS_\perp$ for the data item x_k is normalized to the interval $[0, 1]$.

The *Overlap and Separation Index (OSI)* of a partitioning of a data set is defined as the average overlap-separation value of data points in the data set:

$$V_{OSI_\perp}(U) = \frac{1}{n}\sum_{k=1}^{n} OS_\perp(u_k(x_k), c). \tag{4.14}$$

Le Capitaine and Frélicot have shown in [CF11] that the overlap and separation index (OSI) is normalized to the interval $[0, 1]$. Since in the optimal partitioning of a data set the clusters should be well separated and the overlap between clusters should be minimal, the overlap and separation index (OSI) should be minimized to find the optimal number of clusters.

The overlap and separation index (OSI) determines the optimal number of clusters regarding the geometrical properties of a partitioning using only the membership matrix. Therefore, this index can be applied to incomplete data without any changes. On the other hand, the high computational costs of this index limits its applicability on large data sets. Like the other cluster validity indexes, OSI has to be computed for several partitionings for a different number of clusters but the computational costs exponentially increase with the increasing number of clusters. While the computation of OSI using the standard t-norm can be drastically reduced, there is no shortcut for the computation of OSI using other t-norms.

4.2.2 Cluster Validity based on Compactness

Cluster validity indexes of this category focus only on the consideration of the similarity between data items within clusters. Since data objects are represented by metric objects in the feature space and the similarity between the data items is represented by the distances between them, the idea of CVIs of this category is to measure the geometrical compactness of clusters as point clouds in space. As we will see later, cluster validity functions define compactness of clusters in different ways. While the *Fuzzy Hypervolume* calculates the volume of clusters, the *Partition Density* includes the number of points within clusters. Since both cluster validity indexes involve the data set in their calculation, we also give the adapted versions of them to incomplete data.

4.2.2.1 Fuzzy Hypervolume

The idea of the cluster validity index proposed by Gath and Geva in [GG89] is that clusters of data points should be of minimal volume in an optimal fuzzy partitioning. In this way the *Fuzzy Hypervolume (FHV)* defines the compactness of a partitioning of a data set as the sum of the volumes of its clusters.

$$V_{FHV}(U, X, V) = \sum_{i=1}^{c} \sqrt{det(Cov_i)}. \tag{4.15}$$

The FHV uses the determinant of the covariance matrix of a cluster as a measure for its volume:

$$Cov_i = \frac{\sum_{k=1}^{n} (u_{ik})^m (x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^{n} (u_{ik})^m} \quad \text{for } 1 \leq i \leq c. \tag{4.16}$$

The advantage of using the covariance matrix is that it describes the size and the shape of clusters between pairs of dimensions. That enables fuzzy hypervolume to recognize clusters of different sizes independent of their expansion, their location in the feature space, and their closeness to each other. On the other hand, FHV suffers from the monotonically decreasing tendency for $c \to n$ because the minimal volume of clusters is achieved when each data point is in its own cluster.

Fuzzy Hypervolume for Incomplete Data As mentioned above, fuzzy hypervolume involves the data set in its calculation. To be precise, the calculation of the covariance matrix demands the calculation of distances between values of data points and cluster prototypes in each dimension. Little and Rubin proposed in [LR02] to calculate the covariance matrix of a Gaussian distribution of incomplete data using only available feature values of incomplete data items and normalizing it by the number of used values. Using this idea in [HHC11] we already adapted the calculation of the covariance matrix for fuzzy clusters of incomplete data in the same way as it was used by Timm et al. in [TDK04]. In this way we adapt the FHV to incomplete data using Formulae (4.15) and (4.17) [HCC12].

$$\mathrm{Cov}_{i(pl)}(U, X, V) = \frac{\sum_{k=1}^{n} (u_{ik})^m i_{kp} i_{kl} (x_{kp} - v_{ip})(x_{kl} - v_{il})}{\sum_{k=1}^{n} (u_{ik})^m i_{kp} i_{kl}} \quad \text{for } 1 \leq i \leq c \text{ and } 1 \leq p, l \leq d, \tag{4.17}$$

where

$$i_{kj} = \begin{cases} 1 & \text{if } x_{kj} \in X_{avl} \\ 0 & \text{else.} \end{cases}$$

4.2.2.2 Partition Density

The fuzzy hypervolume values the quality of a clustering only on the basis of the volume of clusters regardless of their densities. In this way large clusters are automatically rated as "bad" ones. To overcome this drawback, the *Partition Density (PD)* [GG89] uses the idea of *mass density* that is defined as mass divided by volume. It seems natural to use the number of data points belonging to a cluster as a measure for its "mass". Since in a fuzzy partitioning all data items are assigned to all clusters with

membership degrees, partition density relates the sum of membership degrees of data points closely located to cluster prototypes to the volume of clusters:

$$V_{PD}(U, X, V) = \frac{Z}{FHV} = \frac{Z}{\sum_{i=1}^{c} \sqrt{det(Cov_i)}}, \tag{4.18}$$

where

$$Z = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik} \quad \forall\, x_k \in \{x_k \mid (x_k - v_i)^T Cov_i^{-1} (x_k - v_i) < 1\}. \tag{4.19}$$

In this way, partition density maintains all properties of the fuzzy hypervolume and relativizes the volume of clusters eliminating the preference of small clusters. But on the other side the monotonically decreasing tendency is increased in partition density when the number of clusters approaches n. Since the membership degrees of data points to clusters get higher in this case, apart from the minimization of cluster volumes more data points will be included in the calculation of parameter Z. Therefore, it is important to determine an appropriate range $[c_{\min}, c_{\max}]$ where the PD should be maximized.

Partition Density for Incomplete Data As in the case of the fuzzy hypervolume the data set is also involved in the calculation of the partition density. While the covariance matrix can be calculated according to Formula (4.17), Condition (4.19) must be adapted to incomplete data. As we already described in [HHC11], we approximate the distance of incomplete data points to the cluster prototypes by using only available feature values of data items and normalizing the result by the number of used values. In this way the condition of Formula (4.19) is substituted as follows:

$$Z = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik} \quad \forall\, x_k \in \left\{ x_k \;\middle|\; \frac{d\begin{pmatrix} i_{k1}(x_{k1} - v_{i1}) \\ \vdots \\ i_{kd}(x_{kd} - v_{id}) \end{pmatrix}^T Cov_i^{-1} \begin{pmatrix} i_{k1}(x_{k1} - v_{i1}) \\ \vdots \\ i_{kd}(x_{kd} - v_{id}) \end{pmatrix}}{\sum_{j=1}^{d} i_{kj}} < 1 \right\}, \tag{4.20}$$

where i_{kj} is defined as in Formula (4.17).

4.2.3 Cluster Validity based on Compactness and Separation

The idea of cluster validity indexes from this category is based on the the general idea of clustering that data items within the clusters should be similar and data items from different clusters should to be as dissimilar as possible. Since the similarity between data items is expressed by the distances between them, cluster validity indexes of this category consider the compactness within the clusters to meet the first requirement.

They calculate the separation between the clusters to measure the dissimilarity between data items from different clusters. Below we give an overview of several cluster validity indexes that differ from each other in the way how they interpret and combine the compactness and the separation criteria of a partitioning. Since almost all of them involve the data set in their calculation, these cluster validity indexes have to be adapted to incomplete data.

4.2.3.1 Fukuyama-Sugeno Index

One of the oldest cluster validity indexes that combine compactness and separation between clusters is the *Fukuyama-Sugeno index (FS)* [FS89]. The Fukuyama-Sugeno index uses the objective function J_m as a compactness measure and the sum of the fuzzy weighted distances between the cluster prototypes and the grand mean of the data set as a separation measure. According to [FS89] it is defined as follows:

$$V_{FS}(U,X,V) = \sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^m \|x_k - v_i\|^2 - \sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^m \|v_i - \overline{v}\|^2 \quad \text{with } \overline{v} = \frac{\sum_{i=1}^{c} v_i}{c}. \tag{4.21}$$

While the compactness measure in the Fukuyama-Sugeno index measures the distances between data items and cluster prototypes directly, the separation measure measures the distances between cluster centers and the grand mean of the data set. On the one hand, this strategy reduces the complexity, on the other hand, the separation degree between clusters depends on the location of the grand mean.

Since in the optimal partitioning the compactness within clusters should be small and the separation between clusters should be large, a small value of V_{FS} corresponds to a good partitioning.

Fukuyama-Sugeno Index for Incomplete Data While calculation of the separation measure in the Fukuyama-Sugeno index involves only cluster prototypes, calculation of the compactness measure is based on the squared Euclidean distances between data items and cluster centers. In [HCC12] we already adapted the Fukuyama-Sugeno index to incomplete data substituting the Euclidean distance function by the partial distance function. In the case of incomplete data, the Fukuyama-Sugeno index can be calculated according to Formula (4.22).

$$V_{FS}(U,X,V) = \sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^m \frac{d\sum_{j=1}^{d}(x_{kj} - v_{ij})^2 i_{kj}}{\sum_{j=1}^{d} i_{kj}} - \sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^m \|v_i - \overline{v}\|^2 \quad \text{with } \overline{v} = \frac{\sum_{i=1}^{c} v_i}{c}, \tag{4.22}$$

where

$$i_{kj} = \begin{cases} 1 & \text{if } x_{kj} \in X_{avl} \\ 0 & \text{else.} \end{cases}$$

4.2.3.2 Xie-Beni Index

In the cluster validity function of Xie and Beni [XB91], the distances between the data points and the cluster prototypes are related to the distances between the clusters:

$$V_{XB}(U, X, V) = \frac{\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2}{n \min\limits_{1 \le i,j \le c,\ i \ne j} \|v_i - v_j\|^2}, \tag{4.23}$$

where $||.||$ is the Euclidean norm. Like the Fukuyama-Sugeno index, the *Xie-Beni index* uses the objective function as a compactness measure, whereas the Xie-Beni index was initially defined for J_2. The separation between clusters is determined by the distance between the two nearest cluster prototypes. On the one hand, the Xie-Beni index favors partitionings where all clusters are well separated from each other. On the other hand, it tends to recognize only rough clustering structures neglecting groups of clusters. A small value for V_{XB} indicates an optimal c-partitioning of a data set.

The monotony properties of the Xie-Beni index were extensively examined and discussed in the literature [Kwo98, PB95, TSS05]. Even Xie and Beni admitted in [XB91] that their cluster validity index monotonically decreases for $c \to n$. Since

$$\lim_{c \to n} \{\|x_k - v_i\|^2\} = 0 \tag{4.24}$$

holds, $V_{XB}(U, X, V)$ also converges to 0 for $c \to n$:

$$\lim_{c \to n} \{V_{XB}(U, X, V)\} = \lim_{c \to n} \left\{ \frac{\sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2}{n \min\limits_{1 \le i,j \le c,\ i \ne j} \|v_i - v_j\|^2} \right\} = 0. \tag{4.25}$$

Therefore, the authors proposed to plot their index for different c as a function and to choose c as $c_{\max}$ where the function starts to decrease monotonically.

Additionally, to make their cluster validity index compatible for J_m, Xie and Beni suggested to generalize $V_{XB}(U, X, V)$ for any $m > 1$. In [PB95] Pal and Bezdek examined the monotony properties of the Xie-Beni index for $m \to \infty$. They found out that the Xie-Beni index gets unstable and unpredictable for large values of m. Since

$$\lim_{m \to \infty} \left\{ \left[u_{ik} = \sum_{j=1}^{c} \left(\frac{D_{ikA}}{D_{jkA}} \right)^{\frac{2}{(m-1)}} \right]^{-1} \right\} = \frac{1}{c} \tag{4.26}$$

and

$$\lim_{m\to\infty}\left\{\left[v_i=\frac{\sum_{k=1}^{n}(u_{ik})^m x_k}{\sum_{k=1}^{n}(u_{ik})^m}\right]\right\}=\frac{\sum_{k=1}^{n}\left(\frac{1}{c}\right)^m x_k}{\sum_{k=1}^{n}\left(\frac{1}{c}\right)^m}=\frac{\left(\frac{1}{c}\right)^m\sum_{k=1}^{n}x_k}{n\left(\frac{1}{c}\right)^m}=\frac{\sum_{k=1}^{n}x_k}{n}=\overline{x}, \tag{4.27}$$

the separation measure in the Xie-Beni index converges to 0 and consequently the Xie-Beni index converges to infinity for $m\to\infty$:

$$\lim_{m\to\infty}\{Sep(V_{XB})\}=\lim_{m\to\infty}\left\{\min_{i\neq j}\|v_i-v_j\|^2\right\}=\left(\min_{i\neq j}\|\overline{x}-\overline{x}\|^2\right)=0. \tag{4.28}$$

$$\lim_{m\to\infty}\{V_{XB}(U,X,V)\}=\lim_{m\to\infty}\left\{\frac{\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^2\|x_k-v_i\|^2}{n\min\limits_{1\leq i,j\leq c,\ i\neq j}\|v_i-v_j\|^2}\right\}$$

$$\overset{(4.28)}{=}\frac{\lim\limits_{m\to\infty}\left\{\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^2\|x_k-v_i\|^2\right\}}{0}\overset{\substack{(4.26)\\(4.27)}}{=}\frac{\frac{1}{c}\sum_{k=1}^{n}\|x_k-\overline{x}\|^2}{0}=\infty.$$

Xie-Beni Index for Incomplete Data Like in the Fukuyama-Sugeno index the compactness measure of the Xie-Beni index is also based on the squared Euclidean distances between data items and cluster centers. To adapt the Xie-Beni index to incomplete data, we replace the Euclidean distances by the partial distances between the incomplete data items and the cluster prototypes [HHC11]. In this way the Xie-Beni index is calculated on data with missing values for J_2 as follows:

$$V_{XB}(U,X,V)=\frac{\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^2\frac{d\sum_{l=l}^{d}(x_{kl}-v_{il})^2 i_{kl}}{\sum_{l=1}^{d}i_{kl}}}{n\min\limits_{1\leq i,j\leq c,\ i\neq j}\|v_i-v_j\|^2}\quad\text{with}\quad i_{kl}=\begin{cases}1 & \text{if } x_{kl}\in X_{avl}\\ 0 & \text{else.}\end{cases} \tag{4.29}$$

Since the calculation of the separation measure in the Xie-Beni index only involves the cluster prototypes, no further changes are needed.

4.2.3.3 Kwon Index

Since the value of the objective function monotonically decreases with increasing number of clusters, the Xie-Beni index inherits this undesirable property. To avoid the monotonically decreasing tendency of $V_{XB}(U,X,V)$ for a large number of clusters, based on the idea in [Dun77] Xie and Beni recommended to introduce an *ad hoc* punishing function in their index [XB91]. In [Kwo98] Kwon extended the Xie-Beni index by adding such a punishing function to the numerator of Formula (4.23). The punishing

function in the *Kwon index* is another separation measure that calculates the average distance of cluster prototypes to the mean of the data set. According to [Kwo98] the Kwon index is defined as follows:

$$V_{Kwon}(U,X,V) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 + \frac{1}{c}\sum_{i=1}^{c}\|v_i - \overline{x}\|^2}{\min\limits_{1\le i,j\le c,\ i\neq j} \|v_i - v_j\|^2} \quad \text{with} \quad \overline{x} = \frac{\sum_{k=1}^{n} x_k}{n}. \tag{4.30}$$

The *ad hoc* punishing function in the numerator of $V_{Kwon}(U,X,V)$ prevents it from converging to 0 for $c \to n$:

$$\begin{aligned}\lim_{c\to n}\{V_{Kwon}(U,X,V)\} &= \lim_{c\to n}\left\{\frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 + \frac{1}{c}\sum_{i=1}^{c}\|v_i - \overline{x}\|^2}{\min\limits_{1\le i,j\le c,\ i\neq j} \|v_i - v_j\|^2}\right\} \\ &= \frac{\sum_{k=1}^{n}\|x_k - \overline{x}\|^2}{n \min\limits_{i\neq j}\|x_i - x_j\|^2}.\end{aligned}$$

The limit of $V_{Kwon}(U,X,V)$ is a function of X, so it is *constant* and depends only on the data set alone. However, due to Property (4.28) the Kwon index as well as the Xie-Beni index converges to *infinity* for $m \to \infty$ which makes this cluster validity index unstable for large values of m [TSS05].

Kwon Index for Incomplete Data Unlike the Fukuyama-Sugeno and the Xie-Beni indexes the Kwon index uses the data set not only for the calculation of the compactness measure but also for the calculation of the punishing function. While we adapt the calculation of the compactness measure to incomplete data as in the cluster validity indexes mentioned before, we calculate the mean of the data set in the punishing function only on the basis of the available feature values. In this way the Kwon index is calculated for incomplete data as follows:

$$V_{Kwon}(U,X,V) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \frac{d\sum_{l=1}^{d}(x_{kl}-v_{il})^2 i_{kl}}{\sum_{l=1}^{d} i_{kl}} + \frac{1}{c}\sum_{i=1}^{c}\|v_i - \overline{x}\|^2}{\min\limits_{1\le i,j\le c,\ i\neq j} \|v_i - v_j\|^2} \tag{4.31}$$

with

$$\overline{x}_l = \frac{\sum_{k=1}^{n} i_{kl} x_{kl}}{\sum_{k=1}^{n} i_{kl}} \quad \text{and} \quad i_{kl} = \begin{cases} 1 & \text{if } x_{kl} \in X_{avl} \\ 0 & \text{else} \end{cases} \quad \text{for} \quad 1 \le l \le d. \tag{4.32}$$

4.2.3.4 Tang-Sun-Sun Index

In [TSS05] Tang et al. proposed an improved version of the Xie-Beni index in terms of stability for the increasing fuzzification parameter m. Similar to the Kwon index they also use an *ad hoc* punishing function in the numerator of the Xie-Beni index to avoid the decreasing tendency of the index for $c \to n$. To solve the problem of instability for increasing m, the *Tang-Sun-Sun index (TSS)* uses an additional punishing function in the denominator that prevents it from converging to 0 for $m \to \infty$. According to [TSS05] the Tang-Sun-Sun index is defined as follows:

$$V_{TSS}(U,X,V) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 + \frac{1}{c(c-1)} \sum_{i=1}^{c}\sum_{\substack{j=1 \\ j\neq i}}^{c} \|v_i - v_j\|^2}{\min\limits_{1\leq i,j\leq c,\ i\neq j} \|v_i - v_j\|^2 + 1/c}. \tag{4.33}$$

While the Kwon index uses a separation measure similar to the Fukuyama-Sugeno index as a punishing function, the Tang-Sun-Sun index uses a punishing function that calculates the average distance between the cluster prototypes. As mentioned above this approach provides more precise information about the separation between the clusters in the partitioning than the separation measure used in the Fukuyama-Sugeno or in the Xie-Beni index. Regarding the convergence behavior of the Tang-Sun-Sun index the *ad hoc* punishing function in the numerator effectively eliminates the decreasing tendency of this cluster validity index for $c \to n$:

$$\lim_{c\to n}\{V_{TSS}(U,X,V)\} = \lim_{c\to n}\left\{ \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 + \frac{1}{c(c-1)} \sum_{i=1}^{c}\sum_{\substack{j=1 \\ j\neq i}}^{c} \|v_i - v_j\|^2}{\min\limits_{1\leq i,j\leq c,\ i\neq j} \|v_i - v_j\|^2 + 1/c} \right\}$$

$$\overset{(4.24)}{=} \frac{0 + \frac{1}{n(n-1)} \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} \|x_i - x_j\|^2}{\min\limits_{i\neq j} \|x_i - x_j\|^2 + 1/n} = \frac{n \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} \|x_i - x_j\|^2}{n^2(n-1) \min\limits_{i\neq j} \|x_i - x_j\|^2 + 1}.$$

An additional punishing function in the denominator of the Tang-Sun-Sun index reinforces the numerical stability of the index for $m \to \infty$:

$$\lim_{m\to\infty}\{V_{TSS}(U,X,V)\} = \lim_{m\to\infty}\left\{ \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 + \frac{1}{c(c-1)} \sum_{i=1}^{c}\sum_{\substack{j=1 \\ j\neq i}}^{c} \|v_i - v_j\|^2}{\min\limits_{1\leq i,j\leq c,\ i\neq j} \|v_i - v_j\|^2 + 1/c} \right\}$$

$$\overset{\substack{(4.26)\\(4.27)\\(4.28)}}{=} \frac{\frac{1}{c}\sum_{k=1}^{n} \|x_k - \overline{x}\|^2 + 0}{0 + 1/c} = \sum_{k=1}^{n} \|x_k - \overline{x}\|^2.$$

The limits of $V_{TSS}(U, X, V)$ for $c \to n$ and for $m \to \infty$ are both functions of X only, so they depend on the characteristics of the data set alone. In this way, two *ad hoc* punishing functions in the Tang-Sun-Sun index ensure the numerical stability of the validation index for large values of m and prevent the monotonically decreasing tendency of it when c approaches n. As in the Xie-Beni and the Kwon indexes the optimal number of clusters for a data set X can be found by minimizing the Tang-Sun-Sun index over the range $[c_{\min}, c_{\max}]$.

Tang-Sun-Sun Index for Incomplete Data Like the Fukuyama-Sugeno and the Xie-Beni index the cluster validity index of Tang et al. uses the data set for the calculation of the objective function in the compactness measure only. Therefore, we adapted the Tang-Sun-Sun index to data with missing values analogously to other indexes that use the objective function. Thus, the Tang-Sun-Sun index for incomplete data is calculated according to Formula (4.34).

$$V_{TSS}(U, X, V) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \frac{d\sum_{l=l}^{d}(x_{kl}-v_{il})^2 i_{kl}}{\sum_{l=1}^{d} i_{kl}} + \frac{1}{c(c-1)}\sum_{i=1}^{c}\sum_{\substack{j=1 \\ j\neq i}}^{c} \|v_i - v_j\|^2}{\min\limits_{1\leq i,j\leq c,\ i\neq j} \|v_i - v_j\|^2 + 1/c}, \tag{4.34}$$

where i_{kl} is defined as in Formula (4.29).

4.2.3.5 Beringer-Hüllermeier Index

The Xie-Beni index and its improved versions use the objective function as a compactness measure and the minimum distance between two cluster centers as a separation criterion. While the objective function considers the intra-cluster similarity of all clusters, the inter-cluster variance (separation between clusters) is reduced to the distance between the two nearest clusters. In order to involve the distances between all clusters in a partitioning, Beringer and Hüllermeier proposed a new separation measure that is calculated as a sum of weighted pairwise distances between clusters [BH07]. In their separation measure they also consider the variability of clusters that is defined as average squared distance between data points and the cluster center:

$$V_i = \frac{\sum_{k=1}^{n} u_{ik}\|x_k - v_i\|^2}{\sum_{k=1}^{n} u_{ik}} \quad \text{for } i = 1, ..., c. \tag{4.35}$$

The distance between two clusters is defined as a distance between their centroids divided by the cluster variabilities.

$$D(C_i, C_j) = \frac{\|v_i - v_j\|^2}{V_i \times V_j} \quad \text{for } i, j \in \{1, ..., c\}. \tag{4.36}$$

Calculating the separation between clusters only on the basis of distances between their prototypes ignores the overlaps between clusters. The centroids of two clusters might be distant but if clusters have large dispersion range, they might partly overlap. In this case the separation between clusters is small. Including the variability of clusters, the possible overlaps between clusters are better considered in the calculation of the distances between clusters.

According to [BH07] the *Beringer-Hüllermeier index (BH)* for a partitioning of a data set X is defined as follows:

$$V_{BH}(U,X,V) = \frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m}\|x_k - v_i\|^2 \times \sum_{i=1}^{c-1}\sum_{j=i+1}^{c}\frac{1}{D(C_i,C_j)}. \tag{4.37}$$

In the Beringer-Hüllermeier index (BH) the separation between clusters is used to express the similarity between clusters in the partitioning by summing up the reciprocals of distances between each pair of clusters. In other words, the index measures the compactness within clusters as intra-cluster similarity and the overlap between clusters as inter-cluster similarity. Since in an optimal partitioning clusters should be compact and well separated, a low value of the Beringer-Hüllermeier index indicates a good partitioning. Using the objective function J_m as a compactness measure the Beringer-Hüllermeier index inherits the undesirable monotonically decreasing tendency when the number of clusters approaches number of data items in the data set.

Beringer-Hüllermeier Index for Incomplete Data Unlike the aforementioned cluster validity indexes that combine the compactness and the separation between clusters, the Beringer-Hüllermeier index uses the average distance between the data points and the cluster prototypes in both the compactness and the separation measures. We adapt the Beringer-Hüllermeier index to incomplete data by substituting the Euclidean distances between the incomplete data items and the cluster prototypes by the partial distances:

$$V_{BH}(U,X,V) = \frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m}\frac{d\sum_{l=l}^{d}(x_{kl}-v_{il})^2 i_{kl}}{\sum_{l=1}^{d} i_{kl}} \times \sum_{i=1}^{c-1}\sum_{j=i+1}^{c}\frac{1}{D(C_i,C_j)}. \tag{4.38}$$

The distance $D(C_i,C_j)$ between two clusters is calculated according to Formula (4.36), where we adapt the variability of clusters to incomplete data as follows:

$$V_i = \frac{\sum_{k=1}^{n} u_{ik}\frac{d\sum_{l=l}^{d}(x_{kl}-v_{il})^2 i_{kl}}{\sum_{l=1}^{d} i_{kl}}}{\sum_{k=1}^{n} u_{ik}} \quad \text{for } 1 \le i \le c, \tag{4.39}$$

where i_{kl} is defined as in Formula (4.29). An advantage of adapting the Beringer-Hüllermeier index in this way is that the calculation of the compactness and the separation measures remains consistent, i.e. the values for the average distances between the data points and the cluster prototypes are the same in both compactness and separation measures.

4.2.3.6 Zahid-Limouri-Essaid Index

In [ZLE99] Zahid et al. proposed another cluster validity index that combines the compactness and the separation criteria of a partitioning. The ratio of the separation and the compactness is computed twice: involving the structure of the data set and using only the membership matrix produced by the partitioning algorithm. The idea of the *Zahid-Limouri-Essaid index (ZLE)* is to quantify the degree of correspondence between the geometrical and the pure fuzzy partitioning structure of a clustering.

Similar to the Xie-Beni index [XB91] the first function $SC_1(X, U, V)$ measures the compactness within the clusters as the sum of distances between the data items and the cluster prototypes. The separation in $SC_1(X, U, V)$ is calculated similar to the separation measure in the Fukuyama-Sugeno index [FS89] as the scatter of cluster prototypes:

$$SC_1(X, U, V) = \frac{\sum_{i=1}^{c} \|v_i - \overline{x}\|_A^2 / c}{\sum_{i=1}^{c} (\sum_{k=1}^{n} u_{ik}^2 \|x_k - v_i\|^2 / \sum_{k=1}^{n} u_{ik})} \quad \text{with} \quad \overline{x} = \frac{\sum_{k=1}^{n} x_k}{n}. \tag{4.40}$$

$SC_1(X, U, V)$ rates the partitioning of a data set as optimal if clusters are widely scattered in the data space and the data points are close to the cluster centers.

The second function $SC_2(U)$ extracts the information about the separation and the compactness of clustering only using the membership matrix.

$$SC_2(U) = \frac{\sum_{i=1}^{c-1} \sum_{j=i+1}^{c} (\sum_{k=1}^{n} \min(u_{ik}, u_{jk})^2) / \sum_{k=1}^{n} \min(u_{ik}, u_{jk})}{\sum_{k=1}^{n} (\max_{1 \le i \le c} u_{ik})^2 / \sum_{k=1}^{n} \max_{1 \le i \le c} u_{ik}}. \tag{4.41}$$

$SC_2(U)$ rates the compactness of a partitioning regarding how clearly the data items are assigned to clusters. The compactness measure in $SC_2(U)$ is calculated as the sum of the compactness measures of all data items. Similar to the separation measure in the Overlap and Separation Index (OSI) described in Section 4.2.1.4 in the case of the standard t-norm, the compactness measure at a data item $x_k \in X$ is determined by the largest membership degree among clusters. To compute the separation between clusters, $SC_2(U)$ computes the overlap between each pair of clusters using the fuzzy

intersection that is defined as the minimum of the membership degrees [Zad65].

The smaller the overlap between the clusters is and the clearer the data points are assigned to clusters in the partitioning, the smaller is the value for $SC_2(U)$. The Zahid-Limouri-Essaid index combines both functions while function $SC_1(X, U, V)$ should be maximized and function SC_2 should be minimized.

$$V_{ZLE}(U, V, X) = SC_1(U, V, X) - SC_2(U). \tag{4.42}$$

The optimal number of clusters is obtained by maximizing $V_{ZLE}(U, V, X)$ over the range $c_{\min}, ..., c_{\max}$. Since the ZLE index uses the objective function J_2 in the denominator of $SC_1(X, U, V)$, it gets unpredictable for $c \to n$. In this case the Zahid-Limouri-Essaid index is not able to provide useful results.

Zahid-Limouri-Essaid Index for Incomplete Data As mentioned above the Zahid-Limouri-Essaid index calculates the ratio of the compactness and the separation measures for a partitioning twice: involving the data set and only using the membership matrix. Therefore the function $SC_1(U, V, X)$ needs to be adapted to data with missing values for the validation of a partitioning produced on incomplete data. We adapt the function $SC_1(U, V, X)$ in the Zahid-Limouri-Essaid index in the same way as other cluster validity indexes using all available feature values:

$$SC_1(X, U, V) = \frac{\sum_{i=1}^{c} \|v_i - \overline{x}\|_A^2 / c}{\sum_{i=1}^{c} \left(\sum_{k=1}^{n} u_{ik}^2 \frac{d \sum_{l=l}^{d} (x_{kl} - v_{il})^2 i_{kl}}{\sum_{l=1}^{d} i_{kl}} / \sum_{k=1}^{n} u_{ik} \right)} \quad \text{with} \quad \overline{x}_l = \frac{\sum_{k=1}^{n} i_{kl} x_{kl}}{\sum_{k=1}^{n} i_{kl}} \tag{4.43}$$

for $1 \leq l \leq d$ where i_{kl} is defined as in Formula (4.29). In this way the Zahid-Limouri-Essaid index can be calculated for a partitioning of incomplete data according to Formula (4.42) and using Formulae (4.43) and (4.41) for calculation of functions $SC_1(U, V, X)$ and $SC_2(U)$, respectively.

4.2.3.7 Bouguessa-Wang-Sun Index

The cluster validity index proposed by Bouguessa et al. in [BWS06] also combines the compactness and the separation criteria of a partitioning. Unlike other CVIs of this category the *Bouguessa-Wang-Sun index (BWS)* analyzes partitionings of the data set particularly with regard to the overlaps between clusters and the variations in the cluster density, orientation and shape. Therefore, it combines relevant properties of the partitioning regarding the aforementioned requirements for the calculation of the compactness and the separation measures.

As in the fuzzy hypervolume (FHV) the covariance matrix of clusters provides the basis for the calculation of the compactness within the clusters. The advantage of using the covariance matrix is that it considers the shape and the orientation of clusters. Unlike the fuzzy hypervolume the Bouguessa index calculates the compactness of a partitioning as a sum of the traces of the covariance matrices of clusters:

$$\text{Comp}_{BWS}(U, V, X) = \sum_{i=1}^{c} \text{tr}(\text{Cov}_i), \tag{4.44}$$

where Cov_i is calculated according to Formula (4.16). Bouguessa et al. adopted the separation measure from [GSBN00] where, similar to the separation measure in the Fukuyama-Sugeno index [FS89], it is defined as the trace of the fuzzy between-cluster scatter matrix:

$$\text{Sep}_{BWS}(U, V, X) = \text{tr}\left(\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m}(v_i - \overline{x})(v_i - \overline{x})^T\right) \text{ with } \overline{x} = \frac{\sum_{k=1}^{n} x_k}{n}. \tag{4.45}$$

The wider the cluster centers are scattered in the data space and the farther they are from the mean of the data set, the larger is the separation between the clusters.

According to [BWS06] the Bouguessa-Wang-Sun index $V_{BWS}(U, V, X)$ is defined as ratio of the separation and the compactness measures:

$$V_{BWS}(U, V, X) = \frac{\text{Sep}_{BWS}(U, V, X)}{\text{Comp}_{BWS}(U, V, X)}. \tag{4.46}$$

A large value for the separation measure and a small value for the compactness measure results in a large value for $V_{BWS}(U, V, X)$ and indicates compact well-separated clusters which is a characteristic of an optimal partitioning. Therefore, the optimal number of clusters is obtained by maximizing $V_{BWS}(U, V, X)$ over the range $c_{\min}, ..., c_{\max}$.

Although the authors argue in [BWS06] that the normalization of the covariance matrix (see Formula (4.16)) prevents the compactness measure $\text{Comp}_{BWS}(U, V, X)$ of their cluster validity index from the monotonic decreasing for $c \to n$, this assertion is not true. Since the Property (4.24) holds, Cov_i also converges to 0 for $c \to n$:

$$\lim_{c\to n}\{Cov_i\} = \lim_{c\to n}\left\{\frac{\sum_{k=1}^{n}(u_{ik})^m(x_k - v_i)(x_k - v_i)^T}{\sum_{k=1}^{n}(u_{ik})^m}\right\} \overset{(4.24)}{=} 0. \tag{4.47}$$

As a result, the Bouguesse-Wang-Sun index gets unpredictable when the number of clusters c approaches n:

$$\lim_{c\to n}\{V_{BWS}(U,V,X)\} = \lim_{c\to n}\left\{\frac{\operatorname{tr}\left(\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{m}(v_i-\overline{x})(v_i-\overline{x})^T\right)}{\sum_{i=1}^{c}\operatorname{tr}\left(\frac{\sum_{k=1}^{n}(u_{ik})^m(x_k-v_i)(x_k-v_i)^T}{\sum_{k=1}^{n}(u_{ik})^m}\right)}\right\}$$
$$\overset{(4.47)}{=} \frac{\operatorname{tr}\left(\sum_{i=1}^{n}\sum_{k=1}^{n} u_{ik}^{m}(x_i-\overline{x})(x_i-\overline{x})^T\right)}{0} = \infty.$$

Bouguessa-Wang-Sun Index for Incomplete Data Similar to the Fukuyama-Sugeno and the Zahid-Limouri-Essaid indexes the cluster validity index of Bouguessa et al. calculates within-cluster scatter as the compactness measure and between-cluster scatter as the separation measure. Both measures involve the data set in their calculation, so they have to be adapted to data items with missing values in the case of validating partitionings of incomplete data using the Bouguessa-Wang-Sun index. While the calculation of the covariance matrix in the compactness measure can be adapted to incomplete data in the same way as in the fuzzy hypervolume (FHV) using Formula (4.17), the mean of an incomplete data set can be calculated as in the Kwon index according to Formula (4.32). Making these changes in the calculation of the compactness and the separation measures in the Bouguessa-Wang-Sun index, it can be used for validation of partitioning of incomplete data sets.

4.2.3.8 Partition Coefficient and Exponential Separation Index

In [WY05] Wu and Yang proposed the *Partition Coefficient and Exponential Separation Index (PCAES)* that especially pays attention to outliers and noisy data points while validating the partitioning results. This cluster validity index that combines the compactness and the separation criteria of a partitioning, verifies whether all clusters are well identified. Since outliers and noisy items identified as single clusters cause worse results for PCAES, this cluster validity index favors partitionings in which clusters are compact and well-separated, on the one hand, but, on the other hand, they should be large enough to be identified as real clusters.

PCAES validates each single cluster regarding their compactness and separation. According to [WY05] the compactness of a cluster is calculated using the partition coefficient normalized by the partition coefficient of the most compact cluster in the partitioning:

$$PC_i(U) = \frac{\sum_{k=1}^{n} u_{ik}^2}{\max_{1\leq i\leq c}\left(\sum_{k=1}^{n} u_{ik}^2\right)} \quad \text{for } 1 \leq i \leq c. \tag{4.48}$$

Wu and Yang consider a cluster as the most compact one if its partitioning coefficient is maximal among the clusters in the partitioning.[1] The exponential separation measure in PCAES measures the distance between cluster C_i and its nearest neighbouring cluster, where the nearest neighbour is the cluster whose centroid has the minimal distance to the centroid of cluster C_i. The exponential separation measure is defined for $i = \{1, \ldots, c\}$ as follows:

$$ES_i(V) = \exp\left(\frac{-\min_{i\neq j}\{\|v_i - v_j\|^2\}}{\beta_T}\right) \quad \text{with } \beta_T = \frac{\sum_{l=1}^{c} \|v_l - \overline{v}\|^2}{c}, \tag{4.49}$$

where $\overline{v}$ is the grand mean of the data set. The nearest neighbour distance is normalized by the between-cluster scatter β_T which depends on the partitioning of the data set. The farther the cluster prototypes are located from the grand mean of the data set, the larger is the value of the between-cluster scatter β_T. Wu and Yang use the exponential function to strengthen the differences between small and large distances. According to the authors this approach has proven beneficial for clustering. Unlike other cluster validity indexes that use the between-cluster scatter in their separation measure, the between-cluster scatter in PCAES does not use the membership matrix. Therefore, it does not use any proximity and overlap information between the clusters themselves which makes the separation measure of PCAES prone to give preference to rough clustering structures neglecting the groups of clusters.

The PCAES index for cluster C_i is defined as the difference between the compactness and the separation measures for this cluster:

$$PCAES_i(V, U) = PC_i(U) - ES_i(V) \quad \text{for } 1 \leq i \leq c. \tag{4.50}$$

The more compact the cluster is and the larger the distance to its neighbouring cluster is, the larger the value of the PCAES index for this cluster is. The PCAES index for partitioning of a data set is defined as the sum over the PCAES values of all clusters in the partitioning:

$$V_{PCAES}(V, U) = \sum_{i=1}^{c} \mathrm{PCAES}_i(V, U). \tag{4.51}$$

[1] The most compact cluster is determined by the minimal partitioning coefficient in the formula in [WY05] but this is obviously a mistake.

Since a large value of $V_{PCAES}(V,U)$ indicates well-identified clusters which are compact and well separated, the optimal number of clusters is obtained by maximizing PCAES over the range $c_{\min}, ..., c_{\max}$. In [WY05] the authors suggested to choose $c_{\min} = 2$ and $c_{\max} = \sqrt{n}$.

Partition Coefficient and Exponential Separation Index for Incomplete Data
The PCAES index only uses the membership degrees and the cluster prototypes for its calculation. Since all clustering algorithms adapted to incomplete data provide this information, the PCAES index can be used on incomplete data without any changes.

4.2.3.9 Partition Negentropy Criterion

The idea of the *Partition Negentropy Criterion (PNC)* is to find a partitioning of a data set with well separated clusters that conform the Gaussian distributions as much as possible [LFSMC09]. The partition negentropy criterion measures the quality of a partitioning of the data set using two parameters: the partition entropy and the average negentropy of clusters:

$$H(C \mid X) + J(X \mid C) \tag{4.52}$$

As mentioned above the partition entropy measures the amount of uncertainty of a clustering or in other words, the averaged degree of the overlap between the clusters in the partitioning. The second term measures the distance to the normality of clusters in the partitioning by means of their negentropy. According to [Com94] the negentropy of a cluster is defined as the distance between the entropy of the cluster and the entropy of the corresponding Gaussian distribution with the same covariance matrix:

$$J(X \mid C) = \hat{H}(X \mid C) - H(X \mid C), \tag{4.53}$$

where $H(X \mid C)$ is the differential partition entropy and $\hat{H}(X \mid C)$ is the differential entropy of a normal distribution with the same covariance matrix.

$$H(X \mid C) + H(C) = H(C \mid X) + H(X). \tag{4.54}$$

Using Property (4.54) of the conditional entropy partition negentropy criterion can be rewritten as follows:

$$\begin{aligned} H(C \mid X) + J(X \mid C) &= H(C \mid X) + \hat{H}(X \mid C) + H(C) - H(X) - H(C \mid X) \\ &= \hat{H}(X \mid C) + H(C) - H(X). \end{aligned} \tag{4.55}$$

Since the entropy of a given data set X is constant, it can be ignored. According to the definition of the conditional entropy the term $\hat{H}(X \mid C)$ can be expressed for $C = \{C_1, ..., C_c\}$ as:

$$\hat{H}(X \mid C) = \sum_{i=1}^{c} p(C_i)\hat{H}(X \mid C = C_i), \tag{4.56}$$

where $p(C_i)$ is the a-priori probability of cluster C_i, and $\hat{H}(X \mid C = C_i)$ is the differential entropy of a normal distribution with the same covariance matrix as of cluster C_i. According to [AG89] the entropy of the multivariate normal distribution can be estimated as follows:

$$\hat{H}(X \mid C_i) = \frac{1}{2}\log[(2\pi e)^d \det(Cov_{C_i})] = \frac{d}{2}\log(2\pi e) + \frac{1}{2}\log\left(\det(Cov_{C_i})\right) \quad \text{for } 1 \leq i \leq c, \tag{4.57}$$

where Cov_{C_i} is the covariance matrix of cluster C_i and d is the dimension of the data set. Since the first term is constant for a data set X, it can be ignored. Substituting Formula (4.57) into Equation (4.55) the partition negentropy criterion (PNC) can be rewritten as:

$$V_{PNC}(X, U, V) = \frac{1}{2}\sum_{i=1}^{c} p_i \log\left(\det(Cov_i)\right) - \sum_{i=1}^{c} p_i \log p_i, \tag{4.58}$$

where we write the a-priori probability of cluster C_i as p_i for short. According to [GG89] the a-priori probability of a fuzzy cluster C_i can be calculated as:

$$p_i = \frac{\sum_{k=1}^{n} u_{ik}^m}{\sum_{j=1}^{c}\sum_{k=1}^{n} u_{jk}^m}. \tag{4.59}$$

Since the amount of uncertainty of an optimal partitioning should be as low as possible and the normality of clusters should be as high as possible, a low value of the partition negentropy criterion (PNC) indicates a good partitioning.

Partition Negentropy Criterion for Incomplete Data Although the partition negentropy criterion (PNC) calculates the entropy of the partitioning itself which only involves the membership matrix, it also needs the data set for the calculation of the covariance matrix of the corresponding Gaussian distribution. As mentioned above the clustering algorithms adapted to incomplete data provide the cluster prototypes and the membership matrix as output. Therefore, the partition negentropy criterion can be calculated on incomplete data using Formula (4.58), whereas the covariance

matrix of the Gaussian distribution should be estimated in the same way as in the fuzzy hypervolume (FHV) using Formula (4.17).

4.3 Summary

In this chapter we analyzed the different cluster validity indexes from the literature towards using them on incomplete data. Since some of the indexes only use the information provided by the clustering algorithms, they can be used for validating the clusterings of incomplete data without any changes. The other CVIs additionally use the data items for their calculation. We adapted them to incomplete data according to the available case approach. Since in the most cases the problem was to calculate the distances between the incomplete data sets and the cluster prototypes, we replaced the Euclidean distances by the partial distances [Dix79]. Few cluster validity indexes use the mean of the data set for their calculation, so we calculated it on the basis of the available feature values. Furthermore, we discussed the shortages of the original cluster validity indexes regarding the determining the optimal number of clusters. In this way we try to avoid drawing false conclusions about the adaption of CVIs to incomplete data in the case the CVIs fail in the experiments.

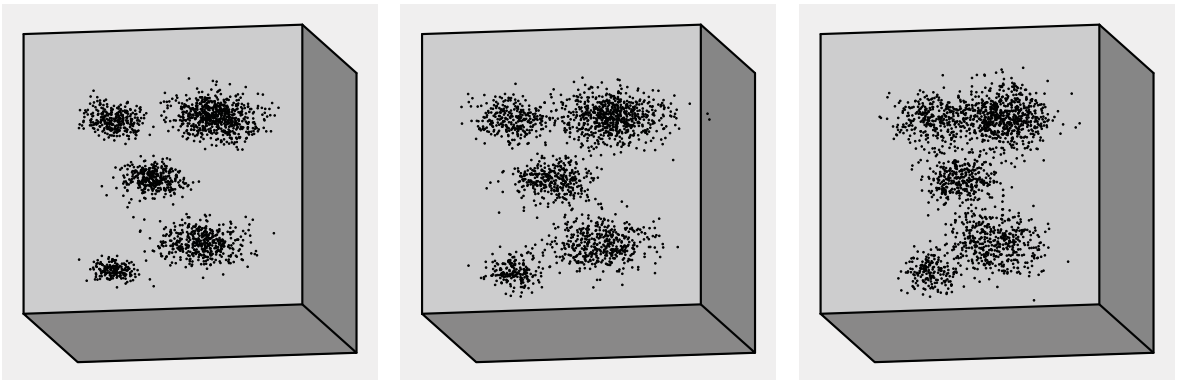
5

Experiments and Evaluation

In this chapter we present the evaluation results of the original and the adapted cluster validity functions on incomplete data. We analyzed them using the partitioning results of several artificial and real data sets produced by the different fuzzy clustering algorithms for incomplete data. Although there are many publications dedicated to the evaluation and the comparison of cluster validity indexes on complete data (see [WZ07] for example), we first tested the described cluster validity functions on complete data to be able to compare whether they perform on the same data sets with missing values as well as on the complete data. Since both the clustering algorithms and the CVIs were adapted to incomplete data, in this chapter, we also address the problem of finding the factors that are crucial for the cluster validity for fuzzy clustering of incomplete data: the adaption of the clustering algorithms, the adaption of the cluster validity functions, or the loss of information in the data itself.

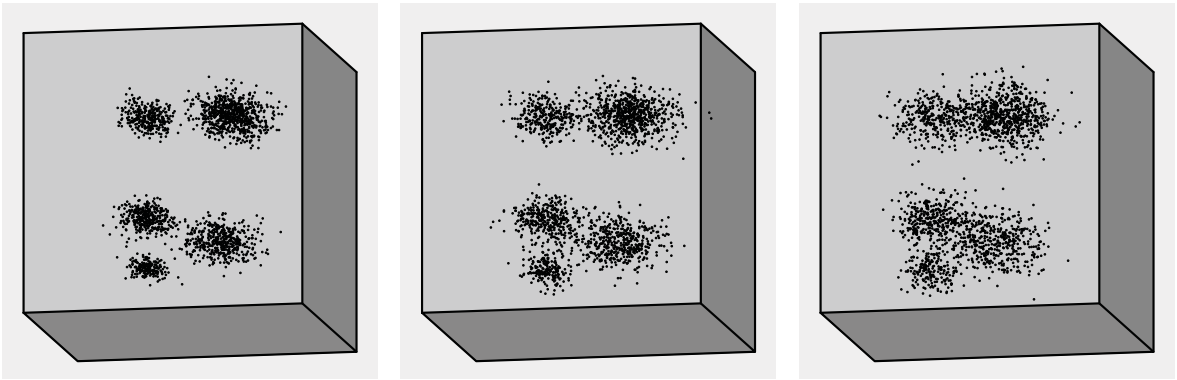
5.1 Test Data

We tested the described cluster validity functions adapted to incomplete data on several artificial and real data sets with different numbers of clusters. Apart from knowing the real number of clusters, the advantage of using artificial data sets is that they can be specifically generated to analyze the interesting properties of CVIs. The first data set series is depicted in Figure 5.1. Each of the data sets consist of 2000 data points generated by the compositions of five 3-dimensional Gaussian distributions. The five clusters have different magnitudes and contain different numbers of items. All data sets in this series have the same mean value. In order to increase the overlap between clusters, the standard deviation was gradually increased while generating the data



(a) (b) (c)

Figure 5.1: Test data: (a) 3D-5-sep, (b) 3D-5-ov, (c) 3D-5-strov.



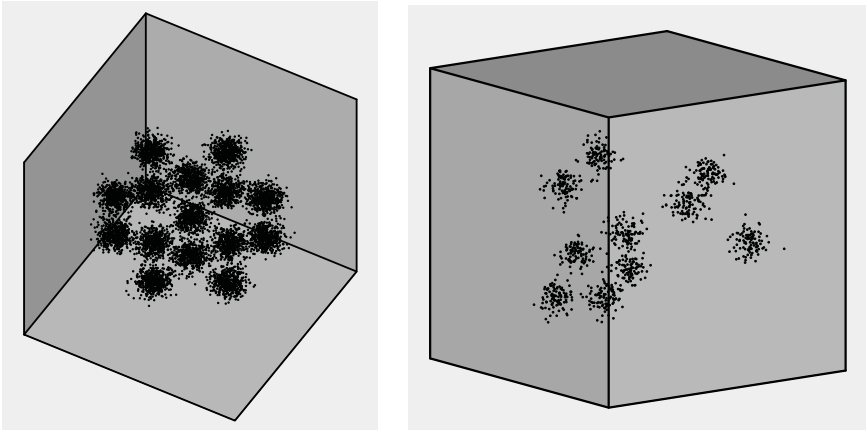
(a) (b) (c)

Figure 5.2: Test data: (a) 3D-5-h-sep, (b) 3D-5-h-ov, (c) 3D-5-h-strov.

sets. While all clusters in the data set *3D-5-sep* are clearly separated from each other, there is some overlap between clusters in the data set *3D-5-strov*. Using this relatively simple data set series, we aim to find out to which degree of overlap the cluster validity functions can determine the correct number of clusters.

Figure 5.2 shows the data sets *3D-5-h-sep*, *3D-5-h-ov* and *3D-5-h-strov* which were generated from the first data set series. These data sets are formed by moving the clusters so that two groups of two and three differently sized clusters build a hierarchical structure in the resulting data sets. We generated this data set series in order to test whether and which cluster validity functions are able to determine the real number of clusters in hierarchically structured data. The gradually increasing overlap between clusters in the data sets should make it even more difficult for CVIs to recognize the five clusters.

In our experiments, we also wanted to find out whether the number of clusters or features in the data set play an important role for determining the real number of clusters. For this purpose, we generated the data set *3D-15* which contains 7500



(a) (b)

Figure 5.3: Test data: (a) 3D-15, (b) 10D-10.

points distributed in 15 equally shaped and sized clusters. Like the other data sets, we generated the data set *3D-15* by the compositions of fifteen 3-dimensional Gaussian distributions. However, unlike the previously described the data sets, *3D-15* contains compact and well-separated clusters. Therefore, it should be easy for the cluster validity indexes to determine the real number of clusters. The data set *3D-15* is depicted in Figure 5.3 (a). To test whether the number of features in the data set is an important factor for determining the real number of clusters, we tested CVIs on the data set *10D-10* which was presented by Havens et al. in [HBP12]. The data set *10D-10* is a 10-dimensional data set which consists of 1000 points distributed in ten clusters. Figure 5.3 (b) shows the first three dimensions of this data set where all clusters are clearly recognizable.

In order to analyze how the cluster validity functions react to the overlaps between clusters without interfering factors, we generated a simple 2-dimensional data set series depicted in Figure 5.4. The basis data set *2D-3-sep* consists of 900 data items which are distributed to three equally sized and shaped clusters. We generated this data set by combining three 2-dimensional Gaussian distributions. Unlike other data sets, we changed this data set only by moving the clusters to each other without changing the standard deviations. In the first data set series, we gradually moved the middle cluster in the direction of the right cluster producing two groups of one and two partly overlapping clusters (cf. Figures 5.4 (b) - (d)). In the second test data series, we wanted to produce a gradual overlap between clusters without building hierarchical structure of clusters. For this purpose, we moved all three clusters to each other producing more and more overlap between clusters (cf. Figures 5.4 (e) - (g)).

One of the most commonly used artificial data sets associated with cluster validity is the *beinsaid* data set presented in [BHB+96]. The authors generated this data set

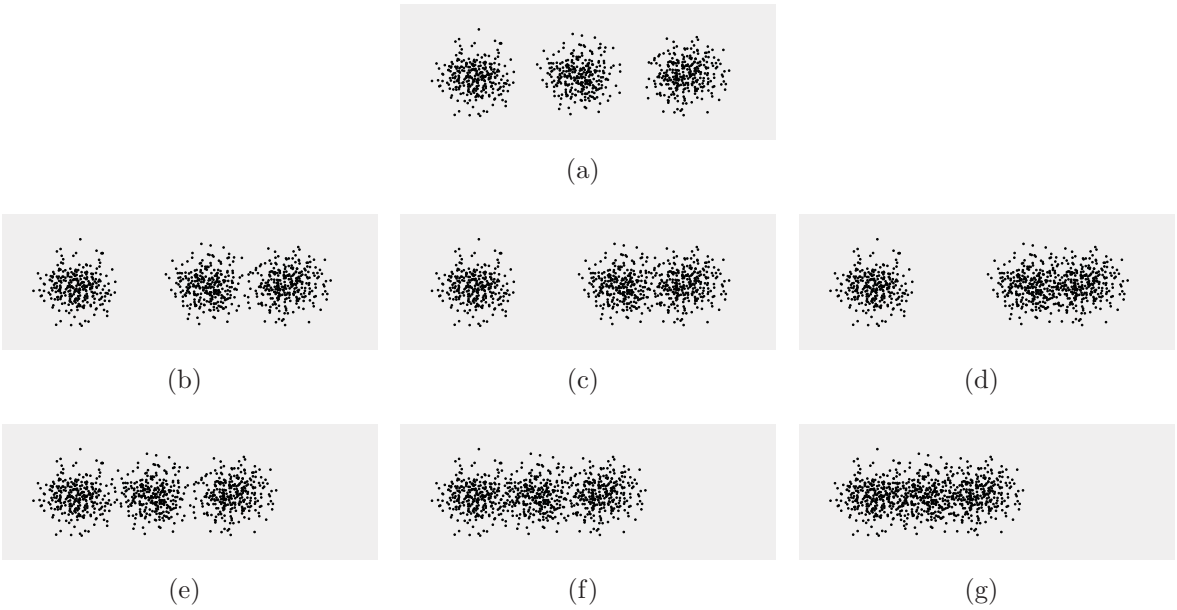


Figure 5.4: Test data: (a) 2D-3-sep, (b) 2D-3-2-tog, (c) 2D-3-2-ov, (d) 2D-3-2-strov, (e) 2D-3-3-tog, (f) 2D-3-3-ov, (g) 2D-3-3-strov.

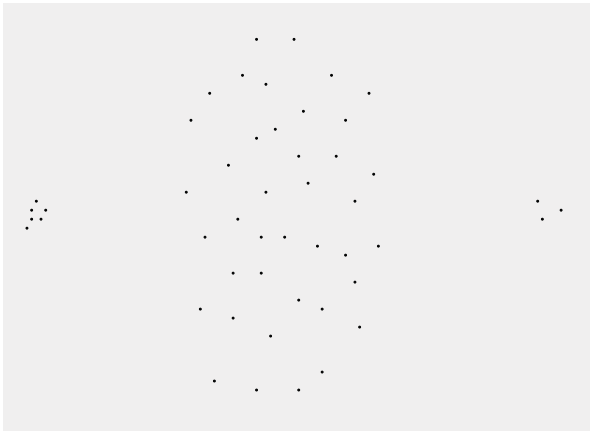


Figure 5.5: *bensaid* data set.

to show the limitations of cluster validity indexes. The *beinsaid* data set is depicted in Figure 5.5. This data set consists of 49 data items that are distributed in three differently sized clusters with 3, 6, and 40 data items, respectively. The clusters are well separated from each other but they and their number are difficult to determine by clustering algorithms and cluster validity functions. The reason is that the data points within clusters are uniformly distributed and not distributed according to the Gaussian distribution. Since clustering algorithms expect clusters with accumulation of data points around the mean value, in this data set, they fail to assign the data items into clusters in the correct way.

In our experiments, we also used eight real data sets from the UCI Machine Learning Repository [AN07]. On the one side, the advantage of using these data sets is that the

number of classes are known. These data sets are widely used in other works for testing and comparing different cluster validity approaches. On the downside, the number of labeled classes does not have to correspond to the number of clusters determined by any clustering algorithm because the class structure of these data sets could be different from the cluster structure. While the relationships between the features in the data set can decide about the class affiliation, the cluster shapes based on the distance relations between the data items are important in the clustering task. Since the number of feature dimensions of these data sets is large, the number of clusters can not be visually assessed.

The data set *ecoli* describes the localization site of proteins. This data set consists of 336 instances each with 7 attributes. The data items are distributed in 8 differently sized classes.

The data set *glass* describes different properties of glass that are used for classification of the type of glass in criminological investigation. The data set consists of 214 data items. The 9 features describe the refractive index and the concentration of different chemical components. The data items are distributed in 6 differently sized classes that represent, for example, window glass, tableware or headlamp glass.

The *ionosphere* data set consists of 351 34-dimensional examples of a radar system that records free electrons in the ionosphere. The data items are divided in two classes that correspond to "good" and "bad" radar returns from the ionosphere depending on whether or not some type of structure of free electrons was detected in the ionosphere.

Maybe the most popular and widely used data set in the cluster validity literature is the *iris* data set. It contains 150 data items. The four features describe the sepal length, the sepal width, the petal length and the petal width of iris plants. The data items are distributed in three equally sized classes, where each class corresponds to a type of iris plant. One class is clearly separated from the two other which partly overlap. In [BKK+99] Bezdek et al. pointed to the fact that there are at least two errors in the original *iris* data set presented by Fischer in [Fis36]. Since most papers do not give any notice of which data set was used, here, we test the cluster validity functions on both *iris* data sets: the original and the corrected one. We refer to the corrected data set as *iris-bezdek*.

The data set *sonar* consists of 208 data items each with 60 features representing sonar signals bounced off a metal cylinder or off a roughly cylindrical rock. Therefore, the data items are divided into two approximately equally sized classes.

The Wisconsin breast cancer data set (abbreviated as *wdbc*) consists of 569 instances with 30 attributes computed from a digitized image of FNA of a breast mass. The data items are divided in two differently sized classes that correspond to malignant and benign tumors. The "benign" class contains 357 data items, the "malignant" class consists of 212 data items.

The data set *wine* consists of 178 data items, each with 13 features representing the results of a chemical analysis of three wine types grown in the same region but derived from three different cultivars. Corresponding to three wine types, the data items are distributed in three classes with 59, 71 and 48 instances. According to [TDK04], only the attributes 7, 10 and 13 are important for dividing 3 clusters. For that reason, we also used the reduced *wine-3D* data set.

5.2 Experimental Setup

We tested the original and the adapted cluster validity functions on synthetic and real data sets. We scaled all feature data in the real data sets to $[-1; 1]$. In order to make a direct comparison between the performance of cluster validity indexes on complete and incomplete data, we first tested described cluster validity functions on complete data. In the next step, we tested the original and the adapted CVIs on the same data sets but after removing some values. We generated incomplete data sets by removing values in all dimensions with the probabilities of 10%, 25% and 40%, according to the most common *missing completely at random (MCAR)* failure mechanism. The percentage of missing values was calculated in relation to all values in the data set.

For each data set, complete and incomplete, we ran the fuzzy clustering algorithms 100 times for each integer c, $c_{\min} \leq c \leq c_{\max}$. We ran the FCM algorithm for complete data and we clustered incomplete data sets using the algorithms PDSFCM, OCSFCM and NPSFCM. To create the testing conditions as real as possible, we initialized the cluster prototypes with random values at the beginning of each trial. The iterations of all fuzzy clustering algorithms were terminated when $\|V_{new} - V_{old}\|_F < 10^{-4}$. As in other experiments, we used the Frobenius norm distance given in Formula (3.6) in chapter 3 for the stopping criterion. We then calculated cluster validity indexes for all partitioning results. For each of the 100 trials of the clustering algorithms, we stored the preferred number of clusters at the respective optimum of CVIs. Finally, for each data set we figured out the optimal number of clusters by the majority voting rule for each cluster validity function. For both data sets *3D-15* and *10D-10* with a large number of clusters, we decided for $c_{\min} = 2$ and $c_{\max} = 17$. Since the computational costs were very high for some CVIs, first of all for the OSI proposed by Le Capitaine and Frélicot, we limited $c_{\min} = 2$ and $c_{\max} = 10$ for all other data sets.

5.3 Experimental Results

We evaluate the experimental results regarding two aspects: the cluster validity index categories and the kind of data. First, we compare different cluster validity strategies

with each other before comparing the single CVIs. We evaluate the cluster validity functions regarding their performance on data sets with different data distributions like the number of attributes, the degree of overlap between clusters, hierarchical structures etc. For the sake of completeness, we tested CVIs on real data sets but we pay particular attention to the experimental results on artificial data because, unlike real data sets, we possess reliable knowledge about their real distribution. However, the tests on the real data sets are also useful to compare the performance of cluster validity functions on complete and incomplete data sets.

5.3.1 Experimental Results on Complete Data Sets

Tables 5.1 and 5.2 show the preferred number of clusters for all CVIs on complete artificial data sets. The subscript numbers in the tables indicate the number of iterations in which the preferred number of clusters was obtained. All cluster validity functions, with a few exceptions, recognized the correct number of clusters in the data set with a large number of clusters *3D-15* and in the multi-dimensional data set *10D-10*. As expected, the high separability degree between compact clusters in the data sets plays a greater role for determining the real number of clusters than the number of dimensions or clusters in the data sets. In contrast, no cluster validity index managed to determine the correct number of clusters in the data set *bensaid*, although the clusters are clearly separated from each other in this data set. This is due to the fact that the data items in this data set are uniformly distributed within the clusters but the CVIs expect compact clusters with accumulations of data points around the mean value. This is a general problem of partitioning clustering algorithms, like the FCM. Generally, they are not able to deal with such data sets in a proper way. A potential approach for recognizing such inappropriate data sets could be computing the average distance for each data item to its k-nearest neighbours. If the kNN distances of data items do not differ much from each other, the data set should not be partitioned with a partitioning clustering algorithm. This approach provides a valuable information about the data set, although it is computationally expensive.

Since there are considerable differences between different kinds of CVIs in the performance results, below we describe the experimental results grouped by category of cluster validity functions.

5.3.1.1 Cluster Validity Indexes using Membership Degrees

The data set series with five differently sized and shaped clusters turned out to be a problem for the most CVIs. While the correct number of clusters in the data set *3D-5-sep* was correctly recognized by the most (normalized) cluster validity indexes using membership degrees, all of them failed on the data sets with overlapping clusters

Table 5.1: Preferred number of clusters for different validity indexes on complete synthetic data sets (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	15_{81}	15_{81}	2_{100}	15_{81}	15_{81}	15_{81}	17_{100}	2_{100}	15_{81}	15_{81}	15_{81}	15_{81}
10D-10	10	10_{89}	10_{89}	10_{89}	10_{89}	10_{89}	10_{89}	17_{96}	2_{100}	13_{34}	10_{89}	10_{89}	7_{43}
3D-5-sep	5	2_{100}	5_{79}	2_{100}	5_{79}	5_{79}	5_{79}	10_{100}	2_{100}	5_{79}	5_{79}	5_{79}	5_{79}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{82}	5_{82}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{82}	5_{82}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	5_{50}	2_{100}	5_{50}	5_{50}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{56}	5_{56}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{63}	5_{63}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	7_{45}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{34}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	6_{29}	2_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{90}	2_{100}	2_{100}	2_{100}	3_{86}	3_{86}
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}
bensaid	3	2_{91}	5_{91}	2_{100}	10_{49}	4_{91}	6_{91}	10_{100}	2_{100}	4_{91}	10_{85}	10_{52}	5_{100}

Table 5.2: Preferred number of clusters for different validity indexes on complete synthetic data sets (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	15_{81}	15_{81}	15_{81}	14_{70}	15_{81}	15_{81}	15_{81}	15_{81}	15_{81}
10D-10	10	10_{89}	10_{89}	10_{89}	6_{45}	10_{89}	10_{89}	10_{89}	10_{89}	10_{82}
3D-5-sep	5	5_{61}	2_{100}	2_{100}	2_{100}	2_{100}	5_{79}	5_{79}	2_{100}	5_{61}
3D-5-ov	5	5_{68}	2_{100}	2_{100}	2_{100}	2_{100}	5_{82}	5_{82}	4_{64}	5_{82}
3D-5-strov	5	5_{74}	2_{100}	2_{100}	2_{100}	2_{100}	5_{82}	2_{100}	4_{68}	5_{82}
3D-5-h-sep	5	5_{26}	2_{100}	2_{100}	2_{100}	2_{100}	5_{50}	5_{50}	2_{100}	$6_{41}(5_{33})$
3D-5-h-ov	5	5_{27}	2_{100}	2_{100}	2_{100}	2_{100}	5_{56}	5_{56}	2_{100}	5_{56}
3D-5-h-strov	5	5_{29}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{63}
2D-3-sep	3	4_{30}	3_{100}	3_{100}	3_{100}	3_{100}	8_{31}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	5_{27}	2_{100}	2_{100}	2_{100}	3_{100}	$7_{25}/10_{25}$	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	5_{30}	2_{100}	2_{100}	2_{100}	3_{100}	9_{24}	3_{100}	3_{100}	3_{100}
2D-3-2-strov	3	5_{24}	2_{100}	2_{100}	2_{100}	2_{100}	10_{31}	3_{86}	2_{100}	3_{86}
2D-3-3-tog	3	3_{54}	3_{100}	3_{100}	3_{100}	3_{100}	6_{50}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	3_{76}	3_{100}	3_{100}	2_{100}	3_{100}	5_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	3_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{100}	3_{100}	3_{100}	3_{100}
bensaid	3	10_{61}	9_{72}	5_{91}	2_{100}	6_{100}	5_{82}	10_{50}	6_{100}	10_{77}

3D-5-ov and *3D-5-strov*. The reason is that membership degrees alone can provide information about the overlap and separation between clusters. This data set series contains differently scattered clusters. In the data sets with overlapping clusters, some data points have a larger distance to their cluster center than to the center of the nearest cluster. In this way, the higher the overlap degree between clusters is in the data set the more of such ambiguous data items exist. Although we can clearly recognize five clusters by a visual assessment, the cluster validity functions based on membership degrees assess the partitioning in five clusters as vague underestimating the correct number of clusters.

The data set series with hierarchical structure of clusters turned out to be an even more challenging problem for the cluster validity functions based on membership degrees. While the most CVIs could correctly determine the correct number of clusters in the data set *3D-5-sep*, only the Overlap and Separation Index $V_{OSI_{H_\gamma}}$ that uses the Hamacher T-norm could recognize the correct number of clusters in the data set *3D-5-h-sep*. The reason for the poor performance of cluster validity functions of this type is that the distance between two groups of clusters and thus the separation in this data set series is larger than the distances between five clusters. Therefore, the partitioning in two clusters is clearer than the partitioning in five clusters.

The correct number of clusters in the two-dimensional data set *2D-3-sep* with three equally sized and shaped clusters was determined by almost all cluster validity functions that only use membership degrees. Most CVIs that recognized the correct number of clusters in the simple data set *2D-3-sep* also determined the correct number of clusters in the data sets where three clusters partly overlap. The indexes V_{NPE} and V_{KKLL} even recognized the correct number of clusters in the data set *2D-3-3-strov* with a high degree of overlap between clusters. The same CVIs also managed to determine the correct number of clusters in the data set *2D-3-2-tog* despite the hierarchical structure of clusters. Despite the same degree of overlap between two clusters in the data sets *2D-3-2-ov* and *2D-3-3-ov* (*2D-3-2-strov* and *2D-3-3-strov*), none of CVIs could recognize the correct number of clusters. Since all clusters in the data set *2D-3-2-tog* are clearly separated from each other (compare Figure 5.4 (b)), the partitioning is clear enough to recognize three clusters. However, with increasing overlap between two clusters, the partitioning in three clusters gets more ambiguous than a partitioning in two clusters. That results in the poor performance of CVIs based on membership degrees.

In summary, the cluster validity indexes based on membership degrees were able to determine the correct number of clusters in the data sets with equally sized and scattered compact and well separated clusters. They also recognized the correct number of clusters in the data sets with hierarchical structure of clusters as long as the overlap degree was low. The CVIs of this category had difficulties determining the correct number of clusters in the data sets with differently sized or scattered clusters and in

the data sets with a high degree of overlap between clusters.

5.3.1.2 Cluster Validity Indexes based on Compactness

Unsurprisingly, the two cluster validity indexes based on compactness managed to determine the correct number of clusters in all data set series. Using the covariance matrix that takes the sizes and the shapes of clusters into account, both the FHV and the PD indexes assessed the partitioning with the correct number of clusters as the most compact one in the data sets with differently sized and scattered clusters. Even the fact, that some data sets contained clusters closely located to each other building distant groups of clusters, did not have much effect on the performance of the FHV and the PD indexes. Since these CVIs only consider the compactness of single clusters, the spatial arrangement of clusters plays a minor role for this cluster validity functions.

Summing up, the cluster validity functions V_{FHV} and V_{PD} determined the correct number of clusters in all data sets as long as the data items within clusters were distributed according to the Gaussian distribution.

5.3.1.3 Cluster Validity Indexes based on Compactness and Separation

The evaluation results on the synthetic data sets showed the differences between the CVIs based on compactness and separation. The experimental results on the data set series with five differently sized and shaped clusters already indicated the weaknesses of some cluster validity functions. While the Xie-Beni index and its derivates V_{Kwon}, V_{TSS} did not recognize the real number of clusters in any of these data sets, the Fukuyama-Sugeno index and the CVIs that use the same separation function managed to determine the correct number of clusters, even for a high degree of overlap between clusters. In our opinion, the reason for the poor performance of the Xie-Beni like indexes is the separation criterion. It is defined as the distance between the two closest cluster centers. The problem is that the clusters were widely distributed in the data space. So the distance between clusters in a partitioning with two clusters was much larger than the distance between two nearest clusters in the partitioning with five clusters. On the other hand, the difference between the compactness criteria is not that large because the clusters partly overlap in the data set. While the Fukuyama-Sugeno index uses the same compactness criterion as the Xie-Beni index, its separation criterion measures the distances between cluster centers and the grand mean of the data set. In our opinion, this method describes the distribution of clusters in this data set series better than the separation criterion in the Xie-Beni index. The cluster validity indexes V_{ZLE} and V_{BWS} use similar compactness and separation measures as V_{FS} and they also determined the correct number of clusters. While the Beringer-Hüllermeier index uses the same compactness measure as the Xie-Beni and the Fukuyama-Sugeno

indexes, it uses its own separation measure that combines the distances between the cluster centers with the variability of clusters. Even so, this separation measure does not seem to be appropriate for data sets with differently scattered clusters and the BH index did not recognize the correct number of clusters in this data set series. The evaluation results of the PCAES index on this data set series demonstrated the problems about the compactness and the separation measures of this CVI. On the one hand, the compactness criterion of the PCAES index is based on the PC index and inherits its problems described above. On the other hand, the separation criterion of the PCAES index is dominated by the distances to the nearest neighboring clusters. Clearly, the distance between two clusters in a partitioning with two clusters is larger than the distance between neighboring clusters in a partitioning with five clusters. Therefore, it is no surprise that the PCAES index did not determine the correct number of clusters in any data set of this series. Since the PNC index aims to find a partitioning of data that conforms Gaussian distribution and this data set series was created by a composition of Gaussian distributions, unsurprisingly, the PNC index determined the correct number of clusters.

The evaluation results on the data set series with five clusters that build two groups of clusters with two and three clusters, respectively, did not differ much from the results on the previous data set series. Only the number of runs in that the CVIs recognized the correct number of clusters decreased. Generally, the tendency to the partitioning with two clusters could have been also observed.

Although the two-dimensional data sets with three equally sized and scattered clusters seem to be an easy job for the cluster validity indexes at first sight, the experimental results on these data set series showed the weak points of CVIs based on compactness and separation. The Fukuyama-Sugeno index, that performed well on the other data sets, was not able to determine the correct number of clusters neither in the data sets with hierarchical structure of clusters nor in the simplest data set *2D-3-sep*. In our opinion, this is due to the fact that the separation function of V_{FS} depends on the spatial position of the grand mean of the data set. Unlike other data sets, in the data set *2D-3-sep* the grand mean is located in the middle cluster. As a result, the distance between the grand mean and the cluster center of the middle cluster minimizes the separation value of the Fukuyama-Sugeno index. Although the compactness value worsened for the partitionings with more than three clusters, the separation value got better. That resulted in the slightly overestimation of the number of clusters by V_{FS} in these data sets. In the data set *2D-3-3-tog* where three clusters are moved together, the separation term did not outweigh the compactness value for a larger number of clusters because all cluster centers were moved closer to the grand mean. The more the clusters were moved together, the more the Fukuyama-Sugeno index tended to the correct number of clusters. While in the data set *2D-3-3-tog* the correct number of clu-

sters was determined in 54 of 100 runs, in the data set *2D-3-3-strov* the correct number of clusters was determined in all 100 runs. As we mentioned before, the cluster validity index proposed by Zahid et al. uses similar compactness and separation functions in its SC_1 as Fukuyama-Sugeno index. So, V_{ZLE} noticeably overestimated the number of clusters for all data sets in this series. In contrast, the Xie-Beni index and its improved versions V_{Kwon} and V_{TSS} recognized the correct number of clusters in the simplest data set *2D-3-sep* and even in the data sets where three clusters are moved together. Due to a high degree of overlap in the data set *2D-3-3-strov*, these indexes underestimated the correct number of clusters. As in the other data sets with hierarchical structure of clusters these CVIs did not determine the correct number of clusters in any of data sets where two clusters are moved together.

While the Beringer-Hüllermeier index did not recognize the correct number of clusters in the data sets with differently sized and scattered clusters, it determined the correct number of clusters in almost all data sets in this series. It underestimated the number of clusters only in the data sets with a high degree of overlap between clusters. As we mentioned before, the CVI proposed by Bouguesse et al. uses similar compactness and separation criteria as the Fukuyama-Sugeno index. Unlike the Fukuyama-Sugeno index, V_{BWS} defines the separation measure as the trace of the fuzzy between-cluster scatter matrix. In other words, it calculates the volume of the between-cluster scatter matrix that is defined by the distances between the cluster centers and the mean of the data set. In contrast to the Fukuyama-Sugeno index, this approach paid off and V_{BWS} determined the correct number of clusters in all data sets of this series. The hierarchical structure of clusters in the data sets influenced the separation measure of the PCAES index counterbalancing. On the one hand, the between-cluster scatter slightly increased because two clusters moved together and the distance to the nearest neighbour of the single cluster increased. On the other hand, the distances to the nearest neighbour of two close clusters decreased. So, the compactness measure played here a crucial role. As a result, the PCAES index determined the correct number of clusters in all data sets in this series except for *2D-3-2-strov*, where the degree of overlap between two clusters is high, which has a negative impact on the compactness measure of the PCAES. The simultaneously moving together of three clusters did not effect the separation criterion of the PCAES index much. On the one hand, the between-cluster scatter decreased but, on the other hand, the nearest neighbour distances decreased too. Therefore, the PCAES index determined the correct number of clusters in all data sets. Although there was a high degree of overlap between clusters in some data sets of this series, the PNC index determined the optimal number of clusters in all data sets. This is due to the fact that the partitioning of the data sets in three clusters conformed to the (compositions of three 2-dimensional) Gaussian distributions slightly better than the partitionings in two or especially more than three

clusters.

In summary, the best cluster validity results were achieved by V_{BWS} and V_{PNC}. Generally, these CVIs were able to determine the correct number of clusters in all types of data sets. We can differentiate the remaining cluster validity indexes in three categories. The cluster validity indexes of the first category determined the correct number of clusters in the data sets with differently sized and scattered clusters but their performance depended on the spatial positioning of the grand mean of the data set. The CVIs V_{FS} and V_{ZLE} belongs to this category. The cluster validity indexes V_{BH} and V_{PCAES} form the second category of CVIs that determined the correct number of clusters on the data sets with equally sized and scattered clusters but failed on the data sets with differently sized and scattered clusters. The Xie-Beni index and its extensions V_{Kwon} and V_{TSS} belong to the third category. These indexes were able to determine the correct number of clusters in the data sets with equally sized and scattered clusters with a flat structure, i.e. the clusters did not build groups of clusters.

5.3.1.4 Experimental Results on Real Data Sets

Tables 5.3 and 5.4 show the performance results for the cluster validity indexes on the complete real data sets. As we mentioned above, the evaluation of the experimental results on the real data sets is difficult because these data sets are object of the classification task and we do not possess knowledge about the real distribution of data in the data sets especially whether it fits to the clustering structure or not. The reason is that these data sets are high dimensional, so the distribution of data could not be found out by a visual assessment.

As shown in the tables, no cluster validity index could determine the real number of clusters in the data sets *ecoli* and *glass*. The most CVIs underestimated the correct number of classes. In contrast, almost all cluster validity indexes managed to determine the correct number of clusters in the data sets *ionosphere*, *sonar*, and *wdbc*. It is important to mention that the first two data sets contain the largest number of classes among the real data sets and the last three data sets contain only two classes. On the one hand, a possible reason for recognizing the correct number of classes in the data sets with two classes may be that the data are clearly distributed in classes. On the other hand, in our experiments we chose $c_{\min} = 2$ as the lower bound of the test range for possible numbers of clusters. Therefore, the correct determination of the number of classes might have been due to the underestimation of the number of classes by chance.

The correct number of clusters in the data sets *iris* and *iris-bezdek* was only recognized by three cluster validity indexes: V_{ZLE}, V_{BWS} and V_{PCAES}. Most of the other CVIs recognized two clusters in these data sets in our experiments. Indeed, in the literature from recent years, one can find the suggestion to assume two or three clusters

Table 5.3: Preferred number of clusters for different validity indexes on complete real data sets (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	5_{100}	5_{100}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{99}	10_{99}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	9_{54}	10_{52}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	9_{51}	10_{58}
sonar	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{51}	10_{54}	10_{67}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{100}

Table 5.4: Preferred number of clusters for different validity indexes on complete real data sets (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{55}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	$10_{38}(8_{30})$
glass	6	10_{99}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{99}
ionosphere	2	10_{97}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
iris	3	4_{39}	2_{100}	2_{100}	2_{100}	2_{100}	3_{79}	3_{100}	3_{100}	10_{82}
iris-bezdek	3	5_{40}	2_{100}	2_{100}	2_{100}	2_{100}	3_{81}	3_{100}	3_{100}	10_{78}
sonar	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{67}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	8_{45}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{49}
wine	3	10_{100}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}

in the *iris* data set. Since this data set is four-dimensional, no one knows for sure how many clusters there are in the data set. On the other side, two- or three-dimensional representations of these data set show three clusters where one cluster is clearly separated from the other two clusters which partly overlap. Since in our study we treat overlapped clusters as autonomous clusters, we assumed three clusters in this data set.

The experimental results for the two *wine* data sets are in general more similar to the results on artificial data sets. A possible reason for that is that the *wine* data set has a more or less simple structure that is similar to the artificial data sets. The clusters, at least in *wine-3D*, only slightly overlap. The Partition Coefficient and the Partition Entropy underestimated the real number of clusters but their normalized versions managed to determine the correct number of clusters. As in the case of the artificial data sets with partly overlapping clusters, V_{KKLL} underestimated the correct number of clusters in the *wine* data sets. Since the clusters in the *wine* data set slightly overlap, it is of little surprise that the Overlap and Separation Index using the standard t-norm recognized the correct number of clusters in the two *wine* data sets. This cluster validity index was able to determine the correct number of clusters in the artificial data sets with overlapping clusters.

Both cluster validity indexes based on compactness overestimated the real number of clusters in the 3-dimensional *wine* data set. That might have been caused by many single data points that are further located from the clusters. The Fuzzy Hypervolume managed to determine the correct number of clusters in the original *wine* data set while the Partition Density underestimated the real number of clusters.

Almost all CVIs based on compactness and separation determined the real number of classes in the *wine* data sets. Only Fukuyama-Sugeno index recognized the correct number of clusters neither in the complete nor in the 3-dimensional *wine* data set. As in the artificial 2-dimensional data sets with three clusters, the grand mean of the *wine* data set was located in one of the clusters. As we already mentioned, that minimized the separation measure of V_{FS}. The two cluster validity indexes V_{BH} and V_{ZLE} determined the correct number of clusters in the 3-dimensional *wine* data set but failed on the complete data set. Since we can not visualize the complete data set, we can not explain this fact. We can only say that the 3-dimensional *wine* data set contains three clusters that neither partly overlap nor differ much in size, nor build groups of clusters like in the *iris* data set. Like the Fuzzy Hypervolume, the Partition Negentropy Criterion calculated the covariance matrices for clusters, it also overestimated the correct number of clusters in the 3-dimensional *wine* data set but was able to determine the real number of classes in the complete data set.

5.3.2 Evaluation of the Adapted CVIs to Incomplete Data

In this section, we want to examine to what extent the adaption of the original cluster validity indexes and the missingness of values in the data sets has a negative impact on the performance of the cluster validity indexes adapted to incomplete data. For this purpose we applied the adapted cluster validity indexes from different categories on the incomplete data using the clustering results, i.e. the membership degrees and the cluster prototypes, obtained by the FCM on the corresponding complete data [HCC12]. In this way we wanted to eliminate the influence of the partitioning results produced by the clustering algorithms adapted to incomplete data. We settled on NPC, the Fuzzy Hypervolume, and the Fukuyama-Sugeno index. Since the CVIs based on the membership degrees only use information provided by the clustering algorithms for their calculation, they are irrelevant in this experiment. Nevertheless, we show the experimental results for NPC for completeness. We decided for the FHV and the FS indexes because they use the membership degrees as well as the cluster prototypes in combination with the data items for their calculation. Furthermore, we performed our experiments on two 3-dimensional data sets with five differently sized and shaped clusters because they turned out to be the most challenging data sets for most cluster validity functions.

Table 5.5 shows the performance results for the original NPC index and the adapted FHV and Fukuyama-Sugeno indexes. In the table we listed the values achieved by the CVIs for different numbers of clusters. Although the experimental settings were rather challenging, both the FHV and the FS indexes could perfectly recognize the correct number of clusters in the data sets. Even for a large percentage of missing values in the data sets, for all numbers of clusters the values for the adapted cluster validity indexes hardly differed from the values obtained on the complete data sets. In this experiment, we showed in exemplary fashion that the adapted versions of the cluster validity indexes maintain the properties of the original CVIs and the loss of values in the data sets did not have much effect on the performance of the cluster validity indexes. In further experiments we examine to what extent the CVIs are affected by the distorted clustering results produced by different clustering algorithms adapted for incomplete data.

5.3.3 Experimental Results on Incomplete Data Sets

Certainly, the performance of the cluster validity functions depends on the clustering results produced by the clustering algorithms adapted to incomplete data. Furthermore, the increasing percentage of missing values in the data has a negative impact on the performance quality of clustering algorithms and CVIs as well. Therefore, we will involve these aspects in our discussion of the experimental results. Tables summarizing

Table 5.5: Cluster validity results of clusterings produced by FCM and using complete (left) and incomplete data (right) [HCC12].

	3D-5-h-sep		
CVIs	NPC	FHV	FS
c = 2	0.797	53.92	-201147
c = 3	0.681	47.00	-226881
c = 4	0.671	36.76	-223743
c = 5	0.708	26.03	-273846
c = 6	0.593	29.58	-224122
c = 7	0.543	33.12	-216209
c = 8	0.503	35.01	-210778
	3D-5-h-strov		
c = 2	0.739	115.32	-170471
c = 3	0.574	127.26	-190852
c = 4	0.513	117.56	-172540
c = 5	0.517	103.55	-202799
c = 6	0.438	113.03	-168654
c = 7	0.401	123.33	-163122
c = 8	0.370	129.23	-148879

	3D-5-h-sep								
	10%			25%			40%		
CVIs	NPC	FHV	FS	NPC	FHV	FS	NPC	FHV	FS
c = 2	0.797	53.69	-201057	0.797	53.04	-202566	0.797	53.28	-202774
c = 3	0.681	46.96	-226893	0.681	46.35	-227429	0.681	46.25	-228392
c = 4	0.671	36.75	-223765	0.671	36.34	-224061	0.671	37.44	-223602
c = 5	0.708	26.02	-273874	0.708	26.17	-273982	0.708	26.25	-273696
c = 6	0.593	29.49	-224104	0.593	29.49	-224225	0.593	29.95	-223999
c = 7	0.543	33.04	-216195	0.543	33.24	-216272	0.543	33.47	-216121
c = 8	0.503	34.87	-210756	0.503	35.25	-210845	0.503	35.56	-210695
	3D-5-h-strov								
c = 2	0.739	115.26	-170322	0.739	114.49	-171419	0.739	113.38	-171207
c = 3	0.574	126.87	-190913	0.574	125.50	-191817	0.574	125.71	-191694
c = 4	0.513	117.67	-172435	0.513	117.02	-172921	0.513	116.67	-172399
c = 5	0.517	103.52	-202719	0.517	104.46	-202751	0.517	102.89	-202688
c = 6	0.438	112.90	-168610	0.438	114.17	-168645	0.438	112.17	-168549
c = 7	0.401	123.64	-163081	0.401	123.99	-163137	0.401	121.72	-163050
c = 8	0.370	129.90	-148835	0.370	129.43	-148922	0.370	126.94	-148835

the complete results of our experiments can be found in Appendix A. For reasons of clarity, here, we only present the relevant parts of the tables.

In our experiments, no cluster validity index recognized the correct number of clusters in the incomplete data set *bensaid* as in the case of complete *bensaid* data set. Despite the high degree of separability between compact clusters, no cluster validity index could reliably recognize the correct number of clusters in the multi-dimensional data set *10D-10* even for a small percentage of missing values. The missing values were homogeneously distributed in all dimensions of this data set, so the "important" dimensions were hit by the missing values as much as all other dimensions. Therefore, the only reason for the poor performance of the cluster validity functions and apparently the clustering algorithms as well we see is the bias in the distance calculation caused by the absence of values in many dimensions. For the same rate of missing values in a multi-dimensional and in a low-dimensional data sets, the data items in the multi-dimensional data set are more affected by the absence of values than the data items in the low-dimensional data set. Thus, we do not mean that the cluster validity functions and the clustering algorithms will fail on the incomplete multi-dimensional data sets in any case, we just state that the absence of values in many dimensions has a negative effect for the estimation of distances between the data points.

Since performance results on incomplete data sets significantly differ depending on the kinds of CVIs, below, we describe the experimental results grouped by the category of cluster validity functions.

5.3.3.1 Cluster Validity Indexes using Membership Degrees

The CVIs of this category only rely on the membership degrees, therefore, their performance strongly depends on the partitioning results produced by the clustering algorithms. Comparing the cluster validity results of the clusterings produced by PDSFCM, OCSFCM, and NPSFCM, we observed that while for a small percentage of missing values in the data sets there was not much difference in the performance of the cluster validity functions, the CVIs performed considerably better on the clustering results produced by the OCSFCM and the NPSFCM when the proportion of missing values increased in the data sets. The reason for this is that the PDSFCM estimates distances while calculating the membership degrees. The distance estimation is based on the available case analysis which is not really aimed at considering the clustering structure. The NPSFCM and OCSFCM substitute missing values either completely by the corresponding feature values of the nearest cluster center or depending on all cluster centers. Therefore, they tend to strengthen the clustering structure which reflects in the less ambiguous membership degrees. That has a positive effect on the performance of the cluster validity indexes based on the membership degrees (compare Tables 5.6

Table 5.6: Performance results of some CVIs based on membership degrees using partitionings produced by PDSFCM and NPSFCM on selected synthetic data sets with 10% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$	V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$
3D-15	15	15_{64}	15_{64}	14_{53}	15_{45}	15_{63}	15_{63}	15_{63}	15_{63}
3D-5-sep	5	5_{83}	5_{83}	4_{68}	5_{83}	5_{82}	5_{82}	5_{82}	5_{82}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	3_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}	2_{100}	2_{100}

Table 5.7: Performance results of some CVIs based on membership degrees using partitionings produced by PDSFCM and NPSFCM on selected synthetic data sets with 40% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$	V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$
3D-15	15	17_{48}	17_{57}	3_{60}	17_{51}	15_{22}	15_{24}	13_{22}	13_{26}
3D-5-sep	5	6_{44}	6_{45}	4_{72}	10_{40}	5_{75}	5_{73}	5_{65}	5_{75}
2D-3-sep	3	3_{100}	3_{95}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	3_{100}	2_{56}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{64}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
2D-3-2-strov	3	2_{85}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
2D-3-3-tog	3	10_{59}	3_{90}	3_{100}	3_{96}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	10_{61}	3_{78}	3_{94}	10_{82}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	10_{77}	2_{100}	2_{100}	10_{99}	10_{67}	3_{100}	2_{100}	10_{72}

and 5.7).

Comparing the performance results of the CVIs for the different kinds of data sets, apart from the data set *10D-10*, there was not much difference between the results on the complete data sets and the data sets with 10% of missing values (compare Table 5.1 and Table 5.6). There were larger differences in the performance to report for a larger number of missing values in the data sets. While the CVIs failed using the partitionings produced by the PDSFCM to recognize the correct number of clusters on the data set *3D-15* with equally sized and well separated clusters, the normalized CVIs managed to recognize the correct number of clusters on the same data set clustered by the OCSFCM and the NPSFCM (compare Table 5.7). As in the case of the complete data sets, the CVIs failed to recognize the correct number of clusters on the data sets with five differently sized and scattered clusters. Only the same CVIs that recognized the correct number of clusters in the complete data set *3D-5-sep* with well separated clusters reliably managed to determine the correct number of clusters using the partitionings produced by the NPSFCM of the same data set even with a large percentage of missing values. Surprisingly, the absence of values in the 2-dimensional data sets with three equally sized and shaped clusters marginally affected the performance of the cluster validity functions regardless the percentage of missing values in the data sets. Here, too, the CVIs performed better on the clusterings produced by the OCSFCM and the NPSFCM. Even the unnormalized CVIs, PC and PE, recognized the correct number of clusters in the data sets *2D-3-sep* and *2D-3-3-tog* with a simple clustering structure.

In summary, it can be stated that there were hardly any differences between the single CVIs on incomplete data. They performed analogously to the experiments on the complete data, whereas the increasing number of missing values in the data sets had a negative effects on the recognition of the correct number of clusters in the data sets with a more complicated structure like large number of clusters or differently sized and scattered clusters. However, there were significant differences in the performance of the CVIs on the clusterings produced by different clustering algorithms adapted to incomplete data. It has been emerged that all cluster validity indexes based on the membership degrees performed better using the partitionings of the data sets produced by the OCSFCM and the NPSFCM due to their tendency to strengthen the clustering structure. The performance differences reinforced with increasing number of missing values in the data sets.

5.3.3.2 Cluster Validity Indexes based on Compactness

Comparable to the CVIs using the membership degrees, both cluster validity functions based on compactness performed on the data sets with a low number of missing values as well as on complete data sets. With a few exceptions, FHV and PD managed to

determine the correct number of clusters in all kinds of data sets, of course apart from the multi-dimensional data set *10D-10*. The performance of the CVIs based on compactness heavily declined with increasing number of missing values in the data sets. Neither FHV nor PD managed to reliably determine the correct number of clusters in any data set with 40% of missing values (compare Table 5.8). As we already stated in [HHC11, HCC12], FHV and hence PD tend to overestimate the number of clusters with increasing number of missing values in the data sets. That can be explained by the fact that the clustering algorithms compute the cluster prototypes close to the available data items that are taken into account for the calculation of both cluster validity indexes. Thus, with increasing number of clusters, the distances between the data items and the cluster prototypes get smaller and the value for FHV as well. In this way, both cluster validity indexes based on compactness that already suffer from the monotonically decreasing tendency for $c \to n$ overestimated the correct number of clusters.

Similar to the CVIs based on the membership degrees, there were some differences in the performance of FHV and PD to account for 25% of missing values in the data sets. While both CVIs quite reliably determined the correct number of clusters in the data set *3D-5-sep* with five differently sized and shaped clusters and in all two dimensional data sets with three equally sized and shaped clusters, apart from the data sets *2D-3-3-strov* using the clustering results produced by the OCSFCM, they failed here and there using the clustering results produced by the PDSFCM and NPSFCM (compare Table 5.8). A somewhat better performance on the clustering results of the OCSFCM, comparing to the NPSFCM, can be explained by the fact that the OCSFCM estimates missing values depending on all cluster prototypes while the NPSFCM replaces missing values by the corresponding values of the nearest cluster center. Considering the fact that in this algorithms the computation of cluster centers and the computation of missing values influence each other, in this way both clustering strategies strengthen the clustering structure but the NPSFCM does it even more. Although, comparing to the PDSFCM, it is a beneficial for determining the correct number of clusters, this property even intensifies the tendency to overestimate the correct number of clusters because both CVIs based on compactness tend to favor a large number of clusters due to their small volume.

As for the other data sets with 25% of missing values, both CVIs performed comparably using the clustering results produced by three clustering algorithms adapted to incomplete data. Both cluster validity functions partially determined the correct number of clusters in the data set *3D-15* with a large number of clusters as a local optimum competing with $c_{\max}$. Although the two CVIs based on compactness are designed to recognize the correct number of clusters in the data sets with differently sized and shaped clusters independently of their arrangement in the data space, neither

Table 5.8: Performance results of CVIs based on compactness using partitionings produced by PDSFCM and OCSFCM on some incomplete synthetic data sets.

data set	c	10%				25%				40%			
		PDSFCM		OCSFCM		PDSFCM		OCSFCM		PDSFCM		OCSFCM	
		V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}
3D-15	15	15_{64}	15_{64}	15_{53}	15_{53}	15_{43}	17_{46}	$17_{37}(15_{30})$	$15_{29}/17_{29}$	17_{59}	17_{77}	17_{31}	17_{30}
3D-5-sep	5	5_{83}	5_{83}	5_{83}	5_{81}	5_{52}	6_{37}	5_{52}	5_{36}	10_{63}	10_{70}	10_{31}	10_{30}
3D-5-ov	5	5_{88}	5_{88}	5_{89}	5_{82}	6_{74}	9_{44}	6_{40}	6_{32}	10_{61}	10_{70}	9_{24}	10_{28}
3D-5-strov	5	5_{87}	5_{87}	5_{87}	5_{57}	6_{71}	6_{61}	5_{37}	6_{28}	10_{83}	10_{85}	10_{36}	10_{35}
3D-5-h-sep	5	5_{56}	5_{56}	5_{67}	5_{67}	6_{50}	6_{52}	6_{33}	7_{35}	10_{73}	8_{56}	10_{44}	8_{29}
3D-5-h-ov	5	5_{73}	5_{73}	5_{71}	5_{65}	6_{60}	6_{54}	6_{49}	6_{44}	10_{76}	10_{60}	10_{54}	8_{35}
3D-5-h-strov	5	5_{62}	6_{64}	5_{65}	5_{52}	10_{29}	8_{70}	7_{28}	8_{42}	10_{79}	8_{52}	10_{41}	8_{45}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{82}	3_{100}	3_{90}	10_{50}	10_{48}	9_{27}	$5_{18}/9_{18}$
2D-3-2-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{89}	10_{60}	10_{64}	10_{49}	10_{17}
2D-3-2-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{83}	3_{100}	3_{80}	10_{64}	10_{62}	10_{42}	10_{31}
2D-3-2-strov	3	3_{86}	3_{86}	3_{85}	3_{85}	3_{75}	10_{49}	3_{74}	3_{35}	10_{80}	10_{82}	10_{51}	10_{40}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{55}	3_{100}	3_{84}	10_{64}	10_{72}	10_{45}	10_{27}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{73}	3_{100}	3_{96}	10_{71}	10_{88}	10_{63}	10_{57}
2D-3-3-strov	3	3_{100}	3_{100}	3_{100}	3_{99}	10_{100}	10_{100}	$10_{35}(3_{21})$	10_{63}	10_{76}	10_{71}	10_{61}	10_{46}

FHV nor PD managed to determine the correct number of clusters in the incomplete 3-dimensional data set with five hierarchically structured clusters for 25% of missing values. That might seem surprising at first glance because the data set *3D-5-h-sep* was created from the data set *3D-5-sep* by building groups of clusters. Even if we independently removed the values from both data sets, the results should have been somewhat comparable. In fact, this is another example that shows how much influence the clustering results have on the cluster validity indexes. If we look back at the cluster validity results on the complete data sets, we see that FHV and PD recognized the correct number of clusters in 79 of 100 runs in the data set *3D-5-sep* but they only determined the correct number of clusters in 50 of 100 runs in the data set *3D-5-h-sep* (compare Table 5.1). This tendency was noticeable by all cluster validity indexes. The reason is that the clustering algorithm produced two different kinds of the partitionings of the data set *3D-5-h-sep* for $c = 5$. One of them was conforming to the original structure of the data set and was also rated as the best partitioning by FHV and PD. In the other partitioning, the cluster partially differed from the original clustering structure. Thus, the moderate performance of the CVIs might have been due to a moderate performance of the clustering algorithms on such kind of data. Since missing values in the data set pose a challenge for the clustering algorithms, the poor performance of the CVIs based on compactness can be partly justified by the poor partitioning results of the clustering algorithms. On the other hand, as we will see later, some of the other CVIs managed to correctly determine the number of clusters in the same data set using the same partitioning results. Therefore, the reason for the poor performance of FHV and PD partly lies in the functionality of these approaches themselves, particularly in their monotonically decreasing tendency for $c \to n$.

Summarizing, it can be said that the two analyzed cluster validity indexes based on compactness performed on the data sets with a small percentage of missing values as well as on the complete data. For 25% of missing values in the data sets, they failed on the data sets with a complicated clustering structure and determined the correct number of clusters in the simple data sets whereas the most reliable results were achieved using the clustering results produced by the OCSFCM. Both CVIs completely failed on all data sets with a large percentage of missing values. They overestimated the correct number of clusters due to their monotonically decreasing tendency for $c \to n$ which was intensified by inaccurate partitioning results produced by the clustering algorithms. This property disqualifies FHV and PD and actually all cluster validity indexes based on compactness for validating clustering results obtained on data with a large percentage of missing values unless the partitioning results of the data sets can be well determined by the clustering algorithms adapted to incomplete data.

5.3.3.3 Cluster Validity Indexes based on Compactness and Separation

Unlike the cluster validity functions of the other categories, there are considerable differences in the performance between the CVIs based on compactness and separation on incomplete data. Some of them hardly lost performance with increasing number of missing values in the data sets, other CVIs totally failed on the data sets with a large percentage of missing values. This is due to the fact that the CVIs of this category use different measures for compactness and separation.

Like in the case of the CVIs of the other categories there were hardly any differences in the performance of the CVIs based on compactness and separation on the data sets with a small percentage of missing values and the complete data. With the increasing number of missing values in the data sets, three groups of the CVIs emerged regarding their performance. One of the groups build cluster validity indexes that use the distances between the cluster prototypes in their separation criteria. We count the Xie-Beni index, its derivates V_{Kwon} and V_{TSS}, the PCAES, and the Beringer-Hüllermeier indexes. The PCAES index additionally uses the distances between the grand mean and the cluster prototypes in its separation criterion but it uses that only for normalization. As a result, these indexes were able to recognize the correct number of clusters only in the relatively simple data sets with equally sized and scattered clusters. On the other hand, as Tables 5.9 and 5.10 show, they hardly lost performance with the increasing number of missing values in the data sets. They managed to determine the correct number of clusters in the 2-dimensional data sets with three clusters even for 40% of missing values as well as in the complete data sets. Like the other CVIs, these cluster validity indexes also had problems to recognize the correct number of clusters in the data set *3D-15* with a large number of clusters, though. Overall, the performance of V_{XB}, V_{Kwon}, and V_{TSS} was inferior to the cluster validity indexes using the membership degrees. So, it is questionable whether the calculation effort pays off toward the performance. Only the PCAES and the Beringer-Hüllermeier indexes outperformed the CVIs based on the membership degrees, although the PCAES index was more sensitive to the partitioning results of the data sets than the other CVIs of this group. In our experiments it totally failed using the clustering results of the data sets even with 25% of missing values produced by the OCSFCM. The Xie-Beni index and its derivates also performed poorer using the partitionings produced by the OCSFCM of the data sets with a large number of missing values (compare Table 5.10). This is due to the fact that OCSFCM tend to displace the cluster centers due to its estimation strategy. That distorts the separation measure of the CVIs. In contrast, the Beringer-Hüllermeier index turned out to be resistant against the different partitioning results produced by the different clustering algorithms adapted to incomplete data. It performed equally using the clusterings produced by the different clustering algorithms adapted to incomplete

Table 5.9: Performance results of some CVIs based on compactness and separation using partitionings produced by PDSFCM and OCSFCM on selected synthetic data sets with 10% of missing values.

data set	c	PDSFCM				OCSFCM			
		V_{XB}	V_{Kwon}	V_{BH}	V_{PCAES}	V_{XB}	V_{Kwon}	V_{BH}	V_{PCAES}
3D-15	15	15_{64}	15_{64}	15_{64}	15_{64}	15_{53}	15_{53}	15_{53}	$14_{35}(15_{33})$
3D-5-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{76}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{86}
2D-3-2-tog	3	2_{100}	2_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{72}
2D-3-2-ov	3	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{87}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{96}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{86}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{87}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	3_{100}	2_{100}	2_{100}	2_{100}	3_{81}

Table 5.10: Performance results of some CVIs based on compactness and separation using partitionings produced by PDSFCM and OCSFCM on selected synthetic data sets with 40% of missing values.

data set	c	PDSFCM				OCSFCM			
		V_{XB}	V_{Kwon}	V_{BH}	V_{PCAES}	V_{XB}	V_{Kwon}	V_{BH}	V_{PCAES}
3D-15	15	14_{46}	14_{46}	14_{47}	14_{33}	10_{22}	10_{22}	4_{46}	9_{16}
3D-5-sep	5	5_{74}	5_{74}	5_{74}	4_{57}	2_{100}	2_{100}	2_{95}	2_{46}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	2_{69}
2D-3-2-tog	3	2_{100}	2_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	2_{89}
2D-3-2-ov	3	2_{100}	2_{100}	3_{100}	3_{53}	2_{100}	2_{100}	3_{95}	2_{99}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{87}	2_{100}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	2_{70}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	2_{61}	2_{81}	3_{99}	2_{70}
2D-3-3-strov	3	2_{100}	2_{100}	10_{46}	3_{100}	2_{100}	2_{100}	2_{69}	2_{64}

data.

The Partition Negentropy Criterion belongs to the second group of cluster validity functions that, like the cluster validity indexes based on compactness, uses the covariance matrices of clusters for its calculation. As Table 5.11 shows, PNC extremely lost performance with the increasing number of missing values in the data sets. As we already stated above, the clustering algorithms adapted to incomplete data tend to strengthen the clustering structure with the increasing number of clusters. In this way, the volumes of clusters get smaller and the covariance matrices as well. As a result, PNC overestimated the correct number of clusters in the data sets with a large number of missing values. As in the case of the cluster validity indexes based on compactness, PNC achieved the best results using the clusterings produced by the OCSFCM. The worst results were obtained using the clusterings produced by the NPSFCM. The PNC index determined $c_{\max}$ as the optimal number of clusters in all data sets even with 25% of missing values (compare Table 5.11).

The last group build the cluster validity indexes that use the distances between the cluster prototypes and the grand mean of the data set in their separation criteria. The indexes FS, ZLE, and BWS belong to this group. On the one hand, this turned out to be a more appropriate separation criterion than the distances between the cluster prototypes. On the other hand, its efficiency depends on the spatial position of the grand mean of the data set. Unlike the other CVIs based on compactness and separation, these three CVIs generally performed better on the 3-dimensional data sets with differently sized and scattered clusters than on the 2-dimensional data sets with three equally sized and shaped clusters. As in the case of complete data, this was due to the weaknesses of the separation criterion. The cluster centers in the 3-dimensional data sets with five clusters were evenly spread in the data space while the cluster centers in the data sets with three clusters were aligned in a line. The separation criterion was adversely affected by this alignment of the cluster centers. Combined with the strengthening of the clustering structure by the clustering algorithms, the CVIs overestimated the correct number of clusters in the data sets with a large number of missing values (compare Table 5.12). Comparing the performance of these three CVIs depending on the clusterings produced by the different clustering algorithms, there was no odds-on favorite. As Table 5.12 shows, the cluster validity indexes reliably determined the correct number of clusters in the data sets with five differently sized and scattered clusters using the partitioning results produced by the NPSFCM. However, they failed on the data sets with three clusters. The CVIs performed better on the data sets with three clusters using the partitioning results produced by the OCSFCM or the PDSFCM, however they performed considerably worse on the data sets with five clusters.

In summary, as the other cluster validity functions the CVIs based on compactness and separation performed on the data sets with a small percentage of missing values

Table 5.11: Performance results of PNC using partitionings produced by PDSFCM, OCSFCM, and NPSFCM on some incomplete synthetic data sets.

data set	c	10%			25%			40%		
		PDSFCM	OCSFCM	NPSFCM	PDSFCM	OCSFCM	NPSFCM	PDSFCM	OCSFCM	NPSFCM
3D-15	15	15_{64}	15_{44}	15_{63}	15_{42}	17_{47}	17_{31}	17_{62}	$16_{34}/17_{34}$	17_{32}
3D-5-sep	5	5_{83}	5_{69}	5_{82}	10_{26}	5_{27}	10_{36}	10_{72}	10_{40}	10_{70}
3D-5-ov	5	5_{88}	5_{86}	5_{93}	7_{60}	$5_{34}/6_{34}$	10_{38}	10_{70}	10_{32}	10_{62}
3D-5-strov	5	5_{87}	5_{84}	5_{84}	10_{45}	$6_{34}(5_{29})$	10_{41}	10_{82}	10_{36}	10_{70}
3D-5-h-sep	5	5_{56}	5_{59}	5_{74}	10_{23}	6_{24}	$9_{36}/10_{36}$	10_{88}	10_{45}	10_{79}
3D-5-h-ov	5	5_{73}	5_{69}	5_{58}	5_{69}	6_{24}	10_{58}	10_{80}	10_{56}	10_{75}
3D-5-h-strov	5	5_{62}	5_{64}	5_{64}	10_{40}	7_{29}	10_{55}	10_{81}	10_{45}	10_{69}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{73}	3_{52}	10_{58}	10_{67}	10_{41}	10_{58}
2D-3-2-tog	3	3_{100}	3_{100}	3_{100}	3_{94}	3_{65}	10_{41}	10_{77}	10_{47}	10_{57}
2D-3-2-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{52}	10_{55}	10_{75}	10_{48}	10_{69}
2D-3-2-strov	3	3_{86}	3_{85}	3_{87}	10_{64}	3_{30}	10_{63}	10_{81}	10_{40}	10_{66}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	10_{63}	3_{50}	10_{41}	10_{71}	10_{51}	10_{73}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	10_{95}	3_{45}	10_{58}	10_{72}	10_{51}	10_{57}
2D-3-3-strov	3	3_{100}	3_{100}	3_{100}	10_{100}	10_{38}	10_{70}	10_{77}	10_{51}	10_{64}

Table 5.12: Performance results of the FS and the BWS indexes using partitionings produced by OCSFCM and NPSFCM on some incomplete synthetic data sets.

data set	c	10%				25%				40%			
		OCSSFCM		NPSFCM		OCSFCM		NPSFCM		OCSFCM		NPSFCM	
		V_{FS}	V_{BWS}	V_{FS}	V_{BWS}	V_{FS}	V_{BWS}	V_{FS}	V_{BWS}	V_{FS}	V_{BWS}	V_{FS}	V_{BWS}
3D-15	15	15_{51}	15_{53}	15_{63}	15_{63}	15_{26}	15_{32}	14_{41}	14_{36}	16_{17}	$16_{21}(15_{18})$	16_{20}	15_{24}
3D-5-sep	5	5_{64}	5_{86}	5_{71}	5_{82}	5_{56}	5_{77}	5_{72}	5_{83}	6_{39}	$6_{38}(5_{34})$	5_{52}	5_{69}
3D-5-ov	5	5_{71}	5_{89}	5_{92}	5_{93}	5_{40}	5_{60}	5_{76}	5_{90}	5_{27}	5_{30}	5_{51}	5_{73}
3D-5-strov	5	5_{67}	2_{94}	5_{57}	5_{84}	5_{42}	2_{82}	5_{73}	5_{88}	7_{27}	2_{53}	$7_{29}(6_{28})$	5_{50}
3D-5-h-sep	5	5_{49}	5_{67}	5_{42}	5_{75}	5_{41}	5_{59}	5_{52}	5_{68}	5_{30}	$6_{41}(5_{39})$	$5_{35}/6_{35}$	5_{47}
3D-5-h-ov	5	5_{48}	5_{67}	5_{33}	5_{58}	5_{34}	5_{42}	5_{36}	5_{58}	5_{27}	2_{29}	5_{38}	5_{53}
3D-5-h-strov	5	5_{40}	2_{100}	5_{40}	2_{100}	5_{34}	2_{100}	5_{49}	2_{100}	5_{28}	2_{87}	5_{27}	5_{38}
2D-3-sep	3	3_{41}	3_{92}	3_{45}	3_{100}	3_{42}	3_{62}	3_{39}	3_{29}	3_{35}	3_{32}	4_{37}	10_{34}
2D-3-2-tog	3	3_{38}	3_{96}	$3_{37}/4_{37}$	3_{100}	3_{38}	3_{53}	4_{31}	3_{28}	3_{27}	3_{19}	4_{33}	10_{46}
2D-3-2-ov	3	3_{48}	3_{79}	4_{50}	3_{100}	$4_{33}/3_{31}$	3_{50}	4_{33}	$7_{17}/9_{17}$	4_{33}	$10_{18}(3_{16})$	4_{38}	10_{48}
2D-3-2-strov	3	3_{31}	3_{80}	$4_{34}/2_{33}$	3_{87}	3_{41}	3_{36}	3_{30}	6_{20}	3_{33}	10_{24}	$4_{22}(3_{21})$	10_{56}
2D-3-3-tog	3	3_{57}	3_{94}	3_{60}	3_{97}	3_{39}	3_{53}	4_{51}	8_{24}	3_{29}	6_{20}	$6_{20}(7_{19})$	10_{54}
2D-3-3-ov	3	3_{48}	3_{79}	4_{42}	3_{72}	3_{39}	3_{43}	7_{23}	7_{25}	3_{24}	9_{30}	8_{23}	10_{58}
2D-3-3-strov	3	3_{32}	3_{43}	3_{32}	3_{36}	$6_{17}/7_{17}$	6_{20}	7_{21}	7_{21}	5_{22}	10_{33}	10_{38}	10_{63}

as well as on the complete data sets. However, no CVI of this category could recognize the correct number of clusters in all data sets with a large number of missing values. However, the Beringer-Hüllermeier index turned out to be the most resistant CVI against the increasing number of missing values in the data sets. It also performed in the same way using the partitioning results of the data sets produced by the different clustering algorithms adapted to incomplete data. The weak point of the Beringer-Hüllermeier index is that it recognized the correct number of clusters only in data sets with a simple clustering structure. Using the partitioning results produced by the NPSFCM, the best results on the data sets with a complicated clustering structure were achieved by the BWS index. Even for a large number of missing values, this CVI determined the correct number of clusters in nearly all 3-dimensional data sets with five differently shaped and sized clusters. The weaknesses of this CVI are that it failed on the simple data sets and its performance depends on the partitioning results produced by the clustering algorithms. The first one is due to the limitations of the separation criterion of the BWS index. The second one indicates the sensitivity of the index to the small deviations in the partitioning results because the clusterings of the data sets produced by the different clustering algorithms generally did not differ that much. The performance of the remaining CVIs based on compactness and separation was much more affected by the missing values in the data sets. Partially, they performed even worse than the CVIs based on the membership degrees. In conclusion, the idea to combine the compactness and separation in a cluster validity index is promising regarding the incomplete data. There is only an index missing that would combine the resistance of the Beringer-Hüllermeier index against a large number of missing values in the data sets and the ability of the BWS index to determine the correct number of clusters in the data sets with a complicated clustering structure.

5.3.3.4 Experimental Results on Real Data Sets

Similar to the experimental results on the incomplete artificial data sets, the cluster validity functions using the membership degrees hardly lost performance with the increasing number of missing values in the data sets. There were only two indexes, NPE and OSI_{D_γ}, that systematically failed to recognize the correct number of clusters in the 3-dimensional version of the *wine* data set with a large number of missing values (compare Table 5.15). Comparing the performance results of the CVIs depending on the clusterings produced by the different clustering algorithms adapted to incomplete data, the KKLL, the OSI_S, and the OSI_{H_γ} indexes even managed to recognize the correct number of clusters in some data sets clustered by the OCSFCM and the NPSFCM that they could not recognize in the complete data sets. That happened sometimes in our other experiments too because the missingness of some values in the data set

Table 5.13: Performance results of some CVIs based on membership degrees using partitionings produced by PDSFCM and NPSFCM on selected real data sets with 10% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$	V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$
ionosphere	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{100}
sonar	2	2_{95}	2_{100}	2_{97}	10_{100}	2_{99}	2_{100}	2_{99}	10_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{100}
wine-3D	3	3_{100}	2_{100}	3_{100}	10_{86}	3_{100}	2_{100}	3_{100}	10_{83}
wine	3	3_{100}	2_{100}	3_{100}	10_{100}	3_{100}	2_{100}	3_{100}	10_{100}

Table 5.14: Performance results of some CVIs based on membership degrees using partitionings produced by PDSFCM and NPSFCM on selected real data sets with 25% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$	V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$
ionosphere	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{100}
sonar	2	2_{98}	2_{100}	2_{99}	10_{100}	2_{100}	2_{100}	2_{100}	10_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	3_{100}	3_{100}	3_{100}	3_{54}	3_{100}	3_{100}	3_{100}	3_{83}
wine	3	3_{100}	2_{100}	3_{100}	10_{100}	3_{100}	2_{100}	3_{100}	10_{100}

Table 5.15: Performance results of some CVIs based on membership degrees using partitionings produced by PDSFCM and NPSFCM on selected real data sets with 40% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$	V_{NPE}	V_{KKLL}	V_{OSI_S}	$V_{OSI_{D_\gamma}}$
ionosphere	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{99}	2_{99}	2_{99}	10_{80}
sonar	2	2_{98}	2_{100}	2_{98}	10_{100}	2_{65}	2_{100}	2_{71}	10_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	10_{50}	2_{100}	2_{100}	10_{74}	10_{49}	2_{56}	3_{80}	10_{70}
wine	3	3_{100}	2_{100}	3_{100}	10_{100}	3_{100}	2_{100}	3_{100}	3_{98}

made the clustering structure clearer. Although the performance differences between the different clustering algorithms were not especially large, as Tables 5.13 - 5.15 show, the results of this experiment confirms the fact that the cluster validity indexes using the membership degrees tend to perform better using the clustering results produced by the OCSFCM and the NPSFCM.

The performance results of the cluster validity functions based on compactness on the incomplete real data sets seem somewhat peculiar at first view. Comparing to complete data, these CVIs failed on few data sets with 10% and 25% of missing values but they overperformed on the data sets with 40% of missing values (compare Table 5.16). They even determined the correct number of clusters in the *glass* data set using the clustering results produced by the NPSFCM. That contradicts the tendency that we observed on the incomplete artificial data sets. On the one hand, this can be due to the fact that FHV and PD determined the correct number of clusters in fewer complete data sets than the other CVIs. On the other hand, the reason could be a simple clustering structure of the real data sets for those the number of clusters was correctly determined. The most real data sets, where the correct number of classes was determined, contained only two classes. Those classes were either compact and well separated like in the data sets *ionosphere* or they were not distinguishable, i.e. there was only a single cluster like in the data set *sonar*. In such data sets the missingness of values does not change the clustering structure much. For those reasons it is not really surprising that the cluster validity indexes based on compactness hardly lost performance with the increasing number of missing values in the data sets.

The performance results of the cluster validity indexes based on compactness and separation on the incomplete real data sets were similar to the results obtained on the incomplete artificial data sets. As Tables 5.17 and 5.18 show, the performance of the CVIs declined with the increasing number of missing values in the data sets. Similar to the experiments on the artificial incomplete data sets, the cluster validity indexes V_{XB}, V_{Kwon}, V_{BH}, and V_{PCAES} hardly lost the performance with the increasing number of missing values in the data sets. Only the index V_{TSS} overestimated the correct number of clusters in few data sets with a large number of missing values especially using the clustering results produced by the OCSFCM and the NPSFCM. Similar to the CVIs based on compactness, V_{PNC} has not lost much performance with the increasing number of missing values in the real data sets. The indexes V_{ZLE} and V_{BWS} sustained the largest performance losses using the clusterings produced by the NPSFCM. As in the experiments described above, they overestimated the correct number of clusters. As the other CVIs using the partitioning results produced by the PDSFCM and the OCSFCM, they hardly lost performance. We cannot comment on the performance of the Fukuyama-Sugeno index because it did not determine the correct number of clusters in any real data sets.

Table 5.16: Performance results of CVIs based on compactness using partitionings produced by PDSFCM and NPSFCM on some incomplete real data sets.

data set	c	10%				25%				40%			
		PDSFCM		NPSFCM		PDSFCM		NPSFCM		PDSFCM		NPSFCM	
		V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}	V_{FHV}	V_{PD}
glass	6	10_{72}	10_{69}	10_{89}	10_{89}	8_{43}	8_{37}	8_{53}	8_{50}	10_{63}	10_{78}	6_{42}	6_{40}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{72}	2_{100}	2_{100}	2_{100}	2_{51}	2_{93}
sonar	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wdbc	2	3_{100}	3_{100}	5_{100}	5_{100}	8_{100}	7_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{93}	2_{95}
wine-3D	3	10_{64}	10_{76}	10_{65}	8_{35}	10_{75}	10_{76}	10_{70}	10_{54}	10_{57}	10_{64}	10_{23}	7_{30}
wine	3	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	2_{88}	3_{100}	2_{94}	3_{100}	3_{100}	3_{93}	3_{65}

Table 5.17: Performance results of some CVIs based on compactness and separation using partitionings produced by PDSFCM and NPSFCM on selected real data sets with 10% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{XB}	V_{BH}	V_{BWS}	V_{PNC}	V_{XB}	V_{BH}	V_{BWS}	V_{PNC}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{99}
iris	3	2_{100}	2_{100}	3_{100}	10_{83}	2_{100}	2_{100}	3_{99}	10_{85}
iris-bezdek	3	2_{100}	2_{100}	3_{100}	10_{80}	2_{100}	2_{100}	3_{100}	10_{82}
sonar	2	2_{98}	2_{99}	2_{95}	2_{100}	2_{99}	2_{100}	2_{99}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	3_{100}	3_{100}	3_{100}	10_{58}	3_{100}	3_{100}	5_{100}	10_{60}
wine	3	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}

Table 5.18: Performance results of some CVIs based on compactness and separation using partitionings produced by PDSFCM and NPSFCM on selected real data sets with 40% of missing values.

data set	c	PDSFCM				NPSFCM			
		V_{XB}	V_{BH}	V_{BWS}	V_{PNC}	V_{XB}	V_{BH}	V_{BWS}	V_{PNC}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{99}	2_{56}
iris	3	2_{100}	2_{100}	3_{93}	9_{31}	2_{100}	2_{100}	4_{43}	9_{32}
iris-bezdek	3	2_{100}	2_{100}	3_{100}	10_{55}	2_{100}	2_{100}	3_{100}	10_{39}
sonar	2	2_{99}	2_{100}	2_{98}	2_{100}	2_{79}	2_{87}	2_{94}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{92}
wine-3D	3	3_{94}	10_{47}	10_{47}	10_{65}	3_{74}	3_{88}	10_{27}	10_{45}
wine	3	3_{100}	2_{100}	3_{100}	9_{83}	3_{100}	2_{100}	3_{100}	5_{59}

In summary, the cluster validity indexes hardly lost performance on the incomplete real data sets comparing to the complete data. This is due to the fact that the real data sets for those the number of clusters was correctly determined on the complete data have a simple clustering structure. In such cases the missingness of values usually does not change the clustering structure much. Otherwise, the performance of the cluster validity indexes on the incomplete real data sets corresponded to the results obtained on the incomplete artificial data sets.

5.4 Conclusions and Future Work

The quality of the partitioning results of data produced by the clustering algorithms strongly depends on the assumed number of clusters. In this chapter, we analyzed the original and the adapted cluster validity functions on different artificial and real incomplete data sets. We found out that the performance of the cluster validity indexes adapted to incomplete data mainly depends on the clustering results produced by the clustering algorithms. In experiments on the incomplete data, using the clustering results obtained on complete data the adapted cluster validity indexes performed as well as on complete data, even for a large number of missing values in the data sets. Also, for a small number of missing values in the data sets, all cluster validity indexes performed as well as on the complete data. With the increasing number of missing values in the data sets, there were large differences in the performance of the CVIs to report. While the two CVIs based on compactness, FHV and PD, and PNC totally failed on the data sets with a large percentage of missing values, the CVIs using the membership degrees and the CVIs based on compactness and separation that use the distances between the cluster prototypes in their separation criteria, hardly lost performance with the increasing number of missing values in the data sets. However, while FHV, PD, and PNC were able to recognize the correct number of clusters in the data sets with a complicated clustering structure, the last CVIs recognized the correct number of clusters only in the data sets with equally sized and shaped clusters.

As mentioned above, the performance of the CVIs depends on the quality of the clustering results produced by the clustering algorithms. Even though, some CVIs turned out to be less sensitive to the variations in the clustering results than the other cluster validity functions. The Beringer-Hüllermeier index performed equally well using the partitionings produced by the different clustering algorithms adapted to incomplete data. The cluster validity indexes using the membership degrees performed considerably better using the clustering results produced by the OCSFCM and the NPSFCM due to their tendency to strengthen the clustering structure. The two CVIs based on compactness and the PNC index obtained the best results using the clusterings produ-

ced by the OCSFCM because it does not tend to compact the clusters as the NPSFCM does. In contrast, the CVIs based on compactness and separation that use the distances between the cluster prototypes in their separation criterion performed poorer using the clusterings produced by the OCSFCM. The reason is that the OCSFCM tend to displace the cluster centers because the computation of missing values and the computation of cluster centers influence each other. Regarding the remaining three CVIs based on compactness and separation, the FS, the ZLE, and the BWS indexes, there were no odds-on favorite among the clustering strategies. While the CVIs performed better on the data sets with a complicated clustering structure using the clustering results produced by the NPSFCM, they favored the OCSFCM and the PDSFCM on the data sets with equally sized and shaped clusters.

The performance of the cluster validity functions also depended on the clustering structure of the incomplete data sets. Even if the most CVIs were able to reliably determine the correct number of clusters in the complete multi-dimensional data set, all CVIs failed on the same incomplete data set even for a small percentage of missing values. We also observed that with the increasing number of missing values in the data sets, the recognition rate of the correct number of clusters faster decreased on the data sets with differently sized and scattered clusters or where some clusters were closely located to each other building groups of clusters. As expected, the performance of the cluster validity functions did not improve using the partitioning results produced by the clustering algorithm FCMCD from chapter 3 on incomplete data sets with differently sized and scattered clusters. This is due to the fact that the clustering algorithm imputed missing values fitting to the partitionings with a particular number of clusters. Therefore, in the future, we focus on the development of a cluster validity index that is able to recognize the correct number of clusters in incomplete data sets with differently sized and scattered clusters.

Considering the overall performance of the cluster validity functions the idea of combining the compactness and separation in CVIs is very promising regarding their application on incomplete data. Therefore, in the future we plan to develop a cluster validity index that uses the same basic idea as the BWS index to be able to determine the correct number of clusters in data sets with a complicated clustering structure. To make our CVI resistant against the large number of missing values, we aim to substitute the computation of the cluster volumes using the covariance matrix through the variability of clusters as in the Beringer-Hüllermeier index. Another idea is to develop a multi-step approach similar to a decision tree for the recognition of the optimal number of clusters. This approach should use the features of the existing cluster validity functions to gain the information about a data set in single steps like whether there are groups of clusters or overlapping clusters in the data set. Approaching in this way, we hope to get more precise information about the structure of the data set using

step-by-step several relatively simple functions instead of bundling them together.

In our work, we focused on the analysis of the data sets with missing values MCAR. Generally, it is the simplest failure mechanism to deal with. Depending on the reason for the missingness of values in the data, there are data sets where the values are missing according to the MAR or the NMAR mechanisms. In [Him09, HC10a, HC10b] we analyzed the clustering algorithms adapted to incomplete data on the data sets with missing values MCAR, MAR and NMAR. In [HHC11] we made the first attempts to analyze some cluster validity functions on few relatively simple data sets with missing values MCAR, MAR and NMAR. All our experiments showed that the clustering results as well as the cluster validity results on the data with missing values MAR and NMAR were noticeably worse than on the data with missing values MCAR. Therefore, in the future, we plan to address the problem of developing a new clustering algorithm and CVIs adapted to data with missing values MAR and NMAR. The fuzzy clustering algorithm that uses a class specific probability presented in [TDK03] provides a good idea for our future research in this direction.

6

Density-Based Clustering using Fuzzy Proximity Relations

Discovering clusters of varying shapes, sizes and densities in a data set is still a challenging problem for density-based algorithms. Recently presented approaches either require the input parameters involving the information about the structure of the data set, or are restricted to two-dimensional data. In this chapter[1], we present a density-based clustering algorithm, which uses the fuzzy proximity relations between data objects for discovering differently dense clusters without any a-priori knowledge of a data set. In experiments, we show that our approach also correctly detects clusters closely located to each other and clusters with wide density variations.

6.1 Introduction

Clustering is one of the primary used data analysis methods, whose task is exploring the distribution of objects in a data set. In general, clustering is defined as a technique for completely partitioning a data set into groups (clusters) of similar data objects. But for some applications, for instance, in image processing, web log analysis and bioinformatics, detection of arbitrarily shaped dense groups of data objects is more useful than just partitioning the complete data set. As a result, the density-based clustering methods become more important.

The basic idea of density-based clustering is that clusters are regarded as dense regions of data points in the feature space separated by regions of lower density. Two established density-based clustering algorithms are DBSCAN [EHPJX96] and DEN-

[1]This chapter is a revised and updated version of [HC11].

CLUE [HK98], which use a global density-threshold for discovering clusters in a data set. While DBSCAN identifies clusters as sets of *density-connected* data points, where the minimal number of points $MinPts$ in the ε-environment defines this concept, DENCLUE uses standard deviation σ in the *influence function* of data points and the minimum density level ξ for distinguishing between clusters and noise. These methods detect clusters of different sizes and shapes, but they are not able to identify differently dense clusters. Using global density parameters, the denser clusters are treated as parts of less dense clusters and not as separate clusters, whereas the sparser clusters are handled as noise. This problem was promptly perceived so that several approaches have been proposed for discovering differently dense clusters in a data set [ABKS99, LS09, FSS$^+$09, ECL00, BG06]. While some of them extend DBSCAN adjusting density parameters for each cluster, other approaches analyze Delaunay Graph of a data set. The algorithms based on DBSCAN have the same weaknesses as DBSCAN: the clustering results strongly depend on the threshold parameters, which are tricky to determine. The most methods based on Delaunay Triangulation do not require any input parameters, but they are restricted to two-dimensional data sets. In this chapter, we present a new density-based algorithm DENCFURE (Density-Based Clustering using Fuzzy Proximity Relations) [HC11] for discovering differently dense clusters in a data set in presence of noise. In our approach, we use the fuzzy proximity relations between data objects to detect clusters as groups of approximately equidistant data points in the feature space. In experiments, we show that DENCFURE is able to detect differently dense clusters without any a-priori knowledge of a data set even if clusters are closely located to each other or if there are wide density variations within clusters.

The remainder of this chapter is organized as follows. We give a brief overview on recent density-based clustering approaches in Section 6.2. In Section 6.3, we present our notion of density-based clustering using fuzzy proximity relations and introduce the algorithm DENCFURE for discovering differently dense clusters in Section 6.4. The evaluation results of our method are presented in Section 6.5. Section 6.6 concludes this chapter with a short summary and discussion of future research.

6.2 Related Work

In the last decade, several density-based algorithms have been proposed for discovering differently dense clusters in a data set. In general, the most effort was done in two directions: extending DBSCAN algorithm and developing graph-based clustering algorithms analyzing minimal spanning trees or Delaunay Graphs. In this section we discuss these approaches.

In [ABKS99], a hierarchical density-based clustering algorithm OPTICS is presen-

ted. Based on DBSCAN this method creates an augmented ordering of density-based clustering structure of a data set using several density parameters. OPTICS is a hierarchical clustering approach, so it is able to identify differently dense clusters only if the clusters are clearly separated and are not contained in each other. Beside OPTICS, we mention here two more algorithms SSDBSCAN and Enhanced DBSCAN, which adjust density parameters for each cluster. The Semi-Supervised DBSCAN (SSDBSCAN) finds the neighborhood radius ε for each cluster determining the length of the largest edge of all density-connection paths between each pair of pre-labeled data points of different clusters [LS09]. The Enhanced DBSCAN determines the neighborhood radius for each cluster as the smallest distance to the $Maxpts$-nearest neighbor of all data points within the cluster [FSS+09]. Another approach LDBSCAN uses the concept of *local outlier factor* (LOF) [BKNS00]. This algorithm discovers clusters as sets of local-density-connected data points with similar *local reachability densities* (LRD) w.r.t. input parameters $LOFUB$, *pct* and $MinPts$ [DXG+07]. The weakness of all these approaches is that the clustering results strongly depend on the input parameters, which involve the information about the structure of the data set and are tricky to determine.

A graph-based clustering algorithm proposed in [Stu03] creates clustering by recursively breaking edges in the minimal spanning tree of a data set based on runt test [HM92]. Apart from heuristically determining the clusters' number, this method has the same weaknesses as OPTICS. Other graph-based clustering algorithms identify clusters as connected components in a Reduced Delaunay Graph [OBS92] of a data set. While the method proposed in [PP05] uses a threshold inputted by user for removing edges in Delaunay Graph, the algorithm AUTOCLUST performs clustering automatically without any a-priori knowledge of the data set [ECL00]. Another approach CRYSTAL is proposed in [BG06]. Instead of reducing the Delaunay Graph, CRYSTAL grows clusters starting from the point in the densest region using proximity information in the Delaunay Triangulation. These three clustering approaches based on Delaunay Triangulation are able to identify sparse and dense clusters closely located to each other as well as clusters with density variations. Since a stable implementation of Delaunay Triangulation in a three-dimensional space is still a challenging problem, these algorithms are restricted to two-dimensional data sets.

6.3 Density-Based Notion of Clusters using Fuzzy Proximity Relations

The basic idea of density-based clustering is that clusters are regarded as dense regions in the feature space separated by regions of lower density. The most approaches to

density-based clustering use the definition proposed by Ester et al. in [EHPJX96], who defined density-based clusters as sets of density-connected data objects w.r.t. density parameters: a radius $\varepsilon \in \mathbb{R}$ and a minimum number of data points $MinPts \in \mathbb{N}$ in ε-neighborhood (compare Figure 6.1(a)). If these density parameters are fixed as global density-threshold such as in DBSCAN [EHPJX96], then the algorithm separates a data set in equally dense clusters and noise (data items that do not belong to any cluster). To detect clusters of different densities in a data set, one or both of these density parameters are adjusted for each cluster [LS09, FSS+09, DXG+07] (compare Figure 6.1 (b)). But the key idea remains the same: for each core data point within a cluster the neighborhood of a given radius has to contain at least a minimum number of data points.

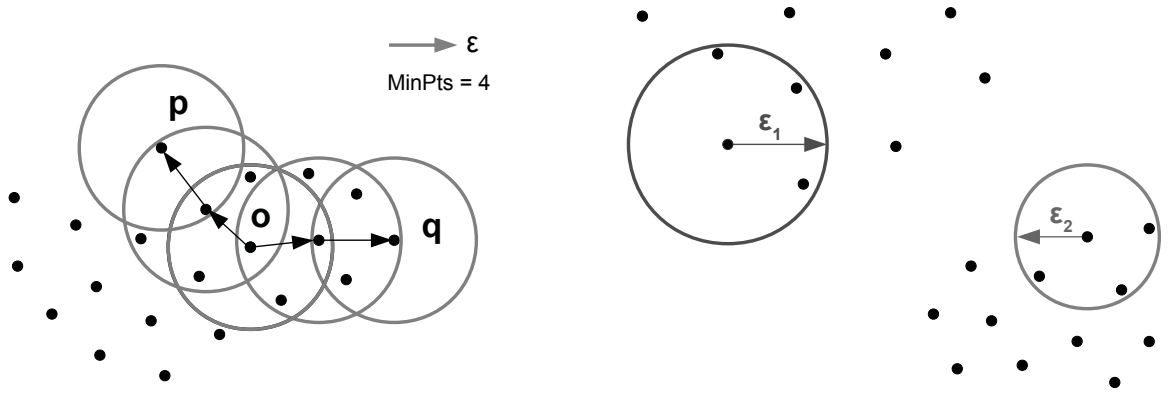


Figure 6.1: (a) p and q are density-connected to each other by o (adapted from [EHPJX96]), (b) two differently based clusters with different ε-radii.

We depart from this definition of density and use the concept of fuzzy proximity relations for the definition of density-connectivity. We proceed on the assumption that human perception of clusters depends rather on the relation of distances between neighboring data points than on the number of data points in a given neighborhood. The basic idea of our approach is that the distances between all neighboring data points within a cluster are approximately equal.

Below we formalize our notion of density-based clustering. We assume that $X = \{x_1, ..., x_n\}$ is a given data set of n metric objects in d-dimensional feature space, and *dist* is a metric distance function on the data objects in X. For each data object $x_i \in X$ we consider $NN_k(x_i)$, the set of its k-nearest neighbors w.r.t. the distance function *dist*. Within the set of the k-nearest neighbors of x_i we detect those data objects $x_j \in X$, which are approximately equidistant to x_i, and refer to them as *directly reachable fuzzy neighbors* of x_i. We determine the equidistances between data objects using a fuzzy membership function u that is also used in the basic fuzzy c-means algorithm (FCM) [Bez81]. For our purpose we adapted the definition of u with $u(x_i, x_j) \in [0, 1]$ for all

$x_i, x_j \in X$ with $1 \leq i, j \leq n$ as follows

$$u(x_i, x_j) = \begin{cases} 1 & \text{if } i = j, \\ \dfrac{(dist_A^2(x_i, x_j))^{\frac{1}{1-m}}}{\sum_{l=1}^{k} (dist_A^2(x_i, x_l))^{\frac{1}{1-m}}} & \text{if } I_{x_i} = \emptyset, \\ \dfrac{1}{\mid I_{x_i} \mid} & \text{if } I_{x_i} \neq \emptyset, x_j \in I_{x_i} \\ 0 & \text{if } I_{x_i} \neq \emptyset, x_j \notin I_{x_i} \end{cases} \tag{6.1}$$

where $I_{x_i} = \{x_j \mid dist_A(x_i, x_j) = 0, i \neq j\}$ and $m > 1$ is the fuzzification parameter. As each application requires an appropriate distance function, we do not restrict our approach to a specific distance function, so it works with any vector A-norm distance function $dist_A(x_i, x_j) = \|x_i - x_j\|_A = \sqrt{(x_i - x_j)^T A (x_i - x_j)}$. Note, that matrix A has to be symmetric positive definite. When $A = I_{d \times d}$, then $dist_A(x_i, x_j) = dist_2(x_i, x_j) = \|x_i - x_j\|_2$ is the Euclidean distance. As required in [Bez81] our adapted fuzzy membership function u satisfies $\sum_{j=1}^{k} u(x_i, x_j) = 1$ for all $i \in \{1, ..., n\}$, $i \neq j$.

Definition 1. *(directly reachable fuzzy neighbor) A data point x_j is a* directly reachable fuzzy neighbor *of a data point x_i if*

1. $x_j \in NN_k(x_i)$ *and*
2. $u(x_i, x_j) \geq \frac{1}{k}$.

For each data point within a cluster its directly reachable fuzzy neighbors are its adjacent data points, i.e. data points in its nearest neighborhood. Obviously, the data points inside a cluster have more directly reachable fuzzy neighbors than data points on the border of a cluster or noise points. For this reason we limit the number of k nearest neighbors for each data point to a maximum possible number of directly reachable fuzzy neighbors for data points inside a perfectly homogeneously dense cluster. This means that for each data point x_i inside a cluster all its directly reachable fuzzy neighbors x_j are equidistant to x_i. Thus, we obtain $u(x_i, x_j) = \frac{1}{k}$ for all $x_j \in NN_k(x_i)$. All directly reachable fuzzy neighbors of a data point x_i on the border of a cluster or of a noise point are data points from the set $NN_k(x_i)$, which are approximately equidistant to x_i. We do not require an exact equidistance between data points because the distances between neighboring data points within clusters are extremely rarely equal in real data sets. The value for the fuzzification parameter m influences the density variance within a cluster. The clusters are perfectly homogeneously dense for values of m near 1 and less uniformly dense for large values of m.

The number k of nearest neighbors of a data point obviously depends on the dimensionality of the data set. For example, a one-dimensional data point can have at most

two different equidistant data points, a two-dimensional data point can have at most six such neighboring data points. In this way we limit the value for k as the maximal number of points at the surface of a hypersphere in d-dimensional space, so that the distances between neighboring points at the surface (i.e. the lengths of the chords) are equal to the radius of the sphere. The value for k is equivalent to the kissing number τ_d in the sphere packing theory [CSB87]. For d-dimensional Euclidean space the kissing numbers are $\tau_1 = 2$, $\tau_2 = 6$, $\tau_3 = 12$, $\tau_4 = 24$ etc. Note, although the data point x_i is a directly reachable fuzzy neighbor of itself by definition, we regard it rather as a special case and do not count it among its k neighbors. In the case the data point x_i is counted as its directly reachable fuzzy neighbor, the number k has to be adjusted and the second condition in Definition 1 would be: $u(x_i, x_j) \geq \frac{1}{k-1}$.

Obviously, the directly reachable fuzzy neighborhood property is not symmetric for each pair of data points. Each data point in a data set has at least one directly reachable fuzzy neighbor, even noise points. But, as mentioned above, the direct neighboring data points within a cluster have to be approximately equidistant. Therefore, the directly reachable fuzzy neighborhood property is required to be symmetric for pairs of data points within a cluster.

Definition 2. *(direct fuzzy density-neighbor) A data point x_j is a* direct fuzzy density-neighbor *of a data point x_i if*

1. *x_j is a directly reachable fuzzy neighbor of x_i, and*

2. *x_i is a directly reachable fuzzy neighbor of x_j.*

Evidently, the direct fuzzy density-neighborhood property is symmetric and does not depend on the distances between data points themselves. This means, if two points are the direct fuzzy density-neighbors of each other, then the distances to their directly reachable fuzzy neighbors are approximately equal. And exactly this property has to be satisfied by the data points within a cluster.

Definition 3. *(fuzzy density-connected) A data point x_j is* fuzzy density-connected *to a data point x_i if there is a chain of data points $x_{i_1}, ..., x_{i_k}$ $(1 \leq i_k \leq n)$, $x_{i_1} = x_i$, $x_{i_k} = x_j$ so that $x_{i_{m+1}}$ is a direct fuzzy density-neighbor of x_{i_m}.*

Fuzzy density-connectivity is a fuzzy proximity relation because it is reflexive and symmetric [Ped12]. Additionally, it is an equivalence relation because it is also transitive. Therefore, we are able to formalize the density-based notion of a cluster and noise. Analogous to [EHPJX96], we define a cluster as a maximal set of fuzzy density-connected data points and noise as the set of data points in X, which do not belong to any cluster.

Definition 4. *(density-based cluster) Let X be a data set of n data points. A* density-based cluster C, $|C| > 1$, *is a non-empty subset of X, satisfying:*

1. $\forall x_i, x_j \in C$: *x_i is fuzzy density-connected to x_j. (Connectivity)*
2. $\forall x_i, x_j \in X$: *if $x_i \in C$ and x_j is fuzzy density-connected to x_i, then $x_j \in C$. (Maximality)*

Definition 5. *(noise) Let $C_1, ..., C_l$ be all clusters of the data set X. The* noise N *is the subset of data points in X not belonging to any cluster C_i, $1 \leq i \leq l$, i.e. $N = \{x \in X \mid \forall i : x \notin C_i\}$.*

According to our notion of density-based clustering, a cluster contains at least two data points. This requirement has to be fulfilled because due to the reflexivity each data item is fuzzy density-connected to itself by definition. Consequently, each data item even noise points that do not have further direct fuzzy density-neighbors apart from themselves would build a cluster. Since this property is undesirable, we require the condition $|C| > 1$ to be fulfilled. Furthermore, our definition of density-based cluster implies that equal data points build their own clusters because all data points equal to a data point x_i (apart from the data point itself) have the same membership degrees to the data point x_i. In the case when the geometrical structure of the data set is of particular interest, it is advisable to remove all duplicates from the data set before clustering. A further property of our notion of density-based clustering is that all data points of a data set X can be completely assigned into clusters. This means that the noise N can be an empty set, $(N = \emptyset)$. But on the other hand, the set noise is a proper subset of any data set X, $(N \subsetneqq X)$. That is because our definition of density-based clustering implicates that any data set X contains at least one cluster, which contains at least two nearest data points in X. In particular, this property means that a data set containing approximately equidistant data points will be regarded as single cluster and not as noise.

6.4 Density-Based Clustering Algorithm using Fuzzy Proximity Relations

In this section, we describe the algorithm DENCFURE (**DEN**sity-Based **C**lustering using **FU**zzy Proximity **RE**lations). DENCFURE is designed to detect varyingly dense clusters and the noise in a given data set according to Definitions 4 and 5. Unlike other density-based clustering algorithms, one of the important properties of DENCFURE is that it does not need any input parameters, which involve information about the structure of the data set. On the contrary, our algorithm not only detects density-based

clusters, but also outputs the information about the average density of the discovered clusters, which can be useful for some applications. Furthermore, DENCFURE is designed to separate sparse and dense clusters, which are closely located to each other. This situation is depicted in Figure 6.2(a), where there is no clear separation between two differently dense clusters. Three border points of a sparse cluster are closer located to the dense cluster than to theirs. This is a challenging problem for density-based algorithms and almost all of them fail to work because the distances between some border points of both clusters are smaller than the average distances between neighboring points in the sparse cluster. In such cases, density-based algorithms based on DBSCAN are not able to separate such clusters correctly. They either assign border data points of the sparse cluster to the dense cluster or merge both clusters into one.

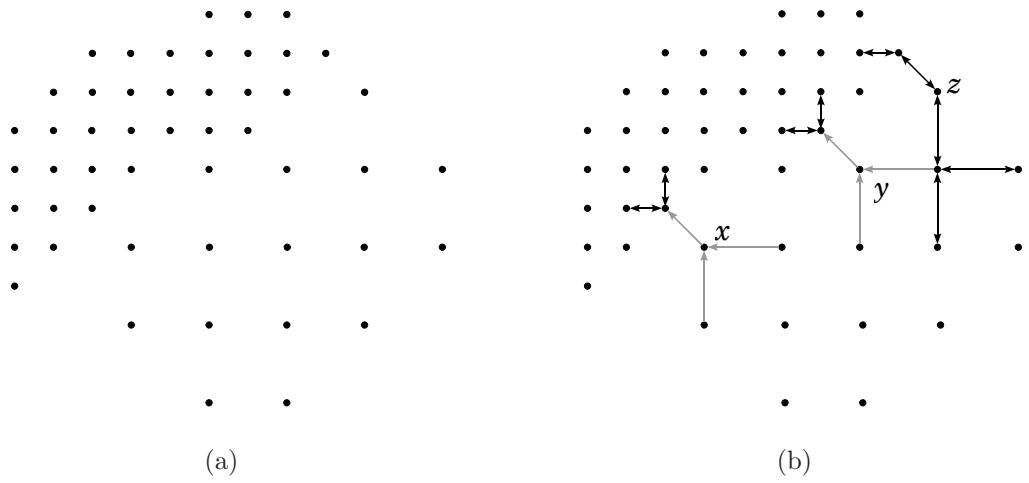


Figure 6.2: Two differently dense clusters closely located to each other.

6.4.1 The Algorithm

The basic idea of the DENCFURE algorithm is to construct a mixed graph with n vertices. The vertices x_i, $1 \leq i \leq n$ represent the data points of a data set X, and there is a directed edge from vertex x_i to vertex x_j if data point x_j is a directly reachable fuzzy neighbor of x_i, and there is an undirected edge between vertices x_i and x_j if x_j and x_i are the direct fuzzy density-neighbors of each other. Then the connected components of the undirected part of the graph represent clusters, and the vertices, which do not belong to any connected component, represent the noise. We use the directed part of the graph to differentiate between noise points and border points of clusters, which are not clearly separated from each other as depicted in Figure 6.2.

The basic version of the DENCFURE algorithm is represented in Algorithm 3. Note, that we focus here predominantly on the presentation of our idea omitting details of efficient data structures and techniques. We refer to Section 6.4.2 for the discussion

Algorithm 3 DENCFURE(X, m, A)

Require: X is a d-dimensional data set with n data points, $m > 1$ is a fuzzification parameter, A is a matrix for the vector distance function

```
1: for all x_i ∈ X calculate DRFN(x_i) and avg_dist_A(x_i);
   // Detect dense and sparse clusters closely located to each other and separate them
2: for all x_i ∈ X do
3:     for all x_j ∈ DFDN(x_i)\x_i do
4:         if (x_i.avg_dist_A < x_j.avg_dist_A && x_i.avg_dist_A < dist_A(x_i, x_j)) then
5:             if (SepClust(x_i, x_j, DFDN, avg_dist_A)) then
6:                 Recalculate x_i.avg_dist_A;
7:             end if
8:         end if
9:     end for
10: end for
   // Detect clusters as sets of fuzzy density-connected points
11: for all x_i ∈ X set x_i.ClusterID = UNCLASSIFIED;
12: ClusterID := nextID(UNCLASSIFIED);
13: for i = 1 to n do
14:     if (x_i.ClusterID = UNCLASSIFIED) then
15:         if (ExpandCluster(x_i, DFDN, ClusterID)) then
16:             ClusterID := nextID(ClusterID);
17:         end if
18:     end if
19: end for
   // Validate noise
20: for all x_i with x_i.ClusterID = UNCLASSIFIED do
21:     if (isNoise(x_i, DRFN, avg_dist_A) = false) then
22:         x_i.ClusterID := NOISE;
23:     end if
24: end for
```

of efficiency improving techniques and data structures. The input parameters required by DENCFURE are a data set X, a fuzzification parameter m and a matrix A for the vector distance function. Note, that the parameters m and A depend rather on a given application than on the distribution of data points in a data set X. If the goal is finding clusters with equidistant data points, then the fuzzification parameter m should be chosen close to 1. If there are no restrictions regarding the density variations within clusters, the fuzzification parameter should be chosen sufficiently large.

At the beginning of the algorithm, for all $x_i \in X$ we calculate the directly reachable fuzzy neighbors $DRFN(x_i)$. Obviously, we can calculate the set of direct fuzzy density-neighbors $DFDN$ from the set $DRFN$. So the inclusion $DFDN \subseteq DRFN$ holds. Additionally, according to Formula (6.2) we calculate $avg\_dist_A(x_i)$ for all $x_i \in X$ with $DFDN(x_i) \backslash x_i \neq \emptyset$ as the average distance from x_i to all its direct fuzzy density-

Algorithm 4 boolean SEPCLUST($x_i, x_j, DFDN, avg\_dist_A$)

1: Calculate $x_i.avg\_dist_A\_diff$ and $x_j.avg\_dist_A\_diff$ according to Formula (6.3);
2: **if** $(|x_j.avg\_dist_A - x_i.avg\_dist_A| > (avg\_dist_A\_diff\_x_i \;\&\&\; avg\_dist_A\_diff\_x_j))$ **then**
3: Remove x_j from $DRFN(x_i)$;
4: Remove x_i from $DRFN(x_j)$;
5: **end if**
6: **return true**;

neighbors apart from the data point itself.

$$avg\_dist_A(x_i) = \frac{\sum\limits_{x_k \in DFDN(x_i)\backslash x_i} dist_A(x_i, x_k)}{|DFDN(x_i)\backslash x_i|}. \tag{6.2}$$

We obtain $DFDN(x_i)\backslash x_i = \emptyset$ for potential noise points that do not have any direct fuzzy density-neighbors apart from themselves and do not calculate avg_dist_A for those.

In the next step of the algorithm, we use the average distances avg_dist_A of data points to detect and break bridges between dense and sparse clusters closely located to each other as depicted in Figure 6.2(a). Here, bridges are termed as direct fuzzy density-neighborhood relations between two border points of different clusters. In Figure 6.2(b), there is a bridge between point z and its direct fuzzy density-neighbor from the dense cluster (the direct fuzzy density-neighborhood relations are illustrated with black arrows). The first indication of such a bridge between two data points x_i and x_j, $i \neq j$, is given if the average distance $avg\_dist_A(x_i)$ of x_i is less than the average distance $avg\_dist_A(x_j)$ of its direct fuzzy density-neighbor x_j, and the distance $dist_A(x_i, x_j)$ between x_i and x_j is greater than $avg\_dist_A(x_i)$. The second condition is tested using the function `SepClust` depicted in Algorithm 4. In this function we go further into the question whether x_i and x_j both belong to an unequally dense cluster or x_i and x_j are border data points of two different varyingly dense clusters. We assume that there is a bridge between data points x_i and x_j, $i \neq j$, if the difference $|avg\_dist_A(x_j) - avg\_dist_A(x_i)|$ between average distances of x_i and x_j is greater than the average differences $avg\_dist_A\_diff(x_i)$ and $avg\_dist_A\_diff(x_j)$ between $avg\_dist_A(x_i)$ (resp. $avg\_dist_A(x_j)$) and the average distances $avg\_dist_A(x_k)$ of all direct fuzzy density-neighbors x_k of x_i, $x_k \in DFDN(x_i)\backslash x_i$ with $x_k \neq x_j$ (resp. $avg\_dist_A(x_l)$ of all $x_l \in DFDN(x_j)\backslash x_j$ with $x_l \neq x_i$). We calculate $avg\_dist_A\_diff(x_i)$ as follows:

$$avg\_dist_A\_diff(x_i) = \frac{\sum\limits_{x_k \in DFDN(x_i)\backslash(x_j \cup x_i)} |avg\_dist_A(x_i) - avg\_dist_A(x_k)|}{|DFDN(x_i)\backslash(x_j \cup x_i)|}. \tag{6.3}$$

If $DFDN(x_i)\backslash(x_j \cup x_i) = \emptyset$, we define $avg\_dist_A\_diff(x_i)$ to be 0. This is primary true

for border points of a cluster or for noise points. If x_i is a border point of a cluster, then the distance between x_i and x_j should not be much larger than the average distance of x_j and its direct fuzzy density neighbours. Otherwise x_i is the noise point not far from a cluster. The calculation of $avg\_dist_A\_diff(x_j)$ is analogous to calculation of $avg\_dist_A\_diff(x_i)$. If both conditions for the presence of a bridge are complied, then we break the bridge removing x_j from the set $DRFN(x_i)$ and x_i from the set $DRFN(x_j)$, respectively.

Algorithm 5 boolean ExpandCluster($x_i, DFDN, ClId$)

```
1: seeds := ∅;
2: for all x_j ∈ DFDN(x_i) do
3:     seeds.add(x_j);
4: end for
5: if (seeds ≠ ∅) then
6:     x_i.ClusterID := ClId;
7:     while (seeds ≠ ∅) do
8:         currentDP := seeds.firstElement();
9:         currentDP.ClusterID := x_i.ClusterID;
10:        seeds.remove(currentDP);
11:        for all x_j ∈ DFDN(currentDP) do
12:            if (x_j.ClusterID = UNCLASSIFIED && seeds.contains(x_j) = false)
               then
13:                seeds.add(x_j);
14:            end if
15:        end for
16:    end while
17: end if
18: return true;
```

After breaking all bridges between closely located clusters, the algorithm assigns data points into clusters using function `ExpandCluster`, which is depicted in Algorithm 5. The algorithm starts with an arbitrary unclassified data point x_i and recursively detects all data points, which are fuzzy density-connected to x_i. This part of DENCFURE is based on a simple algorithm for computing connected components in an undirected graph. First it adds all direct fuzzy density neighbors of x_i to a seed set. The algorithm works the seed set off point by point by labeling them with the cluster ID of x_i. Then it removes the assigned points from the seed set and adds all their unclassified direct fuzzy density neighbors to the seed set. Since some data points within a cluster can share the same direct fuzzy density neighbors, first we check if they are already in the seed set. The algorithm proceeds until the seed set is empty. In this case the complete cluster is found and the function `ExpandCluster` returns to the main algorithm.

After all clusters are detected, there may be still some unclassified data points.

Algorithm 6 boolean isNoise($x_i, DRFN, avg\_dist_A$)

1: **for all** x_j **with** $x_i \in DRFN(x_j)$, $i \neq j$, $DFDN(x_j) \backslash x_j \neq \emptyset$ **and** $(dist_A(x_i, x_j) \leq x_j.avg\_dist_A)$ **do**
2: Find x_j **with** $\max_j \left(\frac{dist_A(x_i, x_j)}{x_j.avg\_dist_A} \right)$;
3: $DRFN(x_j).add(x_i)$;
4: $DRFN(x_i).add(x_j)$;
5: $x_i.avg\_dist_A := dist_A(x_i, x_j)$;
6: $x_i.ClusterID := x_j.ClusterID$;
7: Recalculate $x_j.avg\_dist_A$;
8: **end for**

These data points can be noise points or the endpoints of broken bridges, which do not have any direct fuzzy density-neighborhood relation to their real cluster. But these points can also be border points of a sparse cluster closely located to a dense one so that from the beginning there was no direct fuzzy density-neighborhood relation to neither dense nor sparse cluster. In Figure 6.2(b) we have this situation for points x and y, which do not have any direct fuzzy density-neighbors (grey arrows represent the directly reachable fuzzy neighborhood relations). Since they have more directly reachable fuzzy neighbors in the dense cluster, they do not recognize their neighboring points from the sparse cluster as their directly reachable fuzzy neighbors. In the last step of our algorithm, we detect such undetected border points of clusters using function `isNoise` depicted in Algorithm 6. We assign these data points to clusters testing a criterion of average distance homogeneity. This basically means that if x_i can be assigned to more than one cluster, it will be assigned to the cluster where the distance between x_i and x_j least differs from the average distance of x_j. All other unclassified data points, which could not be assigned to any cluster, are assigned to noise. Concluding, the average density of each discovered cluster can be calculated as the average over average distances of all points containing in the cluster.

6.4.2 Complexity and Efficiency Optimization

The main part of the DENCFURE algorithm for detecting clusters is based on the *depth-first search (DFS)* algorithm. Since we store the direct fuzzy density neighbors of all data points in a data structure, the algorithm visits each data point only one time. So the time complexity is linear for this part of the algorithm. The noise validation takes $O(nk)$ time because the algorithm checks all data points for which an unclassified data point is the direct reachable fuzzy neighbor. However, the runtime of the algorithm is dominated by the construction and adjustment of the graph. The separation of dense and sparse clusters closely located to each other takes $O(nk^2)$ time in the worst case. If all clusters are clearly separated, then this part of the algorithm takes only $O(nk)$ time.

The runtime intensive part is finding the directly reachable fuzzy neighbors for all data points. Without using speed-up indexing structures the distances between all pairs of data points have to be calculated. That results in the runtime complexity $O(n^2)$. Assuming $n \gg k^2$, this part determines the runtime complexity of the entire algorithm. Since the number of directly reachable fuzzy neighbors for each data point is bounded by the parameter k, it does not make sense to calculate the distances between all pairs of data points. In the case that the data set is organized in a spatial tree structure like R-tree [Gut84], only distances to data objects within bounding rectangles or neighboring rectangles have to be calculated. Depending on the organization of the data in the spatial tree, the run time of this part of the algorithm can be reduced to $O(n \log n)$.

6.5 Experimental Results

In this section we evaluate the performance of DENCFURE in terms of accuracy. First, in Figure 6.3 we show the influence of the fuzzification parameter m on the clustering results. Figure 6.3 (a) shows the original data set containing a circular uniformly dense cluster in the middle surrounded by an unequally dense donut-shaped cluster of lower density. Figure 6.3 (b) shows clustering results produced by DENCFURE for $m = 1.1$. As expected, the algorithm separated clusters of perfectly homogeneous density dividing the donut-shaped cluster in many small clusters. In contrast, for $m \geq 2.5$ DENCFURE assigned data points into clusters allowing some degree of tolerance in density variation within clusters (compare Figure 6.3 (c)). In this case, the algorithm assigned data points in six clusters merging small clusters into a donut-shaped one. DENCFURE separated four small clusters on the edge of the donut-shaped cluster. This is due to the fact that the distance between the sets of points of small marginal clusters and the big donut-shaped cluster is greater than the average distances between neighboring data points in both clusters. Therefore, the algorithm did not merge these five clusters into one.

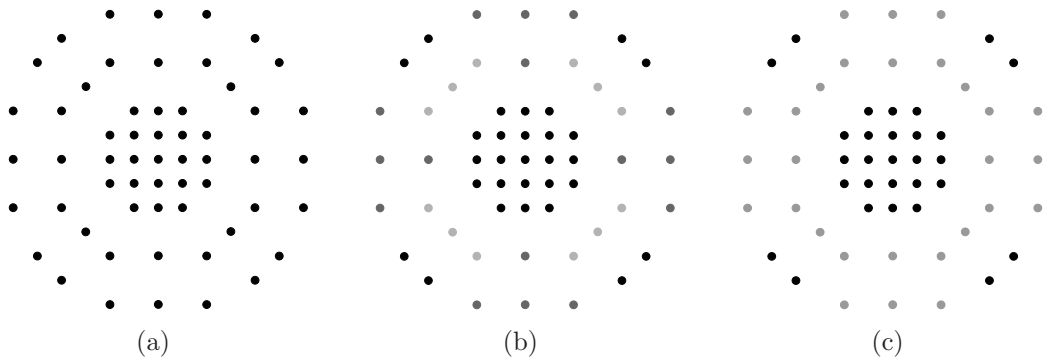


Figure 6.3: Influence of the fuzzification parameter m on the clustering results.

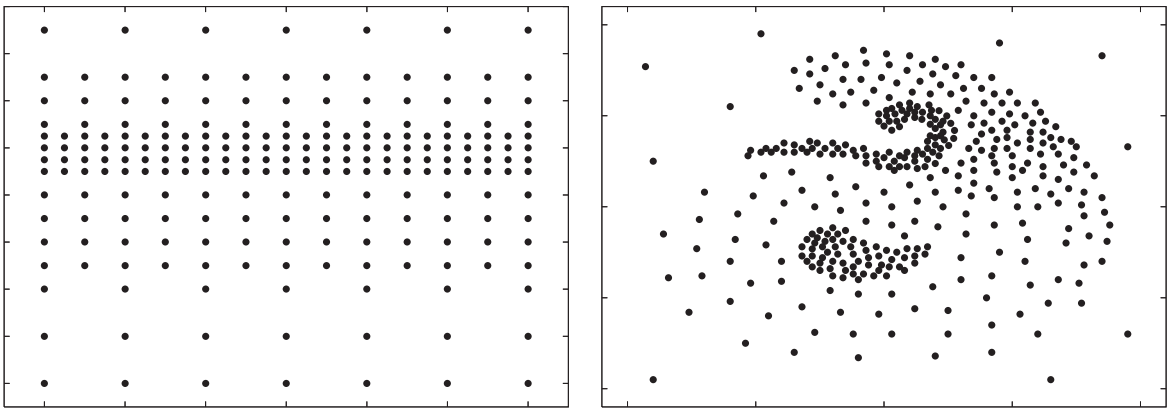
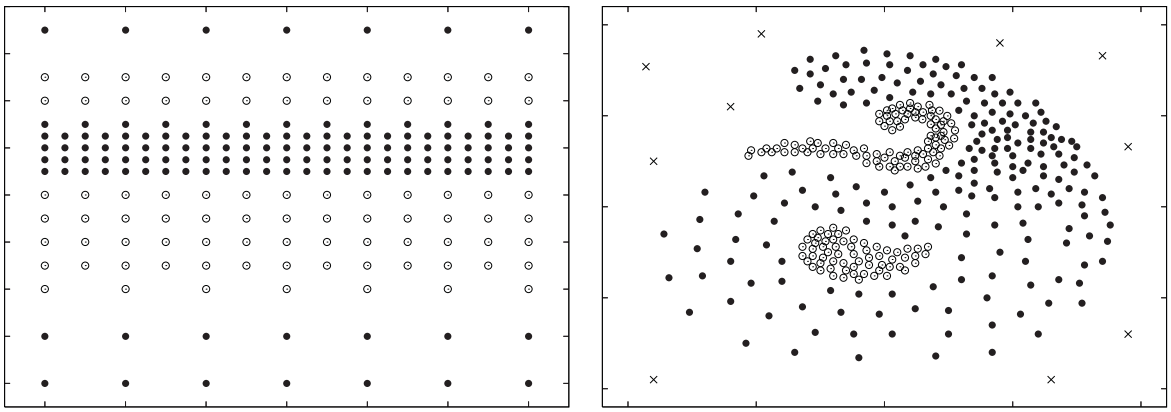


Figure 6.4: Original sample data sets.

Figure 6.5 shows the clustering results produced by DENCFURE with $m = 5$ on two selected sample data sets (the original data sets are shown in Figure 6.4). The first data set contains five differently dense stripe-like clusters with no clear separation between them. The challenge in this data set is that some points between sparse and dense clusters have equal distances to the nearest neighbors of both clusters as the nearest neighbor distances within clusters. The algorithm assigned data points into clusters in the same way as a human would do it by visual assessment (compare Figure 6.5(a)). The contentious points were assigned to the dense clusters. The second data set is depicted in Figure 6.4 (b). There are three clusters of different shapes, sizes and densities with additional noise. In addition to no clear separation between clusters, there are great density variations within a big cluster. This is a challenging problem for density-based algorithms based on DBSCAN, because it is tricky to determine the correct ε for each cluster in the *sorted k-dist graph* in order to separate clusters correctly on the one hand and to discover complete clusters on the other hand. Using the fuzzy proximity relations between data points, DENCFURE correctly discovered all clusters and detected the noise points (represented with crosses) as showed in Figure 6.5(b).

Furthermore, the results of our experiments showed that the value for the fuzzification parameter m, which is an input parameter, does not have to be determined for every data set such as $MinPts$ and ε. In both data sets, which contain differently structured clusters, DENCFURE correctly detected all clusters with the same value for m. Moreover, we obtained the same clustering results for all $m \geq 5$. This means that a greater value for the fuzzification parameter allows the greater degree of tolerance in density variations within clusters, but the clusters are not merged into one for $m \to \infty$.

Figure 6.6 shows the clustering results produced by DENCFURE with $m = 30$ on the *bensaid* data set [BHB+96]. This experiment shows the limitations of our algorithm. While the algorithm detected two small clusters, it divided the large sparse cluster into



(a) (b)

Figure 6.5: Clustering results produced by DENCFURE with $m = 5$ on two sample data sets.

several small clusters and two outliers. This is due to the fact that the distances between small groups of data points are larger than the distances between data points within groups. Although the aim of the DENCFURE algorithm is detecting clusters of approximately equidistant data points, we perceive this issue as a limitation of our algorithm because the density variations within the large cluster are tolerable from human viewpoint. At this point it would have been desirable if the large cluster was detected as a whole one.

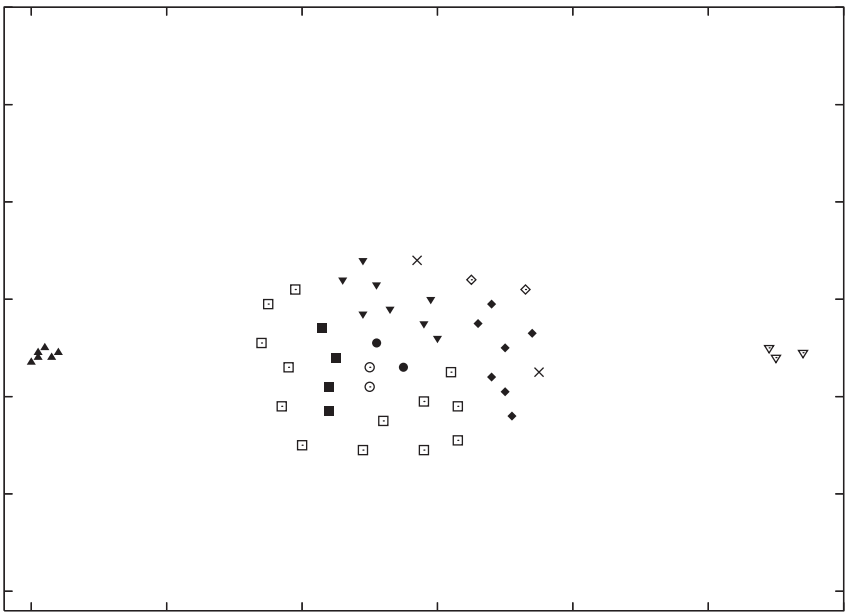


Figure 6.6: Clustering results produced by DENCFURE with $m = 30$ on *bensaid* data set.

6.6 Conclusions and Future Work

Discovering clusters of varying shapes and densities in a data set is still a challenging problem for density-based algorithms. In this chapter, we presented a density-based clustering algorithm DENCFURE using fuzzy proximity relations for detecting differently dense clusters in presence of noise in a data set. The experimental results showed that DENCFURE is able to detect differently dense clusters without any a-priori knowledge of the data set even if clusters are closely located to each other or if there are wide density variations within clusters. The algorithm is designed to separate differently dense clusters independent of their sizes, i.e. DENCFURE also creates clusters of two data points if the distance between them is significantly less than the distances to other points. It makes sense for applications detecting fine structures in a data set. But for other applications clusters matter if they are of certain size even if the distances between some groups of data points are larger than the distances between data points within groups. Therefore, in the future we plan to extend DENCFURE to create a hierarchical ordering for large clusters containing small dense clusters and to ignore small clusters if they are not contained in any other cluster. Furthermore, we plan to analyze and to extend our algorithm working with other distance functions e.g. *p-norm distance* and other fuzzy membership functions. The challenge we see here is that the number of nearest neighbors of a data point depends on the distance function. Moreover, we aim to parallelize our algorithm and to apply known optimization techniques to make DENCFURE applicable to very large data sets.

Another important objective is adapting the DENCFURE algorithm to incomplete data. This is a challenging problem for all algorithms based on distances between data items. In chapter 3 we mentioned the algorithm *VAT* for estimating the number of clusters for partitioning cluster algorithms. The authors proposed estimating the distances between incomplete data points by partial distances [BH02]. As we already mentioned, this strategy does not work when data items do not have values in the same dimensions. To avoid this problem, in our future work we plan to adapt the DENCFURE algorithm to incomplete data introducing a pre-clustering step. The idea is to cluster incomplete data with a partitioning fuzzy clustering algorithm like PDSFCM or OCSFCM and substituting missing values by the corresponding values of their cluster prototypes. The substitution can be a total one or the missing values can be estimated depending on all cluster prototypes taking the membership degrees into account. We expect that estimating missing values in this way will not bias the results of the DENCFURE algorithm much because it does not work with the distances between data points directly to determine the direct fuzzy density-neighborship relations. Much more it uses the relations between distances of neighboring data points for discovering clusters.

7

Conclusion and Future Work

This chapter presents the conclusion of the thesis. We summarize the contributions of this thesis and describe the solutions of addressed problems in Section 7.1. Finally, we discuss some future research directions in Section 7.2.

7.1 Summary

Data analysis methods have gained increasing importance with the development of possibilities for collecting and storing large amounts of data. The volumes of data can contain new useful knowledge that first has to be extracted within the KDD process. Clustering represents an important and one of the primary used techniques for the automatic knowledge extraction from large amounts of data. Its task is to partition a data set into groups of similar data objects. Traditional clustering methods were developed to analyze complete data sets. However, faults during the data collection, data transfer or data cleaning often lead to missing values in data. Since the elimination of incomplete data items or features, or the imputation of missing values may affect the quality of the clustering results, clustering methods that can deal with incomplete data are of great use.

The fuzzy clustering methods for incomplete data proposed in the literature perform well as long as the clusters are similarly scattered. Therefore, in this thesis we proposed an enhanced fuzzy clustering approach for incomplete data. Our method uses a new membership degree for the missing value estimation that takes the cluster dispersions into account. In the original clustering algorithm for incomplete data, the nearest cluster centers had more influence on the imputation of missing values. In our approach, not only the distances between the incomplete data items but also the cluster scatters

are considered during the imputation of missing values. In this way the clusters with large dispersions are fully taken into account during the imputation of missing values. As a result, our approach aims to maintain the clustering structure while the basic method often distorts the original clustering structure splitting the clusters with a low number of incomplete data items in several clusters and distributing the data items of clusters with a high number of incomplete data items to other clusters. Moreover, the experimental results on the artificial and the real data sets with differently scattered clusters have shown that our approach produced less misclassification errors, it was more stable, it required less iterations to termination, and it produced more accurate terminal cluster prototypes as long as the percentage of missing values in the data set was not greater than 40%.

In the second part of the thesis we addressed the problem of finding the optimal number of clusters on the incomplete data using the cluster validity functions. We described and adapted different cluster validity functions to incomplete data according to the available case approach. We evaluated them using the partitioning results of several incomplete artificial and real data sets produced by different fuzzy clustering algorithms for incomplete data. We found out that the determination of the correct number of clusters using the cluster validity indexes adapted to incomplete data mainly depends on the clustering results produced by the clustering algorithms. Even though, there were large differences in the performance of the CVIs on incomplete data. While the CVIs based on compactness totally failed on the data sets with a large percentage of missing values, the CVIs using the membership degrees and the CVIs based on compactness and separation that use the distances between the cluster prototypes in their separation criteria, hardly lost performance with the increasing number of missing values in the data sets. However, the latter CVIs recognized the correct number of clusters only in the data sets with equally sized and shaped clusters, while the CVIs based on compactness were able to recognize the correct number of clusters in the data sets with a complicated clustering structure. Furthermore, we found out that the clustering structure of the incomplete data sets is a crucial factor for finding the optimal number of clusters. While the most CVIs reliably determined the correct number of clusters in the complete multi-dimensional data set, all of them failed on the same incomplete data set even with a small percentage of missing values. We also observed that the recognition rate of the correct number of clusters faster decreased on the data sets with a complicated clustering structure with the increasing number of missing values in the data sets.

Since finding clusters of varying shapes, sizes and densities in a data set is more useful for some applications than just partitioning a data set, in the last part of the thesis, we presented a new density-based algorithm DENCFURE for discovering differently dense clusters in a data set in presence of noise. Our algorithm uses the fuzzy

proximity relations between the data items to detect clusters as groups of approximately equidistant data points in the feature space. The relative equidistance between the data items is expressed using a fuzzy membership function which determines the nearest neighbours of a given data item. In the main, the clusters are defined then as groups of data items that are nearest neighbours of each other. In this way, our clustering algorithm is able to detect differently dense clusters of varying shapes and sizes without any input parameters that involve the information about the structure of the data set. The fuzzy membership function alone specifies the degree of the allowed density variations within the clusters. In experiments on different data sets, we show that our approach is able to correctly detect the clusters closely located to each other and clusters with wide density variations.

7.2 Future Work

In this thesis we proposed an enhanced fuzzy clustering approach for incomplete data that takes the cluster scatters into account during the estimation of missing values. Although the data experiments have shown promising results for our method, the imputation of missing values and the computation of cluster prototypes influence each other. As a result, our clustering algorithm produces less accurate cluster prototypes for a large percentage of missing values in the data set. Therefore, in the future we plan to avoid the influence of the imputed values on the computation of cluster prototypes. Our idea is to completely exclude the missing values from the computation of cluster prototypes calculating them only on the basis of the available feature values. Furthermore, we plan to adapt our method on incomplete data sets with a conditional missingness of values.

Another direction of future research is the development of a multi-step approach similar to a decision tree for determining the optimal number of clusters on incomplete data sets. Our idea is to use parts of the existing cluster validity functions that examine the structure of the data set from different perspectives in single steps. Before determining the optimal number of clusters we aim to ascertain whether there are groups of clusters or overlapping clusters in the data set. In this way, we hope to get more precise information about the structure of the data set using several relatively simple functions step-by-step instead of bundling them together.

In the last part of the thesis, we presented a new density-based algorithm DENCFURE for discovering differently dense clusters in a data set in presence of noise. The experimental results on the *bensaid* data set showed the limitations of our algorithm. Although the data set contains three clusters according to the visual assessment, DENCFURE divided the large cluster into several small ones. The reason is the fuzzy

membership function that determines the allowed density variations within the clusters. In this algorithm we used the fuzzy membership function from the FCM algorithm. Therefore, in the future we plan to analyze and to extend our algorithm working with other fuzzy membership functions and other distance functions. Another important direction of future research is adapting the DENCURE algorithm to incomplete data. This is a challenging problem for all algorithms based on the distances between the data items. The distance estimation using the partial distance function will fail when the incomplete data items do not have values in the same dimensions. Therefore, we plan to adapt the DENCFURE algorithm to incomplete data introducing a pre-clustering step. Our idea is to cluster the incomplete data set with a partitioning fuzzy clustering algorithm adapted to incomplete data and substituting the missing values by the corresponding values of their cluster centers.

References

[ABKS99] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. OPTICS: Ordering Points To Identify the Clustering Structure. In *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania,USA*, pages 46 – 60, 1999.

[AG89] Nabil Ali Ahmed and D. V. Gokhale. Entropy expressions and their estimators for multivariate distributions. *IEEE Transactions on Information Theory*, 35(3):688–692, 1989.

[AGGR98] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. *SIGMOD Record*, 27(2):94–105, June 1998.

[AIS93] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining Association Rules Between Sets of Items in Large Databases. *SIGMOD Record*, 22(2):207–216, June 1993.

[AN07] A. Asuncion and D.J. Newman. UCI Machine Learning Repository, 2007.

[AO01] C. M. Antunes and A. L. Oliveira. Temporal data mining: An overview. In *KDD Workshop on Temporal Data Mining*, pages 1–13, 2001.

[AWY+99] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia Procopiuc, and Jong Soo Park. Fast Algorithms for Projected Clustering. *SIGMOD Record*, 28(2):61–72, June 1999.

[AY00] Charu C. Aggarwal and Philip S. Yu. Finding Generalized Projected Clusters in High Dimensional Spaces. *SIGMOD Record*, 29(2):70–81, May 2000.

[Bac78] E. Backer. *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets*. Delftse Universitaire Pers, 1978.

[Ban95] U. Bankhofer. *Unvollständige Daten- und Distanzmatrizen in der multivariaten Datenanalyse*. Reihe Quantitative Ökonomie. Eul, 1995.

[BEF84] James C. Bezdek, Robert Ehrlich, and William Full. FCM: The fuzzy c-means clustering algorithm. *Computers and Geosciences*, 10(2):191 – 203, 1984.

[Bez74] J. Bezdek. Numerical taxonomy with fuzzy sets. *Journal of Mathematical Biology*, 1(1):57–71, 1974.

[Bez75] J. C. Bezdek. Mathematical models for systematics and taxonomy. In *Proceedings of 8th Annual International Conference on Numerical Taxonomy*, pages 143–166. W. H. Freeman, 1975.

[Bez81] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[BG06] Priyadarshi Bhattacharya and Marina L. Gavrilova. CRYSTAL - A new density-based fast and efficient clustering algorithm. In *In Proceedings of the 3rd International Symposium on Voronoi Diagrams in Science and Engineering, ISVD 2006, Banff, Alberta, Canada, July 2-5, 2006*, pages 102 – 111, 2006.

[BH67] Geoffrey H. Ball and David J. Hall. A clustering technique for summarizing multivariate data. *Systems Research and Behavioral Science*, 12(2):153–155, 1967.

[BH02] J.C. Bezdek and R.J. Hathaway. VAT: A Tool for Visual Assessment of (Cluster) Tendency. In *Proceedings of the 2002 International Joint Conference on Neural Networks*, volume 3, pages 2225 – 2230, 2002.

[BH07] J. Beringer and E. Hüllermeier. Adaptive Optimization of the Number of Clusters in Fuzzy Clustering. In *FUZZ-IEEE 2007, Proceedings of the IEEE International Conference on Fuzzy Systems, Imperial College, London, UK, 23-26 July, 2007*, pages 1–6, 2007.

[BHB+96] Amine Bensaid, Lawrence O. Hall, James C. Bezdek, Laurence P. Clarke, Martin L. Silbiger, John A. Arrington, and F. Reed Murtagh. Validity-guided (re)clustering with applications to image segmentation. *IEEE Transactions on Fuzzy Systems*, 4(2):112 – 123, 1996.

[BHEG01] Asa Ben-Hur, Andre Elisseeff, and Isabelle Guyon. A stability based method for discovering structure in clustered data. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 6–17, 2001.

[BHH07] J. C. Bezdek, R. J. Hathaway, and J. M. Huband. Visual Assessment of Clustering Tendency for Rectangular Dissimilarity Matrices. *IEEE Transactions on Fuzzy Systems*, 15(5):890–903, October 2007.

[BK06] C. Borgelt and R. Kruse. Finding the Number of Fuzzy Clusters by Resampling. In *Proceedings of the IEEE International Conference on Fuzzy Systems*, pages 48 – 54, 2006.

[BKK+99] J.C. Bezdek, J.M. Keller, R. Krishnapuram, L.I Kuncheva, and N.R. Pal. Will the real iris data please stand up? *IEEE Transactions on Fuzzy Systems*, 7(3):368 – 369, Jun 1999.

[BKKP06] J.C. Bezdek, J. Keller, R. Krisnapuram, and N. Pal. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. The Handbooks of Fuzzy Sets. Springer US, 2006.

[BKNS00] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, pages 93 – 104, 2000.

[Bor07] Christian Borgelt. Resampling for Fuzzy Clustering. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 15(05):595 – 614, 2007.

[Bre89] James N. Breckenridge. Replicating Cluster Analysis: Method, Consistency, and Validity. *Multivariate Behavioral Research*, 24(2):147–161, 1989.

[BTK00] Christian Borgelt, Heiko Timm, and Rudolf Kruse. Using fuzzy clustering to improve naive Bayes classifiers and probabilistic networks. In *Proceedings of Ninth IEEE International Conference on Fuzzy Systems (FUZZ IEEE 2000)*, pages 53–58, 2000.

[BWE80] J. C. Bezdek, M. P. Windham, and R. Ehrlich. Statistical parameters of cluster validity functionals. *International Journal of Parallel Programming*, 9(4):323–336, 1980.

[BWS06] Mohamed Bouguessa, Shengrui Wang, and Haojun Sun. An objective approach to cluster validation. *Pattern Recognition Letters*, 27(13):1419–1430, October 2006.

[CBC12] J.P. Carvalho, F. Batista, and L. Coheur. A critical survey on the use of Fuzzy Sets in Speech and Natural Language Processing. In *Proceedings of*

the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2012), pages 1 – 8, June 2012.

[CF11] Hoel Le Capitaine and Carl Frélicot. A Cluster-Validity Index Combining an Overlap Measure and a Separation Measure Based on Fuzzy-Aggregation Operators. *IEEE Transactions on Fuzzy Systems*, 19(3):580–588, 2011.

[Com94] Pierre Comon. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314, 1994.

[CSB87] J. H. Conway, N. J. A. Sloane, and E. Bannai. *Sphere-packings, Lattices, and Groups*. Springer-Verlag New York, Inc., New York, NY, USA, 1987.

[Dav96] R. N. Dave. Validating fuzzy partitions obtained through c-shells clustering. *Pattern Recognition Letters*, 17(6):613–623, May 1996.

[DE84] William H.E. Day and Herbert Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1(1):7–24, 1984.

[DF02] Sandrine Dudoit and Jane Fridlyand. A prediction-based resampling method for estimating the number of clusters in a dataset. *Genome Biology*, 3:1–21, 2002.

[DHS00] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[Dix79] John K. Dixon. Pattern Recognition with Partly Missing Data. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(10):617–621, October 1979.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[DP99] Didier Dubois and Henri Prade. The Place of Fuzzy Logic in AI. In *Selected and Invited Papers from the Workshop on Fuzzy Logic in Artificial Intelligence*, IJCAI '97, pages 9–21, London, UK, UK, 1999. Springer-Verlag.

[Dun77] J. C. Dunn. Indices of partition fuzziness and the detection of clusters in large data sets. In *Fuzzy Automata and Decision Processes*, number 2, pages 271–284. Elsevier, New York, 1977.

[DXG+07] Lian Duan, Lida Xu, Feng Guo, Jun Lee, and Baopin Yan. A local-density based spatial clustering algorithm with noise. *Information Systems*, 32(7):978 – 986, 2007.

[EC02] Vladimir Estivill-Castro. Why So Many Clustering Algorithms: A Position Paper. *ACM SIGKDD Explorations Newsletter*, 4(1):65–75, June 2002.

[ECL00] Vladimir Estivill-Castro and Ickjai Lee. AUTOCLUST: Automatic Clustering via Boundary Extraction for Mining Massive Point-Data Sets. In *Proceedings of the 5th International Conference on Geocomputation*, pages 23 – 25, 2000.

[Efi12] Arthur Efimow. Bestimmung der optimalen Clusteranzahl für Fuzzy Clustering von unvollständigen Daten anhand der Partitionierungsstabilität. Bachelor's thesis (Bachelorarbeit), Heinrich-Heine-Universität Düsseldorf, April 2012.

[EHPJX96] M. Ester, Kriegel H.-P., Sander J., and Xu X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226 – 231, 1996.

[FGP+10] M. Falasconi, A. Gutierrez, M. Pardo, G. Sberveglieri, and S. Marco. A stability based validity method for fuzzy clustering. *Pattern Recognition*, 43(4):1292–1305, April 2010.

[Fis36] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugenics*, 7:179 – 188, 1936.

[FK97] Hichem Frigui and Raghu Krishnapuram. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7):1109 – 1119, 1997.

[FM04] Jerome H. Friedman and Jacqueline J. Meulman. Clustering objects on subsets of attributes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(4):815–849, 2004.

[FPP98] D. Freedman, R. Pisani, and R. Purves. *Statistics*. Norton International Student Edition Series. W.W. Norton, New York, NY, USA, 1998.

[FPS96a] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3):37–54, 1996.

[FPS96b] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Knowledge Discovery and Data Mining: Towards a Unifying Framework. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 82–88, 1996.

[FPS96c] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39(11):27–34, 1996.

[FS89] Y. Fukuyama and M. Sugeno. A new method for choosing the number of clusters for the fuzzy c-means method. In *Proceedings of the 5th Fuzzy Systems Symposium*, pages 247–250, 1989.

[FSS+09] Ahmed M. Fahim, Gunter Saake, Abdel-Badeeh M. Salem, Fawzy A. Torkey, and Mohamed A. Ramadan. An Enhanced Density Based Spatial clustering of Applications with Noise. In *Proceedings of The 2009 International Conference on Data Mining, DMIN 2009, July 13-16, 2009, Las Vegas, USA*, pages 517 – 523, 2009.

[GG89] Isak Gath and Amir B. Geva. Unsupervised Optimal Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 11(7):773–780, 1989.

[GHJ91] A. Guénoche, P Hansen, and B. Jaumard. Efficient algorithms for divisive hierarchical clustering with the diameter criterion. *Journal of Classification*, 8(1):5–30, 1991.

[GK78] D.E. Gustafson and W.C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proceedings of the 1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, pages 761 – 766, Jan 1978.

[GRS98] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: An Efficient Clustering Algorithm for Large Databases. *SIGMOD Record*, 27(2):73–84, June 1998.

[GSBN00] Amir B. Geva, Yossef Steinberg, Shay Bruckmair, and Gerry Nahum. A comparison of cluster validity criteria for a mixture of normal distributed data. *Pattern Recognition Letters*, 21(6–7):511–529, June 2000.

[Gut84] Antonin Guttman. R-trees: A Dynamic Index Structure for Spatial Searching. In *Proceedings of the 1984 ACM SIGMOD International Conference*

on Management of Data, SIGMOD '84, pages 47–57, New York, NY, USA, 1984. ACM.

[HB01] Richard J. Hathaway and James C. Bezdek. Fuzzy c-means Clustering of Incomplete Data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31(5):735 – 744, 2001.

[HBH05] J. M. Huband, J. C. Bezdek, and R. J. Hathaway. bigVAT: Visual Assessment of Cluster Tendency for Large Data Sets. *Pattern Recognition*, 38(11):1875–1886, November 2005.

[HBH06] R. J. Hathaway, J. C. Bezdek, and J. M. Huband. Scalable Visual Assessment of Cluster Tendency for Large Data Sets. *Pattern Recognition*, 39(7):1315–1324, July 2006.

[HBP12] Timothy C. Havens, James C. Bezdek, and Marimuthu Palaniswami. Cluster validity for kernel fuzzy clustering. In *FUZZ-IEEE 2012, Proceedings of the IEEE International Conference on Fuzzy Systems, Brisbane, Australia, June 10-15, 2012*, pages 1 – 8, 2012.

[HC10a] Ludmila Himmelspach and Stefan Conrad. Clustering Approaches for Data with Missing Values: Comparison and Evaluation. In *Proceedings of the Fifth IEEE International Conference on Digital Information Management, ICDIM 2010, July 5-8, 2010, Lakehead University, Thunder Bay, Canada*, pages 19 – 28, 2010.

[HC10b] Ludmila Himmelspach and Stefan Conrad. Fuzzy Clustering of Incomplete Data Based on Cluster Dispersion. In *Proceedings of the 13th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2010, Dortmund, Germany, June 28 - July 2, 2010*, Lecture Notes in Computer Science, vol 6178, Springer, pages 59–68, 2010.

[HC11] Ludmila Himmelspach and Stefan Conrad. Density-Based Clustering using Fuzzy Proximity Relations. In *Proceedings of the 2011 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, pages 1 – 6, 2011.

[HCC12] Ludmila Himmelspach, João Paulo Carvalho, and Stefan Conrad. On Cluster Validity for Fuzzy Clustering of Incomplete Data. In *Proceedings of the 6th International Conference on Scalable Uncertainty Management, SUM 2012, Marburg, Germany, September 17-19, 2012*, Lecture Notes in Computer Science, vol 7520, Springer, pages 612–618, 2012.

[HCJJ11] Timothy C. Havens, Radha Chitta, Anil K. Jain, and Rong Jin. Speedup of fuzzy and possibilistic kernel c-means for large-scale clustering. In *FUZZ-IEEE 2011, Proceedings of the IEEE International Conference on Fuzzy Systems, Taipei, Taiwan, 27–30 June, 2011*, pages 463–470, 2011.

[HFH+09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explorations Newsletter*, 11(1):10–18, November 2009.

[HGN00] Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for Association Rule Mining - a General Survey and Comparison. *SIGKDD Explorations Newsletter*, 2(1):58–64, June 2000.

[HHC11] Ludmila Himmelspach, Daniel Hommers, and Stefan Conrad. Cluster Tendency Assessment for Fuzzy Clustering of Incomplete Data. In *Proceedings of 6th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT-2011)*, pages 290–297. Atlantis Press, 2011.

[Him08] Ludmila Himmelspach. Clustering mit fehlenden Werten: Analyse und Vergleich. Master's thesis (Masterarbeit), University of Dusseldorf, Germany, 2008.

[Him09] Ludmila Himmelspach. Vergleich von Strategien zum Clustern von Daten mit fehlenden Werten. In *Proceedings of the 21. GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), Rostock-Warnemünde, Mecklenburg-Vorpommern, Germany, June 2-5, 2009.*, pages 129 – 133, 2009.

[HK98] Alexander Hinneburg and Daniel A. Keim. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, New York, USA, August 27-31, 1998*, pages 58 – 65, 1998.

[HK00] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.

[HKK96] Frank Höppner, Frank Klawonn, and Rudolf Kruse. *Fuzzy-Clusteranalyse. Verfahren für die Bilderkennung, Klassifikation und Datenanalyse*. Vieweg, 1996.

[HLZS99] Keyun Hu, Yuchang Lu, Lizhu Zhou, and Chunyi Shi. Integrating Classification and Association Rule Mining: A Concept Lattice Framework.

In *Proceedings of the 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, RSFDGrC '99, Yamaguchi, Japan, November 9-11, 1999*, pages 443–447, 1999.

[HM92] J. Hartigan and Surya Mohanty. The runt test for multimodality. *Journal of Classification*, 9(1):63 – 70, 1992.

[Höf11] David Höfig. Vergleich von Strategien zur Bestimmung der optimalen Clusteranzahl. Bachelor's thesis (Bachelorarbeit), Heinrich-Heine-Universität Düsseldorf, November 2011.

[Höp99] Frank Höppner. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. Jossey-Bass higher and adult education series. Wiley, 1999.

[Jan07] J. Jantzen. *Foundations of Fuzzy Control*. Wiley, 2007.

[KAH96] Krzysztof Koperski, Junas Adhikary, and Jiawei Han. Spatial Data Mining: Progress and Challenges. In *Proceedings of the SIGMOD Workshop on Research Issues on data Mining and Knowledge Discovery (DMKD)*, pages 1 – 10, 1996.

[KDL07] Rudolf Kruse, Christian Döring, and Marie-Jeanne Lesot. Fundamentals of Fuzzy Clustering. In *Advances in Fuzzy Clustering and its Applications*, pages 1–30. John Wiley & Sons, Ltd, 2007.

[KF92] R. Krishnapuram and C.-P. Freg. Fitting an unknown number of lines and planes to image data through compatible cluster merging. *Pattern Recognition*, 25(4):385 – 400, 1992.

[KK93] R. Krishnapuram and J.M. Keller. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems*, 1(2):98 – 110, May 1993.

[KK96] R. Krishnapuram and J.M. Keller. The possibilistic C-means algorithm: insights and recommendations. *IEEE Transactions on Fuzzy Systems*, 4(3):385 – 393, Aug 1996.

[KKK04] Peer Kröger, Hans-Peter Kriegel, and Karin Kailing. Density-Connected Subspace Clustering for High-Dimensional Data. In *Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24, 2004*, pages 246–256, 2004.

[KKLL04] Young-Il Kim, Dae-Won Kim, Doheon Lee, and Kwang H. Lee. A cluster validation index for GK cluster analysis based on relative degree of sharing. *Information Sciences*, 168(1–4):225–242, December 2004.

[KKW15] Frank Klawonn, Rudolf Kruse, and Roland Winkler. Fuzzy clustering: More than just fuzzification. *Fuzzy Sets and Systems*, 2015.

[KNB99] Rudolf Kruse, Detlef Nauck, and Christian Borgelt. Data Mining with Fuzzy Methods: Status and Perspectives. In *Proceedings of 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99)*, 1999.

[KR90] L. Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990.

[Kwo98] S.H. Kwon. Cluster Validity Index for Fuzzy Clustering. *Electronics Letters*, 34(22):2176–2177, October 1998.

[LFSMC09] Luis F. Lago-Fernández, Manuel A. Sánchez-Montañés, and Fernando J. Corbacho. Fuzzy Cluster Validation Using the Partition Negentropy Criterion. In *Proceedings of the 19th International Conference on Artificial Neural Networks - ICANN 2009, Part II, Limassol, Cyprus, September 14-17, 2009*, Lecture Notes in Computer Science, vol 5769, Springer, pages 235–244, 2009.

[Lin73] Robert L. Ling. A computer generated aid for cluster analysis. *Communications of the ACM*, 16(6):355 – 361, June 1973.

[Lit88] Roderick JA Little. A test of missing completely at random for multivariate data with missing values. *Journal of the American Statistical Association*, 83(404):1198–1202, 1988.

[LJ03] Martin Law and Anil K. Jain. Cluster Validity by Bootstrapping Partitions. Technical Report MSU-CSE-03-5, University of Washington, February 2003.

[Llo82] Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–136, 1982.

[LM91] D. Lucarella and R. Morara. FIRST: Fuzzy Information Retrieval SysTem. *Journal of Information Science*, 17(2):81–91, May 1991.

[LOW02] Weiqiang Lin, Mehmet A. Orgun, and Graham Williams. An Overview of Temporal Data Mining. In *Proceedings of the 1st Australian Data Mining Workshop (AusDM02)*, pages 83–90. University of Technology, Sydney, 2002.

[LR02] R.J.A. Little and D.B. Rubin. *Statistical analysis with missing data*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 2002.

[LRBB04] T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann. Stability-based validation of clustering solutions. *Neural Comput.*, 16(6):1299–1323, June 2004.

[LS09] Levi Lelis and Jörg Sander. Semi-supervised Density-Based Clustering. In *Proceedings of the Ninth IEEE International Conference on Data Mining (ICDM 2009), Miami, Florida, USA, 6-9 December 2009*, pages 842 – 847, 2009.

[LXY00] Bing Liu, Yiyuan Xia, and Philip S. Yu. Clustering Through Decision Tree Construction. In *Proceedings of the Ninth International Conference on Information and Knowledge Management*, CIKM '00, pages 20–29, New York, NY, USA, 2000. ACM.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[Mah36] Prasanta Chandra Mahalanobis. On the Generalised Distance in Statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49—55, 1936.

[MBF08] Laurent Mascarilla, Michel Berthier, and Carl Frélicot. A k-order fuzzy or operator for pattern classification with k -order ambiguity rejection. *Fuzzy Sets and Systems*, 159(15):2011–2029, August 2008.

[MFK+14] G. Molenberghs, G. Fitzmaurice, M.G. Kenward, A. Tsiatis, and G. Verbeke. *Handbook of Missing Data Methodology*. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. Taylor & Francis, 2014.

[Mie05] Taneli Mielikäinen. Summarization Techniques for Pattern Collections in Data Mining. *The Computing Research Repository (CoRR)*, abs/cs/0505071, 2005.

[MV13] Michael Mampaey and Jilles Vreeken. Summarizing categorical data by clustering attributes. *Data Mining and Knowledge Discovery*, 26(1):130–173, 2013.

[MWZ94] Willi Meier, Richard Weber, and Hans-Jurgen Zimmermann. Fuzzy data analysis – Methods and industrial applications. *Fuzzy Sets and Systems*, 61(1):19 – 28, 1994.

[NH94] Raymond T. Ng and Jiawei Han. Efficient and Effective Clustering Methods for Spatial Data Mining. In *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 144–155, 1994.

[Nov92] Vilém Novák. Fuzzy Sets in Natural Language Processing. In Ronald R. Yager and Lotfi A. Zadeh, editors, *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, volume 165 of *The Springer International Series in Engineering and Computer Science, Springer US*, pages 185–200. 1992.

[OBS92] Atsuyuki Okabe, Barry Boots, and Kokichi Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley & Sons, Inc., New York, NY, USA, 1992.

[PB95] N.R. Pal and J.C. Bezdek. On Cluster Validity for the Fuzzy c-Means Model. *IEEE Transactions on Fuzzy Systems*, 3(3):370–379, aug 1995.

[Ped12] W. Pedrycz. *Fuzzy Modelling: Paradigms and Practice*. International Series in Intelligent Technologies. Springer US, 2012.

[PP05] Giuseppe Papari and Nicolai Petkov. Algorithm That Mimics Human Perceptual Grouping of Dot Patterns. In *Proceedings of the First International Symposium on Brain, Vision, and Artificial Intelligence (BVAI 2005), Naples, Italy, October 19-21, 2005*, pages 497 – 506, 2005.

[PPKB05] Nikhil R. Pal, Kuhu Pal, James M. Keller, and James C. Bezdek. A Possibilistic Fuzzy c-Means Clustering Algorithm. *IEEE Transactions on Fuzzy Systems*, 13(4):517–530, 2005.

[Rou78] M. Roubens. Pattern classification problems and fuzzy sets. *Fuzzy Sets and Systems*, 1(4):239–253, October 1978.

[Rub04] D.B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley Classics Library. Wiley, 2004.

[Rus69] Enrique H. Ruspini. A new approach to clustering. *Information and Control*, 15(1):22 – 32, 1969.

[SC03] Shashi Shekhar and Sanjay Chawla. *Spatial databases - a tour*. Prentice Hall, 2003.

[Sch97] J.L. Schafer. *Analysis of Incomplete Multivariate Data*. Chapman & Hall/-CRC Monographs on Statistics & Applied Probability. CRC Press, 1997.

[Sha48] C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27:379–423, 1948.

[SHB08] I. J. Sledge, J. M. Huband, and J. C. Bezdek. (Automatic) Cluster Count Extraction from Unlabeled Data Sets. In *Proceedings of the 2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery – Volume 01*, FSKD '08, pages 3–13, Washington, DC, USA, 2008. IEEE Computer Society.

[SL01] Manish Sarkar and Tze-Yun Leong. Fuzzy K-means clustering with missing values. In *AMIA 2001, Proceedings of the American Medical Informatics Association Annual Symposium, Washington, DC, USA, November 3-7, 2001*, pages 588 – 592, 2001.

[Stu98] C. Stutz. Partially Supervised Fuzzy c-Means Clustering with Cluster Merging. In *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing*, pages 1725–1729, 1998.

[Stu03] Werner Stuetzle. Estimating the Cluster Tree of a Density by Analyzing the Minimal Spanning Tree of a Sample. *Journal of Classification*, 20(1):025 – 047, 2003.

[SW08] Min Song and Yi-Fang Brook Wu. *Handbook of Research on Text and Web Mining Technologies*. Information Science Reference - Imprint of: IGI Publishing, Hershey, PA, 2008.

[TBDK04] Heiko Timm, Christian Borgelt, Christian Döring, and Rudolf Kruse. An extension to possibilistic fuzzy cluster analysis. *Fuzzy Sets and Systems*, 147(1):3–16, 2004.

[TDK02] Heiko Timm, Christian Döring, and Rudolf Kruse. Fuzzy cluster analysis of partially missing datasets. In *Proceedings of the European Symposium on Intelligent Technologies, Hybid Systems and Their Implementation on Smart Adaptive Systems (EUNITE 2002), Albufeira, Portugal*, pages 426 – 431, 2002.

[TDK03] Heiko Timm, Christian Döring, and Rudolf Kruse. Differentiated Treatment of Missing Values in Fuzzy Clustering. In *Proceedings of the 10th International Fuzzy Systems Association World Congress on Fuzzy Sets and Systems, IFSA 2003*, pages 354 – 361, Istanbul, 2003. LNAI 2715.

[TDK04] H. Timm, C. Döring, and R. Kruse. Different Approaches to Fuzzy Clustering of Incomplete Datasets. *International Journal of Approximate Reasoning*, 35(3):239–249, 2004.

[Thu01] Bhavani M. Thuraisingham. *Managing and Mining Multimedia Databases*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2001.

[Tim02] Heiko Timm. *Fuzzy-Clusteranalyse: Methoden zur Exploration von Daten mit fehlenden Werten sowie klassifizierten Daten*. Dissertation, Otto-von-Guericke-University of Magdeburg, Germany, 2002.

[TK98] Heiko Timm and Frank Klawonn. Classification of data with missing values. In *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT'98), Aachen, Deutschland*, pages 1304 – 1308, 1998.

[TSS05] Y. Tang, F. Sun, and Z. Sun. Improved Validation Index for Fuzzy Clustering. In *Proceedings of the American Control Conference, 2005*, volume 2, pages 1120–1125, june 2005.

[VP08] H. Vanderijt and F.X. Plooij. *The Wonder Weeks: How to Stimulate Your Baby's Mental Development and Help Him Turn His 8 Predictable, Great, Fussy Phases Into Magical Leaps Forward*. Kiddy World Promotions, 2008.

[Wag04] K. Wagstaff. Clustering with Missing Values: No Imputation Required. In *Proceedings of the Meeting of the International Federation of Classification Societies*, pages 649 – 658, 2004.

[WGB+10] L. Wang, X. Geng, J.C. Bezdek, C. Leckie, and K. Ramamohanarao. Enhanced Visual Analysis for Cluster Tendency Assessment and Data Partitioning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1401–1414, 2010.

[WKN10] Yingying Wen, Kevin B. Korb, and Ann E. Nicholson. Generating Incomplete Data with DataZapper. In *Proceedings of the International Conference on Agents and Artificial Intelligence, ICAART 2009*, Communications in Computer and Information Science, vol 67, Springer, pages 110–123. 2010.

[WY05] Kuo-Lung Wu and Miin-Shen Yang. A cluster validity index for fuzzy clustering. *Pattern Recognition Letters*, 26(9):1275–1291, July 2005.

[WZ07] Weina Wang and Yunjie Zhang. On fuzzy cluster validity indices. *Fuzzy Sets and Systems*, 158(19):2095–2117, October 2007.

[XB91] X.L. Xie and G. Beni. A Validity Measure for Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):841–847, August 1991.

[YZ12] R.R. Yager and L.A. Zadeh. *An Introduction to Fuzzy Logic Applications in Intelligent Systems*. The Springer International Series in Engineering and Computer Science. Springer US, 2012.

[Zad65] Lotfi A. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965.

[Zad75] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning – I. *Information Sciences*, 8(3):199 – 249, 1975.

[Zim12] H.J. Zimmermann. *Practical Applications of Fuzzy Technologies*. The Handbooks of Fuzzy Sets. Springer US, 2012.

[ZL04] Jiang-She Zhang and Yiu-Wing Leung. Improved possibilistic C-means clustering algorithms. *IEEE Transactions on Fuzzy Systems*, 12(2):209–217, 2004.

[ZLE99] N. Zahid, M. Limouri, and A. Essaid. A new cluster-validity for fuzzy clustering. *Pattern Recognition*, 32(7):1089–1097, 1999.

[ZLSE08] Abdelhamid Zemirline, Laurent Lecornu, Basel Solaiman, and Ahmed Ech-Cherif. An Efficient Association Rule Mining Algorithm for Classification. In *Proceedings of the 9th International Conference on Artificial Intelligence and Soft Computing - ICAISC 2008, Zakopane, Poland, June 22-26, 2008*, pages 717–728, 2008.

[ZRL96] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. *SIGMOD Record*, 25(2):103–114, June 1996.

List of Figures

List of Tables

A

Appendix

In chapter 5, the performance results of cluster validity indexes adapted to incomplete data were not fully presented in the evaluation discussion. The following tables list the complete evaluation results.

Table A.1: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	2_{100}	15_{64}	2_{100}	15_{64}	15_{64}	14_{53}	17_{100}	2_{100}	15_{64}	15_{45}	15_{64}	15_{64}
10D-10	10	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	17_{27}	17_{26}
3D-5-sep	5	2_{100}	5_{83}	2_{100}	5_{83}	5_{83}	4_{68}	10_{100}	2_{100}	5_{77}	5_{83}	5_{83}	5_{83}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	5_{88}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{88}	5_{88}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{87}	5_{87}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{56}	5_{56}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{73}	5_{73}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{62}	6_{64}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{86}	3_{86}
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	3_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}
bensaid	3	2_{100}	4_{74}	2_{100}	10_{78}	5_{86}	5_{82}	10_{99}	2_{100}	5_{85}	10_{83}	10_{84}	10_{90}

Table A.2: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 10%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	15_{64}	15_{64}	15_{64}	15_{64}	15_{64}	2_{100}	15_{64}	15_{64}	15_{64}
10D-10	10	2_{100}	3_{78}	3_{100}	2_{100}	4_{22}	17_{45}	11_{19}	2_{100}	$16_{32}/17_{32}$
3D-5-sep	5	5_{46}	2_{100}	2_{100}	2_{100}	2_{100}	5_{83}	5_{83}	2_{100}	5_{83}
3D-5-ov	5	5_{53}	2_{100}	2_{100}	2_{100}	2_{100}	5_{88}	5_{88}	4_{59}	5_{88}
3D-5-strov	5	5_{49}	2_{100}	2_{100}	2_{100}	2_{100}	2_{73}	2_{100}	4_{68}	5_{87}
3D-5-h-sep	5	6_{32}	2_{100}	2_{100}	2_{100}	2_{100}	5_{56}	5_{56}	2_{100}	5_{56}
3D-5-h-ov	5	5_{40}	2_{100}	2_{100}	2_{100}	2_{100}	5_{73}	2_{100}	2_{100}	5_{73}
3D-5-h-strov	5	5_{38}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{62}
2D-3-sep	3	$4_{34}(3_{33})$	3_{100}	3_{100}	3_{100}	3_{100}	9_{25}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	4_{45}	2_{100}	2_{100}	2_{100}	3_{100}	8_{28}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	4_{65}	2_{100}	2_{100}	2_{100}	3_{100}	8_{30}	3_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{60}	2_{100}	2_{100}	2_{100}	2_{100}	9_{35}	3_{86}	2_{100}	3_{86}
2D-3-3-tog	3	$4_{55}(3_{45})$	3_{100}	3_{100}	3_{100}	3_{100}	7_{36}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	3_{76}	3_{100}	3_{100}	2_{100}	3_{100}	5_{100}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	3_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{100}	3_{100}	3_{100}	3_{100}
bensaid	3	10_{43}	5_{49}	5_{86}	2_{100}	6_{100}	6_{76}	10_{68}	6_{100}	10_{72}

Table A.3: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	5_{98}	10_{54}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{72}	10_{69}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{63}	10_{62}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{95}	9_{35}
sonar	2	2_{100}	2_{96}	2_{100}	2_{95}	2_{100}	2_{97}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	3_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{86}	10_{64}	10_{76}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{100}

Table A.4: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 10% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{49}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{76}
glass	6	10_{58}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{90}
ionosphere	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
iris	3	4_{34}	2_{100}	2_{100}	2_{100}	2_{100}	3_{75}	3_{100}	2_{100}	10_{83}
iris-bezdek	3	5_{38}	2_{100}	2_{100}	2_{100}	2_{100}	3_{84}	3_{100}	3_{100}	10_{80}
sonar	2	10_{100}	2_{98}	2_{98}	2_{98}	2_{99}	2_{100}	2_{95}	2_{65}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	9_{63}	3_{100}	3_{100}	9_{63}	3_{100}	3_{100}	3_{100}	3_{100}	10_{58}
wine	3	10_{100}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}

Table A.5: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	2_{100}	7_{25}	2_{100}	$14_{29}(15_{28})$	15_{41}	12_{22}	17_{100}	2_{100}	16_{36}	$12_{19}/13_{19}$	15_{43}	17_{46}
10D-10	10	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	17_{35}	$15_{26}/17_{26}$
3D-5-sep	5	2_{100}	3_{100}	2_{100}	$4_{42}(5_{39})$	3_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{98}	5_{52}	6_{37}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	$4_{42}(5_{35})$	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{77}	6_{74}	9_{44}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{99}	6_{71}	6_{61}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	6_{50}	6_{52}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	6_{60}	6_{54}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{29}	8_{70}
2D-3-sep	3	3_{100}	3_{100}	2_{100}	3_{100}	$4_{44}(3_{43})$	3_{55}	10_{100}	2_{100}	3_{97}	$3_{45}/4_{45}$	3_{100}	3_{82}
2D-3-2-tog	3	2_{100}	3_{100}	2_{100}	3_{67}	2_{100}	2_{97}	10_{100}	2_{100}	2_{100}	2_{50}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{83}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{75}	10_{49}
2D-3-3-tog	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{61}	10_{100}	2_{100}	2_{100}	10_{95}	3_{100}	3_{55}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	10_{98}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	10_{73}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	10_{100}	10_{100}
bensaid	3	2_{100}	10_{34}	2_{100}	10_{59}	4_{69}	4_{41}	10_{91}	2_{100}	4_{72}	10_{66}	10_{58}	10_{64}

Table A.6: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 25%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	13_{28}	$14_{42}(15_{41})$	$14_{42}(15_{41})$	$14_{42}(15_{41})$	$14_{42}(15_{41})$	2_{100}	15_{43}	15_{42}	15_{42}
10D-10	10	2_{100}	4_{74}	3_{100}	2_{100}	4_{74}	17_{24}	16_{14}	2_{100}	17_{33}
3D-5-sep	5	5_{39}	5_{84}	5_{84}	5_{84}	5_{84}	5_{61}	5_{84}	4_{65}	10_{26}
3D-5-ov	5	4_{42}	2_{100}	2_{100}	2_{100}	2_{100}	5_{93}	5_{93}	4_{58}	7_{60}
3D-5-strov	5	5_{43}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	4_{56}	10_{45}
3D-5-h-sep	5	6_{35}	2_{100}	2_{100}	2_{100}	2_{100}	5_{63}	5_{63}	2_{100}	10_{23}
3D-5-h-ov	5	5_{41}	2_{100}	2_{100}	2_{100}	2_{100}	5_{69}	2_{100}	2_{100}	5_{69}
3D-5-h-strov	5	6_{33}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{40}
2D-3-sep	3	3_{37}	3_{100}	3_{100}	3_{100}	3_{100}	10_{36}	3_{100}	3_{100}	3_{73}
2D-3-2-tog	3	4_{39}	2_{100}	2_{100}	2_{100}	3_{100}	10_{37}	3_{94}	3_{100}	3_{94}
2D-3-2-ov	3	4_{39}	2_{100}	2_{100}	2_{100}	3_{100}	10_{49}	3_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{55}	2_{100}	2_{100}	2_{100}	2_{100}	10_{50}	3_{60}	2_{100}	10_{64}
2D-3-3-tog	3	4_{61}	3_{100}	3_{100}	3_{100}	3_{100}	8_{31}	3_{100}	3_{100}	10_{63}
2D-3-3-ov	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	9_{58}	9_{52}	3_{100}	10_{95}
2D-3-3-strov	3	5_{100}	2_{100}	2_{100}	2_{100}	3_{100}	6_{100}	6_{51}	3_{100}	10_{100}
bensaid	3	7_{38}	7_{41}	2_{64}	2_{100}	7_{47}	10_{41}	10_{49}	2_{100}	10_{56}

Table A.7: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	10_{69}	10_{58}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	8_{43}	8_{37}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{59}	8_{74}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{51}	10_{74}
sonar	2	2_{100}	2_{99}	2_{100}	2_{98}	2_{100}	2_{99}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	8_{100}	7_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{54}	10_{75}	10_{76}
wine	3	2_{100}	2_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{88}

Table A.8: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 25% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{49}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{71}
glass	6	7_{30}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{53}
ionosphere	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
iris	3	6_{63}	2_{100}	2_{100}	2_{100}	2_{100}	6_{72}	3_{100}	3_{100}	10_{77}
iris-bezdek	3	5_{42}	2_{100}	2_{100}	2_{100}	2_{100}	3_{80}	3_{100}	3_{100}	10_{77}
sonar	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{99}	2_{68}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	9_{43}	3_{100}	3_{100}	3_{70}	3_{100}	3_{100}	3_{34}	3_{100}	10_{82}
wine	3	10_{100}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	4_{100}

Table A.9: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	2_{100}	7_{38}	2_{100}	17_{48}	17_{57}	3_{60}	17_{100}	2_{100}	2_{99}	17_{51}	17_{59}	17_{77}
10D-10	10	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{99}	17_{96}	2_{100}	2_{67}	2_{39}	17_{27}	15_{21}
3D-5-sep	5	2_{100}	4_{92}	2_{100}	6_{44}	6_{45}	4_{72}	10_{100}	2_{100}	2_{100}	10_{40}	10_{63}	10_{70}
3D-5-ov	5	2_{100}	3_{100}	2_{100}	7_{43}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{86}	10_{61}	10_{70}
3D-5-strov	5	2_{100}	4_{100}	2_{100}	6_{33}	2_{77}	4_{98}	10_{100}	2_{100}	2_{100}	10_{87}	10_{83}	10_{85}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{98}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{73}	8_{56}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{98}	10_{76}	10_{60}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{71}	2_{100}	2_{62}	10_{100}	2_{100}	2_{100}	10_{91}	10_{79}	8_{52}
2D-3-sep	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{95}	3_{100}	10_{100}	2_{100}	3_{50}	3_{100}	10_{50}	10_{48}
2D-3-2-tog	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{56}	3_{100}	10_{100}	2_{100}	3_{54}	3_{100}	10_{60}	10_{64}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{64}	2_{100}	2_{100}	10_{100}	2_{100}	10_{36}	2_{100}	10_{64}	10_{62}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{85}	2_{100}	2_{100}	10_{100}	2_{100}	10_{54}	2_{100}	10_{80}	10_{82}
2D-3-3-tog	3	2_{100}	3_{100}	2_{100}	10_{59}	3_{90}	3_{100}	10_{100}	2_{100}	10_{45}	3_{96}	10_{64}	10_{72}
2D-3-3-ov	3	2_{100}	3_{94}	2_{100}	10_{61}	3_{78}	3_{94}	10_{100}	2_{100}	10_{42}	10_{82}	10_{71}	10_{88}
2D-3-3-strov	3	2_{100}	2_{85}	2_{100}	10_{77}	2_{100}	2_{100}	10_{100}	2_{100}	2_{43}	10_{99}	10_{76}	10_{71}
bensaid	3	2_{41}	9_{31}	2_{100}	10_{41}	$7_{26}/9_{26}$	9_{32}	10_{88}	2_{100}	9_{30}	10_{49}	7_{25}	7_{25}

Table A.10: Cluster validity results of clusterings produced by PDSFCM on synthetic data sets with missing values 40%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	8_{31}	14_{46}	14_{46}	14_{46}	14_{47}	3_{100}	17_{32}	14_{33}	17_{62}
10D-10	10	2_{100}	4_{100}	3_{100}	3_{100}	4_{100}	17_{33}	10_{14}	3_{100}	17_{43}
3D-5-sep	5	4_{41}	5_{74}	5_{74}	2_{100}	5_{74}	4_{57}	5_{74}	4_{57}	10_{72}
3D-5-ov	5	5_{39}	2_{100}	2_{100}	2_{100}	2_{100}	5_{74}	5_{85}	4_{67}	10_{70}
3D-5-strov	5	6_{34}	2_{100}	2_{100}	2_{100}	2_{100}	5_{51}	5_{89}	4_{59}	10_{82}
3D-5-h-sep	5	5_{52}	2_{100}	2_{100}	2_{100}	2_{100}	5_{61}	6_{58}	2_{100}	10_{88}
3D-5-h-ov	5	5_{36}	2_{100}	2_{100}	2_{100}	2_{100}	6_{55}	2_{100}	2_{100}	10_{80}
3D-5-h-strov	5	5_{38}	2_{100}	2_{100}	2_{100}	2_{100}	5_{61}	2_{100}	2_{100}	10_{81}
2D-3-sep	3	3_{82}	3_{100}	3_{100}	3_{100}	3_{100}	10_{43}	10_{39}	3_{100}	10_{67}
2D-3-2-tog	3	4_{36}	2_{100}	2_{100}	2_{100}	3_{100}	10_{68}	10_{53}	3_{100}	10_{77}
2D-3-2-ov	3	4_{37}	2_{100}	2_{100}	2_{100}	3_{100}	10_{73}	10_{54}	3_{53}	10_{75}
2D-3-2-strov	3	2_{50}	2_{100}	2_{100}	2_{100}	2_{100}	10_{79}	10_{71}	2_{100}	10_{81}
2D-3-3-tog	3	3_{71}	3_{100}	3_{100}	3_{100}	3_{100}	10_{50}	10_{55}	3_{100}	10_{71}
2D-3-3-ov	3	8_{30}	3_{100}	3_{100}	2_{100}	3_{100}	10_{69}	10_{58}	3_{100}	10_{72}
2D-3-3-strov	3	10_{38}	2_{100}	2_{100}	2_{100}	10_{46}	10_{87}	10_{81}	3_{100}	10_{77}
bensaid	3	6_{23}	7_{36}	2_{31}	2_{100}	7_{37}	10_{40}	10_{48}	2_{48}	10_{33}

Table A.11: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	10_{85}	10_{90}
glass	6	2_{100}	2_{100}	2_{100}	5_{95}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	10_{63}	10_{78}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	7_{36}	7_{27}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{32}	10_{41}
sonar	2	2_{100}	2_{98}	2_{100}	2_{98}	2_{100}	2_{98}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	10_{50}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{74}	10_{57}	10_{64}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	3_{100}

Table A.12: Cluster validity results of clusterings produced by PDSFCM on real data sets with missing values 40% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{54}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{81}	10_{77}
glass	6	9_{39}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{100}	2_{100}	6_{35}
ionosphere	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
iris	3	4_{43}	2_{100}	2_{100}	2_{100}	2_{100}	5_{40}	3_{93}	3_{100}	9_{31}
iris-bezdek	3	5_{34}	2_{100}	2_{100}	2_{100}	2_{100}	3_{70}	3_{100}	3_{100}	10_{55}
sonar	2	10_{100}	2_{99}	2_{99}	2_{99}	2_{100}	2_{100}	2_{98}	2_{65}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	10_{43}	3_{94}	3_{95}	9_{52}	10_{47}	9_{32}	10_{47}	3_{57}	10_{65}
wine	3	8_{55}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	9_{83}

Table A.13: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	15_{53}	15_{53}	2_{100}	15_{51}	15_{48}	15_{53}	17_{100}	2_{100}	$16_{39}(15_{35})$	15_{53}	15_{53}	15_{53}
10D-10	10	3_{92}	3_{92}	3_{92}	3_{92}	3_{92}	3_{92}	3_{92}	2_{100}	3_{92}	3_{92}	13_{91}	13_{91}
3D-5-sep	5	2_{100}	2_{89}	2_{100}	5_{86}	5_{84}	5_{84}	10_{100}	2_{100}	5_{76}	5_{86}	5_{83}	5_{81}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{89}	5_{82}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{87}	5_{57}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	6_{53}	2_{100}	5_{67}	5_{67}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{71}	5_{65}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{65}	5_{52}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	7_{32}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	2_{95}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	7_{42}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{98}	3_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{85}	3_{85}
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{95}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{98}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	3_{100}	3_{97}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{99}
bensaid	3	2_{94}	4_{75}	2_{100}	10_{41}	4_{79}	5_{43}	10_{97}	2_{100}	4_{85}	10_{80}	10_{71}	10_{51}

Table A.14: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 10%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	15_{51}	15_{53}	15_{53}	15_{53}	15_{53}	$2_{50}/15_{50}$	15_{53}	$14_{35}(15_{33})$	15_{44}
10D-10	10	2_{100}	3_{92}	3_{100}	2_{100}	13_{92}	13_{91}	13_{91}	2_{100}	15_{28}
3D-5-sep	5	5_{64}	2_{100}	2_{100}	2_{100}	2_{100}	5_{83}	5_{86}	2_{76}	5_{69}
3D-5-ov	5	5_{71}	2_{100}	2_{100}	2_{100}	2_{100}	5_{89}	5_{89}	2_{68}	5_{86}
3D-5-strov	5	5_{67}	2_{100}	2_{100}	2_{100}	2_{100}	5_{84}	2_{94}	2_{44}	5_{84}
3D-5-h-sep	5	5_{49}	2_{100}	2_{100}	2_{100}	2_{100}	5_{67}	5_{67}	2_{100}	5_{59}
3D-5-h-ov	5	5_{48}	2_{100}	2_{100}	2_{100}	2_{100}	5_{71}	5_{67}	2_{100}	5_{69}
3D-5-h-strov	5	5_{40}	2_{100}	2_{100}	2_{100}	2_{100}	2_{64}	2_{100}	2_{100}	5_{64}
2D-3-sep	3	3_{41}	3_{100}	3_{100}	3_{100}	3_{100}	7_{28}	3_{92}	3_{86}	3_{100}
2D-3-2-tog	3	3_{38}	2_{100}	2_{100}	2_{100}	3_{100}	7_{31}	3_{96}	3_{72}	3_{100}
2D-3-2-ov	3	3_{48}	3_{100}	3_{100}	2_{100}	3_{100}	6_{53}	3_{79}	3_{87}	3_{100}
2D-3-2-strov	3	3_{31}	2_{100}	2_{100}	2_{100}	2_{100}	9_{31}	3_{80}	2_{96}	3_{85}
2D-3-3-tog	3	3_{57}	3_{100}	3_{100}	3_{100}	3_{100}	7_{37}	3_{94}	3_{86}	3_{100}
2D-3-3-ov	3	3_{48}	3_{100}	3_{100}	2_{100}	3_{100}	6_{53}	3_{79}	3_{87}	3_{100}
2D-3-3-strov	3	3_{32}	2_{100}	2_{100}	2_{100}	2_{100}	6_{67}	3_{43}	3_{81}	3_{100}
bensaid	3	7_{23}	5_{26}	2_{46}	2_{100}	6_{25}	5_{33}	10_{62}	2_{42}	10_{64}

Table A.15: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	5_{100}	10_{74}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{88}	10_{87}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{72}	10_{65}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{85}	10_{41}
sonar	2	2_{100}	2_{97}	2_{100}	2_{96}	2_{100}	2_{98}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	4_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	3_{89}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{68}	10_{59}	10_{47}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{100}

Table A.16: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 10% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{78}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{71}
glass	6	4_{54}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{78}
ionosphere	2	10_{80}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
iris	3	5_{43}	2_{100}	2_{100}	2_{100}	2_{100}	3_{77}	3_{100}	3_{56}	10_{72}
iris-bezdek	3	5_{38}	2_{100}	2_{100}	2_{100}	2_{100}	3_{79}	3_{100}	3_{100}	10_{75}
sonar	2	10_{100}	2_{99}	2_{99}	2_{99}	2_{100}	2_{100}	2_{97}	2_{71}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	9_{35}	3_{100}	3_{100}	9_{46}	3_{100}	3_{100}	5_{84}	3_{100}	10_{56}
wine	3	10_{100}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}

Table A.17: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	2_{87}	15_{26}	2_{100}	15_{28}	17_{35}	15_{32}	17_{98}	2_{100}	2_{68}	15_{35}	$17_{37}(15_{30})$	$15_{29}/17_{29}$
10D-10	10	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	2_{100}	3_{98}	3_{98}	14_{18}	$14_{14}/16_{14}$
3D-5-sep	5	2_{100}	2_{100}	2_{100}	5_{82}	5_{64}	5_{57}	10_{100}	2_{100}	5_{65}	5_{75}	5_{52}	5_{36}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	6_{40}	6_{32}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{37}	6_{28}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	6_{50}	2_{100}	6_{33}	7_{35}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	6_{49}	6_{44}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	7_{28}	8_{42}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	9_{41}	3_{100}	3_{100}	3_{90}
2D-3-2-tog	3	2_{93}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	8_{33}	3_{100}	3_{100}	3_{89}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	8_{40}	2_{100}	3_{100}	3_{80}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{74}	3_{35}
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{59}	3_{100}	3_{100}	3_{84}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{60}	3_{100}	3_{100}	3_{96}
2D-3-3-strov	3	2_{100}	2_{98}	2_{100}	3_{100}	3_{100}	2_{100}	10_{100}	2_{100}	2_{94}	2_{96}	$10_{35}(3_{21})$	10_{63}
bensaid	3	2_{100}	4_{35}	2_{100}	10_{33}	3_{48}	3_{48}	10_{86}	2_{100}	4_{57}	10_{61}	10_{45}	10_{38}

Table A.18: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 25%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	15_{26}	$12_{34}/13_{34}$	$12_{34}/13_{34}$	$12_{34}/13_{34}$	15_{27}	2_{99}	15_{32}	12_{27}	17_{47}
10D-10	10	2_{100}	3_{98}	3_{100}	2_{100}	3_{27}	10_{27}	12_{15}	2_{100}	16_{26}
3D-5-sep	5	5_{56}	2_{100}	2_{100}	2_{100}	2_{99}	5_{68}	5_{77}	2_{48}	5_{27}
3D-5-ov	5	5_{40}	2_{100}	2_{100}	2_{100}	2_{100}	5_{65}	5_{60}	2_{47}	$5_{34}/6_{34}$
3D-5-strov	5	5_{42}	2_{100}	2_{100}	2_{100}	2_{100}	5_{56}	2_{82}	2_{64}	$6_{34}(5_{29})$
3D-5-h-sep	5	5_{41}	2_{100}	2_{100}	2_{100}	2_{100}	6_{47}	5_{59}	2_{98}	6_{24}
3D-5-h-ov	5	5_{34}	2_{100}	2_{100}	2_{100}	2_{100}	5_{52}	5_{42}	2_{99}	6_{24}
3D-5-h-strov	5	5_{34}	2_{100}	2_{100}	2_{100}	2_{100}	$2_{37}(5_{34})$	2_{100}	2_{100}	7_{29}
2D-3-sep	3	3_{42}	3_{100}	3_{100}	3_{100}	3_{100}	10_{39}	3_{62}	2_{65}	3_{52}
2D-3-2-tog	3	3_{38}	2_{100}	2_{100}	2_{100}	3_{100}	10_{50}	3_{53}	2_{79}	3_{65}
2D-3-2-ov	3	$4_{33}(3_{31})$	2_{100}	2_{100}	2_{100}	3_{100}	10_{40}	3_{50}	2_{94}	3_{52}
2D-3-2-strov	3	3_{41}	2_{100}	2_{100}	2_{100}	2_{97}	10_{53}	3_{36}	2_{99}	3_{30}
2D-3-3-tog	3	3_{39}	3_{100}	3_{100}	3_{100}	3_{100}	8_{37}	3_{53}	2_{60}	3_{50}
2D-3-3-ov	3	3_{39}	3_{100}	3_{100}	2_{93}	3_{100}	10_{33}	3_{43}	$2_{50}(3_{47})$	3_{45}
2D-3-3-strov	3	$6_{17}/7_{17}$	2_{100}	2_{100}	2_{100}	2_{66}	8_{38}	6_{20}	2_{60}	10_{38}
bensaid	3	4_{31}	3_{48}	$2_{52}(3_{48})$	2_{100}	$5_{24}/6_{24}$	$9_{26}/10_{26}$	10_{49}	2_{62}	10_{48}

Table A.19: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{98}	9_{39}	9_{42}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	9_{76}	9_{73}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{90}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{44}	10_{40}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{35}	10_{50}
sonar	2	2_{100}	2_{98}	2_{100}	2_{95}	2_{100}	2_{99}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	3_{100}
wine-3D	3	2_{95}	3_{100}	2_{96}	3_{98}	3_{98}	3_{100}	10_{100}	2_{100}	3_{87}	3_{89}	10_{69}	10_{56}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	4_{100}	4_{100}

Table A.20: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 25% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{38}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{75}
glass	6	8_{34}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{92}
ionosphere	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{93}
iris	3	7_{40}	2_{100}	2_{100}	2_{100}	2_{100}	$5_{36}(3_{30})$	3_{95}	3_{100}	10_{48}
iris-bezdek	3	4_{44}	2_{100}	2_{100}	2_{100}	2_{100}	5_{55}	3_{99}	3_{66}	10_{57}
sonar	2	10_{100}	2_{99}	2_{99}	2_{99}	2_{100}	2_{100}	2_{97}	2_{63}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	8_{33}	3_{100}	3_{100}	3_{53}	3_{100}	3_{99}	3_{50}	3_{88}	10_{69}
wine	3	10_{99}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	4_{100}

Table A.21: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	2_{100}	$11_{16}/12_{16}$	2_{100}	$16_{17}(14_{15}/15_{15})$	2_{98}	16_{18}	17_{99}	2_{100}	2_{86}	14_{24}	17_{31}	17_{30}
10D-10	10	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	3_{98}	2_{100}	3_{98}	3_{98}	6_{57}	6_{51}
3D-5-sep	5	2_{100}	2_{77}	2_{100}	2_{44}	3_{72}	4_{37}	10_{100}	2_{100}	6_{30}	3_{53}	10_{31}	10_{30}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{99}	2_{95}	2_{90}	10_{100}	2_{100}	2_{100}	2_{98}	9_{24}	10_{28}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{36}	10_{35}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{99}	2_{100}	2_{100}	10_{100}	2_{100}	6_{33}	2_{100}	10_{44}	8_{29}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{54}	8_{35}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{41}	8_{45}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{80}	2_{100}	10_{46}	3_{100}	9_{27}	$5_{18}/9_{18}$
2D-3-2-tog	3	3_{58}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{91}	2_{100}	10_{58}	3_{100}	10_{49}	10_{17}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{98}	2_{100}	2_{100}	2_{100}	2_{100}	10_{58}	2_{100}	10_{42}	10_{31}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{40}	2_{100}	10_{51}	10_{40}
2D-3-3-tog	3	3_{100}	3_{100}	2_{97}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	$9_{40}/10_{40}$	3_{100}	10_{45}	10_{27}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	9_{44}	3_{100}	10_{63}	10_{57}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	2_{48}	3_{69}	2_{100}	10_{100}	2_{100}	$7_{25}/8_{25}$	2_{100}	10_{61}	10_{46}
bensaid	3	2_{82}	10_{32}	2_{100}	10_{42}	9_{23}	9_{32}	10_{77}	2_{100}	9_{27}	10_{54}	9_{22}	10_{23}

Table A.22: Cluster validity results of clusterings produced by OCSFCM on synthetic data sets with missing values 40%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	16_{17}	10_{22}	10_{22}	10_{22}	4_{46}	3_{55}	$16_{21}(15_{18})$	9_{16}	$16_{34}/17_{34}$
10D-10	10	2_{52}	3_{98}	3_{100}	3_{100}	3_{98}	7_{60}	6_{50}	2_{100}	17_{28}
3D-5-sep	5	6_{39}	2_{100}	2_{100}	2_{100}	2_{95}	6_{39}	$6_{38}(5_{34})$	2_{46}	10_{40}
3D-5-ov	5	5_{27}	2_{100}	2_{100}	2_{100}	2_{100}	5_{50}	5_{30}	2_{49}	10_{32}
3D-5-strov	5	7_{27}	2_{100}	2_{100}	2_{100}	2_{100}	5_{47}	2_{53}	2_{48}	10_{36}
3D-5-h-sep	5	5_{30}	2_{100}	2_{100}	2_{100}	2_{100}	6_{37}	$6_{41}(5_{39})$	2_{90}	10_{45}
3D-5-h-ov	5	$5_{27}(6_{26})$	2_{100}	2_{100}	2_{100}	2_{100}	5_{40}	$2_{29}(4_{28})$	2_{86}	10_{56}
3D-5-h-strov	5	5_{28}	2_{100}	2_{100}	2_{100}	2_{100}	$3_{23}/4_{23}$	2_{87}	2_{84}	10_{45}
2D-3-sep	3	3_{35}	3_{100}	3_{100}	3_{100}	3_{100}	10_{47}	3_{32}	2_{69}	10_{41}
2D-3-2-tog	3	3_{27}	2_{100}	2_{100}	2_{100}	3_{100}	10_{61}	3_{19}	2_{89}	10_{47}
2D-3-2-ov	3	4_{33}	2_{100}	2_{100}	2_{100}	3_{95}	10_{54}	$10_{18}(3_{16})$	2_{99}	10_{48}
2D-3-2-strov	3	3_{33}	2_{100}	2_{100}	2_{100}	2_{87}	10_{67}	10_{24}	2_{100}	10_{40}
2D-3-3-tog	3	3_{29}	3_{100}	3_{100}	3_{100}	3_{100}	10_{51}	6_{20}	2_{70}	10_{51}
2D-3-3-ov	3	3_{24}	2_{61}	2_{81}	2_{100}	3_{99}	10_{53}	9_{30}	2_{70}	10_{51}
2D-3-3-strov	3	5_{22}	2_{100}	2_{100}	2_{100}	2_{69}	10_{58}	10_{33}	2_{64}	10_{51}
bensaid	3	9_{27}	10_{19}	2_{52}	2_{100}	9_{30}	10_{35}	9_{28}	2_{73}	10_{30}

Table A.23: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	2_{100}	2_{100}	3_{85}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{97}	10_{56}	10_{58}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	$4_{25}(6_{22})$	$4_{32}(6_{22})$
ionosphere	2	2_{100}	2_{99}	2_{100}	2_{99}	2_{99}	2_{99}	10_{100}	2_{100}	2_{100}	10_{99}	2_{99}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	8_{30}	9_{33}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	9_{25}	9_{31}
sonar	2	2_{100}	2_{71}	2_{100}	2_{64}	2_{100}	2_{72}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	2_{69}	2_{69}
wine-3D	3	2_{89}	3_{63}	2_{100}	10_{27}	2_{70}	3_{74}	10_{100}	2_{100}	2_{63}	10_{66}	7_{25}	8_{30}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	3_{98}

Table A.24: Cluster validity results of clusterings produced by OCSFCM on real data sets with missing values 40% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{36}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{61}
glass	6	7_{22}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{85}	2_{100}	6_{22}
ionosphere	2	10_{99}	2_{100}	2_{100}	2_{99}	2_{100}	2_{100}	2_{99}	2_{100}	2_{99}
iris	3	4_{31}	2_{100}	2_{100}	2_{100}	2_{99}	5_{30}	3_{39}	3_{92}	10_{39}
iris-bezdek	3	10_{24}	2_{100}	2_{100}	2_{100}	2_{100}	3_{48}	3_{100}	3_{92}	10_{38}
sonar	2	10_{100}	2_{81}	2_{81}	$10_{32}(2_{24})$	2_{84}	2_{100}	2_{97}	2_{65}	2_{100}
wdbc	2	10_{87}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	10_{32}	3_{67}	3_{75}	9_{54}	3_{54}	3_{65}	7_{23}	3_{44}	10_{37}
wine	3	10_{42}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	4_{77}

Table A.25: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	15_{63}	15_{63}	2_{100}	15_{63}	15_{63}	15_{63}	17_{100}	2_{100}	15_{63}	15_{63}	15_{63}	15_{63}
10D-10	10	3_{99}	3_{99}	3_{99}	3_{99}	3_{99}	3_{99}	3_{99}	2_{100}	3_{99}	3_{99}	17_{34}	17_{42}
3D-5-sep	5	2_{100}	5_{82}	2_{100}	5_{82}	5_{82}	5_{82}	10_{100}	2_{100}	5_{82}	5_{82}	5_{82}	5_{82}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{93}	5_{93}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{84}	6_{57}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	6_{62}	2_{100}	5_{75}	5_{75}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{58}	5_{50}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	5_{63}	5_{50}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	8_{34}	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	$7_{37}(8_{35})$	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	7_{46}	2_{100}	3_{100}	3_{100}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{87}	3_{87}
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{98}	3_{100}	3_{100}	3_{100}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{43}	3_{100}	3_{100}	3_{100}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}
bensaid	3	2_{93}	4_{47}	2_{100}	10_{66}	4_{53}	5_{49}	10_{99}	2_{100}	4_{75}	10_{91}	10_{74}	10_{58}

Table A.26: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 10%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	15_{63}	15_{63}	15_{63}	15_{63}	15_{63}	15_{63}	15_{63}	15_{63}	15_{63}
10D-10	10	2_{100}	3_{99}	3_{100}	2_{100}	$14_{20}/15_{20}$	17_{43}	9_{26}	2_{99}	17_{51}
3D-5-sep	5	5_{71}	2_{100}	2_{100}	2_{100}	2_{100}	5_{82}	5_{82}	2_{100}	5_{82}
3D-5-ov	5	5_{92}	2_{100}	2_{100}	2_{100}	2_{100}	5_{93}	5_{93}	4_{71}	5_{93}
3D-5-strov	5	5_{57}	2_{100}	2_{100}	2_{100}	2_{100}	5_{84}	5_{84}	4_{71}	5_{84}
3D-5-h-sep	5	5_{42}	2_{100}	2_{100}	2_{100}	2_{100}	5_{75}	5_{75}	2_{100}	5_{74}
3D-5-h-ov	5	5_{33}	2_{100}	2_{100}	2_{100}	2_{100}	5_{58}	5_{58}	2_{100}	5_{58}
3D-5-h-strov	5	5_{40}	2_{100}	2_{100}	2_{100}	2_{100}	2_{64}	2_{100}	2_{100}	5_{64}
2D-3-sep	3	3_{45}	3_{100}	3_{100}	3_{100}	3_{100}	$7_{24}/8_{24}$	3_{100}	3_{100}	3_{100}
2D-3-2-tog	3	$3_{37}/4_{37}$	2_{100}	2_{100}	2_{100}	3_{100}	9_{32}	3_{100}	3_{100}	3_{100}
2D-3-2-ov	3	4_{50}	2_{100}	2_{100}	2_{100}	3_{100}	9_{37}	3_{100}	3_{98}	3_{100}
2D-3-2-strov	3	$4_{34}/2_{33}$	2_{100}	2_{100}	2_{100}	2_{100}	9_{38}	3_{87}	2_{82}	3_{87}
2D-3-3-tog	3	3_{60}	3_{100}	3_{100}	3_{100}	3_{100}	7_{44}	3_{97}	3_{100}	3_{100}
2D-3-3-ov	3	4_{42}	3_{100}	3_{100}	2_{100}	3_{100}	6_{85}	3_{72}	3_{100}	3_{100}
2D-3-3-strov	3	3_{32}	2_{100}	2_{100}	2_{100}	2_{100}	6_{68}	3_{36}	3_{100}	3_{100}
bensaid	3	7_{35}	6_{37}	5_{50}	2_{100}	6_{73}	5_{52}	10_{68}	6_{60}	10_{80}

Table A.27: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 10% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	5_{59}	10_{85}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{89}	10_{89}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{84}	10_{48}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{76}	10_{55}
sonar	2	2_{100}	2_{99}	2_{100}	2_{99}	2_{100}	2_{99}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	5_{100}	5_{100}
wine-3D	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{83}	10_{65}	8_{35}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{100}

Table A.28: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 10% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{85}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{78}
glass	6	10_{68}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{93}
ionosphere	2	10_{68}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{99}
iris	3	5_{47}	2_{100}	2_{100}	2_{100}	2_{100}	3_{80}	3_{99}	3_{99}	10_{85}
iris-bezdek	3	6_{31}	2_{100}	2_{100}	2_{100}	2_{100}	3_{75}	3_{100}	3_{100}	10_{82}
sonar	2	10_{100}	2_{99}	2_{99}	2_{99}	2_{100}	2_{100}	2_{99}	2_{62}	2_{100}
wdbc	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	9_{35}	3_{100}	3_{100}	3_{58}	3_{100}	3_{100}	5_{100}	3_{100}	10_{60}
wine	3	10_{100}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	3_{100}

Table A.29: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	14_{48}	14_{49}	2_{100}	14_{41}	14_{33}	14_{45}	17_{100}	2_{100}	$15_{29}/14_{29}$	14_{50}	$14_{29}/15_{28}$	15_{28}
10D-10	10	4_{75}	4_{75}	4_{75}	4_{75}	4_{75}	4_{75}	4_{75}	2_{100}	4_{75}	4_{75}	5_{31}	5_{33}
3D-5-sep	5	2_{100}	5_{83}	2_{100}	5_{83}	5_{83}	5_{83}	10_{100}	2_{100}	5_{65}	5_{83}	6_{41}	10_{24}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	6_{51}	10_{36}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{32}	10_{46}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	6_{61}	2_{100}	6_{44}	6_{44}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{39}	8_{51}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{38}	8_{64}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	10_{29}	3_{100}	3_{64}	$10_{37}(3_{29})$
2D-3-2-tog	3	2_{94}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	8_{31}	3_{100}	3_{100}	3_{78}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	7_{33}	2_{100}	3_{55}	3_{36}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{56}	$9_{28}/10_{28}$
2D-3-3-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{46}	3_{100}	3_{61}	3_{36}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	6_{53}	3_{100}	3_{65}	3_{50}
2D-3-3-strov	3	2_{100}	2_{66}	2_{100}	3_{100}	3_{100}	2_{100}	10_{100}	2_{100}	2_{56}	3_{77}	10_{67}	10_{82}
bensaid	3	2_{100}	4_{34}	2_{100}	10_{61}	3_{52}	3_{52}	10_{91}	2_{100}	4_{54}	10_{70}	9_{31}	9_{29}

Table A.30: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 25%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	14_{41}	14_{49}	14_{49}	14_{49}	14_{49}	14_{42}	14_{36}	14_{37}	17_{31}
10D-10	10	2_{100}	4_{75}	3_{100}	2_{100}	4_{75}	$16_{23}(15_{22})$	5_{59}	2_{100}	17_{45}
3D-5-sep	5	5_{72}	2_{100}	2_{100}	2_{100}	5_{83}	5_{71}	5_{83}	2_{67}	10_{36}
3D-5-ov	5	5_{76}	2_{100}	2_{100}	2_{100}	2_{100}	5_{90}	5_{90}	2_{74}	10_{38}
3D-5-strov	5	5_{73}	2_{100}	2_{100}	2_{100}	2_{100}	5_{88}	5_{88}	$2_{44}(4_{43})$	10_{41}
3D-5-h-sep	5	5_{52}	2_{100}	2_{100}	2_{100}	2_{100}	5_{57}	5_{68}	2_{100}	$9_{36}/10_{36}$
3D-5-h-ov	5	5_{36}	2_{100}	2_{100}	2_{100}	2_{100}	5_{58}	5_{58}	2_{100}	10_{58}
3D-5-h-strov	5	5_{49}	2_{100}	2_{100}	2_{100}	2_{100}	5_{65}	2_{100}	2_{100}	10_{55}
2D-3-sep	3	3_{39}	3_{100}	3_{100}	3_{100}	3_{100}	10_{42}	3_{29}	3_{100}	10_{58}
2D-3-2-tog	3	4_{31}	2_{100}	2_{100}	2_{100}	3_{100}	10_{57}	3_{28}	3_{100}	10_{41}
2D-3-2-ov	3	4_{33}	2_{100}	2_{100}	2_{100}	3_{100}	10_{43}	$7_{17}/9_{17}(3_{16}/6_{16})$	2_{63}	10_{55}
2D-3-2-strov	3	3_{30}	2_{100}	2_{100}	2_{100}	2_{100}	10_{55}	6_{20}	2_{100}	10_{63}
2D-3-3-tog	3	4_{51}	3_{100}	3_{100}	3_{100}	3_{100}	8_{38}	8_{24}	3_{92}	10_{41}
2D-3-3-ov	3	7_{23}	3_{100}	3_{100}	3_{67}	3_{100}	10_{48}	7_{25}	3_{100}	10_{58}
2D-3-3-strov	3	7_{21}	2_{100}	2_{100}	2_{100}	$3_{51}(2_{49})$	9_{35}	7_{21}	3_{100}	10_{70}
bensaid	3	4_{31}	3_{52}	3_{52}	2_{100}	7_{27}	10_{36}	10_{60}	2_{86}	10_{37}

Table A.31: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 25% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{91}	10_{62}	10_{65}
glass	6	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	8_{53}	8_{50}
ionosphere	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{72}	2_{100}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{51}	10_{61}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{47}	10_{70}
sonar	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{100}	3_{100}
wine-3D	3	2_{94}	3_{100}	2_{94}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	3_{100}	3_{83}	10_{70}	10_{54}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	10_{100}	3_{100}	2_{94}

Table A.32: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 25% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{58}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	10_{88}
glass	6	10_{38}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{51}	2_{100}	10_{69}
ionosphere	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	3_{65}
iris	3	6_{40}	2_{100}	2_{100}	2_{100}	2_{100}	6_{57}	6_{58}	3_{100}	10_{77}
iris-bezdek	3	$5_{43}(4_{42})$	2_{100}	2_{100}	2_{100}	2_{100}	5_{41}	3_{100}	3_{100}	10_{56}
sonar	2	10_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{59}	2_{100}
wdbc	2	10_{86}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}
wine-3D	3	$10_{27}(7_{26})$	3_{100}	3_{100}	3_{49}	3_{100}	3_{74}	7_{38}	3_{99}	10_{75}
wine	3	10_{91}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	4_{97}

Table A.33: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
3D-15	15	$9/10/12_{14}$	$12/13_{18}$	2_{100}	15_{22}	15_{24}	13_{22}	17_{100}	2_{100}	$15/16/17_{25}$	13_{26}	$16_{28}(15_{26})$	15_{28}
10D-10	10	4_{99}	4_{99}	4_{99}	4_{99}	4_{99}	4_{99}	4_{99}	2_{100}	4_{99}	4_{99}	4_{51}	4_{50}
3D-5-sep	5	2_{100}	5_{70}	2_{100}	5_{75}	5_{73}	5_{65}	10_{100}	2_{100}	$7_{34}(5_{32})$	5_{75}	10_{57}	10_{59}
3D-5-ov	5	2_{100}	2_{100}	2_{100}	$2_{53}(5_{41})$	2_{57}	4_{57}	10_{100}	2_{100}	2_{90}	5_{57}	10_{52}	10_{67}
3D-5-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{66}	10_{73}
3D-5-h-sep	5	2_{100}	2_{100}	2_{100}	2_{81}	2_{100}	2_{100}	10_{100}	2_{100}	7_{49}	2_{100}	10_{67}	10_{67}
3D-5-h-ov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{71}	10_{55}
3D-5-h-strov	5	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{66}	10_{42}
2D-3-sep	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	10_{49}	3_{100}	10_{53}	10_{45}
2D-3-2-tog	3	3_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	10_{51}	3_{100}	10_{50}	10_{42}
2D-3-2-ov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	10_{49}	2_{100}	10_{60}	10_{69}
2D-3-2-strov	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	9_{43}	2_{100}	10_{63}	10_{61}
2D-3-3-tog	3	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	10_{40}	3_{100}	10_{65}	10_{59}
2D-3-3-ov	3	2_{100}	3_{100}	2_{100}	3_{100}	3_{100}	3_{100}	10_{100}	2_{100}	9_{44}	3_{100}	10_{53}	10_{64}
2D-3-3-strov	3	2_{100}	2_{100}	2_{100}	10_{67}	3_{100}	2_{100}	10_{100}	2_{100}	8_{30}	10_{72}	10_{67}	10_{69}
bensaid	3	10_{27}	10_{41}	2_{96}	10_{43}	9_{25}	$9_{28}/10_{28}$	10_{82}	2_{100}	9_{24}	10_{53}	2_{25}	8_{23}

Table A.34: Cluster validity results of clusterings produced by NPSFCM on synthetic data sets with missing values 40%(Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
3D-15	15	$16_{20}(13_{18})$	11_{29}	11_{29}	11_{29}	9_{16}	15_{19}	15_{24}	$12_{17}(10_{16})$	17_{32}
10D-10	10	2_{70}	4_{99}	3_{100}	3_{100}	4_{99}	4_{74}	4_{92}	2_{57}	17_{22}
3D-5-sep	5	5_{52}	2_{83}	2_{87}	2_{90}	5_{75}	$7_{29}(5_{28})$	5_{69}	$2_{38}(4_{36})$	10_{70}
3D-5-ov	5	5_{51}	2_{100}	2_{100}	2_{100}	2_{100}	5_{49}	5_{73}	2_{42}	10_{62}
3D-5-strov	5	$7_{29}(6_{28})$	2_{100}	2_{100}	2_{100}	2_{100}	5_{43}	5_{50}	4_{61}	10_{70}
3D-5-h-sep	5	$5_{35}/6_{35}$	2_{100}	2_{100}	2_{100}	2_{100}	7_{41}	5_{47}	2_{100}	10_{79}
3D-5-h-ov	5	5_{38}	2_{100}	2_{100}	2_{100}	2_{100}	6_{48}	5_{53}	2_{100}	10_{75}
3D-5-h-strov	5	5_{27}	2_{100}	2_{100}	2_{100}	2_{100}	5_{51}	5_{38}	2_{100}	10_{69}
2D-3-sep	3	4_{37}	3_{100}	3_{100}	3_{100}	3_{100}	10_{47}	10_{34}	10_{46}	10_{58}
2D-3-2-tog	3	4_{33}	2_{100}	2_{100}	2_{100}	3_{100}	10_{67}	10_{46}	3_{100}	10_{57}
2D-3-2-ov	3	4_{38}	2_{100}	2_{100}	2_{100}	3_{100}	10_{74}	10_{48}	2_{79}	10_{69}
2D-3-2-strov	3	$4_{22}(3_{21})$	2_{100}	2_{100}	2_{100}	2_{100}	10_{81}	10_{56}	2_{100}	10_{66}
2D-3-3-tog	3	$6_{20}(7_{19})$	3_{100}	3_{100}	3_{100}	3_{100}	10_{62}	10_{54}	3_{100}	10_{73}
2D-3-3-ov	3	8_{23}	3_{100}	3_{100}	2_{100}	3_{100}	10_{67}	10_{58}	3_{100}	10_{57}
2D-3-3-strov	3	10_{38}	2_{100}	2_{100}	2_{100}	5_{86}	10_{70}	10_{63}	3_{100}	10_{64}
bensaid	3	6_{19}	10_{23}	2_{48}	2_{100}	10_{35}	10_{38}	10_{39}	2_{58}	9_{32}

Table A.35: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 40% (Part I).

Data Set	c_{real}	V_{PC}	V_{NPC}	V_{PE}	V_{NPE}	V_{KKLL}	V_{OSI_S}	V_{OSI_A}	V_{OSI_L}	$V_{OSI_{H_\gamma}}$	$V_{OSI_{D_\gamma}}$	V_{FHV}	V_{PD}
ecoli	8	2_{100}	2_{95}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{90}	10_{70}	10_{71}
glass	6	2_{100}	2_{100}	2_{100}	2_{67}	2_{100}	2_{100}	10_{98}	2_{100}	2_{100}	2_{98}	6_{42}	6_{40}
ionosphere	2	2_{99}	2_{99}	2_{100}	2_{99}	2_{99}	2_{99}	10_{100}	2_{100}	2_{100}	10_{80}	2_{51}	2_{93}
iris	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	7_{41}	9_{29}
iris-bezdek	3	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	10_{26}	10_{32}
sonar	2	2_{100}	2_{67}	2_{100}	2_{65}	2_{100}	2_{71}	10_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}
wdbc	2	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	10_{100}	2_{100}	2_{100}	2_{100}	3_{93}	2_{95}
wine-3D	3	2_{58}	3_{87}	2_{94}	10_{49}	2_{56}	3_{80}	10_{100}	2_{100}	3_{88}	10_{70}	10_{23}	7_{30}
wine	3	2_{100}	3_{100}	2_{100}	3_{100}	2_{100}	3_{100}	10_{100}	2_{100}	2_{100}	3_{98}	3_{93}	3_{65}

Table A.36: Cluster validity results of clusterings produced by NPSFCM on real data sets with missing values 40% (Part II).

Data Set	c_{real}	V_{FS}	V_{XB}	V_{Kwon}	V_{TSS}	V_{BH}	V_{ZLE}	V_{BWS}	V_{PCAES}	V_{PNC}
ecoli	8	10_{50}	3_{100}	3_{100}	3_{100}	2_{100}	2_{99}	3_{97}	3_{99}	10_{78}
glass	6	5_{49}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	5_{71}	2_{86}	10_{53}
ionosphere	2	10_{51}	2_{100}	2_{100}	2_{99}	2_{100}	2_{99}	2_{99}	2_{100}	2_{56}
iris	3	5_{33}	2_{100}	2_{100}	2_{100}	2_{100}	5_{33}	4_{43}	3_{100}	9_{32}
iris-bezdek	3	5_{24}	2_{100}	2_{100}	2_{100}	2_{100}	5_{35}	3_{100}	3_{99}	10_{39}
sonar	2	10_{100}	2_{79}	2_{79}	$4_{21}(6_{20})$	2_{87}	2_{100}	2_{94}	2_{60}	2_{100}
wdbc	2	10_{91}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{100}	2_{92}
wine-3D	3	10_{45}	3_{74}	3_{84}	9_{49}	3_{88}	$9_{20}(3_{19})$	10_{27}	3_{37}	10_{45}
wine	3	10_{27}	3_{100}	3_{100}	3_{100}	2_{100}	2_{100}	3_{100}	3_{100}	5_{59}

B

List of Own Publications

HCC12 Ludmila Himmelspach, João Paulo Carvalho, and Stefan Conrad. On Cluster Validity for Fuzzy Clustering of Incomplete Data. In *Proceedings of the 6th International Conference on Scalable Uncertainty Management, SUM 2012, Marburg, Germany, September 17-19, 2012*, Lecture Notes in Computer Science, vol 7520, Springer, pages 612 – 618, 2012.

Ludmila Himmelspach's contributions: idea, implementation, design and conduction of experiments, writing of the manuscript

HHC11 Ludmila Himmelspach, Daniel Hommers, and Stefan Conrad. Cluster Tendency Assessment for Fuzzy Clustering of Incomplete Data. In *Proceedings of 6th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT – 2011)*, pages 290 – 297. Atlantis Press, 2011.

Ludmila Himmelspach's contributions: idea, writing of the manuscript

HC11 Ludmila Himmelspach and Stefan Conrad. Density-Based Clustering using Fuzzy Proximity Relations. In *Proceedings of the 2011 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, pages 1 – 6, 2011.

Ludmila Himmelspach's contributions: idea, implementation, design and conduction of experiments, writing of the manuscript

HC10b Ludmila Himmelspach and Stefan Conrad. Fuzzy Clustering of Incomplete Data Based on Cluster Dispersion. In *Proceedings of the 13th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, IPMU 2010, Dortmund, Germany, June 28 – July 2, 2010*, Lecture Notes in Computer Science, vol 6178, Springer, pages 59 – 68, 2010.

Ludmila Himmelspach's contributions: idea, implementation, design and conduction of experiments, writing of the manuscript

HC10a Ludmila Himmelspach and Stefan Conrad. Clustering Approaches for Data with Missing Values: Comparison and Evaluation. In *Proceedings of the Fifth IEEE International Conference an Digital Information Management, ICDIM 2010, July 5 – 8, 2010, Lakehead University, Thunder Bay, Canada*, pages 19 – 28, 2010.

Ludmila Himmelspach's contributions: idea, implementation, design and conduction of experiments, writing of the manuscript

Him09 Ludmila Himmelspach. Vergleich von Strategien zum Clustern von Daten mit fehlenden Werten. In *Proceedings of the 21. GI-Workshop on Foundations of Databases (Grundlagen von Datenbanken), Rostock-Warnemünde, Mecklenburg-Vorpommern, Germany, June 2-5, 2009*, pages 129 – 133, 2009.

Ludmila Himmelspach's contributions: idea, implementation, design and conduction of experiments, writing of the manuscript