DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

CONSTRUCTION AND ANALYSIS OF CLUSTERING ALGORITHMS BASED ON FUZZY RELATIONS AND THEIR APPLICATIONS TO EEG DATA

by
Gözde ULUTAGAY

July, 2009
İZMİR

CONSTRUCTION AND ANALYSIS OF CLUSTERING ALGORITHMS BASED ON FUZZY RELATIONS AND THEIR APPLICATIONS TO EEG DATA

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfilment of the Requirements for the Degree of Doctor of Philosophy
in Statistics Program

by
Gözde ULUTAGAY

July, 2009
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"CONSTRUCTION AND ANALYSIS OF CLUSTERING ALGORITHMS BASED ON FUZZY RELATIONS AND THEIR APPLICATIONS TO EEG DATA"** completed by **GÖZDE ULUTAGAY** under supervision of **PROF. DR. EFENDİ NASİBOĞLU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosphy.

....

Prof. Dr. Efendi NASİBOĞLU

Supervisor

....	....
Prof. Dr. Serdar KURT	Prof. Dr. Tatyana YAKHNO
Thesis Committee Member	Thesis Committee Member
....	....
Prof. Dr. Novruz ALLAHVERDİ	Assoc. Prof. Dr. C. Cengiz ÇELİKOĞLU
Examining Committee Member	Examining Committee Member

______________________________-

Prof. Dr. Cahit HELVACI
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to express my deep and sincere gratitude to my supervisor, Prof. Dr. Efendi NASİBOĞLU, for his wisdom, support, advice, and patience during the whole period of my dissertation. His encouragement and supervision lead me to a good start of my academic career. It has been a privilege for me to work with him.

I would like to thank Prof. Dr. Serdar KURT for his confidence in me and for providing constructive suggestions during my dissertation process. I also owe thanks to my dissertation committee member, Prof. Dr. Tatyana YAKHNO for her helpful comments.

I would like to thank my dear roommate Alper VAHAPLAR for sharing all his experience and time with me generously. I would also like to thank to my dear friends Övgü KINAY, Burcu ÜÇER, and Selma GÜRLER for their support in this process.

I would like to thank my dear parents, Nevin & Coşgun TEKİN, who have always been with me whenever I am in need, for their encouragement and trust in me. I also owe thanks to my grandparents, Perihan & Zeki YÜREKLİTÜRK, for their love, care, and prays through the years and for their confidence in me.

I would like to express my great appreciation to my invaluable husband, Murat ULUTAGAY, for his infinite patience, support, and understanding. His love has always given me unlimited strength. Finally, I would like to extend my special thanks to my little princess, Zeynep Naz ULUTAGAY, who has enriched my life more than I could have ever imagined, and who shows great understanding and patience more than expected from a six-year old child.

Gözde ULUTAGAY

CONSTRUCTION AND ANALYSIS OF CLUSTERING ALGORITHMS BASED ON FUZZY RELATIONS AND THEIR APPLICATIONS TO EEG DATA

ABSTRACT

In this work, fundamentally two algorithms have been proposed. The first one is the NRFJP (Noise-Robust FJP) algorithm which is a robust version of the known fuzzy neighborhood-based FJP (Fuzzy Joint Points) clustering algorithm. In the NRFJP algorithm each point for which certain eps1 fuzzy neighborhood cardinality is smaller than certain eps2 threshold is perceived as noise. Moreover, in case eps2 is zero, the sensitivity of the NRFJP through noises is turned off, consequently NRFJP algorithm transforms into FJP algorithm.

The second algorithm is the FN-DBSCAN (Fuzzy Neighborhood DBSCAN) algorithm which is a mixture of FJP and density-based DBSCAN (Density Based Spatial Clustering Applications with Noise) algorithms. In the study, the effects of fuzzy neighborhood relation in density-based clustering have been investigated. Besides being a more general algorithm, the FN-DBSCAN algorithm transforms into the DBSCAN algorithm when the crisp neighborhood function is used.

The modified version of the FN-DBSCAN algorithm has been developed so as to apply cluster analysis to BIS data. As a result of the computational experiments, it has been observed that FN-DBSCAN based approach gives closer results to the expert's opinion than the well-known FCM (Fuzzy c-means) clustering algorithm.

The codes for the proposed algorithms, NRFJP, FN-DBSCAN and the modified version of FN-DBSCAN to analyze BIS data, have been developed in Borland C++ Builder SDK and they have been designed as an integrated software system.

Keywords: Fuzzy relation, clustering, NRFJP, FN-DBSCAN, EEG, BIS index.

BULANIK İLİŞKİLERE DAYALI KÜMELEME ALGORİTMALARININ OLUŞTURULMASI, ANALİZİ VE EEG VERİLERİNE UYGULANMASI

ÖZ

Bu çalışmada temel olarak iki algoritma önerilmektedir. Birincisi, bulanık komşuluğa dayalı FJP (Fuzzy Joint Points) algoritmasının sapan değerlere dayanıklı versiyonu olan NRFJP (Noise-Robust FJP) algoritmasıdır. NRFJP algoritmasında, belirli bir eps1 için, bulanık komşuluk kardinalitesi eps2 eşiğinden düşük olan her bir nokta sapan değer olarak ele alınır. Algoritmada, eps2 değeri sıfır olarak seçildiğinde, NRFJP algoritmasının sapan değerlere karşı duyarlılığı yok olur ve NRFJP algoritması FJP algoritmasına dönüşür.

Çalışmada önerilen ikinci algoritma ise, FJP ve yoğunluğa dayalı DBSCAN (Density Based Spatial Clustering Applications with Noise) algoritmalarının karışımı olan FN-DBSCAN (Fuzzy Neighborhood DBSCAN) algoritmasıdır. Çalışmada, yoğunluğa dayalı kümelemede kullanılan bulanık komşuluk ilişkilerinin etkisi incelenmiştir. FN-DBSCAN daha genel bir algoritma olmasının yanında, klasik komşuluk fonksiyonu kullanıldığında DBSCAN algoritmasına dönüşmektedir.

BIS verilerine kümeleme analizi uygulamak için FN-DBSCAN algoritmasının modifiye edilmiş versiyonu geliştirilmiştir. Yapılan hesaplama deneyleri sonucunda, iyi bilinen FCM (Fuzzy c-means) kümeleme algoritmasına kıyasla, FN-DBSCAN temelli yaklaşımın uzman görüşüne daha yakın sonuçlar verdiği gözlenmiştir.

Önerilen NRFJP, FN-DBSCAN ve FN-DBSCAN temelinde BIS verilerinin analizi için geliştirilmiş algoritmanın Borland C++ Builder programlama dilinde kodları yazılmış ve entegre bir yazılım sistemi halinde tasarlanmıştır.

Anahtar Sözcükler: Bulanık ilişki, kümeleme, FCM, DBSCAN, FJP, NRFJP, FN-DBSCAN, EEG, BIS indeksi.

CONTENTS

Page

Ph.D. THESIS EXAMINATION RESULT FORM ii

ACKNOWLEDGMENTS iii

ABSTRACT iv

ÖZ v

CHAPTER ONE - INTRODUCTION 1

CHAPTER TWO - PRELIMINARIES OF CLUSTERING ALGORITHMS 4

2.1 Classification of Major Clustering Algorithms 4

2.1.1 Partitioning Methods 6

2.1.2 Hierarchical Methods 6

2.1.3 Density-Based Methods 7

2.1.4 Grid-Based Methods 8

2.1.5 Model-Based Methods 8

2.1.6 Fuzzy Clustering 9

2.2 FCM Algorithm 12

2.2.1 Initialization of Clusters 12

2.2.2 Cluster Validity 12

2.3 DBSCAN Algorithm 13

CHAPTER THREE - FUZZY RELATIONS 18

3.1 Crisp Relations and Their Properties 18

3.1.1 Properties of Relation on a Single Set 19

3.2 Fuzzy Relations and Their Properties 22

3.2.1 Characteristics of Fuzzy Relation 27

3.2.2 Classification of Fuzzy Relation 29

3.2.3 Dissimilitude Relation 32

CHAPTER FOUR - FUZZY NEIGHBORHOOD-BASED CLUSTERING ALGORITHMS 33

4.1 FJP Algorithm 33

4.1.1 FJP Cluster Validity Index 41

4.1.2 Analysis of Clusters' Structure in FJP Clustering 43

4.2 NRFJP Algorithm 49

4.2.1 Adjusting the Optimal Values of the Parameters of NRFJP 51

4.3 FN-DBSCAN Algorithm 54

CHAPTER FIVE - DATA COLLECTION TECHNIQUES 71

5.1 Electroencephalography 71

5.2 Bispectral Index 75

CHAPTER SIX - BIS-CLUSTERING AND APPLICATIONS 78

6.1 Determining BIS Stages by FCM-based algorithm 78

6.2 Determining BIS Stages by FN-DBSCAN-based algorithm 79

6.3 Experimental Results 81

CHAPTER SEVEN - SOFTWARE FOR FUZZY NEIGHBORHOOD BASED CLUSTERING 97

7.1 Software of NRFJP Algorithm 98

7.1.1 Forms 98

7.1.2 Functional Modules 103

7.1.3 Informative Components 103

7.2 Software of FN-DBSCAN Algorithm 104

7.2.1 Forms 105

7.2.2 Functional Modules 109

7.2.3 Informative Components 110

7.3 Software of FN-DBSCAN Algorithm for EEG 111

7.3.1 Forms 111

7.3.2 Functional Modules 115

7.3.3 Informative Components 116

CHAPTER EIGHT - CONCLUSION 117

REFERENCES 120

CHAPTER ONE
INTRODUCTION

Brains do not reason as computers do. Computers reason in clear steps with statements that are black or white. They reason with strings 0s or 1s. Humans reason with the vague terms of common sense as in “The air is *cool*” or “The speed is *fast*” or “He is *young*”. These fuzzy or gray facts are true only to some degree between 0 or 1 and they are false to some degree. Brains work with these fuzzy patterns with ease and computers may not work with them at all. Fuzzy logic tries to change that.

The key idea of fuzziness comes from the multivalued logic of the 1920s: *Everything is a matter of degree.* A statement of fact like “The sky is *blue*” or “The angle is *small*” does not have a binary truth value. It has a *vague* or *fuzzy* truth value between 0 and 1. And so does its negation “The sky is *not blue*.” So the sky is both blue and not blue to some degree. This simple point of fact violates the either-or laws of logic that extend from the first formal logic of ancient Greece to the foundations of modern math and science.

Fuzzy logic builds gray truth into complex schemes of formal reasoning. It is a new branch of machine intelligence that tries to make computers reason with our gray common sense. The earlier uses of the term *fuzzy logic* were the same as *continuous truth* or *vagueness*. It meant matters of degree and gray borders and thus breaking the either-or law of binary logic. Today, fuzzy logic refers to a fuzzy system or mapping from input to output that depends on fuzzy rules. The rules in turn depend on fuzzy sets or vague concepts like *cool air* or *blue sky* or *small angle* and these terms depend on fuzzy degrees of truth or set membership. Fuzzy logic means reasoning with vague concepts. In practice it can mean computing with words.

Fuzziness began as vagueness in the late nineteenth century. Pragmatist philosopher Charles Sanders Peirce seems the first logician to have dealt with vagueness (Peirce, 1931). Logician Bertrand Russell first identified vagueness at the level of symbolic logic (Russell, 1923). In the 1920s, logician Jan Lukasiewicz worked out the first fuzzy or multivalued logic (Lukasiewicz, 1970).

In 1965 Lotfi A. Zadeh, from the University of California at Berkeley, published the landmark paper “Fuzzy Sets” (Zadeh, 1965). This paper first used the word *fuzzy* to mean “vague” in the technical literature. The name *fuzzy* has not only persisted but largely replaced the prior term *vague* (Zadeh, 1987). Zadeh’s 1965 paper applied Lukasiewicz’s logic to each object in a set to work out a complete fuzzy set algebra and to extend the convex separation theorem of pattern recognition.

Since Lotfi A. Zadeh (1965) introduced the concept of fuzzy sets that produced the idea of allowing to have membership functions to all clusters, fuzzy clustering has been widely studied and applied in a variety of substantial areas. In general, the process of grouping a set of objects into classes of similar objects is called clustering (Hartigan, 1975). By clustering, one can identify dense and sparse regions, and therefore, discover overall distribution patterns and interesting correlations among data attributes (Kaufmann & Rousseeuw, 1990). As it is well-known, clustering has its roots in many areas, including statistics, data mining, biology, image processing, machine learning, *etc.*

The main subject of the study is to analyze and evaluate new clustering algorithms based on the fuzzy neighborhood relations. As a real-world application, the algorithms have been applied to BIS (bispectral index) data which are recorded by using EEG (electroencephalography).

The rest of this dissertation work is as follows: In the second chapter, preliminaries about cluster analysis and a categorization of major clustering algorithms are handled. Also, among various clustering methods, FCM (Fuzzy c-means) and DBSCAN (Density Based Spatial Clustering Applications with Noise) algorithms are explained.

In the third chapter, some basic concepts of relations are given and they are all investigated in view of both crisp and fuzzy situations.

In the fourth chapter, fuzzy neighborhood-based clustering methods, which form the basics of the dissertation, are investigated. First of all, Fuzzy Joints Points (FJP) algorithm and some of its basic concepts are explained since the two proposed methods are based on the FJP algorithm. Then, Noise-Robust FJP (NRFJP) algorithm, which is a modified form of FJP algorithm to handle noise points, is proposed and an entropy-based method to adjust one of its parameters is discussed. Then, the second proposed method, Fuzzy-Neighborhood DBSCAN (FN-DBSCAN) which is a mixture of fuzzy relation-based FJP and fast-running DBSCAN algorithms is explained in detail.

In the fifth chapter, in order to form a basis for the real-world application, some basic notions of two of the data collection techniques, electroencephalography (EEG) and Bispectral Index (BIS) are mentioned.

In the sixth chapter, in order to handle the problem of determining BIS stages for 21 people, by modifying the mentioned algorithms according to BIS data, FCM-based and FN-DBSCAN-based approaches are explained and compared both analytically and graphically.

In the seventh chapter, a software “The FJP Family”, coded in Borland C++ Builder 6.0 SDK, for fuzzy neighborhood-based clustering methods is introduced and some examples are given.

Finally, conclusions are stated in the last chapter.

CHAPTER TWO
PRELIMINARIES OF CLUSTERING ALGORITHMS

Clustering and classification tasks are among the most important problems in modern data mining technologies used in processing large databases (Han & Kamber, 2001; Larose, 2005). Clustering analyzes data objects without consulting a known class label different from classification. In general, the class labels are not present in the training data simply because they are not known to begin with. Clustering can be used to generate such labels. The objects are clustered or grouped based on the principle of maximizing the intra-class similarity and minimizing the interclass similarity. That is, clusters of objects are formed so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters. Each cluster that is formed can be viewed as a class of objects, from which rules can be derived (Grabmaier & Rudolph, 2002).

2.1 Classification of Major Clustering Algorithms

In general, major clustering methods can be classified into the categories (Han & Kamber, 2001):

- Partitioning methods,
- Hierarchical methods,
- Density-based methods,
- Grid-based methods,
- Model-based methods.

Some clustering algorithms integrate the ideas of several clustering methods, so that it is difficult to classify a given algorithm as uniquely belonging to only one clustering method category. Furthermore, some applications may have clustering criteria that require the integration of several clustering techniques. A more detailed relationship between these categories are given in Figure 2.1.

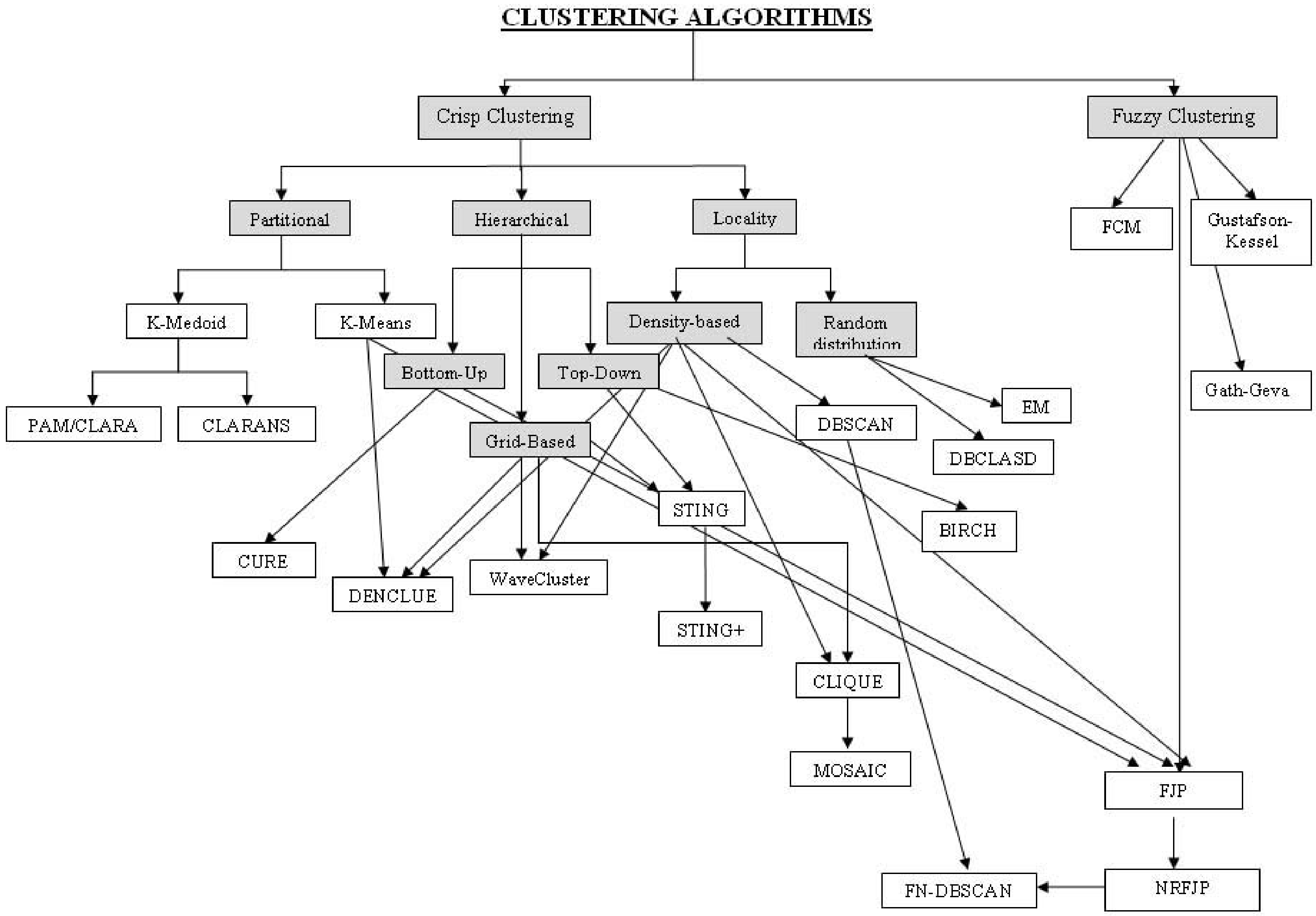


Figure 2.1 Categorization of clustering algorithms.

2.1.1 Partitioning Methods

Partitioning methods aim to directly obtain a single partition of the collection of items into clusters. Many of these methods are based on the iterative optimization of a criterion function reflecting the “agreement” between the data and the partition. Methods using the squared error rely on the possibility to represent each cluster by a prototype and attempt to minimize a cost function that is the sum over all the data items of the squared distance between the item and the prototype of the cluster it is assigned to. In general, the prototypes are the cluster centroids, as in the popular k-means algorithm (MacQueen, 1967). Several solutions were put forward for cases where a centroid cannot be defined, such as the k-medoid method (Kaufmann & Rousseeuw, 1990), where the prototype of a cluster is an item that is “central” to the cluster, or the k-modes method (Huang, 1997) that is an extension to categorical data.

The above-mentioned heuristic clustering methods work well for finding spherical-shaped clusters in small to medium-sized databases. To find clusters with complex shapes and for clustering very large data sets, partitioning-based methods need to be extended.

2.1.2 Hierarchical Methods

Hierarchical methods aim to obtain a hierarchy of clusters, called *dendrogram*, that shows how the clusters are related to each other. These methods proceed either by iteratively merging small clusters into larger ones (agglomerative algorithms, by far the most common) or by splitting large clusters (divisive algorithms). A partition of the data items can be obtained by cutting the dendrogram at a desired level. Agglomerative algorithms need criteria for merging small clusters into larger ones. Most of the criteria concern the merging of pairs of clusters (thus producing binary trees) and are variants of the classical single-link (Sneath & Sokal, 1973), complete-link (King, 1967) or minimum-variance criteria (Ward, 1963; Murtagh, 1984). The use of the single-link criterion can be related to density-based methods but often produces upsetting effects: clusters that are “linked” by a “line” of items cannot be separated or most items are individually merged to one (or a few) cluster(s). The use of the complete-link or of the minimum-variance criterion relates more to squared error methods.

Hierarchical methods suffer from the fact that once a step (merge or split) is done, it can never be undone. This rigidity is useful in that it leads to smaller computation costs by not worrying about a combinatorial number of different choices. However, a major problem of such techniques is that they cannot correct erroneous decisions. There are two approaches to improving the quality of hierarchical clustering: (1) perform careful analysis of object “linkages” at each hierarchical partitioning, such as in CURE (Guha et al., 1998) and Chameleon (Karypis et al., 1999), or (2) integrate hierarchical agglomeration and iterative relocation by first using a hierarchical agglomerative algorithm and then refining the result using iterative relocation, as in BIRCH (Zhang et al., 1996).

2.1.3 Density-Based Methods

Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes. Other clustering methods have been developed based on the notion of density. These methods consider that clusters are dense sets of data items separated by less dense regions; clusters may have arbitrary shape and data items can be arbitrarily distributed. Many methods, such as DBSCAN (Brecheisen et al., 2003) (further improved in (Brecheisen et al., 2003; Daszykowski et al., 2004)), rely on the study of the density of items in the neighborhood of each item. DBSCAN (Density-Based Spatial Clustering Applications with Noise) is typical density-based method that grows clusters according to a density threshold (Ester et al., 1996). DBSCAN is a kind of clustering algorithm based on intra-cluster densities. In this algorithm, distance queries is made for each point in data set for pre-determined ε value and it is investigated whether the points in ε-neighborhood of the point is larger than the *MinPts* value or not . It is possible to form a set with points that have values larger than *MinPts* and for each element of this set, complex-shaped cluster is obtained by repeating the same process.

There are other density-based clustering algorithms as GDBSCAN (Generalized DBSCAN), and OPTICS (Ordering Points to Identify the Clustering Structure) in the literature (Sander et al., 1998; Daszykowski et al., 2004; Ankerst et al., 1999). GDBSCAN algorithm is proposed for the density-skewed case. In this method, α and *MinPts* values are determined by the user according to the densities. Set densities are arranged in increasing order and the sets with fewer densities are joined by using Greedy algorithm. DBSCAN calculates many distance functions that increases the complexity of the algorithm. In order to reduce this complexity, OPTICS algorithm is recommended. In this algorithm, distance queries of $\acute{\varepsilon}$ which are smaller than ε are made and distinct distance functions are used only if it is desired to obtain real clustering. A data set can be represented in OPTICS while multidimensional projection is not possible in DBSCAN.

Some interesting recent work on density-based clustering is using 1-class support vector machines (Ben-Hur et al., 2002).

2.1.4 Grid-Based Methods

Grid-based methods quantize the object space into a finite number of cells that form a grid structure. All of the clustering operations are performed on the grid structure, *i.e.* on the quantized space. The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space.

STING (Wang et al., 1997) is a typical example of a grid-based method. CLIQUE (Agrawal et al., 1998) and Wave-Cluster (Sheikholeslami et al., 1998) are two clustering algorithms that are both grid-based and density-based.

2.1.5 Model-Based Methods

Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. A model-based algorithm may locate clusters by constructing a density function that reflects the spatial distribution of the data points.

It also leads to a way of automatically determining the number of clusters based on standard statistics, taking noise or outliers into account and thus yielding robust clustering methods.

Model-based clustering methods follow two major approaches: a *statistical approach* and *a neural network approach.* Examples of the statistical approach include COBWEB (Fisher, 1987), CLASSIT (Gennari et al., 1989), and AutoClass (Cheeseman & Stutz, 1996). Studies of the neural network approach include competitive learning by Russell (1923) and SOM (self organizing feature maps) by Kohonen (1982).

2.1.6 Fuzzy Clustering

In classical (hard/crisp) clustering, the boundary of different clusters is crisp such that each pattern is assigned to exactly one class. On the other hand, the boundary between clusters may not be precisely defined in real life such that some of the patterns can belong to more than one cluster with different positive degrees of membership. This case is represented by fuzzy clustering instead of crisp clustering (Höppner et al., 1999; Dumitrescu et al., 2000).

In the fuzzy clustering literature, FCM algorithm is the best-known fuzzy clustering method and its variants are found in the literature (Dunn, 1973; Bezdek, 1973). Most of these approaches suppose the fuzziness of clustering with respect to possibility of membership of some elements into some classes. But in this work, a different approach of fuzziness based on a Fuzzy Joint Points (FJP) method is considered. Basic difference of this method is its comprehension of fuzziness in a hierarchical point of view, *i.e.* it considers the elements by constructing homogenous groups in detail. It is obvious that the elements are more dissimilar when they are discussed in more detail. The fuzzier the elements, more similar they are. In this case, fuzziness of clustering points out the investigation of the considered properties in more detail. Since all of the elements will be dissimilar from each other in minimal fuzziness degree of zero, each element can be considered as an individual cluster. On the other hand, in maximal degree of fuzziness, all of the elements can be considered to be similar to each other in such a way that they belong to one class.

Finding the optimal cluster number, specifying initial clusters and direct methods for clustering with iterative development are fundamental problems of FCM-type clustering algorithms. Among these methods, K-nearest neighbor (KNN) and Mountain method are used widely (Zahid et al., 2001; Yager & Filev, 1994; Velthuizen et al., 1997). But these methods have some disadvantages. For instance, the basic disadvantages of KNN are necessity to a priori given number of clusters and to assign equal number of elements to each class. The basic disadvantage of Mountain method is necessity to set up its parameters and without correct set up, the method may give bad results.

Another approach to fuzzy clustering is the Fuzzy Joints Points (FJP) method (Nasibov & Ulutagay, 2005a,b). Unlike FCM, FJP method is able to recognize clusters with arbitrary structure. Furthermore, FJP method does not have a disadvantage such as predetermining the number of clusters or constructing initial clusters. On the other hand, FJP method has an integrated cluster validity mechanism to determine the optimal number of clusters. From this view, FJP method is more advantageous than both hierarchical clustering algorithms and density-based DBSCAN algorithm.

The fundamental idea of the FJP method is to compute the fuzzy relation matrix based on the distance between points. Then, for certain $\alpha \in [0,1]$, α-level sets and equivalence classes are constructed. At the same time, these α-degree equivalence classes determine each α-level set of the fuzzy clusters. Also note that, these α-level sets are not computed for all $\alpha \in [0,1]$ degrees, instead they are computed only for α-levels in which the number of clusters are affected. Then, the final level set is computed based on the maximal change interval of the α's. In other words, the α-level degree that reflects the cluster structure optimally and α-level set appropriate for these level are found simultaneously.

2.2 FCM Algorithm

As a partitioning method, the *k*-means algorithm was first introduced by Mac-Queen (MacQueen, 1967). The *k*-means algorithm takes the input parameter, *k*, and partitions a set of *n* objects into *k* clusters so that the resulting intra-cluster similarity is high but the inter-cluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's center of gravity. Fuzzy c-means (FCM) algorithm is a generalization of the *k*-means algorithm. It was first introduced by Dunn and then generalized by Bezdek (Dunn, 1973; Bezdek, 1973).

FCM algorithm partitions a collection of *n* vectors ($X = \{x_0, x_1, \ldots, x_n\} \subset R^p$) into *c* fuzzy groups such that the weighted within-groups sum of squared error objective function is minimized. The objective function and constraints for FCM are defined as

$$J_m(u,v) = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^m d\,(v_i, x_j) \rightarrow Min \tag{2.1}$$

subject to:

$$\sum_{i=1}^{c} u_{ij} = 1, \quad u_{ij} \in [0,1], \quad 0 < \sum_{j=1}^{n} u_{ij} < n$$

In Equation (2.1), u_{ij} is the membership of the j^{th} data point in the i^{th} cluster, v_i is the i^{th} cluster center, and $d\ (v_i, x_j)$ is the distance between v_i and x_j, *i.e.*

$$d\,(v_i, x_j) = \left[\sum_{k=1}^{p} (x_{jk} - v_{ik})^2 \right]^{1/2} \tag{2.2}$$

The necessary conditions for J_m to reach its minimum are given below:

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{2.3}$$

$$u_{ij} = \frac{1}{\sum_{l=1}^{c} \left[\frac{d\left(v_i, x_j\right)}{d\left(v_l, x_j\right)} \right]^{\frac{2}{m-1}}} \tag{2.4}$$

FCM Algorithm.

Step 1. Given unlabeled data set $X = \{x_0, x_1, \ldots, x_n\}$;

Fix $c, m, ||.||_A$ and $\varepsilon > 0$;

Choose initial cluster centers $\{v_{10}, v_{20}, \ldots, v_{c0}\}$ arbitrarily;

Set $t = 1$.

Step 2. Compute all memberships $u^t = [u^t_{ij}]$, $i = 1, 2, \ldots, c$; $j = 1, 2, \ldots, n$ using Equation (2.4);

Step 3. Update all c fuzzy cluster centers v^t_i using Equation (2.3).

Step 4. Compute $E_t = ||v^t - v^{t-1}||^2$

Step 5. If $E_t < \varepsilon$ stop, else $t = t + 1$ and go to Step 2.

End.

2.2.1 Initialization of Clusters

Initialization of clusters is one of the most crucial steps of FCM clustering algorithm. Speed of resulting and shapes of resultant clusters may differ with respect to this. Therefore, initial cluster construction methods are important. Some of the well known initial cluster construction methods are Mountain method, Modified Mountain method and K-Nearest-Neighbors rule (Yager & Filev, 1994; Velthuizen et al., 1997; Zahid et al., 2001).

2.2.2 Cluster Validity

Cluster validation is an important issue in cluster analysis since the correct structure of a data set is unknown. Once the partition is obtained by a clustering method, the validity function can help us to validate whether it accurately presents the data structure or not. Hence, detecting the cluster validity is the basic problem of cluster analysis. Cluster validity indices may be defined as identifying the optimal cluster number. It is impossible to detect the real structure of the cluster if a little mistake is made in determining the number of clusters. Some of the widely used cluster validity criteria are given Table 2.1 (Bezdek, 1974, 1975; Dunn, 1974; Fukuyamo & Sugeno, 1989; Xie & Beni, 1991; Kwon, 1998).

Table 2.1 Some of the widely used cluster validity indices.

Criteria	**Functional description**	**Optimal number**
PC	$V_{PC} = \frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{2}$	$max\,(V_{PC}, U, c)$
CE	$V_{CE} = -\frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}\log_{a} u_{ij}$	$min\,(V_{CE}, U, c)$
FS	$V_{FS_m} = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m}[d^{2}(x_j, v_i) - d^{2}(m_x, v_i)]$	$min\,(V_{FS}, U, c)$
SI	$V_{SI} = \frac{\min\limits_{i \neq j} d(u_i, u_j)}{\max\limits_{i} \delta(u_i)}$	$max\,(V_{SI}, U, c)$
XB	$V_{XB} = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{2}\lVert x_j - v_i\rVert^{2}}{n(\min\limits_{i \neq k}\lVert v_i - v_k\rVert^{2})}$	$min\,(V_{XB}, U, c)$

2.3 DBSCAN Algorithm

To discover clusters with complex shape, density/neighborhood-based clustering methods have been developed. These typically regard clusters as dense regions of objects in the data space that are separated by regions of low density (representing noise).

DBSCAN (A Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm (Ester et al., 1996). The algorithm grows regions with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial data-bases with noise. It defines a cluster as a maximal set of density-connected points.

Consider a data set $X = x_1, x_2, ..., x_n$. Each object x_i has m properties. Thus, each datum x_i could be handled as a point of m-dimensional space, *i.e.* $x_i = (x_{i1}, x_{i2}, ..., x_{im})$. In this sense, the Euclidean distance $d(x_i, x_j)$ between any points $x_i, x_j \in X$ can be determined as follows:

$$d(x_i, x_j) = \left[\sum_{k=1}^{m} (x_{ik} - x_{jk})^2 \right]^{1/2} \tag{2.5}$$

First of all, let us define some concepts used in the DBSCAN algorithm. The neighborhood set of point $x \in X$ detected by using any of the membership function is determined as follows (Figure 2.2).

Definition 2.1. The neighborhood set of point $x \in X$ with parameter ε (ε-neighborhood set) is as follows:

$$N(x, \varepsilon) = \{y \in X \mid d(x, y) \leq \varepsilon\}. \tag{2.6}$$

Definition 2.2. $x \in X$ is called a core point with parameters ε and *MinPts* if

$$|N(x, \varepsilon)| \geq MinPts \tag{2.7}$$

is satisfied where $|N(x, \varepsilon)|$ is the cardinality of the set $N(x, \varepsilon)$.

Definition 2.3. Let $p, q \in X$. A point p is directly density-reachable from a point q with respect to the ε and *MinPts* if q is a core point and $p \in N(q, \varepsilon)$.

Note that other points can only be directly density-reachable from core points.

Definition 2.4. Let $p_i \in X$, $i = 1, ..., n$. A point p is density reachable from a point q with respect to ε and *MinPts* if there is a chain of points $p_1, ..., p_n$, $p_1 = q$, $p_n = p$, such that p_{i+1} is directly density-reachable from p_i.

Definition 2.5. Let $p, q, o \in X$. A point p is density connected to a point q with respect to ε and *MinPts* if there is a core point o such that both p and q are density-reachable from o with respect to ε and *MinPts*.

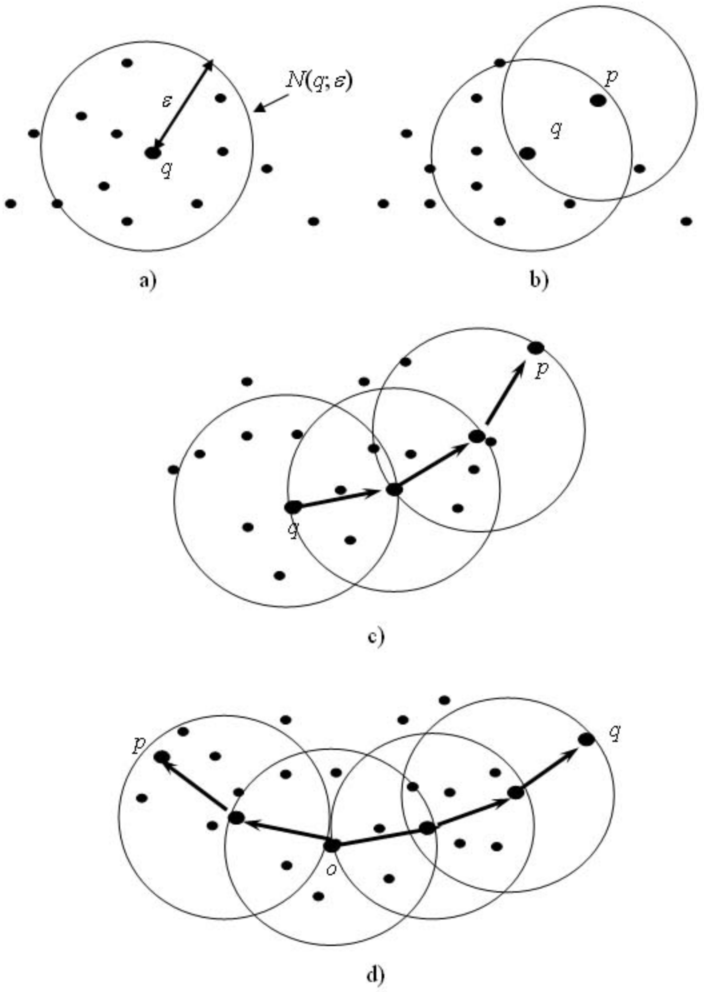


Figure 2.2 Illustration of some concepts used in DBSCAN a) core point, b) direct density reachability, c) density reachability, d) density connectivity.

Density reachability is the transitive closure of direct density reachability, and this relationship is asymmetric. Only core objects are mutually density reachable. Density connectivity, however, is a symmetric relation.

Definition 2.6. Let *D* be a database of points. A cluster *C* with respect to ε and *MinPts* is a non-empty subset of *D* satisfying the following conditions:

a) Maximality: $\forall p,q$: if $p \in C$ and q is density-reachable from p with respect to ε and *MinPts*, then $q \in C$.

b) Connectivity: $\forall p,q \in C$: p is density-connected to q with respect to ε and *MinPts*.

Definition 2.7. Let $C_1,...,C_k$ be the clusters of the database D with respect to parameters ε and *MinPts*. Then, noise is defined as the set of points in the database D not belonging to any cluster C_i, *i.e.* $noise = \{p \in D \mid \forall i : p \notin C_i\}$.

The main idea of DBSCAN algorithm is that each core point must have a certain minimum number of neighbors (*MinPts*) in a certain ε radius. The running principle of the algorithm is as follows: starting from each core point, every core point and points in its neighborhood which are directly density reachable from it (so called seed points) form a set of seeds. Then, the process continues by starting from another core point and a new set of seeds is formed until each core point is handled in this sense. The pseudocode of the DBSCAN algorithm is as follows:

DBSCAN Algorithm.

Step 1. Specify *Eps* and *MinPts* .

Step 2. Mark all the points in the data set as unclassified.

Step 3. Find an unclassified core point p with *Eps* and *MinPts*. Mark p to be classified. Start a new cluster to be the current cluster and assign p to the current cluster.

Step 4. Find all the unclassified points in the *Eps*-neighborhood of p. Create a set of seeds and put all these points into the set.

Step 5. Get a point q in the seeds, mark q to be classified, assign q to the current cluster, and remove q from the seeds.

Step 6. Check if q is a core-point with *Eps* and *MinPts*, if so, add all the unclassified points in the *Eps*-neighborhood of q to the set of seeds.

Step 7. Repeat step 5 through 6 until the set of seeds is empty.

Step 8. Start a new cluster and repeat step 3 through 7 until no more core-points can be found.

Step 9. Output all the clusters found so far, and mark all the points, which do not belong to any cluster, as noise.

End.

In the following chapters, by using above mentioned clustering algorithms, various modifications of fuzzy neighborhood relation based clustering algorithm are constructed and comparative analysis is performed.

CHAPTER THREE
FUZZY RELATIONS

A *relation* represents the presence or absence of association, interaction, or interconnectedness between the elements of two or more sets. This concept can be generalized to allow for various degrees or strengths of relation or interaction between elements. Degrees of association can be represented by membership grades in a *fuzzy relation* in the same way as degrees of set membership are represented in the fuzzy set. In fact, just as the crisp set can be viewed as a restricted case of the more general fuzzy set concept, the crisp relation can be considered to be a restricted case of the fuzzy relation (Klir & Folger, 1988; Pedrycz & Gomide, 1998).

3.1 Crisp Relations and Their Properties

Definition 3.1. If A and B are two sets and there is a specific property between elements x of A and y of B, this property can be described using the ordered pair (x,y). A set of such (x,y) pairs, $x \in A$ and $y \in B$, is called a relation R.

$$R = \{(x,y) \mid x \in A, y \in B\} \tag{3.1}$$

R is a *binary relation* and a subset of $A \times B$.

If $(x,y) \notin R$, x is not in relation R with y. If $A = B$ or R is a relation from A to A, it is written

$$(x,x) \in R \quad or \quad xRx, \quad R \subseteq A \times A. \tag{3.2}$$

Definition 3.2. For sets $A_1, A_2, A_3, ..., A_n$, the relation among elements $x_1 \in A_1, x_2 \in A_2, x_3 \in A_3, ..., x_n \in A_n$ can be described by n-tuple $(x_1, x_2, ..., x_n)$. A collection of such n-tuples $(x_1, x_2, ..., x_n)$ is a relation R among $A_1, A_2, A_3, ..., A_n$ which is called *n-ary relation*. That is

$$(x_1, x_2, ..., x_n) \in R, \quad R \subseteq A_1 \times A_2 \times \cdots \times A_n. \tag{3.3}$$

Definition 3.3. Let R stand for a relation between A and B. The domain and range of this relation are defined as follows

$$dom(R) = \{x \mid x \in A, (x,y) \in R \quad \text{for some} \quad y \in B\} \tag{3.4}$$

$$ran(R) = \{y \mid y \in B, (x,y) \in R \quad \text{for some} \quad x \in A\}. \tag{3.5}$$

Here we call set A as support of $dom(R)$ and B as support of $ran(R)$. $dom(R) = A$ results in completely specified and $dom(R) \subseteq A$ incompletely specified. The relation $R \subseteq A \times B$ is a set of ordered pairs (x,y). Thus, if we have a certain element x in A, we can find y of B, *i.e.*, the mapped image of A. We say "y is the mapping of x".

If we express this mapping as f, y is called the image of x which is denoted as $f(x)$

$$R = \{(x,y) \mid x \in A, y \in B, y = f(x)\} \quad or \quad f : A \to B. \tag{3.6}$$

3.1.1 Properties of Relation on a Single Set

The fundamental properties of relation defined on a set, that is, $R \subseteq A \times A$ such as reflexive relation, symmetric relation, transitive relation, closure, equivalence relation, compatibility relation, pre-order relation and order relation is handled in detail.

1. Reflexive Relation: If for all $x \in A$, the relation xRx or $(x,x) \in R$ is established, we call it *reflexive relation*. The reflexive relation might be denoted as

   $$x \in A \to (x,x) \in R \quad or \quad \mu_R(x,x) = 1, \quad \forall x \in A$$

   where the symbol "$\to$" means *implication*. If it is not satisfied for some $x \in A$, the relation is called *irreflexive*. If it is not satisfied for all $x \in A$, the relation is *antireflexive*.

2. Symmetric Relation: For all $x, y \in A$, if $xRy = yRx$, R is said to be a symmetric

relation and expressed as

$$(x,y) \in R \rightarrow (y,x) \in R, \quad \mu_R(x,y) = \mu_R(y,x), \quad \forall x,y \in A.$$

The relation is *asymmetric* or *nonsymmetric* when for some $x,y \in A$, $(x,y) \in R$ and $(y,x) \notin R$. It is an *antisymmetric* relation if for all $x,y \in A$, $(x,y) \in R$ and $(y,x) \notin R$.

3. Transitive Relation: This concept is achieved when a relation defined on A verifies the following property.

$$(x,y) \in R, (y,z) \in R \rightarrow (x,z) \in R, \quad \forall x,y,z \in A.$$

4. Closure: When relation R is defined in A, the requisites for closure are,

   a) Set A should satisfy a certain specific property.

   b) Intersection between A's subsets should satisfy the relation R.

   The smallest relation $\hat{R}$ containing the specific property is called *closure of* R.

Definition 3.4. A relation $R \subseteq A \times A$ is an *equivalence relation* if reflexivity, symmetry, and transitivity conditions are satisfied.

If an equivalence relation R is applied to a set A, we can perform a partition of A into n disjoint subsets $A_1, A_2, ..., A_n$ which are equivalence classes of R. At this time in each equivalence class, the above three conditions are verified. Assuming equivalence relation R in A is given, equivalence classes are obtained. The set of these classes is a partition of A by R and denoted as $\pi(A/R)$.

Definition 3.5. If a relation satisfies reflexivity and symmetry conditions for every $x,y \in A$, the relation is called *compatibility relation*.

If a compatibility relation R is applied to set A, we can decompose the set A into disjoint subsets which are compatibility classes. In each compatibility class, the above two conditions are satisfied. Therefore, a compatibility relation on a set A gives a partition. But the only difference from the equivalence relation is that transitive relation is not completed in the compatibility relation.

Definition 3.6. For any $x,y,z \in A$, if a relation $R \subseteq A \times A$ satisfies reflexivity and transitivity conditions, it is called *pre-order relation*.

We can assure that if a pre-order exists, it implies that an order exists between classes, and that the number of members in a class can be more than 1. If the property of antisymmetric relation is added to the pre-order, the number of member in a class should be 1 and it becomes an order relation.

Definition 3.7. If a binary relation $R \subseteq A \times A$ satisfies *i)* reflexivity, *ii)* antisymmetry, and *iii)*transitivity conditions for any $x,y,z \in A$, it is called *order relation* or *partial order relation* .

When relation R is given to an arbitrary set A, an order according to R is defined among the elements of A. If the condition (i) is replaced by

(i') Antireflexive relation

$$x \in A \rightarrow (x,x) \notin R$$

we apply the term *strict order relation* for it.

In the order relation, when the following condition (iv) is added, we call this relation a *total order* or *linear order relation*.

$$\text{iv)} \forall x,y \in A, (x,y) \in R \quad or \quad (y,x) \in R$$

The total order is also termed as a *chain* since it can be drawn in a line. Comparing to the total order, the order following only conditions *i)* , *ii)* and *iii)* is called a *partial order*, and a set defining the partial order is called *partial order set*.

Definition 3.8. For all $x,y \in A$, $(x \neq y)$,

i) If $(x,y) \in R$, xRy or $x > y$, $f(x) = f(y) + 1$.

ii) If reachability relation exists in x and y, *i.e.* if $x\hat{R}y$, $f(x) > f(y)$.

Now we can summarize as follows :

(1) In the pre-order, the symmetry or nonsymmetry is allowed. But in the case of order, only the antisymmetry is allowed. In other words, adding the antisymmetry to the pre-order, we get an order.

(2) A pre-order is said to be an order between classes. In other words, an order is a pre-order restricting that the number of class is 1.

(3) An equivalence relation has symmetry, so it can be obtained by adding the symmetry to the pre-order relation.

Characteristics so far discussed are summarized in Table 3.2.

Table 3.2 Comparison of relations.

Relation	**Property**				
	Reflexive	Antireflexive	Symmetric	Antisymmetric	Transitive
Equivalence	✓		✓		✓
Compatibility	✓		✓		
Pre-order	✓				✓
Order	✓			✓	✓
Strict order		✓		✓	✓

3.2 Fuzzy Relations and Their Properties

If a crisp relation R represents that of from sets A to B, for $x \in A$ and $y \in B$, its membership function $\mu_R(x,y)$ is,

$$\mu_R = \begin{cases} 1 & \text{if } (x,y) \in R \\ 0 & \text{if } (x,y) \notin R \end{cases} . \tag{3.7}$$

This membership function maps $A \times B$ to set $\{0,1\}$, *i.e.*

$$\mu_R : A \times B \to [0,1] \tag{3.8}$$

Definition 3.9. *Fuzzy relation* has degree of membership whose value lies in $[0,1]$,

$$R = \{((x,y),\mu_R(x,y)) \mid \mu_R(x,y) \geq 0, x \in A, y \in B\}. \tag{3.9}$$

Here $\mu_R(x,y)$ is interpreted as strength of relation between x and y. When $\mu_R(x,y) \geq \mu_R(x',y')$, (x,y) is more strongly related than (x',y'). When a fuzzy relation $R \subseteq A \times B$ is given, this relation R can be thought as a fuzzy set in the space $A \times B$.

Assume a Cartesian product space $X_1 \times X_2$ composed of two sets X_1 and X_2. This space makes a set of pairs (x_1,x_2) for all $x_1 \in X_1, x_2 \in X_2$. Given a fuzzy relation R between two sets X_1 and X_2, this relation is a set of pairs $(x_1,x_2) \in R$. Consequently, this fuzzy relation can be presumed to be a fuzzy restriction to the set $X_1 \times X_2$. Therefore, $R \subseteq X_1 \times X_2$.

Fuzzy binary relation can be extended to n-ary relation. If $X_1, X_2, ..., X_n$ are assumed to be fuzzy sets, fuzzy relation $R \subseteq X_1 \times X_2 \times \ldots \times X_n$ can be said to be a fuzzy set of tuple elements $(x_1,x_2,\ldots,x_n)$, where $x_1 \in X_1, x_2 \in X_2, \ldots, x_n \in X_n$.

When crisp relation R represents the relation from crisp sets A to B, its domain and range can be defined as,

$$dom(R) = \{x \mid x \in A, y \in B, \mu_R(x,y) = 1\}$$

$$ran(R) = \{y \mid x \in A, y \in B, \mu_R(x,y) = 1\}$$

Definition 3.10. When fuzzy relation R is defined in crisp sets A and B, the domain and range of this relation are defined as:

$$\mu_{dom(R)}(x) = \max_{y \in B} \mu_R(x,y)$$

$$\mu_{ran(R)}(y) = \max_{x \in A} \mu_R(x,y)$$

Set A becomes the support of $dom(R)$ and $dom(R) \subseteq A$. Set B is the support of

$ran(R)$ and

$$ran(R) \subseteq B.$$

Given a certain vector, if an element of this vector has its value between 0 and 1, this vector is called a *fuzzy vector*. *Fuzzy matrix* is a gathering of such vectors. Given a fuzzy matrix $A = (a_{ij})$ and $B = (b_{ij})$, operations can be performed on these fuzzy matrices.

i) Sum: $A + B = \max[a_{ij}, b_{ij}]$

ii) Max product: $A \bullet B = AB = \max_k[min(a_{ij}, b_{ij})]$

iii) Scalar product: λA where $0 \leq \lambda \leq 1$.

Definition 3.11. If a fuzzy relation R is given in the form of fuzzy matrix, its elements represent the membership values of this relation. That is, if the matrix is denoted by M_R, and membership values by $\mu_R(i, j)$, then $M_R = (\mu_R(i, j))$ and it is called a *fuzzy relation matrix*.

It is obvious that a relation is one kind of sets. Therefore operations of fuzzy set to the relation can be applied. Assume $R \subseteq A \times B$ and $S \subseteq A \times B$.

i) Union Relation: Union of two relations R and S is defined as follows:

$$\mu_{R \cup S}(x, y) = \max[\mu_R(x, y), \mu_S(x, y)] = \mu_R(x, y) \vee \mu_S(x, y), \quad \forall (x, y) \in A \times B$$

In general, the sign $\vee$ is used for max operation. For n relations, it is extended to the following:

$$\mu_{R_1 \cup R_2 \cup \ldots \cup R_n}(x, y) = \vee_{R_i} \mu_{R_i}(x, y).$$

ii) Intersection Relation : The intersection relation $R \cap S$ of set A and B is defined by the following membership function:

$$\mu_{R\cap S}(x,y) = \min[\mu_R(x,y), \mu_S(x,y)] = \mu_R(x,y) \wedge \mu_S(x,y), \quad \forall (x,y) \in A \times B$$

The symbol $\wedge$ is for the min operation. In the same manner, the intersection relation for n relations is defined by

$$\mu_{R_1 \cap R_2 \cap \ldots \cap R_n}(x,y) = \wedge_{R_i} \mu_{R_i}(x,y).$$

iii) Complement Relation : Complement relation R for fuzzy relation R shall be defined by the following membership function:

$$\mu_{\overline{R}}(x,y) = 1 - \mu_R(x,y), \quad \forall (x,y) \in A \times B$$

iv) Inverse Relation: When a fuzzy relation $R \subseteq A \times B$ is given, the inverse relation of R^{-1} is defined by the following membership function:

$$\mu_R^{-1}(y,x) = \mu_R(x,y), \quad \forall (x,y) \subseteq A \times B$$

Definition 3.12. Two fuzzy relations R and S are defined on sets A, B and C. That is, $R \subseteq A \times B, S \subseteq B \times C$. The composition $S \bullet R$ of two relations R and S is expressed by the relation from A to C, and this composition is defined by the following:

$$\mu_{S\bullet R}(x,z) = \max_y [\min(\mu_R(x,y), \mu_S(y,z))]$$

$$= \vee_y [\mu_R(x,y) \wedge \mu_S(y,z))], \quad \text{for}\,(x,y) \in A \times B,\, (y,z) \in B \times C.$$

$S \bullet R$ from this elaboration is a subset of $A \times C$. That is, $S \bullet R \subseteq A \times C$.

If the relations R and S are represented by matrices M_R and M_S, the matrix $M_{S\bullet R}$ corresponding to $S \bullet R$ is obtained from the *max-min* product of M_R and M_S, *i.e.*

$$M_{S \bullet R} = M_R \bullet M_S.$$

Presuming that the relations R and S are the expressions of rules that guide the occurrence of event or fact. Then the possibility of occurrence of event B when event A is happened is guided by the rule R. And rule S indicates the possibility of C when B is existing. For further cases, the possibility of C when A has occurred can be induced from the composition rule $S \bullet R$. This manner is named as an *inference* which is a process producing new information.

Definition 3.13. We can obtain α-cut relation from a fuzzy relation by taking the pairs which have membership degrees no less than α. Assume $R \subseteq A \times B$, and R_α is a α-*cut relation*. Then,

$$R_\alpha = \{(x,y) \mid \mu_R(x,y) \geq \alpha, \quad x \in A, y \in B\}.$$

Note that R_α is a crisp relation.

Definition 3.14. Fuzzy relation can be said to be composed of several R_α's as following:

$$R = \bigcup_\alpha \alpha R_\alpha$$

where α is a value in the level set; R_α is a α-cut relation; αR_α is a fuzzy relation. The membership function of αR_α is defined as,

$$\mu_{\alpha R_\alpha}(x,y) = \alpha \bullet \mu_{R_\alpha}(x,y), \quad \text{for}\,(x,y) \in A \times B.$$

Thus we can decompose a fuzzy relation R into several αR_α, so called *decomposition of relation*.

Definition 3.15. The *projection* of a fuzzy relation $R \subseteq A \times B$ with respect to A or B is as follows:

$$\mu_{R_A}(x) = \max_y \mu_R(x,y) : \text{projection to A}, \quad \forall x \in A, y \in B$$

$$\mu_{R_B}(y) = \max_x \mu_R(x,y): \text{ projection to B}, \quad \forall x \in A, y \in B$$

Definition 3.16. Extending the projection in 2-dimensions to n-dimensional fuzzy set, assume relation R is defined in the space of $X_1 \times X2 \times \cdots \times X_n$. Projecting this relation to subspace of $X_{i1} \times X_{i2} \times \ldots \times X_{ik}$ is called *projection in n-dimension* and it gives a projected relation given below:

$$\mu_{R_{X_{i1} \times X_{i2} \times \cdots \times X_{ik}}}(x_{i1}, x_{i2}, \ldots, x_{ik}) = \max_{X_{j_1}, X_{j_2}, \ldots, X_{jm}} \mu_R(x_1, x_2, \ldots, x_n)$$

where $X_{j_1}, X_{j_2}, \ldots, X_{j_m}$ represent the omitted dimensions, and $X_{i1} \times X_{i2} \times \cdots \times X_{ik}$ the remained dimensions, and thus

$$\{X_1, X_2, \ldots, X_n\} = \{X_{i1} \times X_{i2} \times \cdots \times X_{ik}\} \cup \{X_{j_1}, X_{j_2}, \ldots, X_{j_m}\}.$$

Definition 3.17. As the opposite concept of projection, *cylindrical extension* is possible. If a fuzzy set or fuzzy relation R is defined in space $A \times B$, this relation can be extended to $A \times B \times C$ and we can obtain a new fuzzy set. This fuzzy set is written as $C(R)$.

$$\mu_{C(R)}(a,b,c) = \mu_R(a,b), \quad a \in A,\, b \in B,\, c \in C.$$

3.2.1 Characteristics of Fuzzy Relation

Assume that fuzzy relation R is defined on $A \times A$. The followings are some properties of a fuzzy relation.

1. Reflexive Relation: For all $x \in A$, if $\mu_R(x,x) = 1$, we call this relation *reflexive*.
2. Symmetric Relation: When fuzzy relation R is defined on $A \times A$, it is called *symmetric* if it satisfies the following condition:

   $$\mu_R(x,y) = \mu \Rightarrow \mu_R(y,x) = \mu, \quad \forall (x,y) \in A \times A.$$

   If we express this symmetric relation as a matrix, we get a symmetric matrix. So we easily see that our previous relation “x is close to y” is a symmetric relation.

We say "antisymmetric" for the following case.

$$\mu_R(x,y) \neq \mu_R(y,x) \quad \text{or} \quad \mu_R(x,y) = \mu_R(y,x) = 0, \quad \forall (x,y) \in A \times A, x \neq y.$$

We can also define the concept of "asymmetric" or "nonsymmetric" as follows.

$$\mu_R(x,y) \neq \mu_R(y,x), \quad \exists (x,y) \in A \times A, x \neq y.$$

"Perfect antisymmetry" can be thought to be the special case of antisymmetry satisfying:

$$\mu_R(x,y) > 0 \Rightarrow \mu_R(y,x) = 0, \quad \exists (x,y) \in A \times A, x \neq y.$$

3. Transitive Relation: Transitive relation is defined as,

$$\mu_R(x,z) \geq \max_y[\min(\mu_R(x,y), \mu_R(y,z))], \forall (x,y), (y,x), (x,z) \in A \times A. \quad (3.10)$$

If we use the symbol $\vee$ for max and $\wedge$ for min, the last condition becomes

$$\mu_R(x,z) \geq \vee_y[\mu_R(x,y) \wedge \mu_R(y,z)].$$

If the fuzzy relation R is represented by fuzzy matrix M_R, we know that left side in the above formula corresponds to M_R and right one to M_{R^2}. That is, the right side is identical to the composition of relation R itself. So the previous condition becomes,

$$M_R \geq M_{R^2} \quad \text{or} \quad R \supseteq R^2.$$

4. Transitive Closure : As we have referred the expression of fuzzy relation by matrix M_R, fuzzy matrix M_{R^2} corresponding composition R^2 shall be calculated by the max-min composition of M_R, *i.e.*

$$\mu_{R^2}(x,z) = M_R \bullet M_R = \max_y[\min(\mu_R(x,y), \mu_R(y,z))].$$

Transitive relation was referred to as $R \supseteq R^2$ and thus the relation between M_R

and M_{R^2} holds

$$M_R \geq M_{R^2},$$

then again, the relation $R \supseteq R^3$ may well be satisfied, and by the method of generalization we know

$$R \supseteq R^k, \quad k = 1,2,3,\ldots$$

from the property of closure, the transitive closure of R shall be,

$$\hat{R} = R \cup R^2 \cup R^3 \cup \ldots$$

Generally, if we go on multiplying fuzzy matrices (i.e, composition of relation), the following equation is held:

$$R^k = R^{k+1}, \quad k \leq n$$

where $R \subseteq A \times A$ and the cardinality of A is n. So, $\hat{R}$ is easily obtained

$$\hat{R} = R \cup R^2 \cup R^3 \cup \ldots \cup R^k, \quad k \leq n$$

3.2.2 Classification of Fuzzy Relation

In this section, the concepts of equivalence, compatibility, pre-order and order relations of crisp relations is generalized to those of fuzzy relations. We assume relation R is defined on $A \times A$.

Definition 3.18. If a fuzzy relation $R \subseteq A \times A$ satisfies reflexivity, symmetry, and transitivity conditions, it is called a *fuzzy equivalence relation* or *similarity relation.*

Using this similarity relation, the following three applications can be performed.

(1) Partition of sets : Just like crisp set A is done partition into subsets $A_1, A_2, \ldots$ by the equivalence relation, fuzzy set A also can be performed partition.

(2) Partition by α-cut : If α-cut is done on a fuzzy relation, we get crisp relations. By performing α-cut on fuzzy equivalence relation, we get crisp equivalence relations and thus the set A can be partitioned. For instance, if a partition is done on set A into subsets $A_1, A_2, A_3, \ldots$, the similarity among elements in A_i is no less than α. The α-cut equivalence relation R_α is defined by

$$\mu_R(x,y) = \begin{cases} 1, & \text{if } \mu_R(x,y) \geq \alpha, \forall x,y \in A_i \\ 0, & \text{otherwise} \end{cases} .$$

If α-cut is applied according to α_1 in level set $\{\alpha_1, \alpha_2, \ldots\}$, the partition by this procedure is denoted by $\pi(R_{\alpha_1})$ or $\pi(A/R_{\alpha_1})$. In the same manner, $\pi(R_{\alpha_2})$ is obtained by the procedure of α_2-cut. Then, it is known that if $\alpha_1 \geq \alpha_2$, $R_{\alpha_1} \subseteq R_{\alpha_2}$ and it can be said that $\pi(R_{\alpha_1})$ is more refined than $\pi(R_{\alpha_2})$.

(3) Set similar to element x : If similarity relation R is defined on set A, elements related to arbitrary member $x \in A$ can make up "set similar to x". Certainly this set shall be fuzzy one.

Definition 3.19. If fuzzy relation R in set A satisfies reflexivity and symmetry conditions, it is called *fuzzy compatibility relation* or *resemblance relation*.

If fuzzy compatibility relation is given on set A, a partition can be processed into several subsets. Subsets from this partition are called the *fuzzy compatibility classes* and if α-cut is pplied to the fuzzy compatibility relation, α-cut crisp compatibility relation R_α is obtained. A compatibility class A_i in this relation is defined by,

$$\mu_R = \begin{cases} 1, & \text{if } \mu_R(x,y) \geq \alpha, \quad \forall x,y \in A_i \\ 0, & \text{otherwise} \end{cases}$$

the collection of all compatibility classes from a α-cut is called *complete α-cover*. Note the differences of the cover and partition.

Definition 3.20. Given fuzzy relation R in set A, if the reflexivity and transitivity conditions are well kept for all $x, y, z \in A$, this relation is called *pre-order relation*.

Also if certain relation is transitive but not reflexive, this relation is called *semi-pre-order* or *nonreflexive fuzzy pre-order*.

Definition 3.21. If relation R satisfies the reflexivity, antisymmetry, and transitivity conditions for all $x, y, z \in A$, it is called *fuzzy order relation*.

Definition 3.22. A *corresponding crisp relation* R_1 from given fuzzy order relation R by arranging the value of membership function can be obtained as follows:

i) if $\mu_R(x,y) \geq \mu_R(y,x)$ then $\mu_{R_1}(x,y) = 1,\ \mu_{R_1}(y,x) = 0$

ii) if $\mu_R(x,y) = \mu_R(y,x)$ then $\mu_{R_1}(x,y) = \mu_{R_1}(y,x) = 0$.

If the corresponding order relation of a fuzzy order relation is total order or linear order, this fuzzy relation is named as *fuzzy total order*, and if not, it is called *fuzzy partial order*. When the antisymmetry relation condition of the fuzzy order relation is transformed into *perfect antisymmetric*, the fuzzy order relation becomes a *perfect fuzzy order*, where perfect antisymmetry is defined as follows:

$$\mu_R(x,y) > 0 \Rightarrow \mu_R(y,x) = 0, \quad \forall (x,y)\, nA \times A, x \neq y.$$

When the reflexivity relation condition of the fuzzy order relation does not exist, the fuzzy order relation is called *fuzzy strict order*.

In the fuzzy order relation, if $R(x,y) > 0$ holds, let us say that x dominates y and denote $x \geq y$. With this concept, two fuzzy sets are associated.

Definition 3.23. Dominating class $R_{\geq[x]}$ which dominates x is defined as,

$$\mu_{R_{\geq[x]}}(y) = \mu_R(y,x).$$

Definition 3.24. Dominated class $R_{\leq[x]}$ with elements dominated by x is defined as,

$$\mu_{R_{\leq[x]}}(y) = \mu_R(x,y).$$

3.2.3 Dissimilitude Relation

The reflexivity, symmetry, and transitivity conditions for the similarity relation were mentioned above. Especially, the transitivity is defined as given in Formula (3.10). *Dissimilitude relation* maintains the opposite position in the concept of similarity relation. As a result of applying the complement relation $\overline{R}$, instead of relation R, we can think of the transitivity of $\overline{R}$.

For any $(x,y) \in A \times A$, since $\mu_{\overline{R}}(x,y) = 1 - \mu_R(x,y)$, transitivity of R shall be,

$$\mu_R(x,z) \geq \vee_y[(1-\mu_{\overline{R}}(x,y)) \wedge (1-\mu_{\overline{R}}(y,z))].$$

The right part of this relation can be transformed by $\overline{A} \cap \overline{B} = \overline{A \cup B}$, *i.e.*

$$(1-\mu_{\overline{R}}(x,y)) \wedge (1-\mu_{\overline{R}}(y,z)) = 1-(\mu_R(x,y) \vee \mu_R(y,z)).$$

Consequently,

$$\mu_R(x,z) \geq \vee_y[1-(\mu_R(x,y)) \wedge \mu_R(y,z))]$$

i.e.

$$\mu_R(x,z) \leq \vee_y[\mu_R(x,y)) \wedge \mu_R(y,z)].$$

So, this property is called *transitivity of min-max operation.*

Definition 3.25. Given fuzzy relation R in set $A \times B$, if the antireflexivity, symmetry, and min-max transitivity conditions are well kept, this relation is called *dissimilitude relation.*

In the next chapter, fuzzy neighborhood relation is constructed on the basis of distance between data points. Furthermore, the clustering process is performed via construction of equivalency sets by using the transitive closure of this fuzzy relation.

CHAPTER FOUR
FUZZY NEIGHBORHOOD-BASED CLUSTERING ALGORITHMS

4.1 FJP Algorithm

As abovementioned, in classical fuzzy clustering the matter of fuzziness is usually a possibility of membership of each element into different classes with different positive degrees from [0,1]. In Fuzzy Joint Points (FJP) approach, the fuzziness of clustering is evaluated as how much in detail the properties of classified elements are investigated (Nasibov & Ulutagay, 2005b). The main advantage of the FJP algorithm is that it combines determination of initial clusters, cluster validity and direct clustering, which are the fundamental stages of a clustering process. Moreover, it also uses a more sensitive neighborhood analysis compared to DBSCAN algorithm since it benefits the fuzzy sets theory (Nasibov & Ulutagay, 2005a, 2006a,b).

It is possible to handle the fuzzy properties with various level-degrees of details and to recognize individual outlier elements as independent classes by the FJP method. This situation could be important in biological, medical, *etc.* problems in order to recognize new forms of living objects.

Let $F(E^p)$ denote the set of whole p-dimensional fuzzy sets of the space $E^p$and $\mu_A \rightarrow [0,1]$ denote the membership function of the fuzzy set $A \in F(E^p)$.

Definition 4.1. A conical fuzzy point $A = (a,R) \in F(E^p)$ of the space E^p is a fuzzy set with membership function (Figure 4.1)

$$\mu_A(x) = \begin{cases} 1 - \dfrac{d(x,a)}{R} & \text{if } d(x,a) \leq R \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

where $a \in E^p$ is the center of fuzzy point A, and $A = (a,R)$ is $R \in E^1$ is the radius of its support $suppA$, where

$$suppA = \{x \in E^p \,|\, \mu_A(x) > 0\}.$$

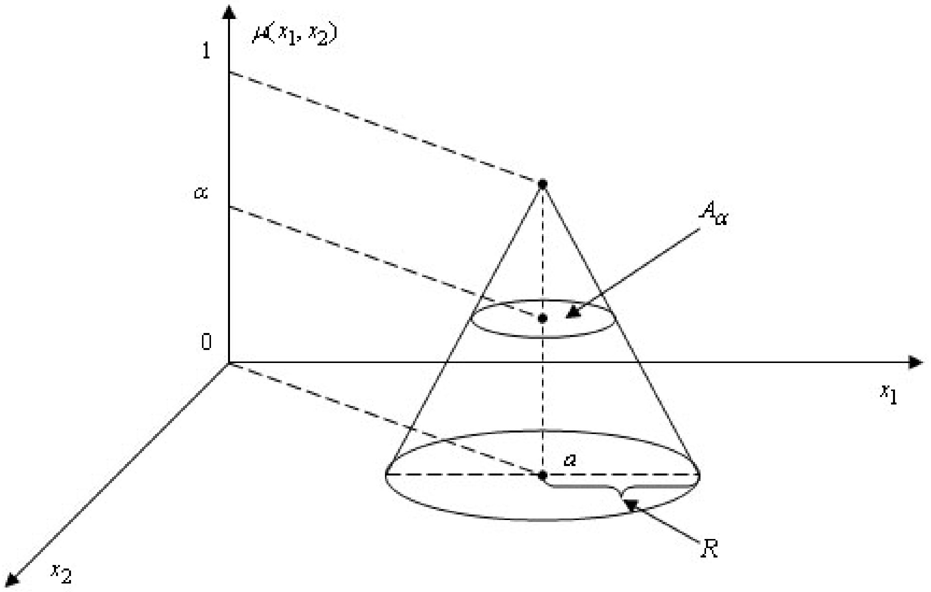


Figure 4.1 Fuzzy conical point $A = (a,R) \in F(E^2)$.

The α-level set of conical fuzzy point $A = (a,R)$ is calculated as

$$A_\alpha = \{x \in E^p \,|\, \mu_A(x) \geq \alpha\} = \{x \in E^p \,|\, d(x,a) \leq R \cdot (1-\alpha)\}. \tag{4.2}$$

Note that an analogue of conical fuzzy point $A = (a,R) \in F(E^1)$ of space E^1 is a triangular symmetrical fuzzy number $A = (a,R,R)$.

Let $A = (a,R)$ and $B = (b,R)$ be fuzzy points from the set $X \subset F(E^1)$ and let $T : X \times X \to [0,1]$ denote a fuzzy similarity relation on the set X as follows:

$$T(A,B) = 1 - \frac{d(a,b)}{2R}, \tag{4.3}$$

where $a \in E^p$ and $b \in E^p$ are the centers of the fuzzy points A and B respectively as shown in Figure 4.2.

Equation (4.3) can be rewritten as

$$d(a,b) = 2R(1 - T(A,B)). \tag{4.4}$$

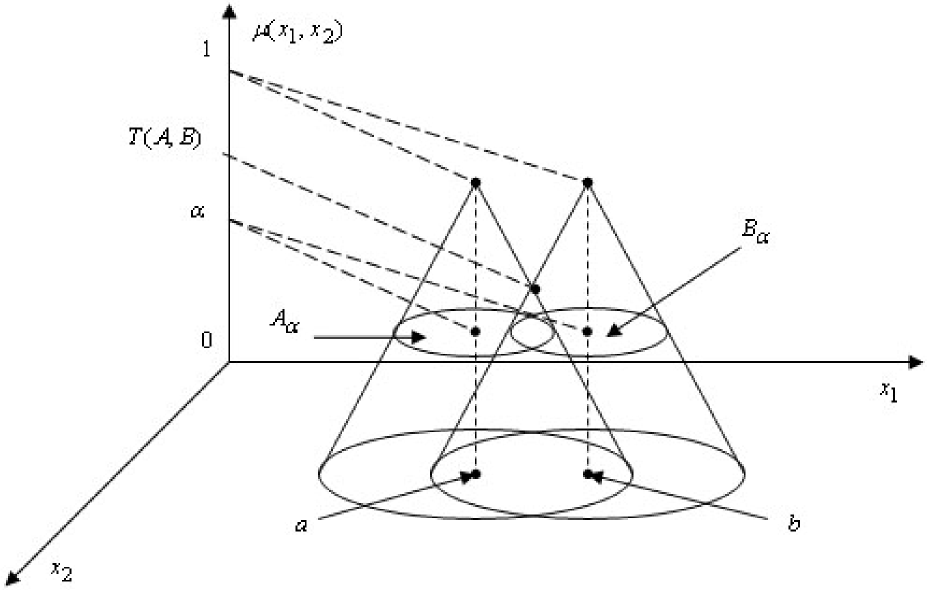


Figure 4.2 Fuzzy α-neighbor points $A = (a,R)$ and $B = (b,R)$ in the space E^2.

It is obvious that the relation T is reflexive, *i.e.* $\forall A \in X$, $T(A,A) = 1$ is provided.

Definition 4.2. Let A and B be fuzzy points on the set $X \subset F(E^1)$. If

$$T(A,B) \geq \alpha \tag{4.5}$$

is provided for fixed $\alpha \in (0,1]$, then the points A and B are called fuzzy α-neighbor points and it is denoted by $A \sim_\alpha B$ (Figure 4.2).

Lemma 4.1. *(Nasibov & Ulutagay, 2005a) The fuzzy points $A = (a,R)$ and $B = (b,R)$ are α-neighbor for fixed $\alpha \in (0,1]$ if and only if the inequality*

$$d(a,b) \leq 2R(1-\alpha) \tag{4.6}$$

is provided, where $d(a,b)$ denotes the distance between the centers of the fuzzy points A and B.

Proof. Suppose that for some $\alpha \in (0,1]$, the fuzzy points $A = (a,R)$ and $B = (b,R)$ are α-neighbor points. Then, by definition, the inequality (4.5) is provided. Hence, with

$\alpha \in (0,1]$, recalling (4.3) the following is obtained,

$$1-\frac{d(a,b)}{2R} \geq \alpha \Rightarrow d(a,b) \leq 2R(1-\alpha). \tag{4.7}$$

Now, suppose inequality (4.6) holds. We then find

$$\alpha \leq 1-\frac{d(a,b)}{2R} = T(A,B), \tag{4.8}$$

i.e. relation (4.5) is provided. This completes the proof of the lemma. □

Definition 4.3. If there is a chain of α-neighbor fuzzy points $C^1,\ldots,C^2,k \geq 0$, for fixed $\alpha \in (0,1]$, between the points A and B, *i.e.*

$$A \sim_\alpha C^1, C^1 \sim_\alpha C^2, \ldots, C^{k-1} \sim_\alpha C^k \quad \text{and} \quad C^k \sim_\alpha B, \tag{4.9}$$

then the fuzzy points A and B are called *fuzzy α-joint points*.

Definition 4.4. Let $X \subset F(E^p)$ be a set of fuzzy points. If the fuzzy points A and B are α-joint for $\alpha \in (0,1]$ and $\forall A,B \in X$, then the set X is called *fuzzy α-joint set*.

Let $d(A_\alpha,B_\alpha)$ be the classical distance between the level sets A_α and B_α, *i.e.*

$$d(A_\alpha,B_\alpha) = min\{d(x,y) \mid x \in A_\alpha, y \in B_\alpha\}. \tag{4.10}$$

Let the relation $\hat{T} : X \times X \to [0,1]$ be the transitive closure of relation $T : X \times X \to [0,1]$, which is obtained by using max-min composition.

Theorem 4.1. *(Nasibov & Ulutagay, 2005a) The fuzzy points A and B are fuzzy α-neighbor points for fixed $\alpha \in (0,1]$ if and only if the following relation holds:*

$$A_\alpha \cap B_\alpha \neq \emptyset \tag{4.11}$$

Proof. Suppose that the fuzzy points A and B are α-neighbor points, consequently, inequality (4.5) is satisfied. First, assume that (4.11) is not satisfied, *i.e.*,

$$A_\alpha \cap B_\alpha = \emptyset. \tag{4.12}$$

Then on the line connecting the points $a \in E^p$ and $b \in E^p$ there exists $x \in E^p$, $x \neq A_\alpha,\ x \neq B_\alpha$, such that the inequalities

$$d(a,x) > R(1-\alpha) \quad \text{and} \quad d(b,x) > R(1-\alpha) \tag{4.13}$$

hold.

In view of the fact that a, x, and b are collinear from (4.13), the following can be written:

$$d(a,b) = d(a,x) + d(x,b) > 2R(1-\alpha). \tag{4.14}$$

But by the assertion of Lemma 4.1, the latter inequality contradicts the condition of the fact that points A and B are α-neighbor.

Now, suppose that 4.11 holds. Then $\exists x : x \in A_\alpha, x \in B_\alpha$. Hence, in view of 4.13, the following is obtained:

$$d(a,x) \leq R(1-\alpha) \quad \text{and} \quad d(b,x) \leq R(1-\alpha). \tag{4.15}$$

In view of the triangle property of the distance, it follows from 4.15 that

$$d(a,b) \leq d(a,x) + d(x,b) \leq 2R(1-\alpha) \Rightarrow d(a,b) \leq 2R(1-\alpha). \tag{4.16}$$

By the statement of the Lemma 4.1, the latter inequality asserts that the fuzzy points A and B are α-neighbor points. This completes the proof of the theorem. □

Theorem 4.2. *(Nasibov & Ulutagay, 2005a) Any points $A,B \in X$ of the finite set X are fuzzy α-joint points if and only if*

$$\hat{T}(A,B) \geq \alpha \tag{4.17}$$

holds, where $\hat{T} : X \times X \to [0,1]$ is the transitive closure of the fuzzy relation T.

Proof. First, assume that the fuzzy sets A and B are α-joint sets. Then by Definition 4.3, a sequence of fuzzy points $C^1,\ldots,C^k$, $k \geq 0$ between the points A and B exists, *i.e.*

$$T(A,C^1) \geq \alpha, \quad T(A,C^2) \geq \alpha, \quad \ldots, \quad T(C^{k-1},C^k) \geq \alpha, \quad T(C^k,B) \geq \alpha. \tag{4.18}$$

Recall that $\hat{T}$ of any relation T is the minimal transitive relation containing the relation T, *i.e.* (Pedrycz & Gomide, 1998):

a) $\forall A,B \in X$, the relation $\hat{T}(A,B) \geq T(A,B)$ is satisfied,

b) $\forall A,B,C \in X$, it follows from $\hat{T}(A,B) \geq \alpha$ and $\hat{T}(B,C) \geq$ that $\hat{T}(A,C) \geq \alpha$.

Then, in view of property (a), it follows from 4.18 that

$$\hat{T}(A,C^1) \geq \alpha, \quad \hat{T}(A,C^2) \geq \alpha, \quad \ldots, \quad \hat{T}(C^{k-1},C^k) \geq \alpha, \quad \hat{T}(C^k,B) \geq \alpha. \tag{4.19}$$

Recalling property (b), from the latter inequalities it follows that inequality (4.17) is satisfied.

Now, let us prove that points A and B are α-joint fuzzy points.

By the definition of transitive closure (Pedrycz & Gomide, 1998),

$$\hat{T} = T \cup T^2 \cup \ldots \cup T^{k-1} \cup T^k \cup \ldots \tag{4.20}$$

and for a reflexive relation T on an n-element set,

$$T \subset T^2 \subset \ldots \subset T^{n-1} = T^n = T^{n+1} = \ldots. \tag{4.21}$$

Then for some $1 \leq k \leq n-1$,

$$\hat{T} = T^k. \tag{4.22}$$

Since inequality (4.17) is valid, the following is obtained:

$$T^k(A,B) \geq \alpha \tag{4.23}$$

which asserts that the elements of A and B are connected by a chain $(A, C^1, \ldots, C^{k-1}, B)$ of length k and that for all sequences of pairs from this chain, it holds that

$$T(A,C^1) \geq \alpha, \quad T(A,C^2) \geq \alpha, \ldots, T(C^{k-1},C^k) \geq \alpha, \quad T(C^k,B) \geq \alpha. \tag{4.24}$$

By Definition 4.3, the latter inequalities assert that the points A and B are fuzzy α-joint points, which completes the proof. □

Let a data set $\{x_1, x_2, \ldots, x_n\}$, $x_i \in E^p$ be given. It is required to divide the set into homogenous groups, *i.e.* to classify its elements. Number of classes is unknown a priori. Note that, in FJP algorithm, the fuzzy relation $T : X \times X \to [0,1]$ is normalized by calculating the radius of the considered fuzzy points as

$$R = \frac{\max\{d(x_i,x_j) \mid x_i, x_j \in X\}}{2} \equiv \frac{d_{max}}{2}. \tag{4.25}$$

Thus $\forall A, B \in X$ the degree of the relation $T(A,B)$ is defined as

$$T(A,B) = 1 - \frac{d(a,b)}{d_{max}}, \tag{4.26}$$

that implies,

$$d(a,b) = d_{max} \cdot (1 - T(A,B)). \tag{4.27}$$

The following algorithm is suggested in work (Nasibov & Ulutagay, 2006b) in order to solve the abovementioned problem. The value of optimal degree α is calculated and then the initial set $\{x_1, x_2, \ldots, x_n\}$ is partitioned into fuzzy α-joint sets by this algorithm.

FJP Algorithm.

FJP1. Compute:
$d_{ij} := d(x_i, x_j), \quad i, j = 1, \ldots, n;$
$d_{max} := \max d_{ij};$
$\varepsilon := 0.01 \cdot \min d_{ij};$
Set $\alpha_0 := 1$;,

FJP2. Compute the fuzzy relation $T_{ij} := 1 - \dfrac{d_{ij}}{d_{max}}$, $i, j = 1, \ldots, n$;
Compute the transitive closure $\hat{T}$ of the relation T;

FJP3. Set $y_i := x_i$, $i = \overline{1,n}$; $t := 1$; $k := n$;

FJP4. Compute: $d_t := \min d(y_i, y_j); \quad \alpha_t := \max\{1 - \frac{d_t + \varepsilon}{d_{max}}, 0\}$;

FJP5. Call the procedure Clusters (α_t) where the fuzzy α_t-joint sets $X^1, X^2, \ldots, X^k$, and the number k of these sets for α_t are computed;

FJP6. If $k > 1$, then set $y_i := X^i, i = 1, \ldots, k,\ t = t + 1$; and go to FJP4;
If $k = 1$, then go to FJP7;

FJP7. Compute:
$\Delta\alpha_i := \alpha_i - \alpha_{i+1}; \; i = 0, \ldots, t-1;$
$z := \arg\max \Delta\alpha_i;$
$\overline{\alpha} := \alpha_z - \dfrac{\Delta\alpha_z}{2};$

FJP8. Call the procedure Clusters $(\overline{\alpha})$ with parameter $\overline{\alpha}$.

```
FJP9. $\overline{\alpha}$ is the optimal membership degree of clustering;
      $\overline{k}$ is the optimal number of clusters;
      $X^1, X^2, \dots, X^{\overline{k}}$ is the partition of the set $X$.

End.
```

The auxiliary procedure Clusters (α) is used to implement the FJP algorithm. For a fixed input parameter α, this procedure partitions the set $X = \{x_1, x_2, \dots, x_n\}$ into fuzzy α-joint sets and returns these sets and the number of the sets.

```
Procedure Clusters $(\alpha)$
Input parameter: $\alpha$
Output parameters: $\alpha$-fuzzy joint sets $X^1, X^2, \dots, X^k$; $k$- number of these sets;

Cl1. $S := X = \{x_1, x_2, \dots, x_n\}$; $k := 1$;

Cl2. Get the first element $A \in S$ of the set $S$;
     Create sets: $X^k := \{B \in S \mid \hat{T}(A,B) \geq \alpha\}$;    $S := S \setminus X^k$;

Cl3. If $S \neq \emptyset$, then let $k := k+1$ and go to Step 2;
     Otherwise go to Step 4;

Cl4. Return the sets $X^1, X^2, \dots, X^k$; and number $k$ of these sets.

End.
```

4.1.1 FJP Cluster Validity Index

As mentioned in Section 2.2.2, one of the most crucial problems of all clustering algorithms is the validation of clusters obtained. An advantage of the FJP algorithm is that it has an integrated mechanism for cluster validation. Once a clustering structure is obtained for convenient α-level, a validity function is computed. At the end of the clustering process, the clustering structure that gives the maximum value to this function is considered as optimal.

Let $A = (a,R)$ and $B = (b,R)$ be fuzzy points from the set $X \subset F(E^1)$ and let $T : X \times X \to [0,1]$ denote a fuzzy similarity relation on the set X as defined in Equation 4.3.

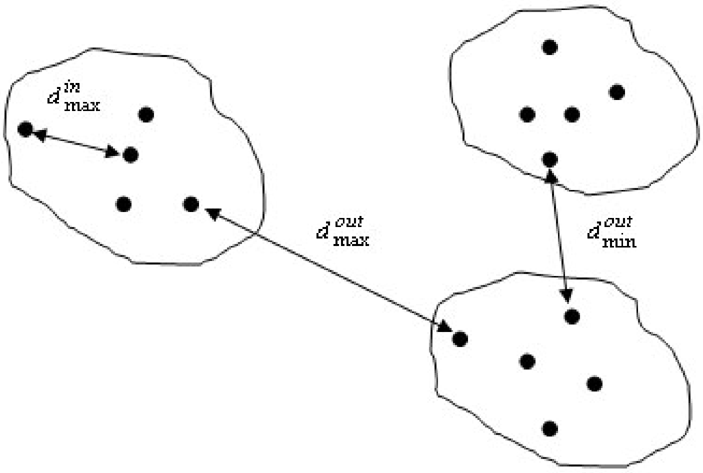


Figure 4.3 Location of homogenous sets obtained by the FJP algorithm.

Let $X^k,\ k = \overline{1,t}$, be homogenous classes created with respect to clustering. The followings can be written (Figure 4.3):

$$d_k^{in} = d_{max} \cdot (1 - \min_{x,y \in X^k} \hat{T}(x,y)), \tag{4.28}$$

$$d_{max}^{in} = \max_k d_k^{in}, \tag{4.29}$$

$$d_{min}^{out} = \min_{i \neq j} \{d(X^i, X^j) \mid i \neq j\}, \tag{4.30}$$

$$d_{max}^{out} = \max_{i,j} d(X^i, X^j). \tag{4.31}$$

As mentioned above, the cluster validity criterion used in FJP algorithm depends on the largest α change interval that does not affect the cluster number. Since the fuzzy point membership function is monotonic, its inverse function exists. Thus, the change interval of α parameter can be evaluated based on distance, and the following cluster validity function can be used (Nasibov & Ulutagay, 2007a):

$$V_{FJP} = d_{min}^{out} - d_{max}^{in} = \min_{i \neq j} \{d\,(X^i, X^j) - \max_k \{d_{max} \cdot (1 - \min_{x,y \in X^k} \hat{T}(x,y))\}. \qquad (4.32)$$

In other words, the clustering structure that gives maximum value to the above function is determined as optimal.

Due to the appropriate optimality structure, the cluster validity criterion given in Equation (4.32) can be rewritten as follows:

$$V'_{FJP} = \min_{i \neq j} d\,(X^i, X^j) - \min_k \min_{x,y \in X^k} \hat{T}(x,y). \qquad (4.33)$$

4.1.2 Analysis of Clusters' Structure in FJP Clustering

In this section, the properties of clustering structures which are formed on the base of the FJP approach are investigated.

The initial data set X can be divided into k fuzzy α-joint clusters each providing

$$\forall i, j : i \neq j \Rightarrow X^i \cap X^j = \emptyset \quad \text{and} \quad \bigcup_{i=1}^{k} X^i = X \qquad (4.34)$$

with a fixed α value.

It is obvious that the clustering structure, determined by FJP method, is based on the α-level degree. Let's designate homogeneity classes as $X^j(\alpha)$, $j = 1, \ldots, k(\alpha)$. Thus, the Formulae (4.28)-(4.31), given in Section 4.1.1, can be rewritten based on α-level as follows:

$$d_k^{in}(\alpha) = d_{max} \cdot (1 - \min_{x,y \in X^k(\alpha)} \hat{T}(x,y)), \qquad (4.35)$$

$$d_{max}^{in}(\alpha) = \max_{k} d_k^{in}(\alpha), \tag{4.36}$$

$$d_{min}^{out}(\alpha) = \min_{i \neq j} d\,(X^i(\alpha), X^j(\alpha)), \tag{4.37}$$

$$d_{max}^{out}(\alpha) = \max_{i \neq j} d\,(X^i(\alpha), X^j(\alpha)), \tag{4.38}$$

The following theorems are proven to explain the relation of this structure with α-degree better.

Theorem 4.3. $d_{max}^{in}(\alpha)$ *is a non-increasing function of the* α *parameter.*

Proof. Consider first $\forall \alpha_1, \alpha_2 : \alpha_2 \leq \alpha_1$. It is obvious that for each $X^i = X^i(\alpha_1)$, $i = 1, 2, \ldots, k(\alpha_1)$, a certain $\exists j \in \{1, 2, \ldots, k(\alpha_2)\} : X^j = X^j(\alpha_2)$ can be obtained, *i.e.*

$$X^i \subset X^j. \tag{4.39}$$

Thus,

$$\min_{x,y \in X^i} \hat{T}(x,y) \geq \min_{x,y \in X^j} \hat{T}(x,y) \tag{4.40}$$

can be written and by using the last inequality,

$$\begin{aligned} d_i^{in}(\alpha_1) &= d_{max} \cdot (1 - \min_{x,y \in X^i} \hat{T}(x,y)) \\ &\leq d_{max} \cdot (1 - \min_{x,y \in X^j} \hat{T}(x,y)) = d_j^{in}(\alpha_2) \end{aligned} \tag{4.41}$$

i.e.

$$d_i^{in}(\alpha_1) \leq d_j^{in}(\alpha_2) \tag{4.42}$$

is obtained. Consequently,

$$d_{max}^{in}(\alpha_1) = \max_{i} d_i^{in}(\alpha_1) \leq \max_{j} d_j^{in}(\alpha_2) = d_{max}^{in}(\alpha_2) \tag{4.43}$$

holds. As a result, it is proved that

$$d_{max}^{in}(\alpha_1) \leq d_{max}^{in}(\alpha_2) \tag{4.44}$$

holds $\forall \alpha_2 \leq \alpha_1$. □

Theorem 4.4. $d_{min}^{out}(\alpha)$ *is a non-increasing function of the* α *parameter.*

Proof. First, consider $\forall \alpha_1, \alpha_2 : \alpha_2 \leq \alpha_1$. It is obvious that each set $X^i = X^i(\alpha_2)$, $i = 1, 2, \ldots, k(\alpha_2)$, will contain at least one $\exists j \in \{1, 2, \ldots, k(\alpha_1)\} : X^j = X^j(\alpha_1)$ set. In other words, some clusters which are different in α_1-level will be in the same cluster in α_2 -level and the number of different clusters for which $min_{i \neq j}$ selection is possessed will decrease. Hence, the following can be written:

$$\min_{i \neq j} d\,(X^i(\alpha_1), X^j(\alpha_1)) \leq \min_{i \neq j} d\,(X^i(\alpha_2), X^j(\alpha_2)) \tag{4.45}$$

By taking (4.37) into account,

$$d_{min}^{out}(\alpha_1) \leq d_{min}^{out}(\alpha_2) \tag{4.46}$$

holds $\forall \alpha_2 \leq \alpha_1$ which completes the proof. □

Theorem 4.5. $d_{max}^{out}(\alpha)$ *is a non-decreasing function of the* α *parameter.*

Proof. Consider $\forall \alpha_1, \alpha_2 : \alpha_2 \leq \alpha_1$. It is clear that each set $X^i = X^i(\alpha_2)$, $i = 1, 2, \ldots, k(\alpha_2)$, will contain at least one $\exists j \in \{1, 2, \ldots, k(\alpha_1)\} : X^j = X^j(\alpha_1)$, set. In other words, some clusters which are different in α_1-level will be in the same cluster in α_2 -level and the number of different clusters for which $max_{i \neq j}$ selection is possessed will decrease. Hence,

$$\max_{i \neq j} d\,(X^i(\alpha_1), X^j(\alpha_1)) \geq \max_{i \neq j} d\,(X^i(\alpha_2), X^j(\alpha_2)). \tag{4.47}$$

can be written. By taking (4.38) into account,

$$d_{max}^{out}(\alpha_1) \geq d_{max}^{out}(\alpha_2) \tag{4.48}$$

holds $\forall \alpha_2 \leq \alpha_1$ which completes the proof. □

Theorem 4.6. *(Nasibov & Ulutagay, 2007c) If for any partition of the set X holds*

$$\frac{d_{max}^{in}}{d_{min}^{out}} < \frac{1}{2} < \frac{d_{min}^{out} - d_{max}^{in}}{d_{max}^{out}}, \tag{4.49}$$

then FJP algorithm will determine this partition as optimal.

Theorem 4.7 given below has a better usage in comparison with Theorem 4.6 since it provides a wider upper bound.

Theorem 4.7. *If for any partition of the set X holds*

$$\frac{d_{max}^{in}}{d_{min}^{out}} < \frac{1}{2} < \frac{d_{min}^{out}}{d_{max}^{out} + d_{max}^{in}}, \tag{4.50}$$

then FJP algorithm will determine this partition as optimal.

Proof. Suppose that a hidden structure with possible partition that holds the inequalities (4.50) exists. Values of α, calculated by the cyclic use of the steps 4-6 of the FJP algorithm, are denoted as $\alpha_0, \alpha_1, \ldots, \alpha_t$. It is obvious that if a partition of the set X is recognized, then it holds that

$$d_{max}^{in} < d_{min}^{out} < d_{max}^{out}. \tag{4.51}$$

Hence, some sequent values α_z and α_{z+1} of the sequence α_i, $i = \{0, 1, \ldots, t\}$, generated by step 5 of the FJP algorithm, will be convenient to the distances d_{max}^{in} and d_{min}^{out}, and there there will be no value of the parameter $\alpha \in (\alpha_{z+1}, \alpha_z)$ which affects the structure of clusters.

Let’s denote the values of the relation T by $1 = \alpha^0, \alpha^1, \alpha^2,$ and α^3, proper to the distances $0 = d_0, d_{max}^{in}, d_{min}^{out},$ and d_{max}^{out}. With respect to the steps 7-9, the above considered partition is calculated in the FJP algorithm when

$$\alpha^1 - \alpha^2 = \Delta\alpha_z = \alpha_z - \alpha_{z+1} = \max_{i=0,\ldots,t-1} \Delta\alpha_i \tag{4.52}$$

Now, (4.52) can be proven when (4.50) holds.

Assume that (4.50) holds. Consider at first the inequality given below:

$$\frac{d_{max}^{in}}{d_{min}^{out}} < \frac{1}{2}. \tag{4.53}$$

Thus following sequence of inequalities holds:

$$\begin{aligned} 2\,d_{max}^{in} < d_{min}^{out} \Rightarrow d_{max}^{in} < d_{min}^{out} - d_{max}^{in} \\ \Rightarrow d_{max}^{in} - d_0 < d_{min}^{out} - d_{max}^{in}. \end{aligned} \tag{4.54}$$

Consequently, by taking (4.3) into account,

$$\alpha^0 - \alpha^1 = (1 - \frac{d_0}{d_{max}}) - (1 - \frac{d_{max}^{in}}{d_{max}}) = \frac{d_{max}^{in} - d_0}{d_{max}} \tag{4.55}$$

$$\alpha^1 - \alpha^2 = (1 - \frac{d_{max}^{in}}{d_{max}}) - (1 - \frac{d_{min}^{out}}{d_{max}}) = \frac{d_{min}^{out} - d_{max}^{in}}{d_{max}} \tag{4.56}$$

are obtained. By taking (4.54) into account in (4.55) and (4.56),

$$\alpha^0 - \alpha^1 < \alpha^1 - \alpha^2 \tag{4.57}$$

is obtained. Then, $\forall \alpha_i, \alpha_j \in [\alpha^1, \alpha^0]$,

$$\mid \alpha_i - \alpha_j \mid \leq \alpha^0 - \alpha^1 < \alpha^1 - \alpha^2 \tag{4.58}$$

holds that implies $\forall \alpha_i, \alpha_{i+1} \in [\alpha^1, \alpha^0]$,

$$\alpha_i - \alpha_{i+1} < \alpha^1 - \alpha^2 \tag{4.59}$$

Now, consider the second part of (4.50). Let

$$\frac{1}{2} < \frac{d^{out}_{min}}{d^{out}_{max} + d^{in}_{max}}, \tag{4.60}$$

holds from which follows

$$d^{out}_{max} + d^{in}_{max} < 2d^{out}_{min} \tag{4.61}$$

i.e.

$$d^{out}_{max} - d^{out}_{min} < d^{out}_{min} - d^{in}_{max}. \tag{4.62}$$

Thus, the following can be written:

$$\alpha^2 - \alpha^3 = (1 - \frac{d^{out}_{min}}{d_{max}}) - (1 - \frac{d^{out}_{max}}{d_{max}}) = \frac{d^{out}_{max} - d^{out}_{min}}{d_{max}} \tag{4.63}$$

Taking (4.55) and (4.63) into account, from (4.62) it follows that

$$\alpha^2 - \alpha^3 < \alpha^1 - \alpha^2. \tag{4.64}$$

Then $\forall \alpha_i, \alpha_j \in [\alpha^3, \alpha^2]$,

$$\mid \alpha_i - \alpha_j \mid \leq \alpha^2 - \alpha^3 < \alpha^1 - \alpha^2, \tag{4.65}$$

holds, *i.e.*

$$\alpha_i - \alpha_{i+1} < \alpha^1 - \alpha^2. \tag{4.66}$$

Thus from (4.59) and (4.66)

$$\alpha^1 - \alpha^2 = \max_{i=0,\ldots,t-1} \Delta\alpha_i \tag{4.67}$$

follows which completes the proof. □

4.2 NRFJP Algorithm

Although FJP algorithm has many advantages, it is unsuccessful in clustering data with noise points. In this section Noise-Robust FJP (NRFJP) algorithm which is robust through noises is proposed (Nasibov & Ulutagay, 2007a). In this algorithm each point for which certain ε_1 fuzzy neighborhood cardinality is smaller than a certain ε_2 threshold is perceived as noise. Note that, by changing the ε_1 and ε_2 parameters, it is possible to change the sensitivity of the NRFJP algorithm through noises. It is obvious that in order to turn off the sensitivity of the NRFJP through noises, it is enough to make $\varepsilon_2 = 0$. Finally, in the result of clustering, it is be possible to assign the noise points to the nearest class.

Let $N(x)$ denote a fuzzy neighborhood set of given point $x \in X$ on the base of the fuzzy relation T , *i.e.*

$$N(x) = \{(y, T(x,y)) | y \in X\}. \tag{4.68}$$

Let

$$N(x,\varepsilon_1) = \{y \in X | T(x,y) \geq \varepsilon_1\} \tag{4.69}$$

be the ε_1-level set of $N(x)$, *i.e.* fuzzy ε_1-neighborhood set of the point $x \in X$.

Definition 4.5. A point $x \in X$ is called a *noise point* with parameters ε_1, ε_2 for given $\varepsilon_1 \geq 0$ and $\varepsilon_2 \geq 0$, if $card\, N(x,\varepsilon_1) < \varepsilon_2$ is satisfied, where

$$card\, N(x,\varepsilon_1) = \sum_{y \in N(x,\varepsilon_1)} T(x,y) \tag{4.70}$$

is the *fuzzy cardinality* of the set $N(x,\varepsilon_1)$.

NRFJP Algorithm.

Step 1. Compute:

$d_{ij} := d(x_i, x_j), \quad i, j = 1, \ldots, n;$

$d_{max} := \max\limits_{i,j=1,\ldots,n} d_{ij};$

$\varepsilon := 0.01 \cdot \min\limits_{i,j=1,\ldots,n} d_{ij}.$

Set up the values ε_1 and ε_2;

Let $\alpha_0 := 1$;

Step 2. Compute the fuzzy relation $T_{ij} := 1 - \dfrac{d_{ij}}{d_{max}}, \; i, j = 1, \ldots, n;$

Compute the transitive closure $\hat{T}$ of the relation T;

Step 3. Call the procedure $NoiseFilter(\varepsilon_1, \varepsilon_2)$ to divide initial data set X into core X_{core} and noise X_{noise} sets, *i.e.*, $X = X_{core} \cup X_{noise}$ and $X_{core} \cap X_{noise} = \emptyset$.

Step 4. Let n_c = count of elements X_{core};

Set: $y_i := x_i, \; i = 1, \ldots, n_c; \; t := 1; \; k := n_c;$

Step 5. Compute: $d(y_i, y_j) = \min\{d(x', x'') | x' \in y_i, x'' \in y_j\}, \; i, j = i, \ldots, k;$

$d_t := \min\limits_{i \neq j} d(y_i, y_j); \quad \alpha_t := \max\left\{1 - \dfrac{d_t + \varepsilon}{d_{max}}, 0\right\};$

Step 6. Call the procedure $Clusters(\alpha_t)$ to calculate fuzzy α_t -joint sets $X^1, X^2, \ldots, X^k$, with conical fuzzy points $\left(x_i, \dfrac{d_{max}}{2}\right)$, $i = 1, \ldots, n_c$, and to constitute number k of these sets with current value α_t;

Step 7. If $k > 1$, then set $y_i : X^i, \; i = i, \ldots, k; \; t = t + 1$ and go to Step 5;

If $k = 1$, then go to Step 8.

Step 8. Compute: $\Delta\alpha_i := \alpha_i - \alpha i + 1, \; i = \overline{0, t-1};$

$z := arg \max\limits_{i=0,\ldots,t-1} \Delta\alpha_i; \quad \bar{\alpha} := \alpha_z - \varepsilon;$

Step 9. Call the procedure $Clusters(\bar{\alpha})$ with parameter $\bar{\alpha}$;

Step10. $\bar{\alpha}$ is the optimal membership degree of clustering;

$\bar{k}$ is the optimal number of clusters;

$X^1, X^2, \ldots, X^k$ is the partition of the set X.

```
Step11. For each element $x \in X_{noise}$ repeat step 12;

Step12. Compute: $k^* = arg\,min\{dist(x,X^k) \mid k = 1,\ldots,\bar{k}\}$;
        Assign $x$ to the $X^{k^*}$.

End.
```

The procedure $NoiseFilter(\varepsilon_1,\varepsilon_2)$ is used to divide initial data set X into two disjunctive core X_{core} and noise X_{noise} sets, *i.e.* $X = X_{core} \cup X_{noise}$ and $X_{core} \cap X_{noise} = \emptyset$.

```
Procedure NoiseFilter $(\varepsilon_1,\varepsilon_2)$.
Input parameters: $\varepsilon_1$ and $\varepsilon_2$;
Output parameters: The sets $X_{core}$ and $X_{noise}$;

Step 1. Let $X \equiv \{x_1,x_2,\ldots,x_n\}$ is the set of initial points $X_{noise} \neq \emptyset$;

Step 2. For each element $x \in X$ repeat the steps 3 and 4:

Step 3. Calculate $CardN(x,\alpha_1) = \sum_{y \in N(x,\varepsilon_1)} T(x_i,y_i)$;

Step 4. If $CardN(x,\alpha_1) < \alpha_2$, then mark $x$ as noise point, i.e. $X_{noise} = X_{noise} \cup \{x\}$;

Step 5. Let $X_{core} = X \backslash X_{noise}$;

Step 6. Return the sets $X_{core}$ and $X_{noise}$.

End.
```

4.2.1 Adjusting the Optimal Values of the Parameters of NRFJP

Prior to the theory of fuzzy sets, two principal measure of uncertainty were recognized. One of them, proposed by Hartley (1928), is based solely on the classical set theory. The other, introduced by Shannon (1948), is formulated in terms of probability theory. Both of these measures pertain to some aspects of ambiguity, as opposed to vagueness or fuzziness. However, each measures a different aspect of ambiguity: Hartley's measure pertains to nonspecificity, Shannon's measure to conflict or dissonance in evidence (Klir & Folger, 1988).

Both Hartley and Shannon introduced their measures for the purpose of measuring information in terms of uncertainty. Therefore, these measures are often referred to as *measures of information*. It has been more common, however, to refer to the measure invented by Shannon as the *Shannon entropy*. The name entropy was suggested by Shannon himself, presumably because of a similarity in the mathematical form between his measure and that of physical entropy as defined in certain formulation of statistical mechanics.

The Shannon entropy, which is a measure of uncertainty and information formulated in terms of probability theory, is expressed by the function

$$H\big(p(x) \mid x \in X\big) = -\sum_{x \in X} p(x) log_2 \, p(x), \tag{4.71}$$

where $p(x) \mid x \in X$ is a probability distribution on a finite set X. It is thus a function of the form

$$H : \mathbb{P} \rightarrow [0, \infty),$$

where $\mathbb{P}$ denotes the set of all probability distributions on finite sets.

Shannon entropy was considered for many years to be the only feasible basis for information theory. It has certainly dominated the literature on information theory since it was proposed by Shannon in 1948. Hartley information, which is in a fact a predecessor of Shannon entropy, is rarely mentioned in the current literature. When it is mentioned, it is almost always given one of two probabilistic interpretations. In the first, it is viewed as a measure that only distinguishes between zero and nonzero probabilities in the given probability distribution, that is, a measure that is totally insensitive to the actual values of the probabilities. It is derived from Shannon entropy by replacing any nonzero probability in the probability distribution with one.

The second probabilistic interpretation views Hartley information as equivalent to

Shannon entropy under the assumption that all elements of the set X are equally probable. In this case, the equal probabilities are $1/|X|$. When we substitute them for $p(x)$ in Formula (4.71), we readily obtain the Hartley information $log_2|X|$.

Based on the adjustment of ε_1 and ε_2 parameters, NRFJP algorithm could result in different partitions. If the parameters are not set well, clustering results could be incorrect. In this part Shannon entropy is applied in order to obtain the optimal value of ε_1 parameter (Nasibov & Ulutagay, 2006c).

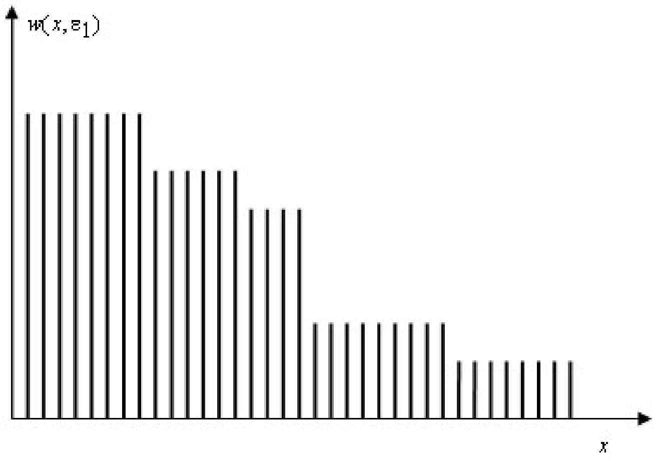


Figure 4.4 Neighborhood-density of points.

In order to evaluate the neighborhood-density of a point based on the ε_1 parameter, the following function can be used:

$$w(x_i,\varepsilon_1) = Card\,N(x_i,\varepsilon_1) = \sum_{y\in N(x,\varepsilon_1)} T(x_i,y_i), \quad i = 1,...,N. \tag{4.72}$$

Then $w(x_i,\varepsilon_1)$ values are normalized (Figure 4.4):

$$w'(x_i,\varepsilon_1) = \frac{w(x_i,\varepsilon_1)}{\max\limits_i w(x_i,\varepsilon_1)}. \tag{4.73}$$

By selecting the ε_1 parameter, it is aimed to divide the data set into core and noise points distinctively. Entropy can be used as an indicator of the division. A greater value of the entropy indicates that the elements are approximately in the same location while a smaller value is an indication of a distinctive division. In this sense, the objective is to find an ε_1^* value that makes the entropy of the $w(x_i, \varepsilon_1)$ minimum based on ε_1 (Figure 4.5):

$$E(\varepsilon_1^*) = \min_{\varepsilon_1 \in [0,1]} E(\varepsilon_1) = \min_{\varepsilon_1 \in [0,1]} \left[-\sum_{i=1}^{N} w'(x_i, \varepsilon_1) \cdot ln\, w'(x_i, \varepsilon_1) \right] \tag{4.74}$$

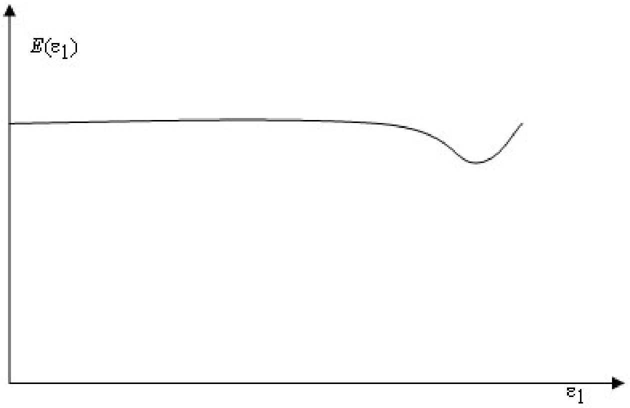


Figure 4.5 Entropy function depending on ε_1.

4.3 FN-DBSCAN Algorithm

The main objective of the crisp neighborhood-based clustering algorithms such as DBSCAN is to grow the interested cluster until its density is greater than a specified threshold (Ester et al., 1996; Sander et al., 1998). Namely, each point in the interested cluster should consist of at least a minimum number of points within a certain threshold. Such a method could be used in order to eliminate outlier points and to determine clusters with irregular shapes.

As mentioned previously, FN-DBSCAN algorithm integrates the advantages of DBSCAN and NRFJP algorithms in such a way that it combines the speed of the DBSCAN algorithm and robustness of the NRFJP algorithm (Nasibov, 2007; Nasibov & Ulutagay, 2008a, 2009).

The parameters ε and *MinPts* are used in crisp DBSCAN algorithm. The parameter ε is selected to provide the condition $0 \leq \varepsilon \leq d_{max}$, where

$$d_{max} = \max_{x_i, x_j \in X} d(x_i, x_j) \tag{4.75}$$

where $d(x_i, x_j)$ is the distance between the points x_i and x_j. However, since ε represents the direct value of the neighborhood radius, it takes values from different intervals corresponding to the scale of data. Such a case causes some problems in adjusting the values of ε. In order to eliminate this problem, we can normalize data $x_i = (x_{i1}, x_{i2}, ..., x_{im})$, $i = 1, ..., m$, and get an ε value from the interval [0,1] by using the following transformation

$$x_{ij} = \frac{x_{ij} - x_j^{min}}{(x_j^{max} - x_j^{min})\sqrt{m}}, \quad j = 1, ..., m \tag{4.76}$$

where

$$x_j^{min} = \min_{i=1,...,n} x_{ij} \quad and \quad x_j^{max} = \max_{i=1,...,n} x_{ij}, \quad j = 1, ..., m.$$

The multiplier $\frac{1}{\sqrt{m}}$ guarantees to agglomerate all of the points into a sphere with radius 1. So the condition $d_{max} \leq 1$ and respectively the condition $0 \leq \varepsilon \leq 1$ will be satisfied.

On the other hand, we use the formula given below in order to invert the value of *MinPts* to the interval [0,1] and indicate it by ξ parameter:

$$\xi = \frac{MinPts}{w^{max}} \tag{4.77}$$

where

$$w^{max} = \max_{i=1,...,n} w_i, \tag{4.78}$$

where w_i is the cardinality of the point x_i within a given radius. In general words, concerning fuzzy situation, w_i is the sum of the membership degrees of points to the neighborhood set within ε radius. Hence,

$$w_i = \sum_{k=1}^{n} N_{x_i}(x_k) \tag{4.79}$$

where $N_{x_i}(x_k)$ is the neighborhood degree of the point x_k to the point x_i. In crisp case, for obtaining $N_{x_i}(x_k)$, the following formula is used:

$$N_{x_i}(x_k) = \begin{cases} 1, & \text{if } d(x_i, x_k) \leq \varepsilon \\ 0, & \text{otherwise} \end{cases} \tag{4.80}$$

If it is enlarged to the fuzzy neighborhood case, N_{x_i} function could be formed as any neighborhood membership function.

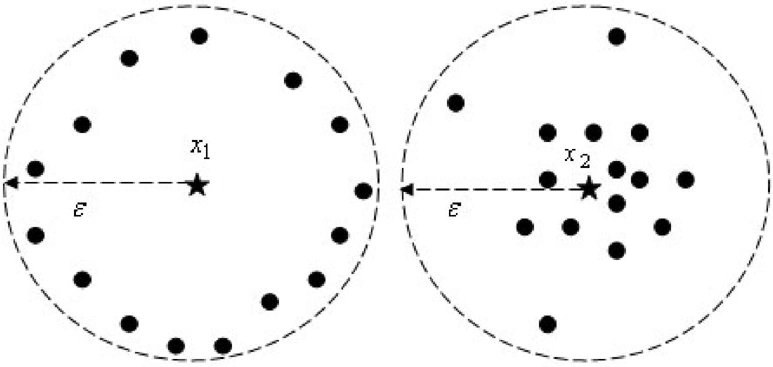


Figure 4.6 Points x_1 and x_2 are similar according to crisp neighborhood cardinality, but dissimilar according to fuzzy neighborhood cardinality.

One of the fundamental advantages of using fuzzy neighborhood function is that the neighborhood membership degrees of the points with different distances from core point also differ. But, in DBSCAN algorithm, there is no difference between points within the same neighborhood radius of core point (Figure 4.6). Because of that it could be more advantageous to use fuzzy neighborhood function instead of crisp neighborhood function.

Now let's investigate points x_1 and x_2 which have the same number of neighbors within a given ε radius (Figure 4.6). It is obvious that points x_1 and x_2 in Figure 4.6 are the same according to the crisp neighborhood relation used in DBSCAN. On the other hand, if fuzzy neighborhood function is used, point x_1 will have a higher membership degree of being a core point than that of point x_2. Such a neighborhood membership function used in NRFJP algorithm is as follows (Nasibov & Ulutagay, 2007a):

$$N_x(y) = \begin{cases} 1 - \dfrac{d(x,y)}{d_{max}}, & \text{if } d(x,y) \le \varepsilon \\ 0, & \text{otherwise} \end{cases} \tag{4.81}$$

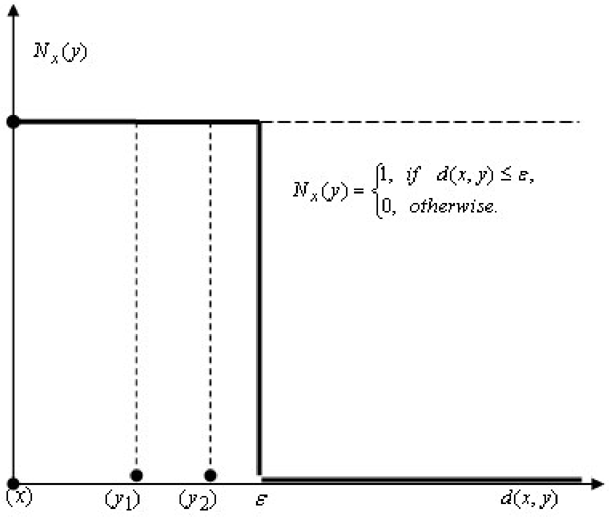
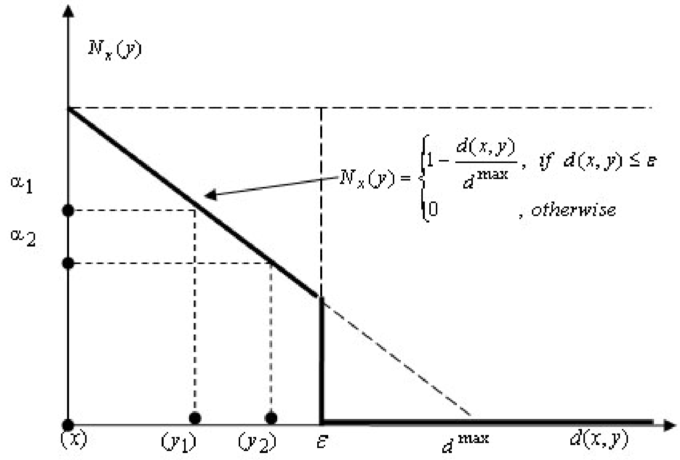


Figure 4.7 Neighborhood relation used in DBSCAN method.

The graphics of the membership functions given by Formulas (4.80) and (4.81) are shown in Figure 4.7 and Figure 4.8, respectively. As it is seen from Figure 4.7, points y_1 and y_2 have the same neighborhood membership degrees to the point x in case of crisp membership function. But they are different in fuzzy membership case, *i.e.* since, y_1 is closer to x than y_2 is, the membership degree of y_1, *i.e.* α_1, is higher than the membership degree of y_2, *i.e.* α_2 (Figure 4.8). Note that we can also deal with other neighborhood membership functions which might take the neighborhood relation into consideration more sensitively (Nasibov & Ulutagay, 2007a).

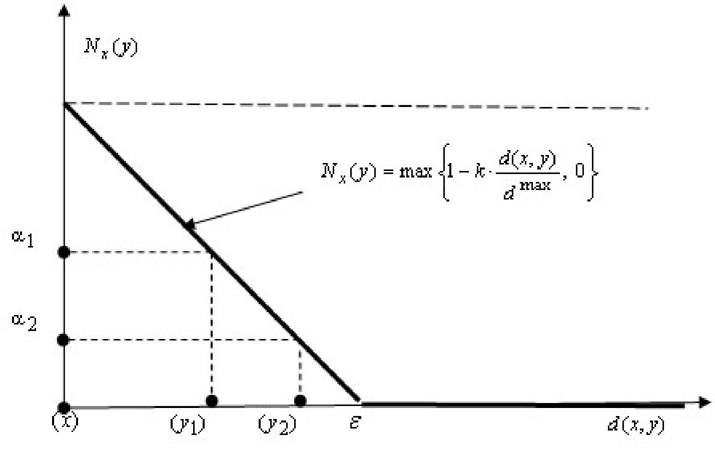
As seen from the figures, different sensitivity can be reached by using different membership functions in neighborhood analysis. For instance, the membership function in Figure 4.7 handles the points within radius ε_1 as identical whereas there is an obvious distinction in Figure 4.8. By using the formula given below, the difference becomes more evident (Figure 4.9):

$$N_x(y) = \max\left\{1 - k \cdot \frac{d(x,y)}{d_{max}}, 0\right\}. \tag{4.82}$$



$N_x(y)$
α_1
α_2
$N_x(y) = \begin{cases} 1 - \frac{d(x,y)}{d^{\max}}, & \text{if } d(x,y) \le \varepsilon \\ 0 & \text{, otherwise} \end{cases}$
(x)
(y_1)
(y_2)
ε
$d^{\max}$
$d(x,y)$

Figure 4.8 Neighborhood relation used in FJP method.



$N_x(y)$
$N_x(y) = \max\left\{1 - k \cdot \frac{d(x,y)}{d^{\max}}, 0\right\}$
α_1
α_2
(x)
(y_1)
(y_2)
ε
$d(x,y)$

Figure 4.9 Linear neighborhood relation.

However, the sensitivity is the same on near and far distance of the reference point. But by using the following exponential membership function shown in Figure 4.10, the sensitivity differs exponentially:

$$N_x(y) = \exp\left(-\left(k \cdot \frac{d(x,y)}{d_{max}}\right)^2\right). \tag{4.83}$$

Note that, in definition of the fuzzy point, if different nonlinear functions are used, different results can be found by the FJP algorithm. For example, consider a data set with 8 elements. If the membership function is linear as in Figure 4.11, the widest change interval in which the α-parameter does not affect the number of clusters is found as the interval number 3 and according to the working principle of the algorithm, such a situation is appropriate for two clusters.

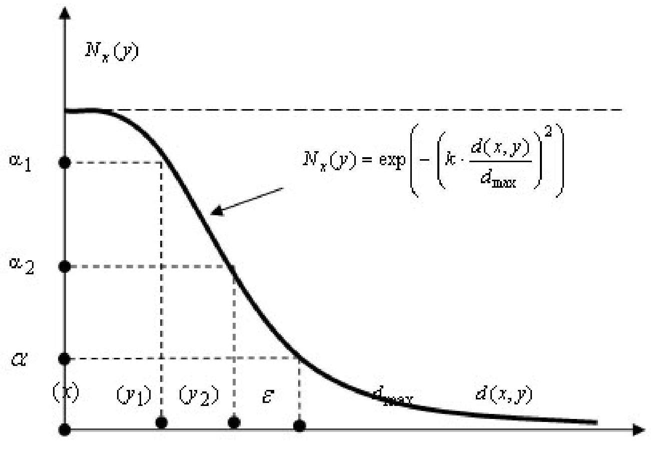


Figure 4.10 Exponential membership function.

However, when the membership function of the fuzzy points looks like bell-shaped as in Figure 4.12, the widest change interval found as number 2 and such a situation is suitable for four clusters. Thus, FJP algorithm finds two clusters if a membership function of a fuzzy point as in Figure 4.11 is used whereas it detects four clusters if a membership function of a fuzzy point as in Figure 4.12 is used.

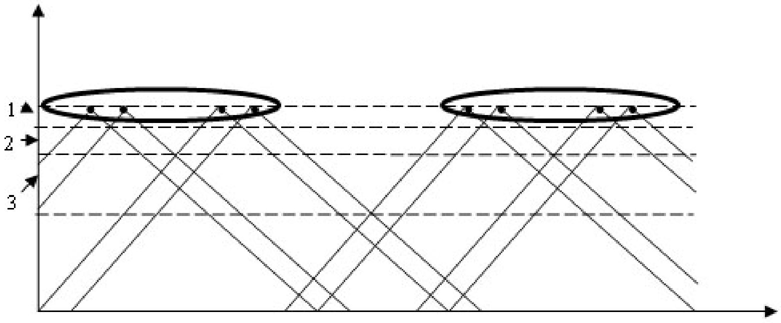


Figure 4.11 Linear membership function.

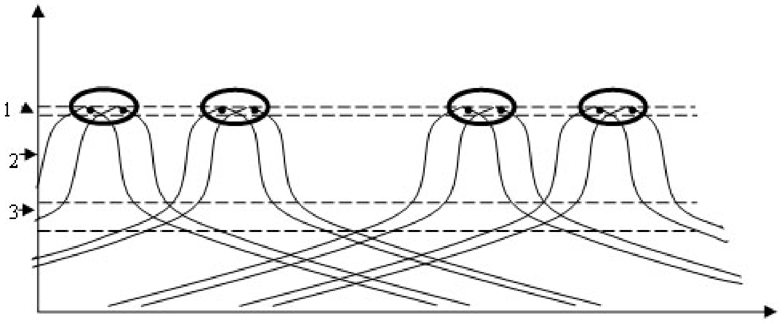


Figure 4.12 Bell-shaped membership function.

Note that in FJP-based algorithms, the lowest layer that is suitable for one cluster is not taken into consideration.

In order to explain the FN-DBSCAN algorithm the concepts given above will be defined for fuzzy sets approach. The main advantage of transforming the DBSCAN algorithm to the FN-DBSCAN algorithm by using the fuzzy sets theory is the usability of various neighborhood membership functions that regularize different neighborhood sensitivities. So the FN-DBSCAN method could be more robust to the variations of the density within clusters and to the scale of the (Nasibov & Ulutagay, 2008a) dataset.

Definition 4.6. The *fuzzy neighborhood* set of point $x \in X$ with ε_1 parameter is a fuzzy set determined as follows:

$$FN(x;\varepsilon_1) = \left\{(y, N_x(y)) \mid y \in X, N_x(y) \geq \varepsilon_1\right\} \tag{4.84}$$

where $N_x : X \longrightarrow [0,1]$ is any membership function that determines neighborhood relation between points.

ε_1 parameter used in Formula (4.84) determines the minimal threshold of neighborhood degrees. Note that if the neighborhood membership function is given in the form (4.81), then the parameter ε that fixes the maximal neighborhood radius and the parameter ε_1 in (4.84) that fixes the minimal neighborhood membership degree have the following relationship:

$$\varepsilon = d_{max}\,(1-\varepsilon_1) \tag{4.85}$$

Definition 4.7. A point x is called a *fuzzy core point* with parameters ε_1 and ε_2 if

$$card\,FN(x;\varepsilon_1) \equiv \sum_{y \in N(x;\varepsilon_1)} N_x(y) \geq \varepsilon_2 \tag{4.86}$$

holds for the point $x \in X$, where

$$N(x;\varepsilon_1) = \{y \in X | N_x(y) \geq \varepsilon_1\}. \tag{4.87}$$

determines the ε_1 level set of the fuzzy neighborhood set of the point x.

Definitions 4.6 and 4.7 in FN-DBSCAN algorithm are used instead of Definitions 2.1 and 2.2 in DBSCAN algorithm, respectively. Definition 4.7 differs from Definition 2.2 in such a way that it uses a level-based neighborhood set instead of a distance-based neighborhood set and it uses the concept of fuzzy cardinality instead of crisp cardinality in the determination of a core point.

In FN-DBSCAN algorithm, other points could be directly density-reachable only from a fuzzy core point as in the DBSCAN algorithm. Definitions 2.3 through 2.7 in DBSCAN algorithm are also used directly in FN-DBSCAN algorithm. By the guidance of these definitions, the pseudocode of the FN-DBSCAN algorithm on the basis of fuzzy neighborhood relation is given below:

FN-DBSCAN algorithm.

```
Step 1. Specify parameters $\varepsilon_1$ and $\varepsilon_2$.
Step 2. Mark all the points in the data set as unclassified. Set $t = 1$.
Step 3. Find an unclassified fuzzy core-point with parameters $\varepsilon_1$ and $\varepsilon_2$.
Step 4. Mark $p$ to be classified. Start a new cluster $C_t$ and assign $p$ to the cluster $C_t$.
Step 5. Create an empty set of seeds $S$. Find all the unclassified points in the set $N(p;\varepsilon_1)$
        and put all these points into the set $S$.
Step 6. Get a point $q$ in the set $S$, mark $q$ to be classified, assign $q$ to the cluster $C_t$, and
        remove $q$ from the set $S$.
Step 7. Check if $q$ is a fuzzy core-point with parameters $\varepsilon_1$ and $\varepsilon_2$; if so, add all the
        unclassified points in the set $N(q;\varepsilon_1)$ to the set $S$.
Step 8. Repeat step 6 through Step 7 until the set of seeds is empty.
Step 9. Find a new fuzzy core point $p$ with parameters $\varepsilon_1$ and $\varepsilon_2$, and repeat Step 4
        through Step 7.
Step10. Mark all the points, which do not belong to any cluster, as noise.
End.
```

Note that the FN-DBSCAN algorithm gives the same results as the DBSCAN algorithm does if the neighborhood membership function is handled as in the Formula (4.80). So the FN-DBSCAN algorithm can always be adjusted to give the better results than the DBSCAN algorithm by using an appropriate neighborhood membership function (Nasibov & Ulutagay, 2007a,b).

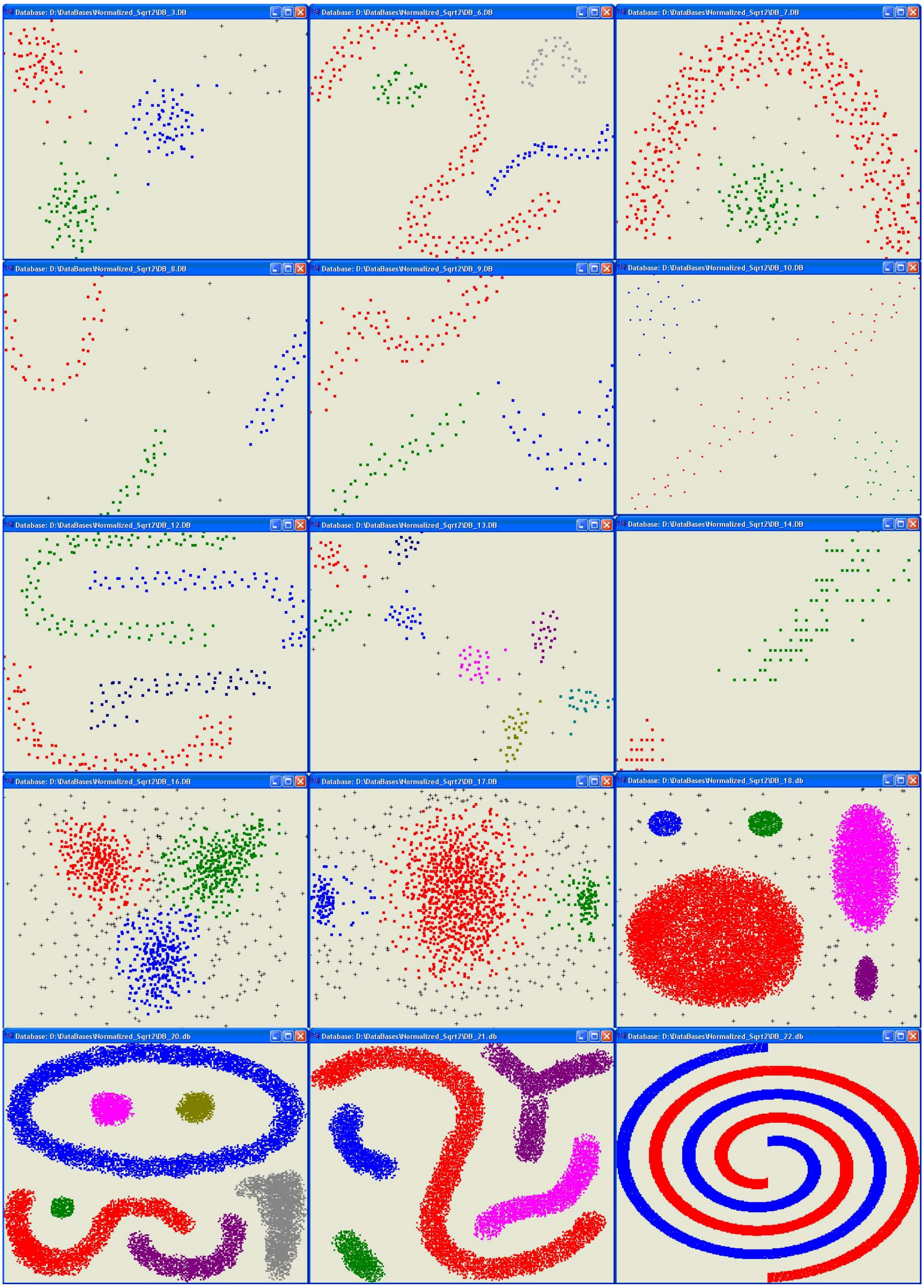

Database: D:\DataBases\Normalized_Sqrt2\DB_3.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_6.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_7.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_8.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_9.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_10.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_12.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_13.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_14.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_16.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_17.DB
Database: D:\DataBases\Normalized_Sqrt2\DB_18.db
Database: D:\DataBases\Normalized_Sqrt2\DB_20.db
Database: D:\DataBases\Normalized_Sqrt2\DB_21.db
Database: D:\DataBases\Normalized_Sqrt2\DB_22.db

Figure 4.13 Some of the data sets with various shapes and densities.

Example 4.1. In order to compare FN-DBSCAN algorithm which is based on fuzzy neighborhood analysis with DBSCAN algorithm which is based on crisp neighborhood analysis, we use 22 data sets with various shapes and densities. The data sets were obtained from the papers (Bensaid et al., 1996; Dong et al., 2006; Nasibov & Ulutagay, 2006b, 2007a) and some of them were simulated. Some of the data sets used in experiments are shown in Figure 4.13.

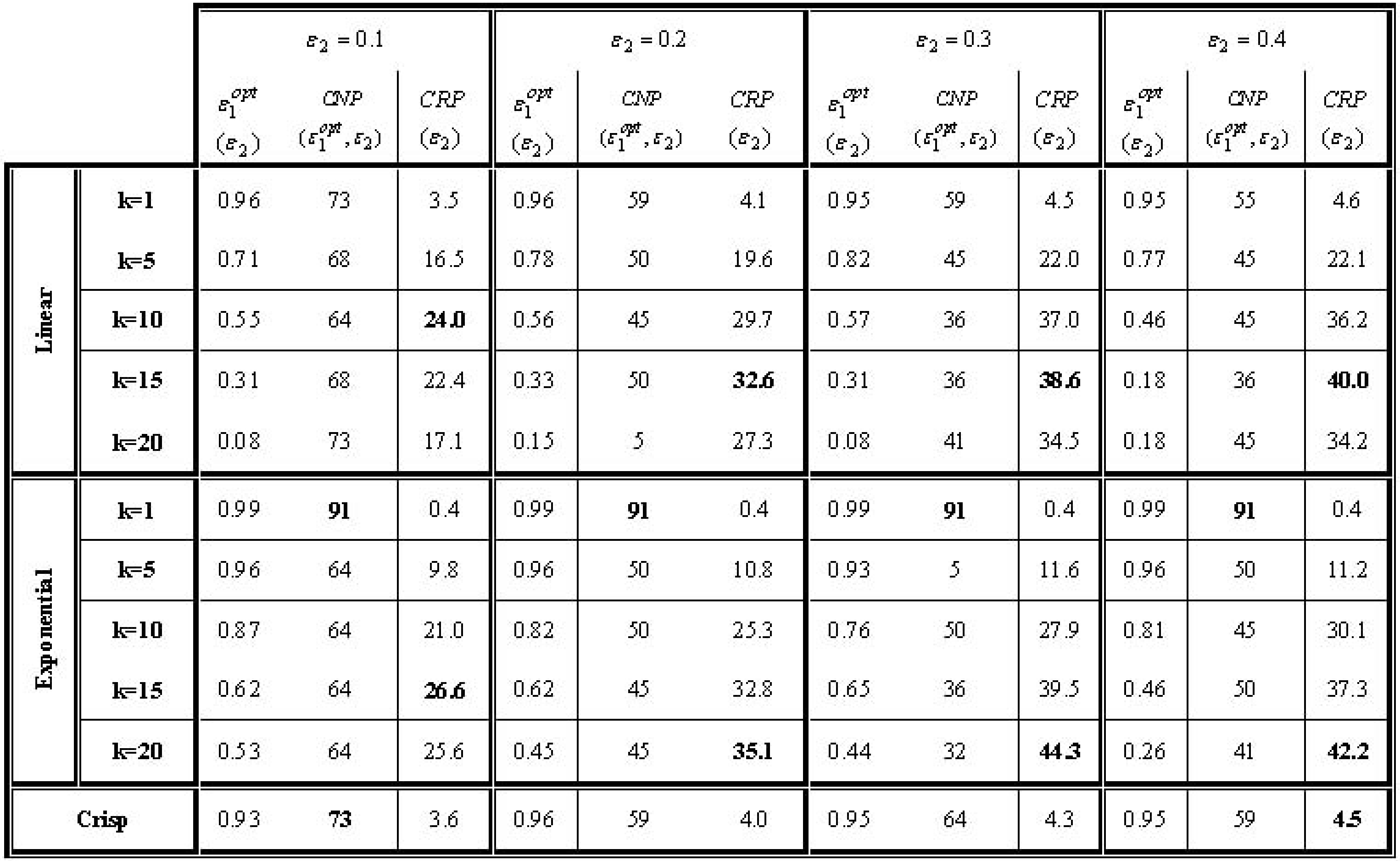
The codes for algorithms were developed in Borland C++ 6.0 SDK and the experiments were computed in Pentium(R)-D, 2.80 GHz, 2 GB RAM computers.

Correctness of the clustering results is validated by the expert visually. To evaluate the performances of the algorithms, we use the indicators given below, as "Correct Number Percent (CNP)" that indicates the percentage of the ratio of number of correct classified data sets to all number of data sets and "Correct Range Percent (CRP)" that indicates the percentage of correct result range of ε_1 parameter to the whole $[0,1]$ interval.

To formulate the CNP and CRP criteria the following notations are used:

- N- number of all data sets;
- $n(\varepsilon_1, \varepsilon_2)$- number of correct clustered data sets with fixed parameters ε_1 and ε_2 of the algorithm;
- $[\varepsilon_{1i}^{L}(\varepsilon_2), \varepsilon_{1i}^{U}(\varepsilon_2)]$- the "Correct Range (CR)", *i.e.* the widest continuous interval of the ε_1 parameter for fixed ε_2, in which algorithm gives correct results for the data set i, $i = 1, ..., N$, where $\varepsilon_{1i}^{L}(\varepsilon_2)$ is the lower bound, and $\varepsilon_{1i}^{U}(\varepsilon_2)$ is the upper bound of the interval;
- $\varepsilon_1^{opt}(\varepsilon_2)$- the value of the ε_1 parameter in which the number of correct classified data sets is maximum for fixed ε_2.

Table 4.1 Indicators of ε_1 parameter for various *k* and ε_2 values for crisp, linear and exponential membership functions.



		$\varepsilon_2 = 0.1$			$\varepsilon_2 = 0.2$			$\varepsilon_2 = 0.3$			$\varepsilon_2 = 0.4$		
		ε_1^{opt} (ε_2)	*CNP* $(\varepsilon_1^{opt}, \varepsilon_2)$	*CRP* (ε_2)	ε_1^{opt} (ε_2)	*CNP* $(\varepsilon_1^{opt}, \varepsilon_2)$	*CRP* (ε_2)	ε_1^{opt} (ε_2)	*CNP* $(\varepsilon_1^{opt}, \varepsilon_2)$	*CRP* (ε_2)	ε_1^{opt} (ε_2)	*CNP* $(\varepsilon_1^{opt}, \varepsilon_2)$	*CRP* (ε_2)
Linear	**k=1**	0.96	73	3.5	0.96	59	4.1	0.95	59	4.5	0.95	55	4.6
	k=5	0.71	68	16.5	0.78	50	19.6	0.82	45	22.0	0.77	45	22.1
	k=10	0.55	64	**24.0**	0.56	45	29.7	0.57	36	37.0	0.46	45	36.2
	k=15	0.31	68	22.4	0.33	50	**32.6**	0.31	36	**38.6**	0.18	36	**40.0**
	k=20	0.08	73	17.1	0.15	5	27.3	0.08	41	34.5	0.18	45	34.2
Exponential	**k=1**	0.99	**91**	0.4	0.99	**91**	0.4	0.99	**91**	0.4	0.99	**91**	0.4
	k=5	0.96	64	9.8	0.96	50	10.8	0.93	5	11.6	0.96	50	11.2
	k=10	0.87	64	21.0	0.82	50	25.3	0.76	50	27.9	0.81	45	30.1
	k=15	0.62	64	**26.6**	0.62	45	32.8	0.65	36	39.5	0.46	50	37.3
	k=20	0.53	64	25.6	0.45	45	**35.1**	0.44	32	**44.3**	0.26	41	**42.2**
Crisp		0.93	**73**	3.6	0.96	59	4.0	0.95	64	4.3	0.95	59	**4.5**

So the CNP and CRP indicators are calculated as follows:

$$CNP(\varepsilon_1,\varepsilon_2) = \frac{n(\varepsilon_1,\varepsilon_2)}{N} \cdot 100\%\,, \tag{4.88}$$

$$CRP_i(\varepsilon_2) = \mid \varepsilon_{1i}^U(\varepsilon_2) - \varepsilon_{1i}^L(\varepsilon_2) \mid \cdot 100\%\,, \tag{4.89}$$

$$CRP(\varepsilon_2) = \frac{\sum_{i=1}^{N} CRP_i(\varepsilon_2)}{N}\,. \tag{4.90}$$

The experimental results for 22 data sets are given in Table 4.1. As it is seen, $k = 1$ for linear case has approximately same results with the crisp case. However, the results vary when different neighborhood membership functions are used. The results of the algorithm ameliorate when we change the value of the parameter k from 1 through 20, and then they deteriorate. The best results are found in linear case for $k = 15$.

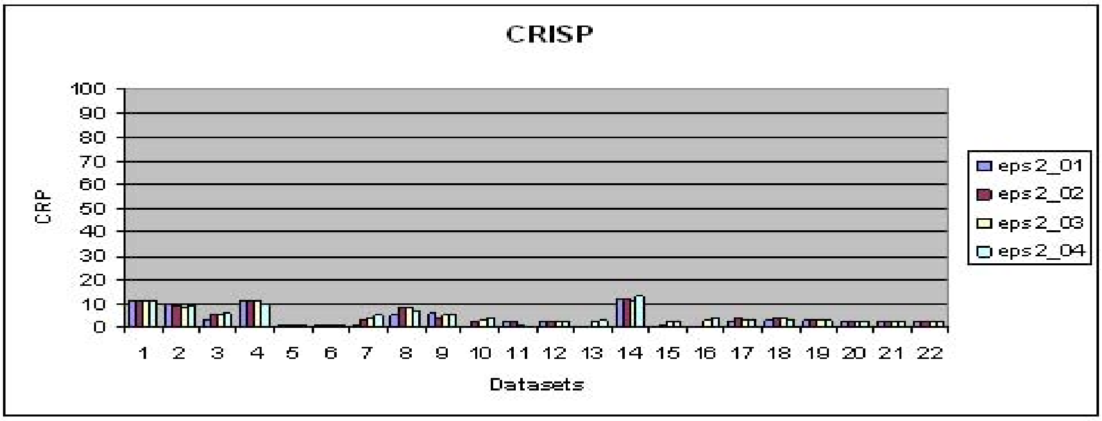


Figure 4.14 $CRP(\varepsilon_2)$ for $\varepsilon_2 = 0.1, 0.2, 0.3, 0.4$ in crisp membership case.

We can say that results generally get better when exponential neighborhood function is used. The best results in exponential case are found for $k = 20$. If we compare this result with the best results of linear case and the best results of the crisp case, we can conclude that exponential membership function is very effective. Good results are marked in bold in Table 4.1. In order to show these results visually, comparisons are given as histograms for various values of the parameters (Figures 4.14-4.17).

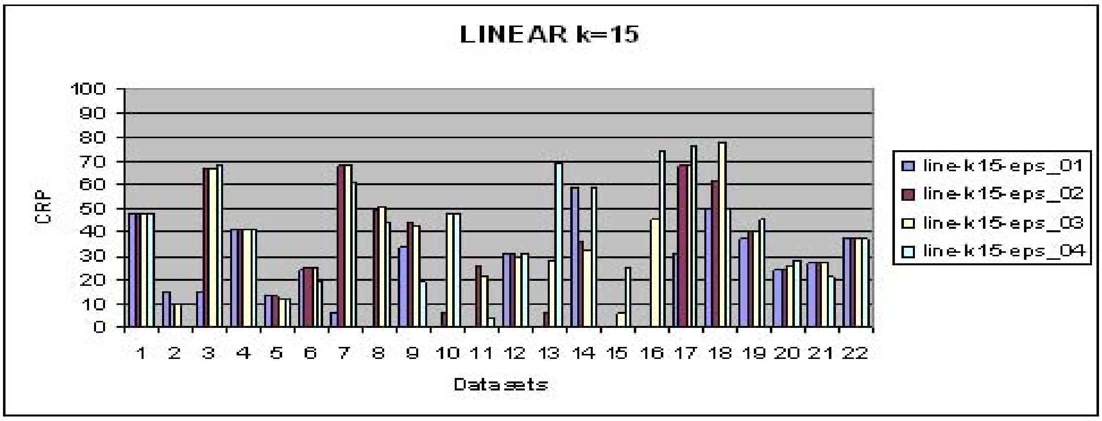


Figure 4.15 $CRP(\varepsilon_2)$ for $\varepsilon_2 = 0.1, 0.2, 0.3, 0.4$ and $k = 15$ in linear membership case.

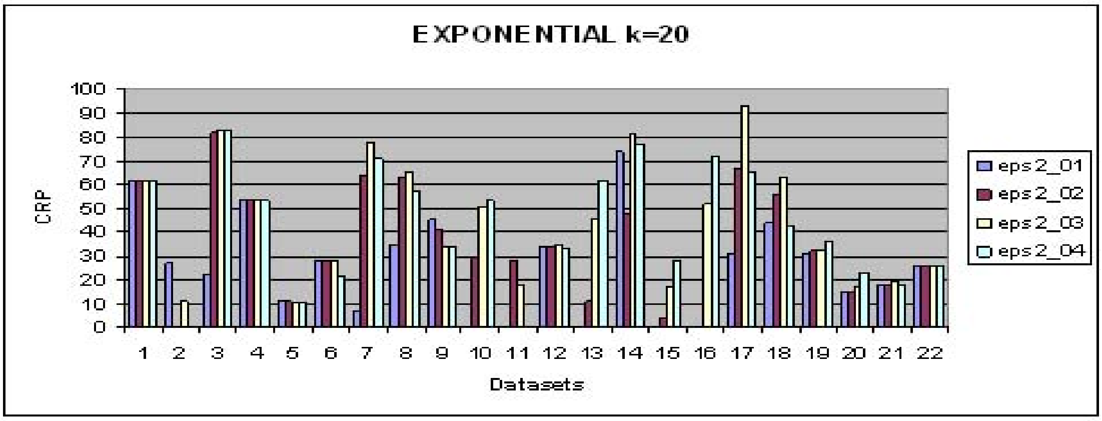


Figure 4.16 $CRP(\varepsilon_2)$ for $\varepsilon_2 = 0.1, 0.2, 0.3, 0.4$ and $k = 20$ in exponential membership case.

As it is seen from the histogram, crisp neighborhood membership function results in correct partitions in a narrow range of ε_1 parameter. However, in fuzzy neighborhood membership function case, the results of FN-DBSCAN algorithm are stable in a wider range of ε_1 parameter. It is obvious that the best results are found by using exponential membership function given in Equation 4.83. Otherwise, the worst results are observed by using crisp membership function given in Equation 4.80. Approximately the same results are obtained by using Equation 4.81 and 4.82 for neighborhood membership functions. In exponential case, CRP indicator for ε_1 parameter is about 25-45% while this index is only 3-5% in crisp case.

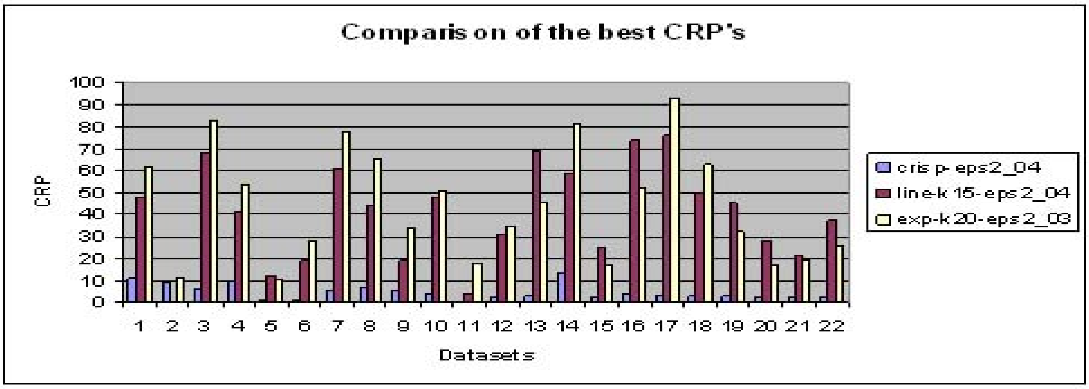


Figure 4.17 Comparison of the best results of the $CRP(\varepsilon_2)$, a) for $\varepsilon_2 = 0.4$ in crisp membership case, b) for $k = 15$, $\varepsilon_2 = 0.4$ in linear membership case, c) for $k = 20$, $\varepsilon_2 = 0.3$ in exponential membership case.

In the second indicator the least squares method of regression is used. Hence, we look for such an optimal value of the ε_1 parameter that its distance defined as follows from its CR will be minimum (Figure 4.18):

$$d(\varepsilon_1, [\varepsilon_{1i}^L, \varepsilon_{1i}^U]) = \begin{cases} 1, & \text{if } \varepsilon_1 \notin [\varepsilon_{1i}^L, \varepsilon_{1i}^U] \\ 0, & \text{otherwise} \end{cases} . \tag{4.91}$$

In fact,

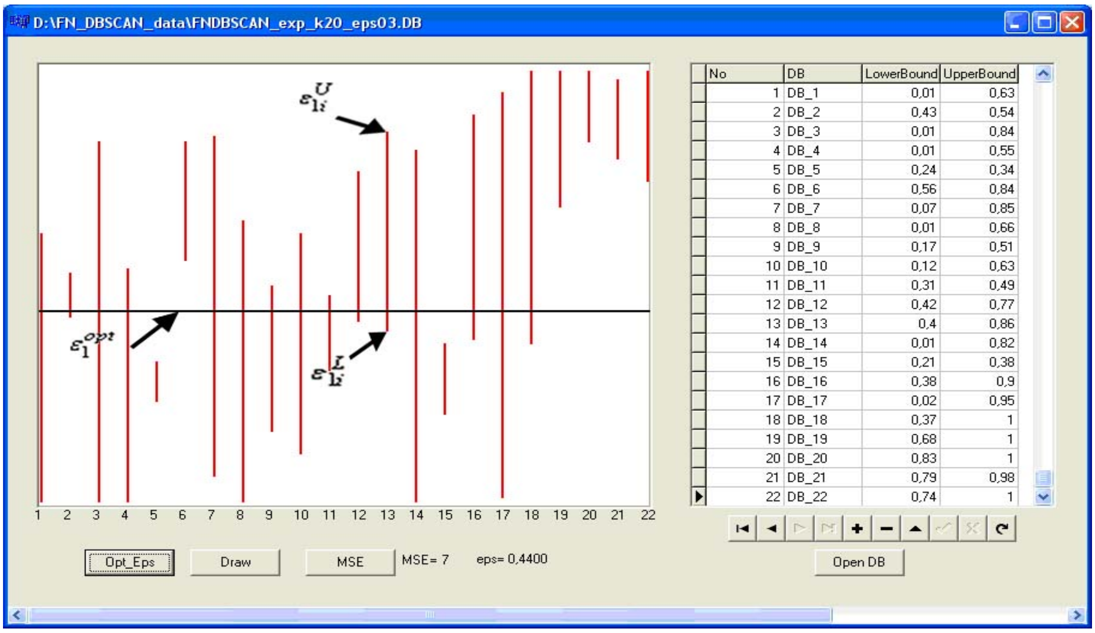
$$f(\varepsilon_1;\varepsilon_2) = N - n(\varepsilon_1;\varepsilon_2). \tag{4.92}$$

The optimal value of the ε_1 parameter at fixed value of the ε_2 parameter is obtained by the following optimization problem:

$$f(\varepsilon_1;\varepsilon_2) = \sum_{i=1}^{N} d\left(\varepsilon_1, [\varepsilon_{1i}^L, \varepsilon_{1i}^U]\right)^2 \quad \rightarrow \quad min \tag{4.93}$$

The solution of the problem (4.93) has been mentioned previously as $\varepsilon_1^{opt}(\varepsilon_2)$.

Corresponding optimal value $f(\varepsilon_1^{opt};\varepsilon_2)$ indicates the number of data sets for which the algorithm gives minimum incorrect results for the fixed value of the ε_2 parameter.

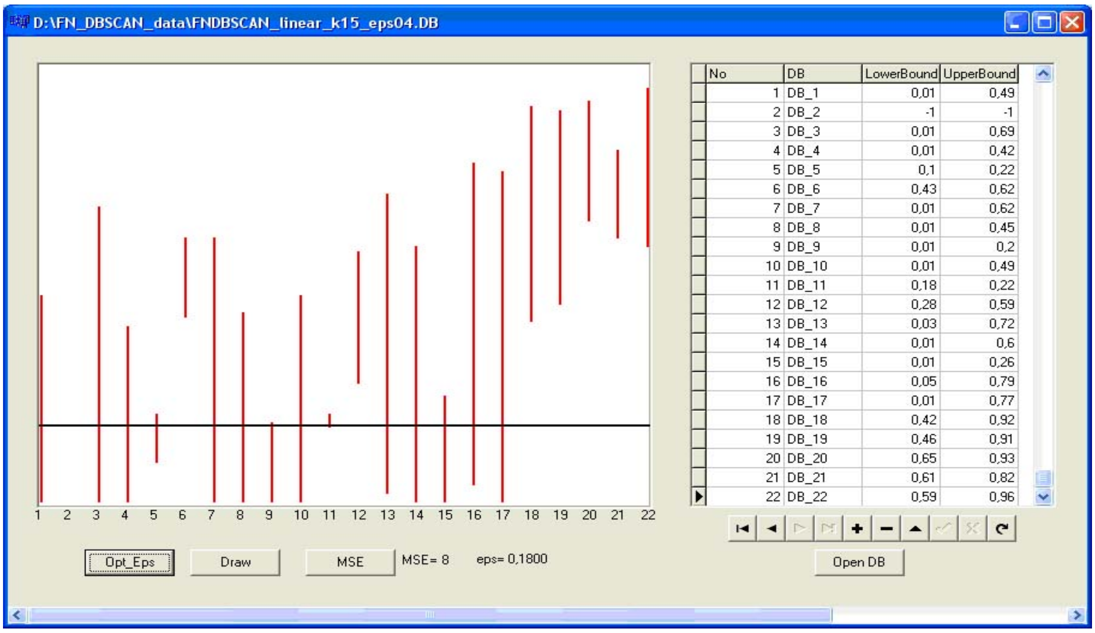


D:\FN_DBSCAN_data\FNDBSCAN_exp_k20_eps03.DB

MSE= 7 eps= 0,4400

No	DB	LowerBound	UpperBound
1	DB_1	0,01	0,63
2	DB_2	0,43	0,54
3	DB_3	0,01	0,84
4	DB_4	0,01	0,55
5	DB_5	0,24	0,34
6	DB_6	0,56	0,84
7	DB_7	0,07	0,85
8	DB_8	0,01	0,66
9	DB_9	0,17	0,51
10	DB_10	0,12	0,63
11	DB_11	0,31	0,49
12	DB_12	0,42	0,77
13	DB_13	0,4	0,86
14	DB_14	0,01	0,82
15	DB_15	0,21	0,38
16	DB_16	0,38	0,9
17	DB_17	0,02	0,95
18	DB_18	0,37	1
19	DB_19	0,68	1
20	DB_20	0,83	1
21	DB_21	0,79	0,98
22	DB_22	0,74	1

Figure 4.18 $[\varepsilon_{1i}^{L},\varepsilon_{1i}^{U}]$ intervals of ε_1^{opt} value for $k = 20$, $\varepsilon_2 = 0.3$ in exponential membership case.



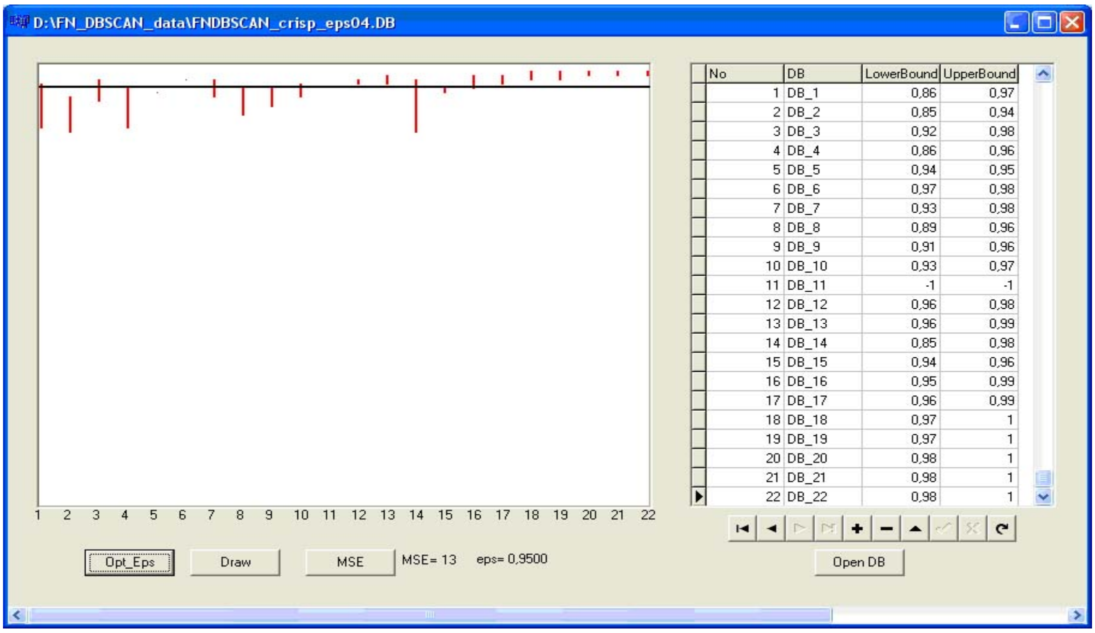
D:\FN_DBSCAN_data\FNDBSCAN_linear_k15_eps04.DB

MSE= 8 eps= 0,1800

No	DB	LowerBound	UpperBound
1	DB_1	0,01	0,49
2	DB_2	-1	-1
3	DB_3	0,01	0,69
4	DB_4	0,01	0,42
5	DB_5	0,1	0,22
6	DB_6	0,43	0,62
7	DB_7	0,01	0,62
8	DB_8	0,01	0,45
9	DB_9	0,01	0,2
10	DB_10	0,01	0,49
11	DB_11	0,18	0,22
12	DB_12	0,28	0,59
13	DB_13	0,03	0,72
14	DB_14	0,01	0,6
15	DB_15	0,01	0,26
16	DB_16	0,05	0,79
17	DB_17	0,01	0,77
18	DB_18	0,42	0,92
19	DB_19	0,46	0,91
20	DB_20	0,65	0,93
21	DB_21	0,61	0,82
22	DB_22	0,59	0,96

Figure 4.19 $[\varepsilon_{1i}^{L},\varepsilon_{1i}^{U}]$ intervals of ε_1^{opt} value for $k = 15$, $\varepsilon_2 = 0.4$ in linear membership case.

It is clear from Table 4.1 that in fuzzy neighborhood function cases, most of the results are better than that of crisp case. Moreover, we get the best results for $k = 15$ and $\varepsilon_2 = 0.4$ in linear case, and $k = 20$ and $\varepsilon_2 = 0.3$ in exponential case (Figures 4.19-4.20). The accuracy rate for the CNP indicator is also better than that of the crisp case, in other words the indicator is 91% for the fuzzy exponential ($k = 1$) case whereas it is less than 73% in all variants of the crisp case.



MSE= 13 eps= 0,9500

No	DB	LowerBound	UpperBound
1	DB_1	0,86	0,97
2	DB_2	0,85	0,94
3	DB_3	0,92	0,98
4	DB_4	0,86	0,96
5	DB_5	0,94	0,95
6	DB_6	0,97	0,98
7	DB_7	0,93	0,98
8	DB_8	0,89	0,96
9	DB_9	0,91	0,96
10	DB_10	0,93	0,97
11	DB_11	-1	-1
12	DB_12	0,96	0,98
13	DB_13	0,96	0,99
14	DB_14	0,85	0,98
15	DB_15	0,94	0,96
16	DB_16	0,95	0,99
17	DB_17	0,96	0,99
18	DB_18	0,97	1
19	DB_19	0,97	1
20	DB_20	0,98	1
21	DB_21	0,98	1
22	DB_22	0,98	1

Figure 4.20 $[\varepsilon_{1i}^{L}, \varepsilon_{1i}^{U}]$ intervals of ε_1^{opt} value for $\varepsilon_2 = 0.4$ in crisp membership case.

To sum up, we can conclude that for data sets with high density, greater values of the parameters k and ε_1 ($k = 15 \div 20$, $\varepsilon_1 = 0.90 \div 0.99$), and in data sets with low density, smaller values of these parameters should be preferred. We can also note that, for data sets with large number of noise points, greater values of the ε_2 parameter ($\varepsilon_2 = 0.3 \div 0.4$) give better results.

So, in this chapter, which forms the theoretical foundation of dissertation, new Noise-Robust FJP and Fuzzy Neighborhood DBSCAN algorithms are presented and comparative analysis that point out the advantages of the algorithms are performed.

CHAPTER FIVE
DATA COLLECTION TECHNIQUES

Living organisms are made up of many component systems- the human body, for example, includes the nervous system, the cardiovascular system, and the muscoloskeletal system, among others. Each system is made up of several subsystems that carry on many physiological processes. For example, the cardiac system performs the important task of rhythmic pumping of blood throughout the pulmonary system for oxygenation of the blood itself.

Physiological processes are complex phenomena, including nervous or hormonal stimulation and control; inputs and outputs that could be in the form of physical material, neurotransmitters, or information; and action that could be mechanical, electrical, or biochemical (Rangayyan, 2002). Most physiological processes are accompanied by or manifest themselves as signals that reflect their nature and activities. Such signals could be of many types, including biochemical in the form of hormones and neurotransmitters, electrical in the form of potential or current, and physical in the form of pressure or temperature (Akay, 2000).

The representation of biomedical signals in electronic form facilitates computer processing and analysis of data.

5.1 Electroencephalography

The summated neuronal activity of the brain recorded as minute electrical potentials from the human scalp is called the electroencephalogram (Akay, 2000). Conventionally, such potentials are recorded with three types of electrodes: scalp, cortiacal, and depth electrodes. For scalp recording, the electrodes are typically placed on the scalp in accordance with some internationally defined geometrical sites (Figure 5.1). Information derived from the depth electrodes and microelectrodes has shown that under normal circumstances conducted action potentials in axons contribute little to the surface EEG because they occur asynchronously in time in large number of axons, which run in many directions relative to the surface. Thus, their net influence on potential at the surface is negligible. Whether recorded from the scalp, cortex, or depths of the brain, these biopotentials represent a superposition of the volume conductor fields produced by a variety of active neuronal current generators.

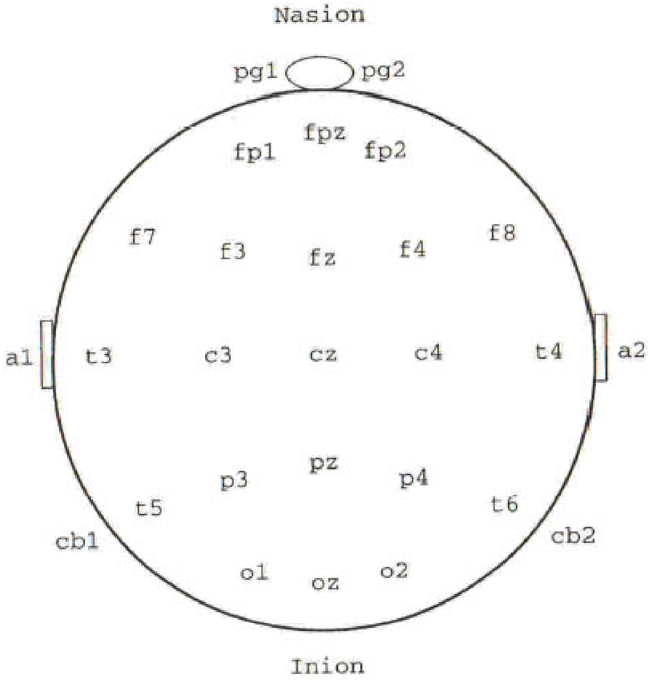


Figure 5.1 The 10-20 system of electrode placement for EEG recording.

The EEG signal contains information regarding changes in the electrical potential of the brain obtained from a given set of recordings. These data include the characteristic waveforms with accompanying variations in amplitude, frequency, phase, and so on as well as brief occurrence of electrical patterns.

The EEG signal patterns are modulated by a wide range of variables, including biochemical, metabolic, circulatory, humoral, neuroelectric, and behavioral factors. The EEG is extremely difficult for an untrained observer to interpret, partially because of the spatial mapping of functions onto different regions of the brain and electrode placement. The EEG is used routinely in the diagnosis of neurological disorders such as epilepsy, stroke, and brain damage; it is also used in sleep and drug research and the investigation of the pscyhiatric disorders.

EEG signals exhibit several patterns of rhythmic or periodic activity. The commonly used terms for EEG frequency f bands are (Figures 5.2-5.4):

- Delta (δ) : $0.5 \leq f \leq 4$ Hz;
- Theta (θ) : $4 \leq f \leq 8$ Hz;
- Alpha (α) : $8 \leq f \leq 13$ Hz;
- Beta (β) : $f \geq 13$ Hz.

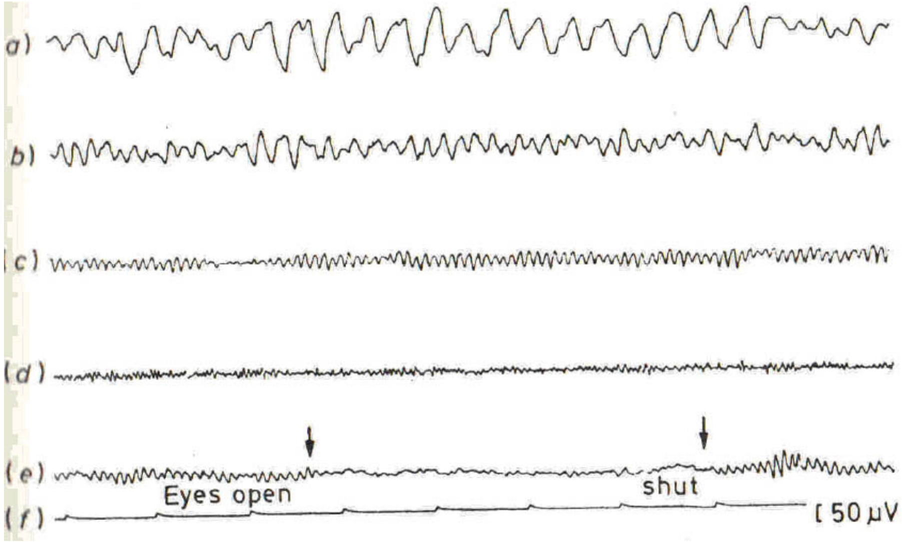


Figure 5.2 From top to bottom: (a) delta rhythm; (b) theta rhythm; (c) alpha rhythm; (d) beta rhythm; (e) blocking of the alpha rhythm by eye opening; (f) 1 s time markers and 50 μv marker.

The alpha wave is replaced by slower rhythms at various stages of sleep. Theta waves appear at the beginning stages of the sleep; delta waves appear at deep-sleep stages. High-frequency beta waves appear as background activity in tense and anxious subjects. The depression or absence of the normal (expected) rhythm in a certain state of the subject could indicate abnormality. The presence of delta or theta (slow) waves in wakeful adult would be considered to be abnormal. Focal brain injury and tumors lead to abnormal slow waves in the corresponding regions. Unilateral depression (left-right asymmetry) of a rhythm could indicate disturbances in cortical pathways. Spikes and sharp waves could indicate the presence of epileptogenic regions in the corresponding parts of the brain.

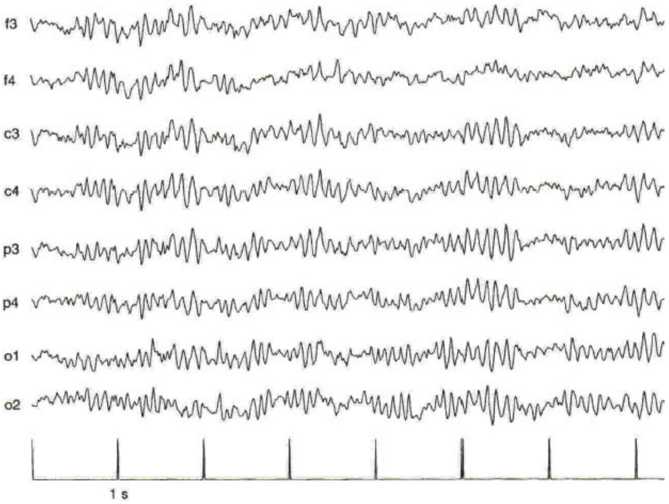


Figure 5.3 Eight channels of the EEG of a subject displaying alpha rhythm.

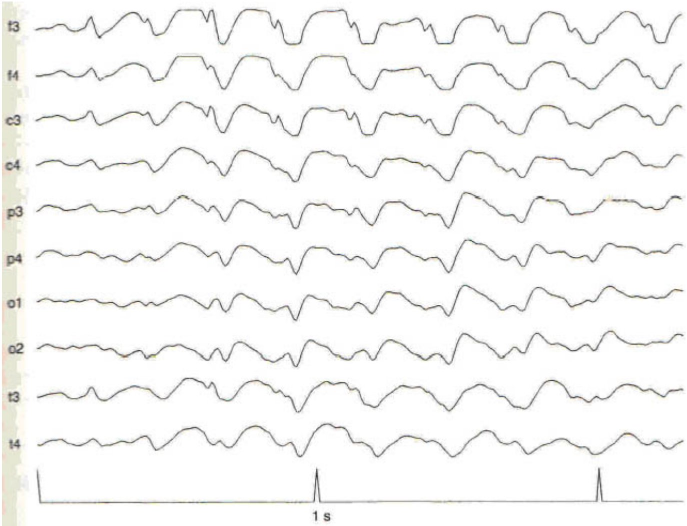


Figure 5.4 Ten channels of the EEG of a subject displaying spike-and-wave complexes.

5.2 Bispectral Index

Bispectral analysis is a statistical technique that allows study of phenomena with nonlinear character, such as surf beats and wave breaking (Johansen & Sebel, 2000). Bispectral analysis provides a description to a continuous pseudo-randomly varying signal (*e.g.*, EEG) that is an alternative to other conventional power spectral analysis techniques derived from fast Fourier transformation. Bispectral analysis is computationally intensive, and it was not until fast microprocessors were developed that online bispectral analysis of the EEG in the operating room became possible.

Conventional analysis of the EEG using fast Fourier transformation produces information regarding the power, frequency, and the phase of the EEG signal. Typical displays, such as the compressed spectral array, graph power and frequency information and discard the phase information. Bispectral analysis represents a different description of the EEG in that interfrequency phase relations are measured, *i.e.*, the bispectrum quantifies relations among the underlying sinusoidal components of the EEG. The data contained in both the bispectral analysis and conventional frequency-power analyses of the EEG are used to create the proprietary parameter of the bispectral index, or BIS.

The BIS integrates various EEG descriptors into a single variable. The mixture of subparameters of EEG activity was derived empirically from a prospectively collected database of anesthetized volunteers with measures of clinically relevant sedative endpoints and hypnotic drug concentrations. The process by which BIS was derived is shown schematically in Figure 5.5.

In order to point out hypnotic or sleep level, BIS stages are used. In humans, 5 BIS stages and the stage awake are defined. Each BIS stage is characterized by a specific pattern of frequency content. BIS stages are defined as follows:

- Stage awake: Signal with continuity alpha activity.
- Stage 1: No presence of alpha activity, low beta and theta activity,

- Stage 2: Less than 20 % of delta activity and presence of K-complexes and spindles. K-complexes are low frequency waves near 1.0 Hz, with an amplitude of at least 75 mV. Spindles are well defined waves in the range 11-15 Hz with a time duration of more than 0.5 seconds. There is no criterion about the amplitude of a spindle.
- Stage 3: More than 20 % and less than 50 % of delta activity,
- Stage 4: More than 50 % of delta activity.
- Stage REM: Low amplitude waves with little theta activity and often sawtooth waves. REM and awake signals might have a similar shape, but REM have little alpha activity.

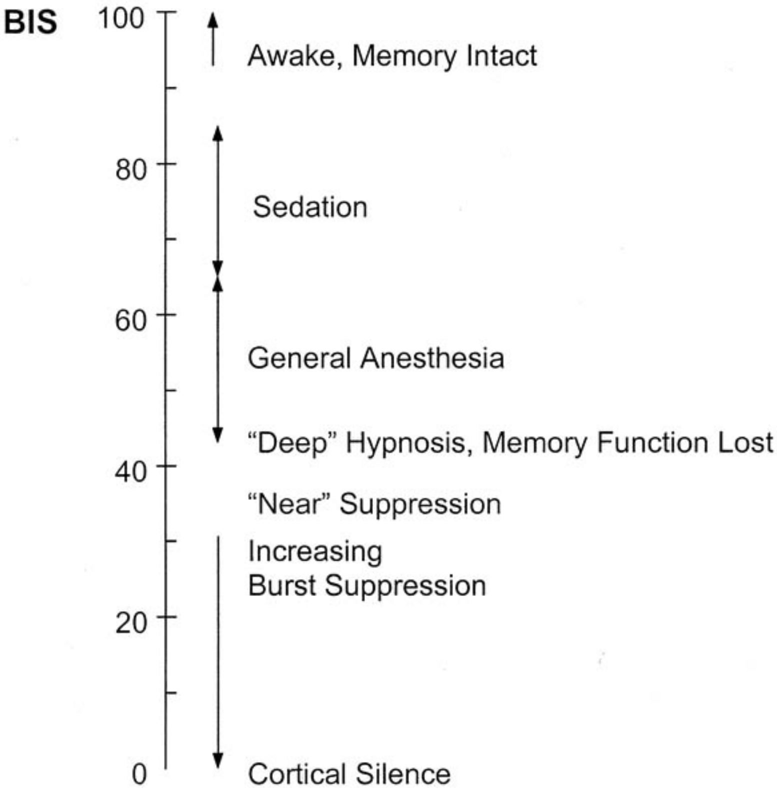


Figure 5.5 The Bispectral Index Scale (BIS versions 3.0 and higher)

The classification of EEG sleep is usually made by a visual scorer, which takes 30-s epochs and give a classification according to the rules of Rechtschaffen and Kale (Rechtschaffen & Kales, 1968). Not every epoch has 100% properties of an specific stage. The decision is made according to which stage properties are present the most and that is sometimes difficult to be carried out.

By using the abovementioned data collection techniques, BIS sleep data are obtained and in the next chapter *fuzzy clustering approach* has been used for detecting the sleep stage levels.

CHAPTER SIX
BIS-CLUSTERING AND APPLICATIONS

Determining BIS Stages by FCM-based algorithm

In this section, an algorithm is constructed to determine BIS stage intervals with respect to the BIS values by using FCM algorithm. At first, all the measurement series are merged in a pooled data set. We have BIS value recorded by EEG and stage level determined by experts, and they are denoted by x_i and s_i, for each measurement (point) in the pooled data set, respectively. Note that BIS stages get integer labels from 1 to 5 and BIS values get continuous values from interval $[0,100]$. Mean, α_k, and standard deviation, σ_k, of the BIS values are computed for each set of points that have the k-th stage ($k = 1,...,5$). Then, by using the axis orthogonal projection method, the fuzzy membership functions of classes for BIS values convenient to each stage are determined. The centers of the classes $k = 1,...,5$, α_k, and their membership functions $\mu_k(\cdot)$ are taken from the FCM results. Then BIS stage level of each point i is stored as

$$s_i = arg \max_{k=1,...,5} \mu_k(x_i), \quad i = 1,...,n \tag{6.1}$$

To summarize, the algorithm to determine the BIS stages is given below.

Algorithm 1.

Step 1. Merge all the data sets to construct a data pool.

Step 2. Run the FCM algorithm with cluster number is equal to 5 with respect to BIS values x_i, $i = 1,...,n$ to form the clusters corresponding to stage levels k, $k = 1,...,5$.

Step 3. Determine a stage label s_i with respect to the maximum membership degree of the BIS value x_i by using Formula (6.1).

End.

Determining BIS Stages by FN-DBSCAN-based algorithm

In this section, an algorithm is constructed determine BIS stage intervals with respect to the BIS values by using FN-DBSCAN method. At first, all the measurement series are merged in a pooled data set. Mean, α_k, and standard deviation, σ_k, of the BIS values are computed for each set of points that have the k-th stage ($k = 1,...,5$). Then, by using the axis orthogonal projection method, the fuzzy membership functions of classes for BIS values convenient to each stage are determined. These classes are assumed to be normally distributed and their parameters are calculated as shown below (Babuska, 1998; Kung & Su, 2007):

$$\alpha_k = \frac{\sum_{i \in I_k} x_i}{n_k} \quad \text{and} \quad \sigma_k = \sqrt{\frac{\sum_{i \in I_k} (x_i - \alpha_k)^2}{n_k}} \tag{6.2}$$

where I_k is the index set of x_i values convenient to the k-th stage, *i.e.* $I_k = \{i \mid s_i = k\}$, and n_k is the number of measurements with BIS stages equal to k. According to the method used in (Babuska, 1998; Kung & Su, 2007), the bell-shaped fuzzy membership function of BIS values convenient to k-th stage is constructed as follows:

$$\mu_k(x) = e^{-\frac{1}{2}\left(\frac{x-\alpha_k}{\sigma_k}\right)^2}, \quad k = 1,...,5. \tag{6.3}$$

Formula 6.2 is then used to determine the stage labels of clusters formed by the FN-DBSCAN algorithm. Suppose that the number of clusters found by using FN-DBSCAN algorithm is M. The average of BIS values of each cluster is calculated and a stage label is determined with respect to this value. Hence,

$$\bar{x}_m = \frac{\sum_{i \in C_m} x_i}{n_m}, \quad m = 1,...,M \tag{6.4}$$

where C_m is the index set of elements in the m-th cluster, and n_m is the number of elements in this cluster. Then for each m-th cluster

$$k_m^* = arg \max_{k=1,...,5} \mu_k(\bar{x}_m), \quad m = 1,...,M \tag{6.5}$$

is calculated in order to determine the appropriate stage label. The stage label of each measurement in cluster m is stored as k_m^*, *i.e.*

$$\bar{s}_i = k_m^*, \quad i \in C_m, \quad m = 1,...,M. \tag{6.6}$$

To summarize, the algorithm to determine the BIS stages is given below.

Algorithm 2.

Step 1. Merge all the data sets to construct a data pool.

Step 2. Calculate mean, α_k, and standard deviation, σ_k, by using Formula (6.2) for each set of BIS values, having the stage k, $k = 1,...,5$.

Step 3. Determine the fuzzy membership functions of classes convenient to each stage level $k = 1,...,5$ by using Formula (6.3).

Step 4. Run the FN-DBSCAN algorithm to part measurement series into connected intervals according to BIS values. Each connected interval is a separate cluster.

Step 5. Calculate the average of BIS values of each cluster by using Formula (6.4).

Step 6. Determine cluster membership degrees of the average value for each cluster by using Formula (6.3).

Step 7. Determine a stage label for each cluster with respect to the maximum membership degree of the average value by using Formula (6.6).

Step 8. Determine a stage label for each point within the cluster with respect to the stage label of the cluster.

End.

6.1 Experimental Results

The main purpose of this study is to show that studies on neighborhood-based cluster analysis gives effective results in predicting stage levels of BIS measurement series (Ulutagay & Nasibov, 2008a,b, 2009; Nasibov et al., 2008, 2009). 21 data sets each of which are registered in every five seconds during sleep for a 25-minute periods are used to demonstrate functionality of the FN-DBSCAN clustering algorithm. Thus, each data set consists 306 BIS measurements containing BIS stages given by experts. For instance, a part of the data set DB1 is given in Table 1. As first two columns, in order to use in learning process, experts determined the BIS stage values corresponding to each measurement moment (Table 6.1, Column 3).

Table 6.1 BIS values and stages for data set DB1.

No	BIS-value	BIS stage (by expert)	Normalized BIS-value	$\bar{x}_k$	BIS stage (by method)
1	39.2	2	0.392	0.391	2
2	39.2	2	0.392	0.391	2
3	39.2	2	0.392	0.391	2
4	40.8	2	0.408	0.391	2
5	40.9	2	0.409	0.391	2
6	38.8	2	0.388	0.391	2
7	38.4	2	0.384	0.391	2
8	39.0	2	0.390	0.391	2
9	38.9	2	0.389	0.391	2
10	37.9	2	0.379	0.391	2
11	33.2	2	0.354	0.391	2
12	33.2	2	0.332	0.391	2
13	36.1	2	0.361	0.391	2
14	37.9	2	0.379	0.391	2
15	39.6	2	0.396	0.391	2
16	38.7	2	0.387	0.391	2
17	39.9	2	0.399	0.391	2
18	42.5	2	0.425	0.391	2
19	42.5	2	0.425	0.391	2
20	43.0	2	0.430	0.391	2
...	...	...	...	...	...

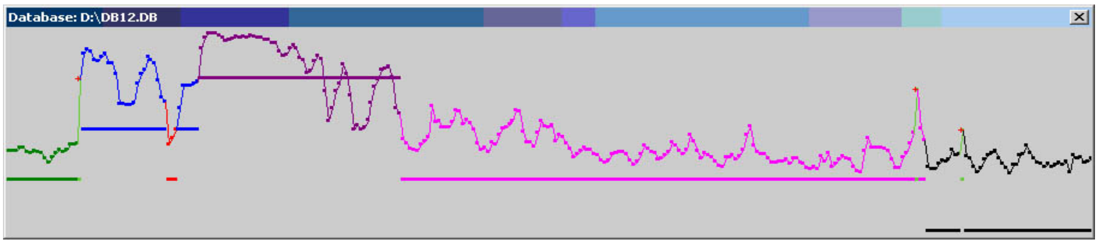
The participants arrived at the laboratory approximately 2h prior to their regular sleep time. They spent the sleep period lying on a bed which was located in a dimly illuminated, acoustically, and electromagnetically shielded chamber. Data for the study comprises first night recordings. Because the current study is related to non-REM sleep, corresponding to the first sleep cycle, approximately 90 min of data was processed both for BIS and EEG data pools.

EEG activity was recorded using an electro-cap for whole scalp with 30 EEG electrodes, which were referenced to linked earlobe electrodes, based on the international 10-10 system.

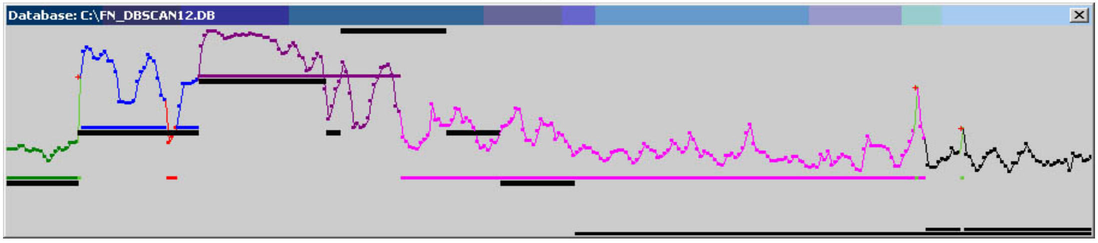
The research was performed in the Sleep Dynamics Laboratory of the Biophysics Department of the Faculty of Medicine, Dokuz Eylül University. BIS recording was performed using a bispectral index monitor (Aspect-A2000) with a sensor (BIS Quatro). BIS sensors were applied to the left forehead as specified by the manufacturer, above and parallel to eyebrow and next to left eye (sensors were checked for signal quality [SQI], assuming impedance below 5 kOhms).

In every five seconds, BIS was recorded via the RS232 cable using a HyperTerminal protocol. The subjects were monitored via a video system and their BIS and EEG recordings were synchronized. Sleep scoring was performed by visual inspection (with second verification by the Sleep Disorders Center in Department of Neurology at DEU, Faculty of Medicine) according to the criteria of Rechtschaffen & Kales (1968) in 30-s time windows (awake 0, stage 1, stage 2, stage 3, stage 4). For every single Rechtschaffen and Kale score there were six BIS values obtained ($6 \times 5s$).

The experiments showed that FN-DBSCAN-based algorithm is more successful than the FCM-based algorithm. The FN-DBSCAN-based algorithm determines only jumping situations, and does not respond to permanent changes (Figure 6.1). Thus, we can say that FN-DBSCAN-based algorithm is more advantageous than the FCM-based algorithm in detecting the sudden reactions and durations of the stable intervals for series in BIS data.



(a)



(b)

Figure 6.1 Clustering results of the FN-DBSCAN algorithm for data set DB1 (a) clustering plot alone, (b) compared with expert opinion (bold black lines under the colored lines).

Example. First of all, by merging 21 data sets each of which has 306 measurements, a data pool of consisting $21 \times 306 = 6426$ data is formed. Mean, α_k, and standard deviation, σ_k, of the BIS values for each k-th ($k = 1, \ldots, 5$) is stage level calculated by using the Formula (6.2) (Table 6.2 and Figure 6.2).

Table 6.2 Mean and standard deviation of each BIS stage.

Stage level (k)	Mean (α_k)	Standard deviation (σ_k)
1	0.311	0.055
2	0.493	0.122
3	0.689	0.130
4	0.836	0.087
5	0.886	0.084

Then, fuzzy membership functions of mean BIS values convenient to each stage level for each stable class is determined by using the Formula 6.3 of the axis orthogonal projection method (Figure 6.3).

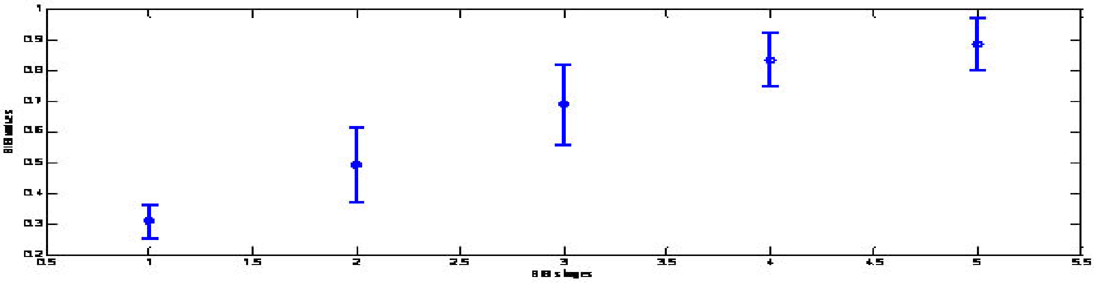


Figure 6.2 Means and standard deviations of the standardized values of the stages.

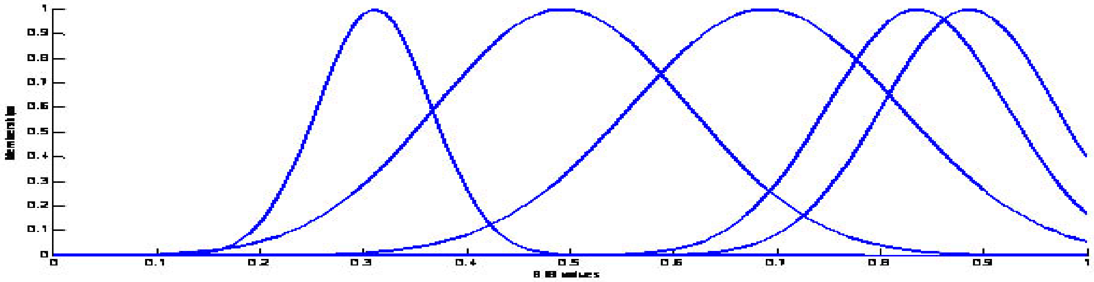


Figure 6.3 Membership functions of the standardized values of the stages.

In order to illustrate the proposed approach, the results of the FN-DBSCAN algorithm for some measurements of DB1 are given in Table 6.1. Naturally, different clustering results were obtained for various values of ε_1 and ε_2 parameters of the algorithm. The concept of distance is used in order to reach the nearest results of the experts' opinions. In this sense, the following formula is used to compute the distance between the predicted serie, ser, and series given by expert, each consisting of 306 data:

$$d(ser, expert) = \sqrt{\sum_{i=1}^{306}(s_i - \overline{s}_i)^2} \tag{6.7}$$

where s_i and $\overline{s}_i$ are the BIS stage levels of the i-th measurement given by the experts and computed by the Formula (6.6), respectively.

Each point x_i, $i = 1, ..., n$, is considered as a two-dimensional point (i, x_i) on the time series plot. The following weighted Euclidean distance is used to calculate the distance between the i-th and the j-th points of a measurement serie:

$$d(i,j) = \sqrt{(i-j)^2 + \eta \cdot (x_i - x_j)^2} \tag{6.8}$$

where $\eta \in [0,1]$ is a scale parameter.

Table 6.3 Optimal results for all data sets.

DB	OptScale (η)	MinBIS distance	OptEps1 (ε_1)	OptEps2 (ε_2)	Clusters (k)
DB1	0.25	4.008	0.88	0.1	6
DB2	1.00	2.031	0.51	0.2	6
DB3	0.20	2.716	0.99	0.0	87
DB4	1.00	2.264	0.62	0.0	7
DB5	0.10	1.541	0.97	0.0	5
DB6	0.90	0.866	0.75	0.2	9
DB7	0.05	2.795	0.99	0.0	5
DB8	0.10	2.716	0.97	0.0	5
DB9	0.20	2.475	0.93	0.0	4
DB10	0.15	1.732	0.98	0.0	16
DB11	0.05	0.000	0.01	0.0	1
DB12	0.35	2.264	0.93	0.0	21
DB13	0.30	2.165	0.75	0.0	4
DB14	0.20	1.920	0.97	0.0	4
DB15	0.35	4.257	0.90	0.0	10
DB16	0.05	3.725	0.99	0.0	4
DB17	0.25	1.500	0.95	0.0	6
DB18	0.80	0.829	0.75	0.0	4
DB19	0.50	2.345	0.84	0.1	4
DB20	0.05	3.132	0.01	0.0	1
DB21	0.05	2.194	0.99	0.0	5

The results for each data set, among the η parameters, optimal η parameter, *i.e.* the nearest results to expert's opinion, which are obtained by running FN-DBSCAN based algorithm, are given in Table 6.3. Moreover, for each data set various η parameters are given in Tables 6.5-6.11. In these tables, for each η parameter, optimal values of ε_1 and ε_2 parameters and the number of clusters obtained for these values are given.

By running FCM-based algorithm for various values of the fuzziness exponent (q), ranging from 1.5 through 15.0 sum and increasing 0.1 in each step, the nearest results to expert's opinion are also obtained. In Table 6.4, for each data set, sum square error results for optimal values of parameters obtained by FN-DBSCAN and FCM-based algorithms are given. Under the "FCM" heading, it is seen that FCM results for q is changed from 1.5 to 2.9, the results for $q = 3$, and q is changed from 3.1 to 15. Since

FCM gives the same sum square of error results except $q = 3$, running ranges of q are given in such a way.

Table 6.4 Sum square error results for expert vs FN-DBSCAN and expert vs FCM.

DB	**FN-DBSCAN**	**FCM**		
		q=1.5-2.9	**q=3.0**	**q=3.1-15.0**
DB1	82	169	159	169
DB2	66	272	213	272
DB3	160	306	273	306
DB4	84	167	149	169
DB5	37	253	233	253
DB6	12	110	175	110
DB7	123	220	202	220
DB8	120	166	151	166
DB9	52	141	122	141
DB10	51	122	125	122
DB11	0	4	6	4
DB12	257	322	332	322
DB13	75	352	337	352
DB14	61	145	137	145
DB15	291	367	330	367
DB16	219	240	208	240
DB17	72	67	64	67
DB18	10	88	70	89
DB19	87	385	354	385
DB20	148	415	368	415
DB21	81	180	177	180

The results obtained for the optimal values of each algorithm are evaluated by using the above formula and the results are shown in Table 6.12.

Table 6.5 Scale independent optimal results for the datasets DB1-DB3.

DB1					DB2					DB3				
Scale (η)	BIS dist	Eps1 (ε_1)	Eps2 (ε_2)	k	Scale (η)	BIS dist	Eps1 (ε_1)	Eps2 (ε_2)	k	Scale (η)	BIS dist	Eps1 (ε_1)	Eps2 (ε_2)	k
0.05	4.670	0.99	0	13	0.05	2.291	0.99	0	6	0.05	4.198	0.01	0	1
0.10	4.623	0.99	0	39	0.10	2.291	0.97	0	5	0.10	2.187	0.99	0	27
0.15	4.437	0.90	0.1	4	0.15	2.332	0.93	0	4	0.15	2.795	0.99	0	62
0.20	4.016	0.90	0.1	6	0.20	2.332	0.88	0	4	0.20	2.716	0.99	0	87
0.25	4.008	0.88	0.1	6	0.25	2.332	0.82	0	4	0.25	2.817	0.99	0	114
0.30	4.352	0.83	0.1	5	0.30	2.332	0.75	0	4	0.30	2.894	0.98	0	71
0.35	4.235	0.96	0	51	0.35	2.332	0.68	0	4	0.35	2.739	0.91	0.1	3
0.40	4.235	0.95	0	49	0.40	2.332	0.61	0	4	0.40	2.850	0.98	0	107
0.45	4.257	0.94	0	48	0.45	2.046	0.78	0	7	0.45	2.937	0.92	0.1	14
0.50	4.257	0.93	0	48	0.50	2.046	0.73	0	5	0.50	2.947	0.97	0	100
0.55	4.265	0.93	0	54	0.55	2.046	0.68	0	5	0.55	2.905	0.97	0	105
0.60	4.265	0.93	0	58	0.60	2.046	0.64	0	5	0.60	2.773	0.87	0.2	9
0.65	4.521	0.93	0	63	0.65	2.046	0.59	0	5	0.65	2.905	0.96	0	99
0.70	4.521	0.93	0	68	0.70	2.046	0.55	0	5	0.70	3.000	0.78	0	27
0.75	4.220	0.91	0	56	0.75	2.046	0.53	0	5	0.75	2.905	0.93	0	64
0.80	4.228	0.90	0	54	0.80	2.046	0.53	0	5	0.80	2.761	0.85	0	19
0.85	4.228	0.89	0	52	0.85	2.046	0.53	0	5	0.85	2.761	0.83	0	18
0.90	4.235	0.88	0	52	0.90	2.046	0.53	0	5	0.90	2.761	0.81	0.1	12
0.95	4.235	0.87	0	51	0.95	2.062	0.55	0	9	0.95	2.761	0.81	0.1	10
1.00	4.257	0.87	0	53	1.00	2.031	0.51	0.2	6	1.00	2.883	0.93	0	95

Table 6.6 Scale-dependent optimal results for the datasets DB4-DB6.

DB4					DB5					DB6				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	2.512	0.94	0	2	0.05	2.646	0.98	0	3	0.05	2.222	0.01	0	1
0.10	2.512	0.79	0	2	0.10	1.541	0.97	0	5	0.10	2.222	0.01	0	1
0.15	2.512	0.60	0	2	0.15	2.121	0.97	0	15	0.15	2.222	0.01	0	1
0.20	2.512	0.40	0	2	0.20	2.194	0.94	0.2	8	0.20	2.222	0.01	0	1
0.25	2.512	0.30	0	2	0.25	2.208	0.93	0	13	0.25	2.208	0.98	0.2	41
0.30	2.462	0.89	0	4	0.30	2.372	0.93	0.2	12	0.30	2.165	0.98	0.2	43
0.35	2.462	0.85	0	4	0.35	2.437	0.90	0.2	8	0.35	1.061	0.84	0.2	2
0.40	2.462	0.82	0	4	0.40	2.475	0.89	0.2	9	0.40	1.061	0.80	0.2	2
0.45	2.462	0.80	0	4	0.45	2.194	0.87	0.1	7	0.45	1.061	0.75	0.1	2
0.50	2.449	0.79	0	5	0.50	1.984	0.84	0.2	4	0.50	1.061	0.72	0.1	2
0.55	2.449	0.75	0	5	0.55	1.984	0.83	0.2	4	0.55	1.061	0.70	0.1	2
0.60	2.449	0.71	0	5	0.60	2.000	0.79	0.2	4	0.60	1.061	0.68	0.1	2
0.65	2.449	0.67	0	5	0.65	2.016	0.81	0.1	5	0.65	1.061	0.65	0.1	2
0.70	2.449	0.64	0	5	0.70	2.016	0.76	0.2	4	0.70	1.061	0.65	0.1	2
0.75	2.449	0.61	0	5	0.75	2.016	0.75	0.2	4	0.75	1.031	0.77	0.2	5
0.80	2.449	0.57	0	5	0.80	2.016	0.75	0.1	6	0.80	1.031	0.74	0.2	8
0.85	2.449	0.54	0	5	0.85	1.820	0.73	0	7	0.85	1.000	0.76	0.2	8
0.90	2.449	0.52	0	4	0.90	1.820	0.70	0	7	0.90	0.866	0.75	0.2	9
0.95	2.449	0.49	0	4	0.95	1.820	0.70	0	7	0.95	0.866	0.73	0.2	6
1.00	2.264	0.62	0	7	1.00	1.750	0.68	0	8	1.00	0.866	0.71	0.2	6

Table 6.7 Scale-dependent optimal results for the datasets DB7-DB9.

DB7					DB8					DB9				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	2.795	0.99	0	5	0.05	2.750	0.99	0	6	0.05	2.839	0.99	0	5
0.10	2.795	0.97	0	5	0.10	2.716	0.97	0	5	0.10	2.537	0.98	0	10
0.15	2.937	0.90	0	2	0.15	2.727	0.98	0	18	0.15	2.550	0.97	0	13
0.20	2.937	0.84	0	2	0.20	2.795	0.96	0	11	0.20	2.475	0.93	0	4
0.25	2.937	0.76	0	2	0.25	2.761	0.95	0	9	0.25	2.475	0.90	0	4
0.30	2.937	0.72	0	2	0.30	2.761	0.92	0	8	0.30	2.475	0.86	0	4
0.35	2.937	0.70	0	3	0.35	2.727	0.89	0.2	4	0.35	2.475	0.82	0	3
0.40	2.937	0.68	0	3	0.40	2.839	0.88	0.2	6	0.40	2.475	0.79	0	3
0.45	2.937	0.66	0	3	0.45	2.958	0.88	0	7	0.45	2.475	0.78	0	3
0.50	2.937	0.63	0	3	0.50	2.969	0.01	0	1	0.50	2.475	0.77	0	3
0.55	2.937	0.61	0	3	0.55	2.969	0.01	0	1	0.55	2.475	0.76	0	3
0.60	2.937	0.58	0	3	0.60	2.969	0.01	0	1	0.60	2.525	0.75	0	4
0.65	2.937	0.55	0	3	0.65	2.969	0.01	0	1	0.65	2.586	0.73	0	3
0.70	2.937	0.52	0	3	0.70	2.969	0.01	0	1	0.70	2.598	0.73	0	5
0.75	2.937	0.49	0	3	0.75	2.969	0.01	0	1	0.75	2.598	0.72	0	5
0.80	2.937	0.46	0	3	0.80	2.969	0.01	0	1	0.80	2.598	0.72	0	5
0.85	2.937	0.43	0	3	0.85	2.969	0.01	0	1	0.85	2.598	0.72	0	7
0.90	2.969	0.01	0	1	0.90	2.969	0.01	0	1	0.90	2.598	0.71	0	7
0.95	2.969	0.01	0	1	0.95	2.969	0.01	0	1	0.95	2.598	0.71	0	9
1.00	2.969	0.01	0	1	1.00	2.969	0.01	0	1	1.00	2.598	0.71	0	10

Table 6.8 Scale-dependent optimal results for the datasets DB10-DB12.

DB10					**DB11**					**DB12**				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	3.800	0.99	0	2	0.05	0.000	0.01	0	1	0.05	3.953	0.99	0	4
0.10	2.107	0.99	0	24	0.10	0.000	0.01	0	1	0.10	2.437	0.99	0	33
0.15	1.732	0.98	0	16	0.15	0.000	0.01	0	1	0.15	2.462	0.98	0	26
0.20	1.969	0.99	0	70	0.20	0.000	0.01	0	1	0.20	2.424	0.97	0	26
0.25	1.920	0.96	0	12	0.25	0.000	0.01	0	1	0.25	2.372	0.96	0	25
0.30	1.936	0.98	0	51	0.30	0.000	0.01	0	1	0.30	2.358	0.95	0	23
0.35	1.904	0.98	0	51	0.35	0.000	0.01	0	1	0.35	2.264	0.93	0	21
0.40	1.953	0.97	0	51	0.40	0.000	0.01	0	1	0.40	2.278	0.91	0	18
0.45	1.936	0.97	0	51	0.45	0.000	0.01	0	1	0.45	2.278	0.89	0	12
0.50	1.920	0.97	0	68	0.50	0.000	0.01	0	1	0.50	2.372	0.89	0.1	15
0.55	1.953	0.95	0	49	0.55	0.000	0.01	0	1	0.55	2.372	0.86	0.1	12
0.60	1.936	0.94	0	42	0.60	0.000	0.01	0	1	0.60	2.372	0.85	0.1	13
0.65	1.936	0.93	0	38	0.65	0.000	0.01	0	1	0.65	2.562	0.84	0.1	14
0.70	1.953	0.93	0	41	0.70	0.000	0.01	0	1	0.70	2.562	0.83	0.1	13
0.75	1.953	0.92	0	39	0.75	0.000	0.01	0	1	0.75	2.562	0.81	0.1	13
0.80	1.936	0.91	0	35	0.80	0.000	0.01	0	1	0.80	2.550	0.80	0.2	9
0.85	1.953	0.91	0	38	0.85	0.000	0.01	0	1	0.85	2.550	0.78	0.2	9
0.90	1.936	0.90	0	34	0.90	0.000	0.01	0	1	0.90	2.550	0.77	0.2	9
0.95	1.936	0.89	0	33	0.95	0.000	0.01	0	1	0.95	2.550	0.77	0.2	9
1.00	1.904	0.87	0	27	1.00	0.000	0.01	0	1	1.00	2.562	0.73	0.1	12

Table 6.9 Scale-dependent optimal results for the datasets DB13-DB15.

DB13					DB14					DB15				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	2.250	0.97	0.2	2	0.05	4.617	0.01	0	1	0.05	4.330	0.92	0	2
0.10	2.264	0.89	0	2	0.10	2.969	0.99	0	7	0.10	4.308	0.96	0	6
0.15	2.264	0.76	0	2	0.15	2.795	0.99	0	22	0.15	4.330	0.46	0	2
0.20	2.264	0.62	0	2	0.20	1.920	0.97	0	4	0.20	4.308	0.91	0	5
0.25	2.264	0.47	0	2	0.25	1.920	0.96	0	5	0.25	4.308	0.90	0	5
0.30	2.165	0.75	0	4	0.30	1.920	0.95	0	5	0.30	4.265	0.90	0	7
0.35	2.165	0.69	0	4	0.35	1.920	0.95	0	6	0.35	4.257	0.90	0	10
0.40	2.264	0.15	0	2	0.40	2.092	0.95	0	8	0.40	4.257	0.90	0	12
0.45	2.264	0.09	0	2	0.45	2.121	0.95	0	11	0.45	4.287	0.89	0	12
0.50	2.264	0.05	0	2	0.50	2.151	0.95	0	15	0.50	4.287	0.87	0	11
0.55	2.264	0.03	0	2	0.55	1.984	0.94	0	12	0.55	4.287	0.85	0	11
0.60	2.264	0.02	0	2	0.60	1.984	0.94	0	12	0.60	4.287	0.82	0	9
0.65	2.264	0.01	0	2	0.65	2.031	0.94	0	14	0.65	4.294	0.79	0	8
0.70	2.264	0.01	0	2	0.70	2.165	0.91	0	7	0.70	4.294	0.77	0	8
0.75	2.264	0.01	0	2	0.75	2.165	0.90	0	7	0.75	4.308	0.74	0	7
0.80	2.264	0.01	0	2	0.80	2.165	0.89	0	7	0.80	4.308	0.71	0	7
0.85	2.194	0.54	0.2	5	0.85	2.165	0.88	0	6	0.85	4.308	0.68	0	7
0.90	2.264	0.01	0	2	0.90	2.165	0.87	0	6	0.90	4.308	0.65	0	7
0.95	2.264	0.01	0	2	0.95	2.165	0.87	0	6	0.95	4.308	0.63	0	6
1.00	2.264	0.01	0	2	1.00	2.165	0.86	0	6	1.00	4.308	0.72	0	11

Table 6.10 Scale-dependent optimal results for the datasets DB16-DB18.

DB16					DB17					DB18				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	3.725	0.99	0	4	0.05	2.817	0.99	0	3	0.05	2.345	0.98	0	2
0.10	4.243	0.98	0	6	0.10	1.785	0.98	0	5	0.10	1.250	0.98	0	9
0.15	4.243	0.96	0	5	0.15	1.561	0.97	0	4	0.15	1.173	0.97	0	12
0.20	3.800	0.93	0	3	0.20	1.521	0.96	0	7	0.20	1.173	0.94	0	5
0.25	3.800	0.89	0	3	0.25	1.500	0.95	0	6	0.25	1.173	0.93	0	7
0.30	3.800	0.84	0	3	0.30	1.500	0.94	0	7	0.30	1.146	0.92	0	10
0.35	3.800	0.79	0	3	0.35	1.500	0.93	0	8	0.35	1.146	0.91	0	11
0.40	3.800	0.77	0	3	0.40	1.541	0.93	0	12	0.40	1.146	0.90	0	12
0.45	4.123	0.88	0	12	0.45	1.732	0.78	0	2	0.45	1.146	0.89	0	13
0.50	4.123	0.85	0	12	0.50	1.732	0.74	0	2	0.50	1.146	0.88	0	13
0.55	4.123	0.82	0	12	0.55	1.732	0.72	0	2	0.55	1.146	0.88	0	16
0.60	4.123	0.80	0	11	0.60	1.696	0.86	0	8	0.60	0.968	0.83	0	10
0.65	4.123	0.78	0	11	0.65	1.696	0.83	0	6	0.65	0.968	0.82	0	10
0.70	4.146	0.97	0	104	0.70	1.696	0.81	0	6	0.70	0.968	0.80	0	9
0.75	4.153	0.97	0	115	0.75	1.696	0.79	0	6	0.75	0.968	0.77	0	6
0.80	4.146	0.96	0	97	0.80	1.696	0.76	0	6	0.80	0.829	0.75	0	4
0.85	4.146	0.96	0	101	0.85	1.696	0.75	0	6	0.85	0.829	0.74	0.1	4
0.90	4.153	0.96	0	102	0.90	1.696	0.97	0	106	0.90	1.118	0.73	0.1	5
0.95	4.153	0.96	0	105	0.95	1.677	0.97	0	109	0.95	1.118	0.73	0.1	5
1.00	4.153	0.95	0	100	1.00	1.696	0.70	0	6	1.00	1.118	0.73	0.1	5

Table 6.11 Scale-dependent optimal results for the datasets DB19-DB21.

DB19					DB20					DB21				
Scale (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**	**Scale** (η)	**BIS dist**	**Eps1** (ε_1)	**Eps2** (ε_2)	**k**
0.05	2.372	0.99	0	5	0.05	3.132	0.01	0.1	1	0.05	2.194	0.99	0	5
0.10	2.411	0.01	0	1	0.10	3.132	0.01	0.1	1	0.10	2.693	0.92	0	3
0.15	2.411	0.01	0	1	0.15	3.132	0.01	0.1	1	0.15	2.693	0.92	0	3
0.20	2.385	0.92	0	2	0.20	3.132	0.01	0.1	1	0.20	2.784	0.97	0	44
0.25	2.385	0.88	0	2	0.25	3.132	0.01	0.1	1	0.25	2.784	0.96	0	36
0.30	2.372	0.87	0.2	2	0.30	3.132	0.01	0.1	1	0.30	2.500	0.92	0	13
0.35	2.358	0.85	0	4	0.35	3.132	0.01	0.1	1	0.35	2.669	0.91	0	17
0.40	2.358	0.84	0	4	0.40	3.132	0.01	0.1	1	0.40	2.817	0.89	0	16
0.45	2.358	0.84	0	4	0.45	3.132	0.01	0.1	1	0.45	2.817	0.88	0	17
0.50	2.345	0.84	0.1	4	0.50	3.132	0.01	0.1	1	0.50	2.727	0.84	0.2	10
0.55	2.358	0.82	0.1	4	0.55	3.132	0.01	0.1	1	0.55	2.704	0.81	0.2	10
0.60	2.358	0.79	0.1	3	0.60	3.132	0.01	0.1	1	0.60	2.693	0.78	0.2	10
0.65	2.358	0.76	0.1	3	0.65	3.132	0.01	0.1	1	0.65	2.693	0.77	0.2	10
0.70	2.372	0.72	0.1	3	0.70	3.132	0.01	0.1	1	0.70	2.716	0.77	0.2	10
0.75	2.372	0.69	0.1	2	0.75	3.132	0.01	0.1	1	0.75	2.795	0.77	0.2	11
0.80	2.372	0.65	0.1	2	0.80	3.132	0.01	0.1	1	0.80	2.861	0.76	0.2	11
0.85	2.372	0.62	0.1	2	0.85	3.132	0.01	0.1	1	0.85	2.761	0.60	0.2	8
0.90	2.372	0.58	0.1	2	0.90	3.132	0.01	0.1	1	0.90	2.761	0.58	0.2	8
0.95	2.372	0.55	0.1	2	0.95	3.132	0.01	0.1	1	0.95	2.761	0.57	0.2	8
1.00	2.372	0.53	0.1	2	1.00	3.132	0.01	0.1	1	1.00	2.761	0.56	0.2	8

In order to evaluate the classification accuracy (CA) for each series (data set), the following formula is used:

$$CA = \frac{\text{correctly detected number of points in the serie}}{\text{total number of points in the serie}}. \tag{6.9}$$

Experimental results showed that FCM-based algorithm found the best results for $q = 3$. The best results of both FCM and FN-DBSCAN-based algorithms are compared statistically by Wilcoxon signed ranks test in SPSS statistical software (Figure 6.5). According to the test results, it is statistically significant with $p < 0.001$ that FN-DBSCAN gives lower SSE than FCM, *i.e.* finds closer results to the expert's opinion. Furthermore, the comparison for the best results, *i.e.* the minimum sum square error results of each algorithm are given graphically in Figure 6.4. Average classification accuracy of FCM and FN-DBSCAN-based algorithms are 58% and 71%, respectively.

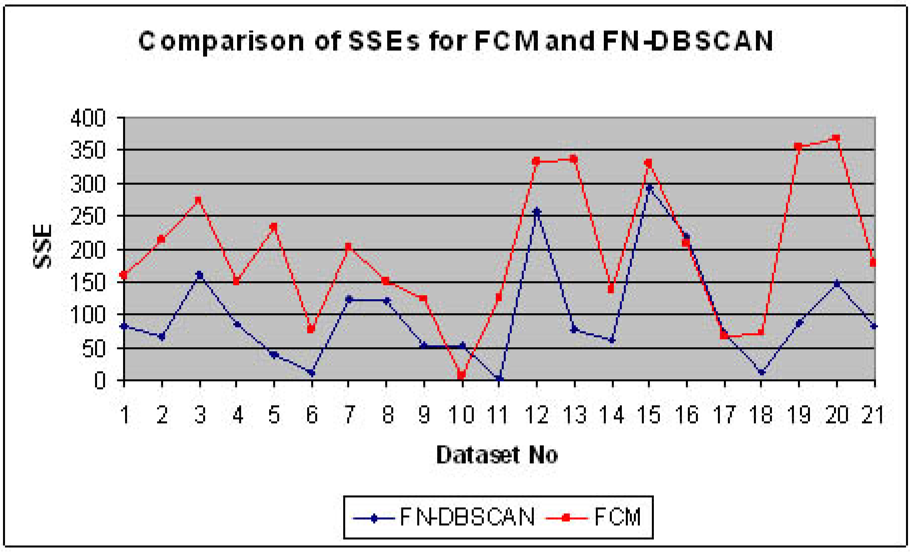
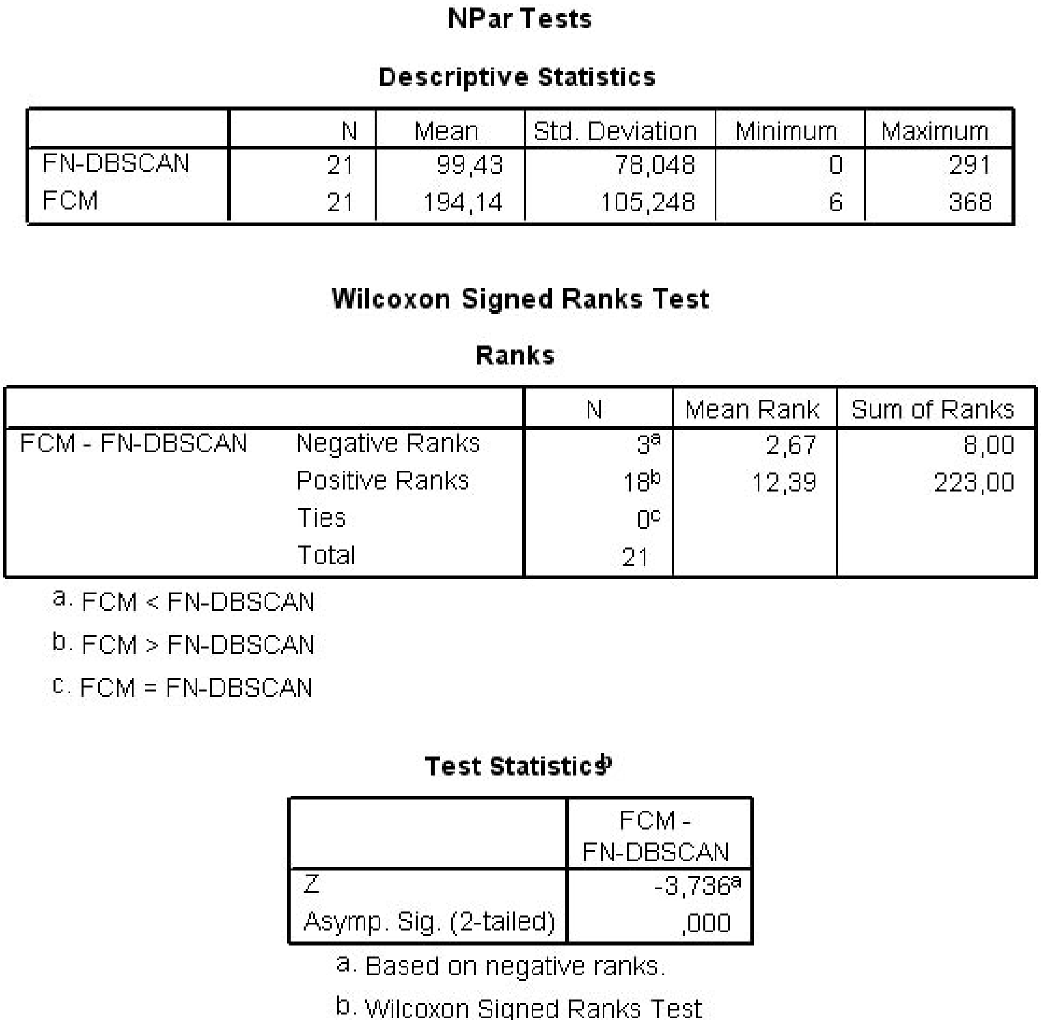


Figure 6.4 Graphical plots for SSEs of FCM and FN-DBSCAN algorithms.

Table 6.12 Classification accuracy of the algorithms.

DB	**FN-DBSCAN**		**FCM**	
	Number of correct points	Classification accuracy	Number of correct points	Classification accuracy
DB1	153	0.500	152	0.497
DB2	240	0.784	162	0.529
DB3	186	0.608	125	0.408
DB4	222	0.725	199	0.650
DB5	269	0.879	144	0.471
DB6	294	0.961	231	0.755
DB7	183	0.598	125	0.408
DB8	186	0.608	173	0.565
DB9	254	0.830	186	0.608
DB10	255	0.833	184	0.601
DB11	306	1.000	302	0.987
DB12	224	0.732	168	0.549
DB13	231	0.755	116	0.379
DB14	245	0.801	214	0.699
DB15	15	0.049	66	0.216
DB16	117	0.382	175	0.572
DB17	261	0.853	251	0.820
DB18	296	0.967	236	0.771
DB19	219	0.716	30	0.098
DB20	158	0.516	28	0.092
DB21	225	0.735	177	0.578
Mean	**216.143**	**0.706**	**164**	**0.536**
Std. dev.	**67.320**	**0.220**	**68.276**	**0.223**



NPar Tests

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
FN-DBSCAN	21	99,43	78,048	0	291
FCM	21	194,14	105,248	6	368

Wilcoxon Signed Ranks Test

Ranks

		N	Mean Rank	Sum of Ranks
FCM - FN-DBSCAN	Negative Ranks	3[a]	2,67	8,00
	Positive Ranks	18[b]	12,39	223,00
	Ties	0[c]		
	Total	21		

a. FCM < FN-DBSCAN

b. FCM > FN-DBSCAN

c. FCM = FN-DBSCAN

Test Statistics[b]

	FCM - FN-DBSCAN
Z	-3,736[a]
Asymp. Sig. (2-tailed)	,000

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

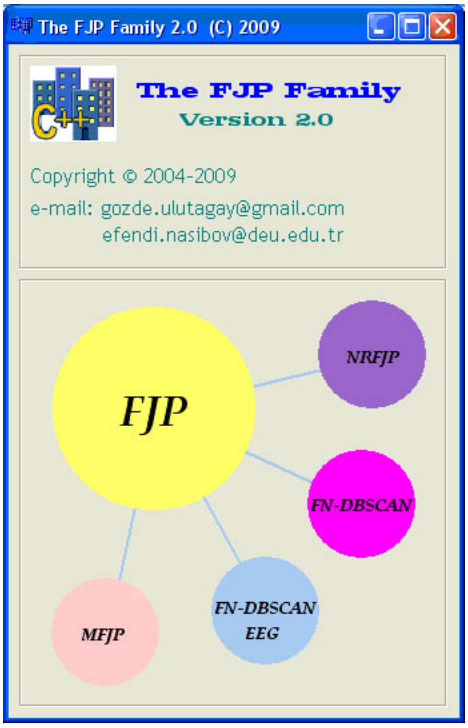
Figure 6.5 Results of Wilcoxon signed rank test to compare SSEs of FCM and FN-DBSCAN algorithms.

CHAPTER SEVEN

SOFTWARE FOR FUZZY NEIGHBORHOOD BASED CLUSTERING

In this chapter, presentation and running principles of the program system constructed for the methods and algorithms which are explained in Chapters 4 and 6 are given. Moreover, detailed information concerning the forms, functional buttons, and informative components constituted in “The FJP Family” Program is given. Code for the program systems are developed in Borland C++ Builder 6.0 SDK and applied on Pentium IV, 2.66 Mhz CPU, 2 Gb RAM computer.

When the program starts to run, the window given in Figure 7.1 appears on the screen. Since all the proposed algorithms are based on the FJP method, we named the opening program as “The FJP Family”. When the mentioned algorithms are clicked from Figure 7.1, corresponding form appears on the screen. Now, let us investigate the methods mentioned in the dissertation in detail.



The FJP Family 2.0 (C) 2009

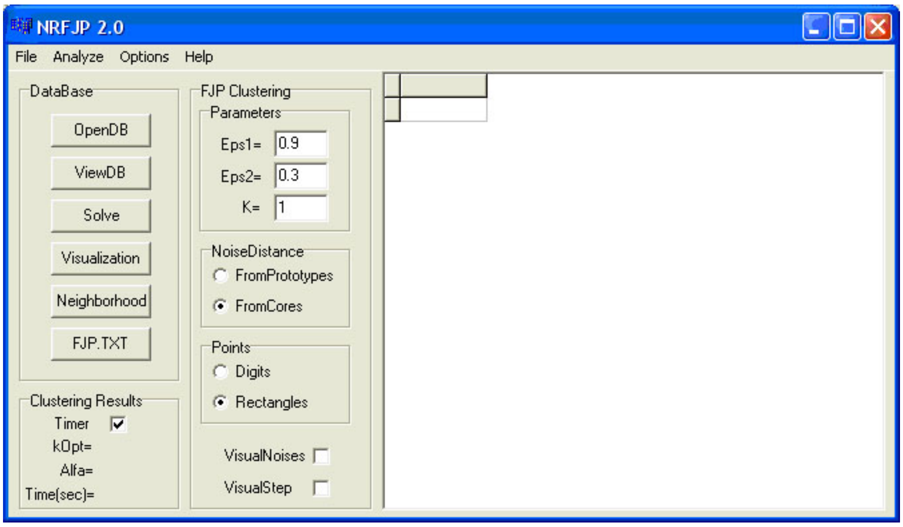
The FJP Family
Version 2.0

Copyright © 2004-2009
e-mail: gozde.ulutagay@gmail.com
efendi.nasibov@deu.edu.tr

Figure 7.1 Program selection window.

7.1 Software of NRFJP Algorithm

In this section, we explain the running principle of the software of NRFJP algorithm explained in Section 4.2.



Eps1= 0.9

Eps2= 0.3

K= 1

○ FromPrototypes

● FromCores

○ Digits

● Rectangles

Timer ☑

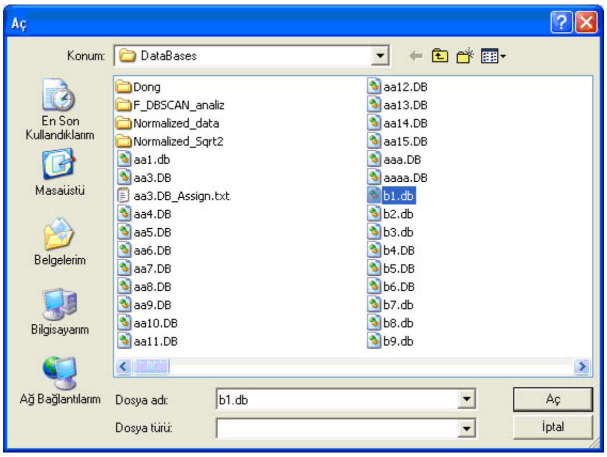
VisualNoises □

VisualStep □

Figure 7.2 Opening window of NRFJP program.

7.1.1 *Forms*

In Figure 7.1, when the NRFJP button is clicked, Figure 7.2 appears on the screen. First of all, it is required to select the table by pressing on the “OpenDB” button (Figure 7.3).



Konum: DataBases

Dosya adı: b1.db

Dosya türü:

Figure 7.3 Data set selection window.

It is possible to see the structure of a two-dimensional data set before starting the clustering process (Figure 7.4). “ViewDB” button is pressed for this purpose in Figure 7.3.

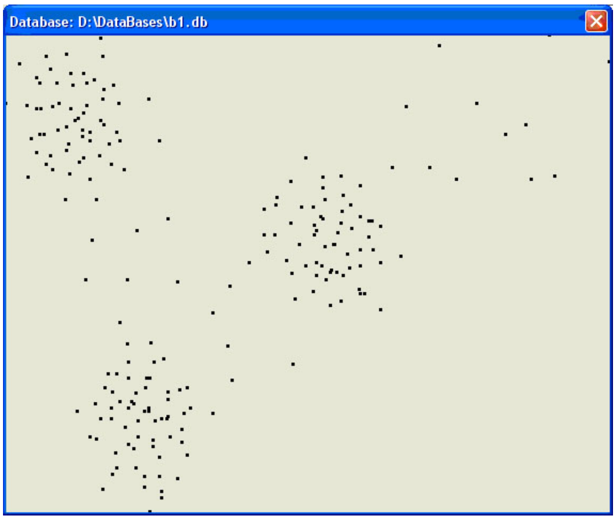
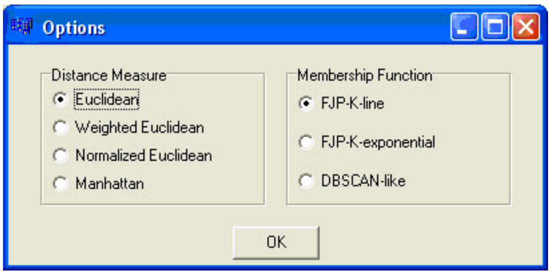


Figure 7.4 Visualization of data set.

The parameters ε_1 and ε_2 of the NRFJP algorithm can be set up from the “Parameters” part of the main program window of NRFJP program. Moreover, the parameter “k” that is used in various neighborhood membership functions can also be determined. In order to select the neighborhood membership function and distance metric which are to be used in clustering, “Options” is clicked from the menu, and the window shown in Figure 7.5 appears on the screen. For instance, we select “Euclidian distance” as distance metric, and “FJP-k-line” as linear neighborhood membership function. Let the value of “k” be 1 for this function. When the “Solve” button is clicked in the main form, clustering process starts. Clustering result for $\varepsilon_1 = 0.90$ and $\varepsilon_2 = 0.2$ is shown in Figure 7.6. On “Clustering Results” part, it is shown that optimal number of clusters for given parameters is “kOpt=3” and the α-level convenient to this cluster number is “$\alpha = 0.9393$”. Furthermore, since we check the “Timer” option, it is shown that the process is realized in 5 seconds in “Time (sec)” part. If we select “Visual Step” option before clicking on the “Solve” button, each clustering structure convenient to α-level, in which number of clusters is affected, is shown one-by-one in a separate form. Then, optimal clustering results are shown at the end of the clustering process.



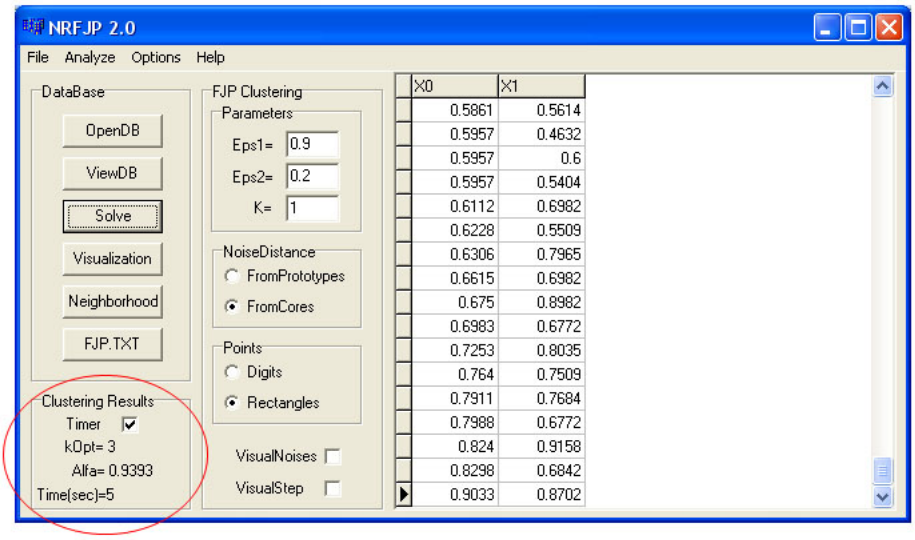
Distance Measure

● Euclidean
○ Weighted Euclidean
○ Normalized Euclidean
○ Manhattan

Membership Function

● FJP-K-line
○ FJP-K-exponential
○ DBSCAN-like

Figure 7.5 Options for distance measure and membership function.



FJP Clustering

Parameters

Eps1= 0.9
Eps2= 0.2
K= 1

NoiseDistance

○ FromPrototypes
● FromCores

Points

○ Digits
● Rectangles

VisualNoises □
VisualStep □

Clustering Results

Timer ☑
kOpt= 3
Alfa= 0.9393
Time(sec)=5

X0	X1
0.5861	0.5614
0.5957	0.4632
0.5957	0.6
0.5957	0.5404
0.6112	0.6982
0.6228	0.5509
0.6306	0.7965
0.6615	0.6982
0.675	0.8982
0.6983	0.6772
0.7253	0.8035
0.764	0.7509
0.7911	0.7684
0.7988	0.6772
0.824	0.9158
0.8298	0.6842
0.9033	0.8702

Figure 7.6 Clustering results of FN-DBSCAN algorithm with parameters $\varepsilon_1 = 0.9$ and $\varepsilon_2 = 0.1$.

Neighborhood degrees for each element is shown on a graphic if the “Neighborhood” button is clicked (Figure 7.7). The neighborhood degrees of the elements are given in decreasing order in this graphic and it is possible to decide for the suitable value of ε_2 parameter by looking at the widest jumping interval. The red line placed on the sticks can be moved upwards and downwards. When this line is moved, the value of ε_2 parameter changes as the name of the window. After the form is being closed, the last value of the parameter is written across the “Eps2” information component placed in the main form.

When “Visualization” button is clicked, the visual clustering result is shown (Figure 7.8). It is also possible to select the style of elements in a cluster. Elements can be shown as digits or as colored rectangles if “Digits” or “Rectangles” item is selected, respectively, from the “Points” part. It is available to decide whether the noise points are assigned to the nearest clusters according to the distance from core point or prototype by selecting “FromCores” or “From Prototypes”, respectively from the “Noise Distance” part. However, if “Visualization” button is clicked after “VisualNoises” option is checked, noise points are checked as “+” and they are not assigned to any cluster. For instance, in Figure 7.8a, noise points are assigned to the nearest clusters according to distance from prototypes while they are not assigned, instead shown as “+”, in Figure 7.8b.

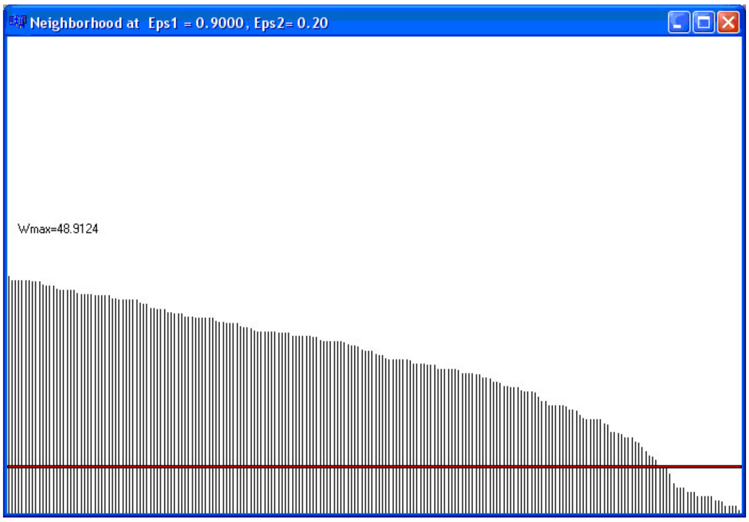
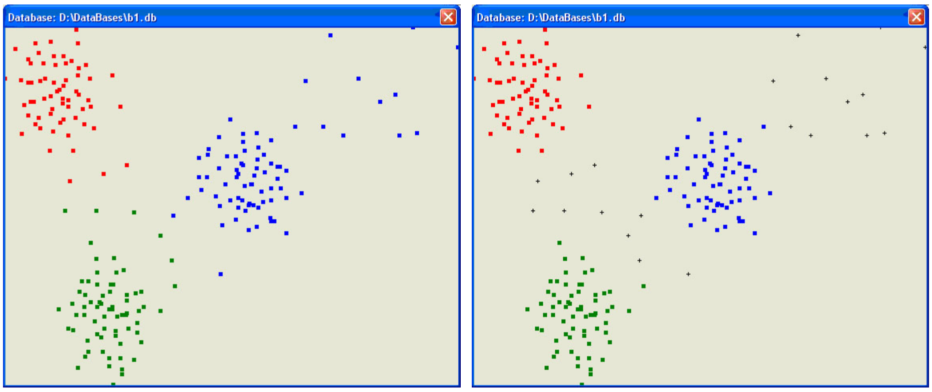


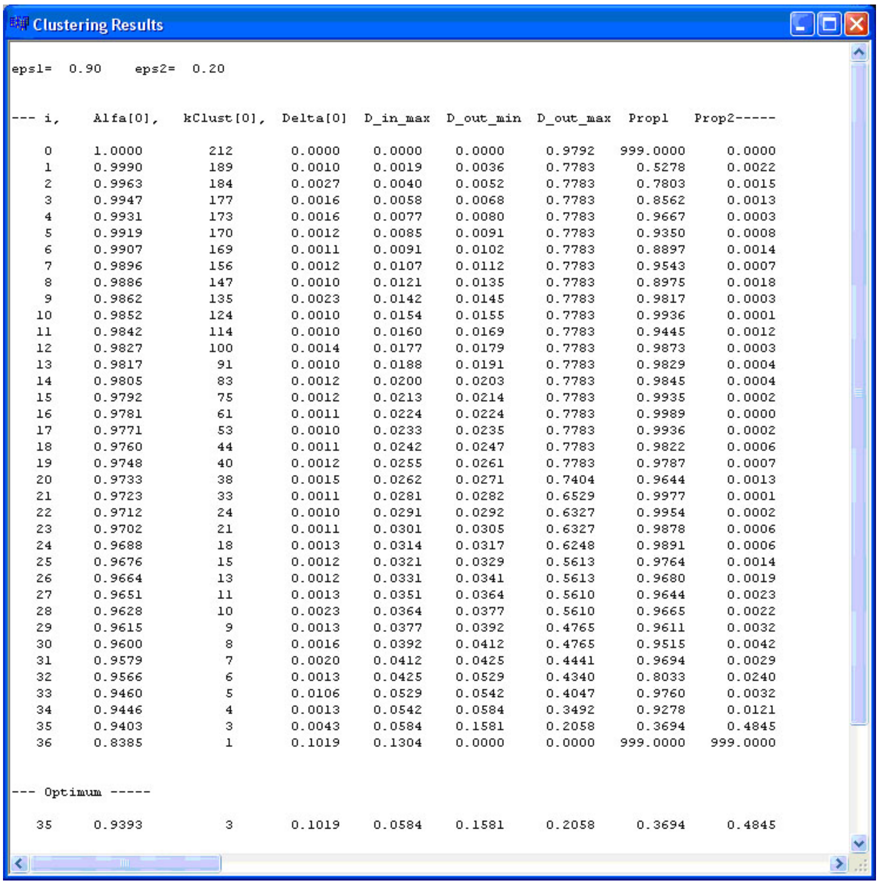
Figure 7.7 Neighborhood degrees of all elements in decreasing order.



(a) (b)

Figure 7.8 Visualization of clustering results with (a) points assigned to nearest clusters, (b) noise points.

The clustering results of each α-level, in which the number of clusters is affected, are written on a flat file. “FJP.txt” button can be used at any time to display the contents of the flat file as shown in Figure 7.9. In the first line of this file, the values of the parameters ε_1 and ε_2 are shown. The columns “i” , “Alfa[0]”, “kClust[0]”, and “Delta[0]” show, by order, the step number, α-level in which the number of clusters is changed, number of these clusters convenient to α-level, and the difference between the preceding and corresponding α-level. “D_in_max”, “D_out_min” and “D_out_max” are the values of formulas given in (4.36), (4.37) and (4.38), respectively, in order to calculate the clustering structure. Columns “Prop1” and “Prop2” are the lower and upper bounds of Theorem 4.7, respectively, which is used to determine the optimal clustering structure. Under the “Optimum” heading, abstract information for the optimal clustering structure obtained in the clustering process is given.



```
eps1=  0.90    eps2=  0.20

--- i,    Alfa[0],   kClust[0],  Delta[0]  D_in_max  D_out_min  D_out_max  Prop1    Prop2-----

    0     1.0000        212       0.0000    0.0000    0.0000     0.9792  999.0000     0.0000
    1     0.9990        189       0.0010    0.0019    0.0036     0.7783    0.5278     0.0022
    2     0.9963        184       0.0027    0.0040    0.0052     0.7783    0.7803     0.0015
    3     0.9947        177       0.0016    0.0058    0.0068     0.7783    0.8562     0.0013
    4     0.9931        173       0.0016    0.0077    0.0080     0.7783    0.9667     0.0003
    5     0.9919        170       0.0012    0.0085    0.0091     0.7783    0.9350     0.0008
    6     0.9907        169       0.0011    0.0091    0.0102     0.7783    0.8897     0.0014
    7     0.9896        156       0.0012    0.0107    0.0112     0.7783    0.9543     0.0007
    8     0.9886        147       0.0010    0.0121    0.0135     0.7783    0.8975     0.0018
    9     0.9862        135       0.0023    0.0142    0.0145     0.7783    0.9817     0.0003
   10     0.9852        124       0.0010    0.0154    0.0155     0.7783    0.9936     0.0001
   11     0.9842        114       0.0010    0.0160    0.0169     0.7783    0.9445     0.0012
   12     0.9827        100       0.0014    0.0177    0.0179     0.7783    0.9873     0.0003
   13     0.9817         91       0.0010    0.0188    0.0191     0.7783    0.9829     0.0004
   14     0.9805         83       0.0012    0.0200    0.0203     0.7783    0.9845     0.0004
   15     0.9792         75       0.0012    0.0213    0.0214     0.7783    0.9935     0.0002
   16     0.9781         61       0.0011    0.0224    0.0224     0.7783    0.9989     0.0000
   17     0.9771         53       0.0010    0.0233    0.0235     0.7783    0.9936     0.0002
   18     0.9760         44       0.0011    0.0242    0.0247     0.7783    0.9822     0.0006
   19     0.9748         40       0.0012    0.0255    0.0261     0.7783    0.9787     0.0007
   20     0.9733         38       0.0015    0.0262    0.0271     0.7404    0.9644     0.0013
   21     0.9723         33       0.0011    0.0281    0.0282     0.6529    0.9977     0.0001
   22     0.9712         24       0.0010    0.0291    0.0292     0.6327    0.9954     0.0002
   23     0.9702         21       0.0011    0.0301    0.0305     0.6327    0.9878     0.0006
   24     0.9688         18       0.0013    0.0314    0.0317     0.6248    0.9891     0.0006
   25     0.9676         15       0.0012    0.0321    0.0329     0.5613    0.9764     0.0014
   26     0.9664         13       0.0012    0.0331    0.0341     0.5613    0.9680     0.0019
   27     0.9651         11       0.0013    0.0351    0.0364     0.5610    0.9644     0.0023
   28     0.9628         10       0.0023    0.0364    0.0377     0.5610    0.9665     0.0022
   29     0.9615          9       0.0013    0.0377    0.0392     0.4765    0.9611     0.0032
   30     0.9600          8       0.0016    0.0392    0.0412     0.4765    0.9515     0.0042
   31     0.9579          7       0.0020    0.0412    0.0425     0.4441    0.9694     0.0029
   32     0.9566          6       0.0013    0.0425    0.0529     0.4340    0.8033     0.0240
   33     0.9460          5       0.0106    0.0529    0.0542     0.4047    0.9760     0.0032
   34     0.9446          4       0.0013    0.0542    0.0584     0.3492    0.9278     0.0121
   35     0.9403          3       0.0043    0.0584    0.1581     0.2058    0.3694     0.4845
   36     0.8385          1       0.1019    0.1304    0.0000     0.0000  999.0000   999.0000


--- Optimum -----

   35     0.9393          3       0.1019    0.0584    0.1581     0.2058    0.3694     0.4845
```

Figure 7.9 Flat file of clustering results.

7.1.2 Functional Modules

Followings are the functional modules assigned to the corresponding buttons on the opening window of the NRFJP program and their functions:

OpenDB - Opens the data set on which the clustering process will be realized (Figure 7.3).

ViewDB - Shows the two-dimensional data set visually (Figure 7.4).

Solve - Runs NRFJP algorithm.

Visualization - Shows the clustering result visually (Figure 7.8).

Neighborhood - Shows neighborhood degree graph for each element in decreasing order. It can be used to set up the value of ε_2 parameter (Figure 7.7).

FJP.TXT - Provides access to the flat file "FJP.txt" in which the clustering results are saved (Figure 7.9).

7.1.3 Informative Components

Timer - Keeps time for the clustering process if it is checked.

kOpt - Shows the optimal number of clusters.

Alfa - Shows the α-level convenient to the optimal cluster number.

Time(sec) - Shows the elapsed time for the clustering process.

Eps1 - Fixes the value of ε_1 parameter which determines the threshold of the neighborhood cardinality.

Eps2 - Fixes the value of ε_2 parameter which determines the threshold of the neighborhood density.

K - Fixes the value of the parameter “k” used in the neighborhood membership function.

FromPrototypes - Takes into account the distance from prototypes in order to assign noise points to the nearest clusters.

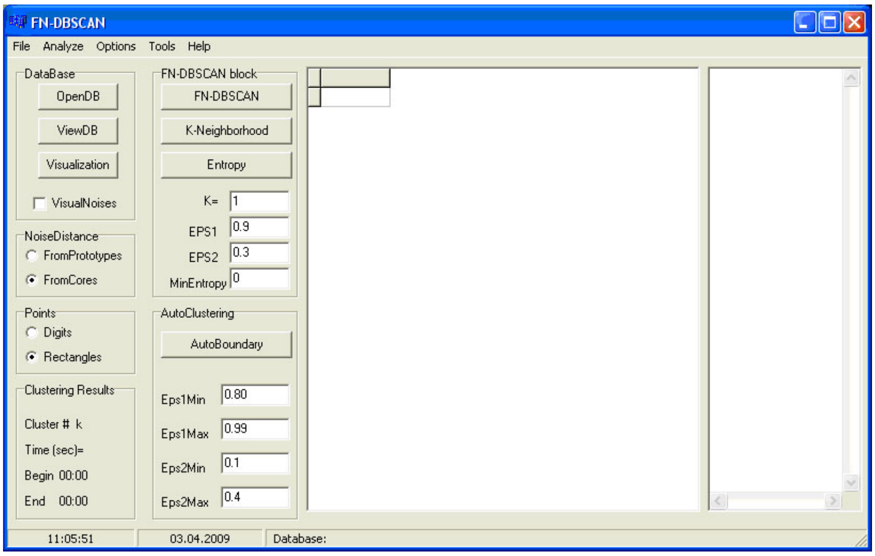
FromCores - Takes into account the distance from core points in order to assign noise points to the nearest clusters.

VisualNoises - If this option is checked before clicking the “Visualization” button, noise points are shown as “+”, *i.e.* they are not assigned to clusters.

VisualStep - If this option is checked before clicking the “Solve” button, clustering results for each affected α-level is shown on the screen one-by-one. In each display, confirmation is needed to continue clustering process.

7.2 Software of FN-DBSCAN Algorithm

In this section, we will explain the running principle of the software of FN-DBSCAN algorithm mentioned in Section 4.3.



□ VisualNoises

○ FromPrototypes
● FromCores

○ Digits
● Rectangles

K= 1
EPS1 0.9
EPS2 0.3
MinEntropy 0

Eps1Min 0.80
Eps1Max 0.99
Eps2Min 0.1
Eps2Max 0.4

Cluster # k
Time (sec)=
Begin 00:00
End 00:00

11:05:51 | 03.04.2009 | Database:

Figure 7.10 Opening window of FN-DBSCAN program.

7.2.1 Forms

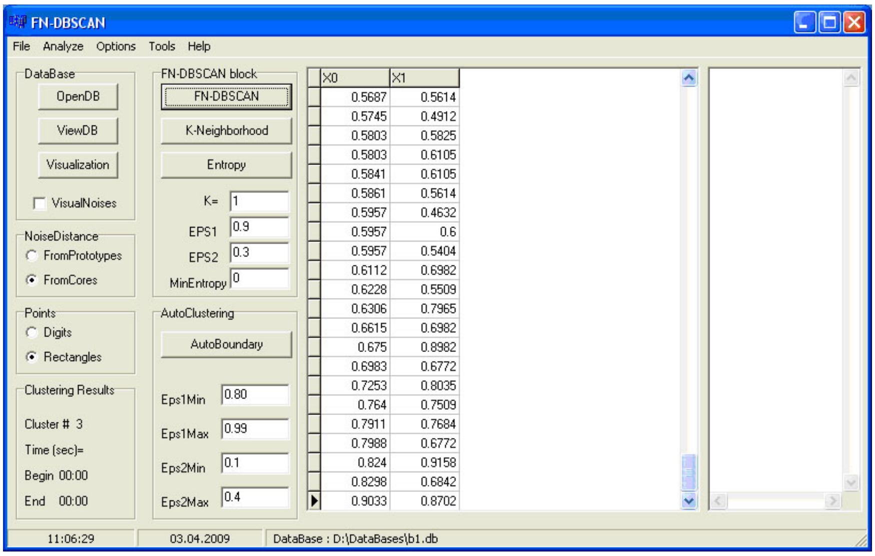
In Figure 7.1, when the FN-DBSCAN button is clicked, Figure 7.10 appears on the screen. First of all, it is required to select the table by pressing on the "OpenDB" button. It is possible to see the shape of the data set by pressing on the "ViewDB"'button given as in NRFJP program. The parameters ε_1 and ε_2 of FN-DBSCAN algorithm can be fixed according to the data set. Furthermore, we can determine the parameter "k" used in various neighborhood membership functions. By clicking on the "Options" menu item from the main menu, the form given in Figure 7.5 appears on the screen that provides selection for distance metric and neighborhood membership function. For instance, select "Euclidian distance" and linear neighborhood membership function from Options and determine "k" as 1. When "FN-DBSCAN" button from "FN-DBSCAN block" panel in the main menu is clicked, then clustering process starts.

In "Clustering Results" panel placed on the lower left hand side of Figure 7.11, it is shown that "Cluster # 3", *i.e.* the optimal number of clusters is found as 3. "Visualization" button again provides a visual presentation for the clustering results. As in NRFJP program system, this program offers various options for the presentation of clustering results in the "Points" and "Noise Distance" panels.

This time, we run FN-DBSCAN algorithm by selecting exponential neighborhood function from the "Options" window given in Figure 7.5. When this option is checked, due to the previous experiments as the most suitable value, the parameter "K" is determined as 8.3 automatically. But another value can also be specified for this parameter.

It is also possible to see the optimal number of clusters for each value of the parameters ε_1 and ε_2 found by the algorithm by specifying the upper and lower bounds in the "AutoClustering" panel. At the beginning program window "Eps1Min", "Eps1Max", "Eps2Min" and "Eps2Max" values are 0.80, 0.99, 0.1 and 0.4, respectively. For example, in Figure 7.12, we specify Eps1Min as 0.01, Eps1Max as 0.99, Eps2Min as 0.1, and Eps2Max as 0.4, which means that the algorithm will start by ε_1 and ε_2

values, and in each step by increasing ε_1 0.01 through 0.99, for each value of ε_2 by increasing 0.1 after pressing "AutoBoundary" button, algorithm will run 400 times. For each specified value of the parameter, we get an optimal clustering structure. Hence, by running FN-DBSCAN algorithm for 400 times, and the results are being written to Memo on the right hand side of the program window for each value given in Figure 7.12. In this manner, for each cluster number, we can detect the working ranges of the parameters. In Figure 7.12, optimal number of clusters is found as "Cluster # 33" which is the result for the last running of the algorithm for $\varepsilon_1 = 0.99$ and $\varepsilon_2 = 0.4$. Besides, on the left hand side part of the form, the elapsed time for the auto-clustering process can be seen. The values across the "Begin" and "End" labels are the starting, and the final second and split second.



□ VisualNoises

NoiseDistance
○ FromPrototypes
● FromCores

Points
○ Digits
● Rectangles

Clustering Results
Cluster # 3
Time (sec)=
Begin 00:00
End 00:00

K= 1
EPS1 0.9
EPS2 0.3
MinEntropy 0

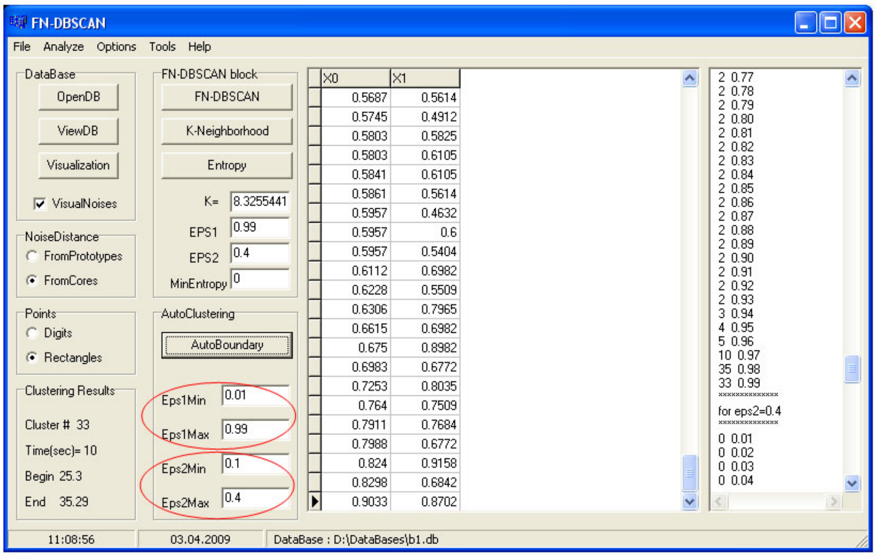
Eps1Min 0.80
Eps1Max 0.99
Eps2Min 0.1
Eps2Max 0.4

X0	X1
0.5687	0.5614
0.5745	0.4912
0.5803	0.5825
0.5803	0.6105
0.5841	0.6105
0.5861	0.5614
0.5957	0.4632
0.5957	0.6
0.5957	0.5404
0.6112	0.6982
0.6228	0.5509
0.6306	0.7965
0.6615	0.6982
0.675	0.8982
0.6983	0.6772
0.7253	0.8035
0.764	0.7509
0.7911	0.7684
0.7988	0.6772
0.824	0.9158
0.8298	0.6842
0.9033	0.8702

11:06:29 | 03.04.2009 | DataBase : D:\DataBases\b1.db

Figure 7.11 Clustering results of FN-DBSCAN algorithm with parameters $\varepsilon_1 = 0.9$ and $\varepsilon_2 = 0.3$.

In this program, simulating clusters from certain distributions and structures for certain number of data are also allowed by selecting "Cluster Generator" from "Tools" menu item (Figures 7.13-7.14). By the help of the "Cluster Generator" auxiliary procedure, it is possible to work on an existing data set by clicking on the "Open" button as well as simulating a data set.



☑ VisualNoises

NoiseDistance
○ FromPrototypes
● FromCores

Points
○ Digits
● Rectangles

Clustering Results
Cluster # 33
Time(sec)= 10
Begin 25.3
End 35.29

K= 8.3255441
EPS1 0.99
EPS2 0.4
MinEntropy 0

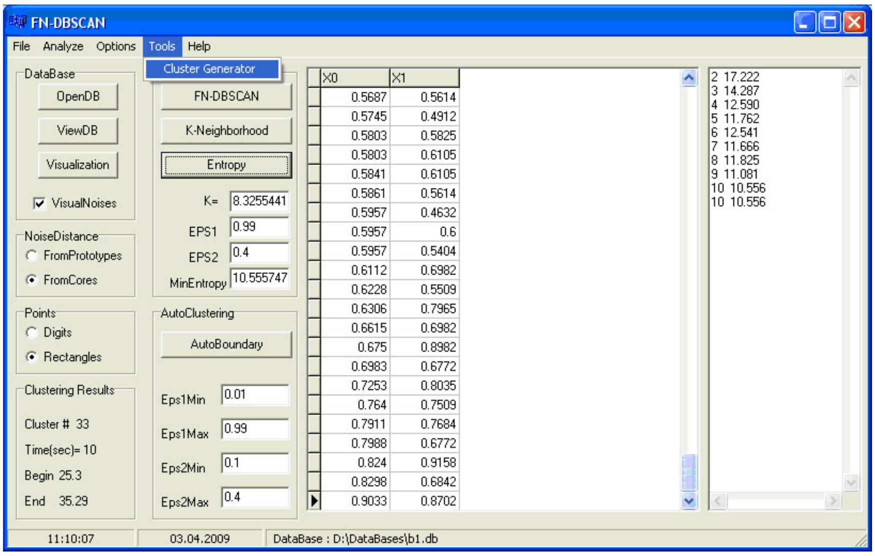
Eps1Min 0.01
Eps1Max 0.99
Eps2Min 0.1
Eps2Max 0.4

X0	X1
0.5687	0.5614
0.5745	0.4912
0.5803	0.5825
0.5803	0.6105
0.5841	0.6105
0.5861	0.5614
0.5957	0.4632
0.5957	0.6
0.5957	0.5404
0.6112	0.6982
0.6228	0.5509
0.6306	0.7965
0.6615	0.6982
0.675	0.8982
0.6983	0.6772
0.7253	0.8035
0.764	0.7509
0.7911	0.7684
0.7988	0.6772
0.824	0.9158
0.8298	0.6842
0.9033	0.8702

```
2 0.77
2 0.78
2 0.79
2 0.80
2 0.81
2 0.82
2 0.83
2 0.84
2 0.85
2 0.86
2 0.87
2 0.88
2 0.89
2 0.90
2 0.91
2 0.92
2 0.93
3 0.94
4 0.95
5 0.96
10 0.97
35 0.98
33 0.99
**************
for eps2=0.4
**************
0 0.01
0 0.02
0 0.03
0 0.04
```

11:08:56 | 03.04.2009 | DataBase : D:\DataBases\b1.db

Figure 7.12 Automatic clustering results of FN-DBSCAN algorithm.



☑ VisualNoises

NoiseDistance
○ FromPrototypes
● FromCores

Points
○ Digits
● Rectangles

Clustering Results
Cluster # 33
Time(sec)= 10
Begin 25.3
End 35.29

K= 8.3255441
EPS1 0.99
EPS2 0.4
MinEntropy 10.555747

Eps1Min 0.01
Eps1Max 0.99
Eps2Min 0.1
Eps2Max 0.4

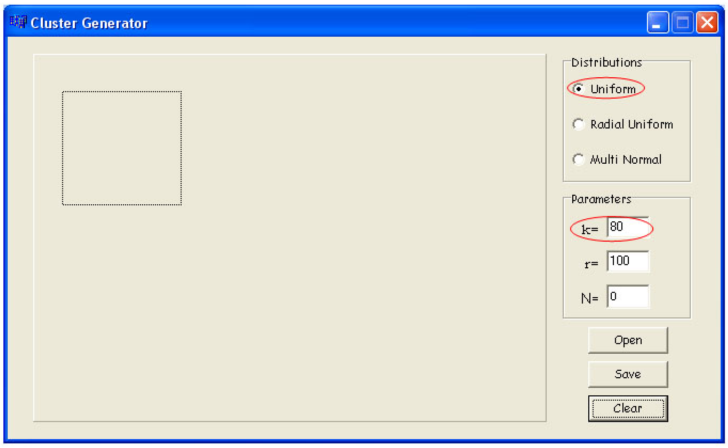
X0	X1
0.5687	0.5614
0.5745	0.4912
0.5803	0.5825
0.5803	0.6105
0.5841	0.6105
0.5861	0.5614
0.5957	0.4632
0.5957	0.6
0.5957	0.5404
0.6112	0.6982
0.6228	0.5509
0.6306	0.7965
0.6615	0.6982
0.675	0.8982
0.6983	0.6772
0.7253	0.8035
0.764	0.7509
0.7911	0.7684
0.7988	0.6772
0.824	0.9158
0.8298	0.6842
0.9033	0.8702

```
2 17.222
3 14.287
4 12.590
5 11.762
6 12.541
7 11.666
8 11.825
9 11.081
10 10.556
10 10.556
```

11:10:07 | 03.04.2009 | DataBase : D:\DataBases\b1.db

Figure 7.13 Selection of Cluster Generator procedure.

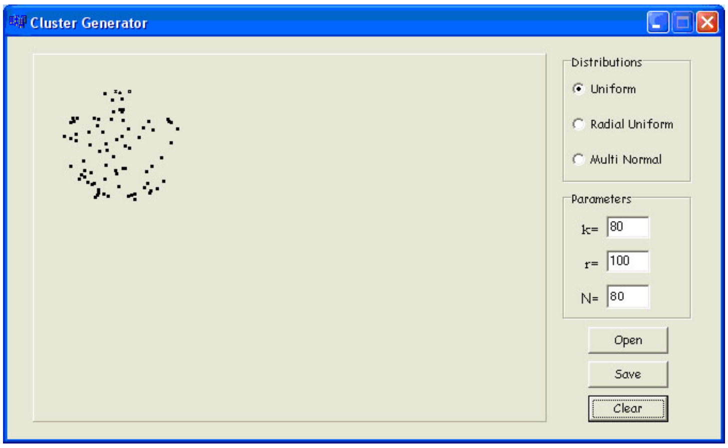
In order to simulate a data set, first of all, the distribution of clusters should be selected from “Distributions” panel located at the upper right hand side of the form. Afterwards, the desired number of elements (k) in each attempt should be determined. The clusters can be simulated in two ways. The first one is realized by determining the radius of the cluster by entering the value on “r” label and then clicking on the screen. The other way is drawing a rectangle within a desired radius on the screen by the help of the mouse (Figure 7.14). Thus, when we drop out clicking on the mouse, the first cluster within a defined radius and with defined number of elements is generated. Let the clusters have a uniform distribution with 80 elements in each cluster. When we draw a rectangle on the screen, the first cluster is placed in this area (Figure 7.15). After each generated cluster, total number of elements of the data set is written across "N" placed at the "Parameters" panel. In Figure 7.16, it is shown that there are 4 clusters each having 80 elements. We can save our data set by clicking on the "Save" button, or we can clear data window by clicking on the "Clear" button.



Distributions
● Uniform
○ Radial Uniform
○ Multi Normal

Parameters
k= 80
r= 100
N= 0

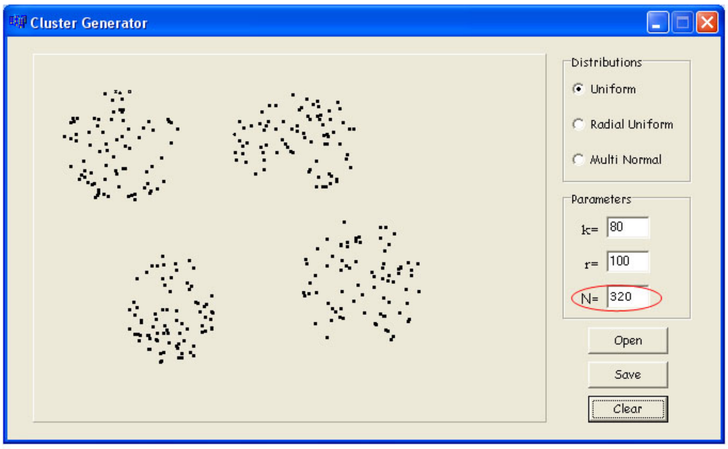
Figure 7.14 Opening window of Cluster Generator procedure and determination of cluster frame.



Distributions
● Uniform
○ Radial Uniform
○ Multi Normal

Parameters
k= 80
r= 100
N= 80

Figure 7.15 The first cluster has been generated within a defined frame.



Distributions

● Uniform

○ Radial Uniform

○ Multi Normal

Parameters

k= 80

r= 100

N= 320

Figure 7.16 Four clusters have been generated.

7.2.2 *Functional Modules*

Followings are the functional modules assigned to the corresponding buttons on the opening window of the FN-DBSCAN program and their functions:

OpenDB - Opens the data set on which the clustering process are realized (Figure 7.3).

ViewDB - Shows the two-dimensional data set visually.

FN-DBSCAN - Runs FN-DBSCAN algorithm.

Visualization - Shows the clustering result visually.

K-Neighborhood - Shows neighborhood degree graph of each element. It can be used to set up the value of ε_2 parameter.

Entropy - Determines the optimal value of ε_1 parameter by using entropy-based methods explained in Section 4.2.1.

AutoBoundary - Enables running FN-DBSCAN algorithm within specified ε_1 and ε_2 ranges, *i.e.* $\left([Eps1Max - Eps1Min] \times 100 + 1\right) \times \left([Eps2Max - Eps2Min] \times 10 + 1\right)$ times, in order to determine the working parameter ranges of the algorithm.

7.2.3 Informative Components

FromPrototypes - Considers the distance from prototypes to assign noise points to the nearest clusters.

FromCores - Considers the distance from core points to assign noise points to the nearest clusters.

VisualNoises - If this option is checked before clicking the "Visualization" button, noise points are shown as "+", *i.e.* they are not assigned to clusters.

Digits - Shows visual clustering results as digits. Note that each different digit represents a cluster.

Rectangles - Shows visual clustering results as rectangles. Note that each different color represents a cluster.

Cluster # - Shows optimal number of clusters at the end of the clustering process.

Time(sec) - Displays the elapsed time for clustering process.

Begin - Displays the starting time in seconds and split seconds.

End - Displays the final time in seconds and split seconds.

K - Specifies the value of parameter *k* used in the neighborhood membership function.

Eps1 - Fixes the value of ε_1 parameter which determines the threshold of the neighborhood cardinality.

Eps2 - Fixes the value of ε_2 parameter which determines the threshold of the neighborhood density.

MinEntropy - Displays the minimum entropy value after clicking on the "Entropy" button.

Eps1Min - Specifies the minimum value of ε_1 parameter that is to be used in automatic clustering.

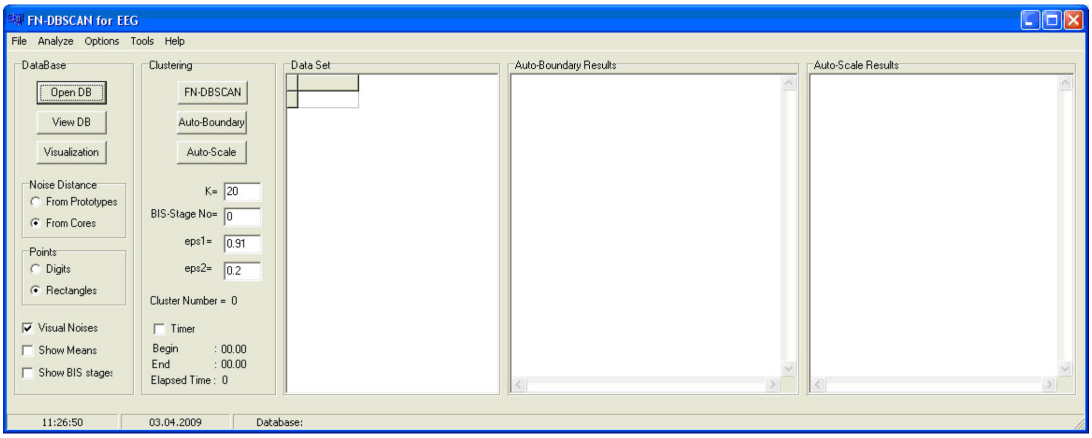
Eps1Max - Specifies the maximum value of ε_1 parameter that is to be used in automatic clustering.

Eps2Min - Specifies the minimum value of ε_2 parameter that is to be used in automatic clustering.

Eps2Max - Specifies the maximum value of ε_2 parameter that is to be used in automatic clustering.

7.3 Software of FN-DBSCAN Algorithm for EEG

In this section, we will explain the running principle of the software of FN-DBSCAN algorithm, designed for EEG, which was explained in Section 5.2.



Noise Distance
○ From Prototypes
● From Cores

Points
○ Digits
● Rectangles

☑ Visual Noises
☐ Show Means
☐ Show BIS stage:

K= 20
BIS-Stage No= 0
eps1= 0.91
eps2= 0.2

Cluster Number = 0

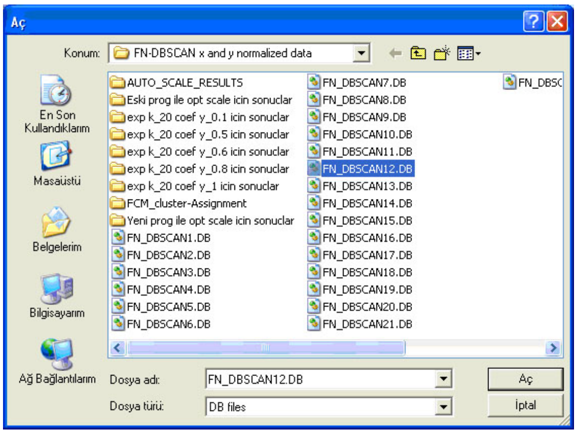
☐ Timer
Begin : 00.00
End : 00.00
Elapsed Time : 0

11:26:50 | 03.04.2009 | Database:

Figure 7.17 Opening window of FN-DBSCAN algorithm for EEG program.

7.3.1 Forms

In Figure 7.1, when the FN-DBSCAN EEG button is clicked, Figure 7.17 appears on the screen. First of all, it is required to select the table by pressing on the “OpenDB” button (Figure 7.18). It is possible to see the shape of the data set by pressing on the “ViewDB” button as we did in previous programmes (Figure 7.19).

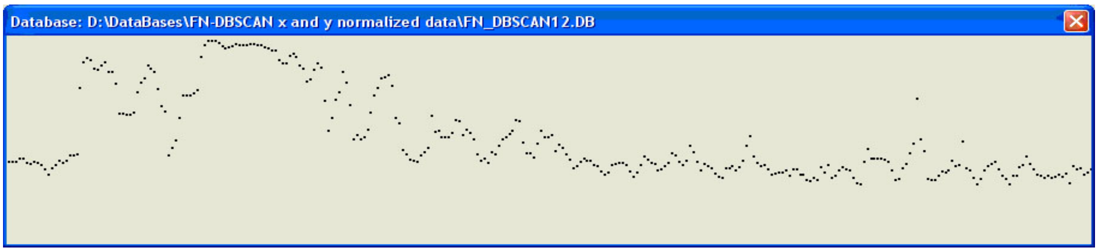


Konum: FN-DBSCAN x and y normalized data

AUTO_SCALE_RESULTS
Eski prog ile opt scale icin sonuclar
exp k_20 coef y_0.1 icin sonuclar
exp k_20 coef y_0.5 icin sonuclar
exp k_20 coef y_0.6 icin sonuclar
exp k_20 coef y_0.8 icin sonuclar
exp k_20 coef y_1 icin sonuclar
FCM_cluster-Assignment
Yeni prog ile opt scale icin sonuclar
FN_DBSCAN1.DB
FN_DBSCAN2.DB
FN_DBSCAN3.DB
FN_DBSCAN4.DB
FN_DBSCAN5.DB
FN_DBSCAN6.DB
FN_DBSCAN7.DB
FN_DBSCAN8.DB
FN_DBSCAN9.DB
FN_DBSCAN10.DB
FN_DBSCAN11.DB
FN_DBSCAN12.DB
FN_DBSCAN13.DB
FN_DBSCAN14.DB
FN_DBSCAN15.DB
FN_DBSCAN16.DB
FN_DBSCAN17.DB
FN_DBSCAN18.DB
FN_DBSCAN19.DB
FN_DBSCAN20.DB
FN_DBSCAN21.DB
FN_DBSC

Dosya adı: FN_DBSCAN12.DB
Dosya türü: DB files

Figure 7.18 Data set selection.

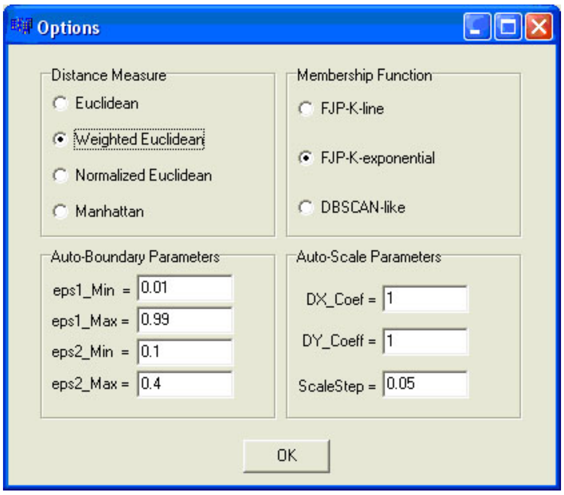


Database: D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.19 Visualization of data set.

It is possible to make various selections by clicking on the “Options” menu item. For instance, distance metric from “Distance Measure”, neighborhood membership function from “Membership Function”, lower and upper bounds of ε_1 and ε_2 parameters to realize automatic clustering from “Auto-Boundary Parameters” panel, the weights of variable and the amount of weight that is to be used to determine scale automatically from “Auto-Scale Parameters” panel are possible. Unless “Weighted Euclidian” distance measure option is checked, “DY_Coef” and “ScaleStep” options are inactive.

As we have just mentioned above, there are options for clustering with various distance metrics. For instance, “Weighted Euclidian” distance can be used to obtain better clustering results by assigning different weights to time (X) and BIS-value (Y) variables. The relative weights of these variables can be set up by using parameter used in Formula (6.8). In order to find out the best value of this parameter in a certain range, “Auto-Scale Parameters” panel in “Options” menu item can be used. In Figure 7.20, “DX_Coef” and “DY_Coef” are the weights of X and Y variables, respectively. “Scale Step” is the incremental step of parameter that reflects the relative weight. Note that the weight of Y variable is the corresponding η parameter.



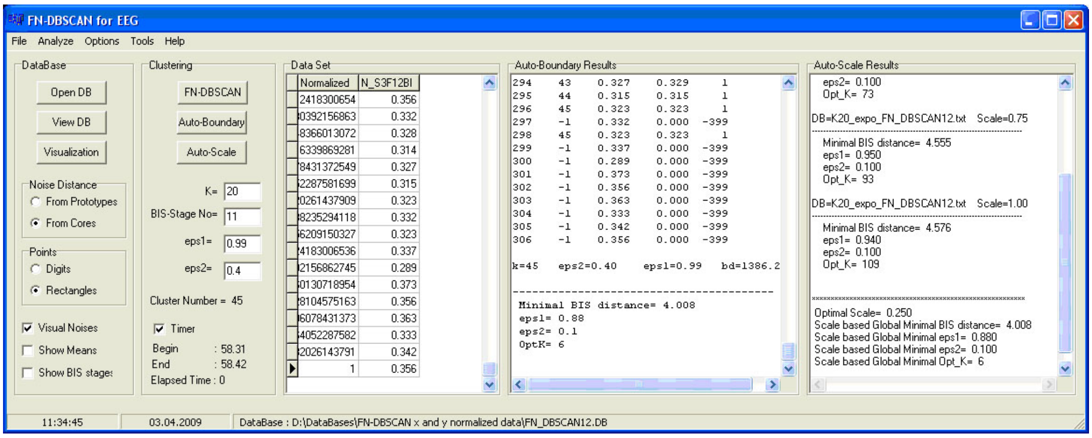
Distance Measure
○ Euclidean
● Weighted Euclidean
○ Normalized Euclidean
○ Manhattan

Membership Function
○ FJP-K-line
● FJP-K-exponential
○ DBSCAN-like

Auto-Boundary Parameters
eps1_Min = 0.01
eps1_Max = 0.99
eps2_Min = 0.1
eps2_Max = 0.4

Auto-Scale Parameters
DX_Coef = 1
DY_Coeff = 1
ScaleStep = 0.05

Figure 7.20 Options window.



Noise Distance
○ From Prototypes
● From Cores

Points
○ Digits
● Rectangles

☑ Visual Noises
□ Show Means
□ Show BIS stages

K= 20
BIS-Stage No= 11
eps1= 0.99
eps2= 0.4
Cluster Number = 45

☑ Timer
Begin : 58.31
End : 58.42
Elapsed Time : 0

Data Set

Normalized	N_S3F12BI
2418300654	0.356
0392156863	0.332
8366013072	0.328
6339869281	0.314
8431372549	0.327
2287581699	0.315
0261437909	0.323
8235294118	0.332
6209150327	0.323
4183006536	0.337
2156862745	0.289
0130718954	0.373
8104575163	0.356
6078431373	0.363
4052287582	0.333
2026143791	0.342
1	0.356

Auto-Boundary Results

```
294    43    0.327    0.329      1
295    44    0.315    0.315      1
296    45    0.323    0.323      1
297    -1    0.332    0.000   -399
298    45    0.323    0.323      1
299    -1    0.337    0.000   -399
300    -1    0.289    0.000   -399
301    -1    0.373    0.000   -399
302    -1    0.356    0.000   -399
303    -1    0.363    0.000   -399
304    -1    0.333    0.000   -399
305    -1    0.342    0.000   -399
306    -1    0.356    0.000   -399

k=45    eps2=0.40    eps1=0.99    bd=1386.2

----------------------------------------
 Minimal BIS distance= 4.008
 eps1= 0.88
 eps2= 0.1
 OptK= 6
```

Auto-Scale Results

```
   eps2= 0.100
   Opt_K= 73

DB=K20_expo_FN_DBSCAN12.txt   Scale=0.75
------------------------------------------------------------
   Minimal BIS distance= 4.555
   eps1= 0.950
   eps2= 0.100
   Opt_K= 93

DB=K20_expo_FN_DBSCAN12.txt   Scale=1.00
------------------------------------------------------------
   Minimal BIS distance= 4.576
   eps1= 0.940
   eps2= 0.100
   Opt_K= 109


************************************************************
 Optimal Scale= 0.250
 Scale based Global Minimal BIS distance= 4.008
 Scale based Global Minimal eps1= 0.880
 Scale based Global Minimal eps2= 0.100
 Scale based Global Minimal Opt_K= 6
```

11:34:45 | 03.04.2009 | DataBase : D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.21 Auto-Scale results.

The value of "DY_Coef" coefficient is increased from beginning through 1 as much as "Scale Step". Then the value of η that gives the nearest result to the expert's opinion is considered as optimal.

After closing "Options" window, if "Auto-Scale" button is clicked, the results are shown in Figure 7.21. In "Auto-Boundary Results" part, given in the middle Memo, the sequence number, its reference cluster number, cluster mean, stage mean given by expert and stage mean found by the method are shown for each element according to each value of ε_1 and ε_2 parameters. In "Auto-Scale Results" part, the nearest result to the expert's opinion, *i.e.* "MinimalBIS distance", optimal values of ε_1 and

ε_2 parameters and the optimal number of clusters for that scale are shown. Finally, in the bottom the optimal values for the optimal scale among the above-mentioned results are given (Figure 7.22). If “Show BIS stages” item is checked, placed at the bottom left hand side of the form, before pressing “Visualization” button, it is possible to show stage values both given by the expert (black lines) and obtained by the FN-DBSCAN method (colored lines) for a visual interpretation and comparison (Figure 7.23).

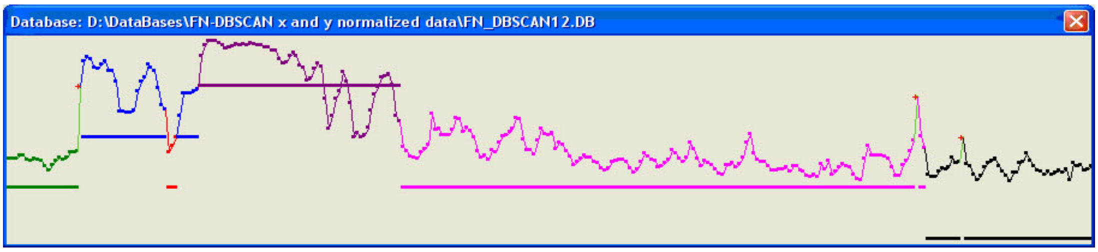

Database: D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.22 Clustering results of FN-DBSCAN based algorithm.

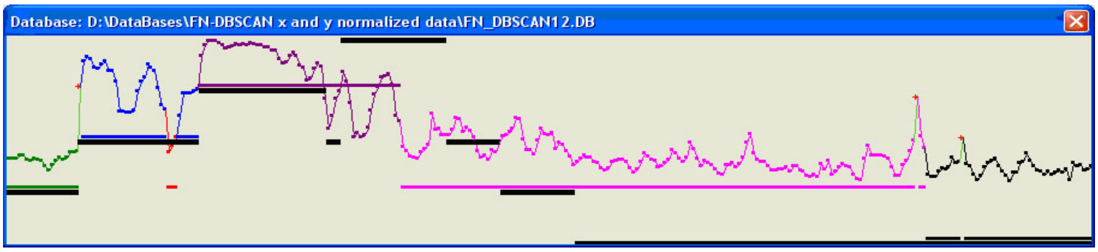
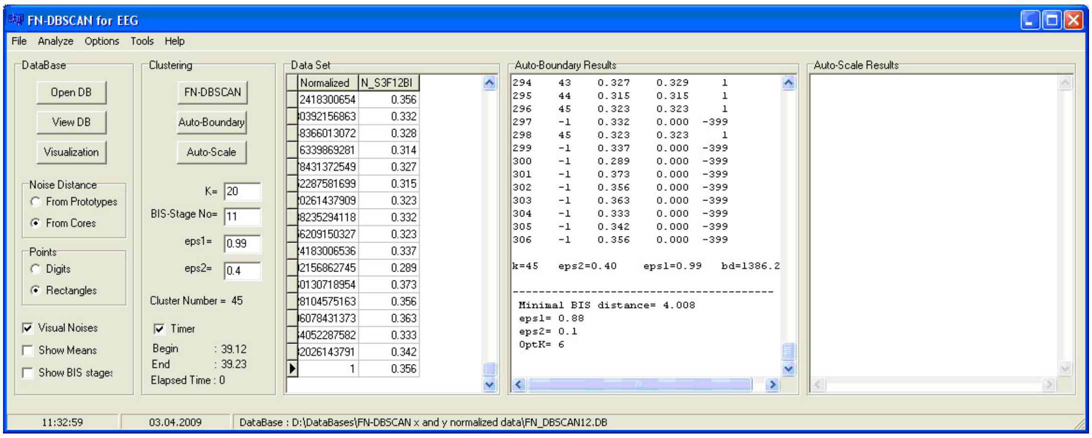

Database: D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.23 Clustering results, stage values given by expert (black lines), stage values obtained by FN-DBSCAN (colored lines).

So that we can see the clustering results by keeping the scale constant and entering the minimum and maximum values of ε_1 and ε_2 in “Options” window, it is enough to press “Auto-Boundary” button. The results are shown only in “Auto-Boundary Results” part. In Figure 7.24, the elapsed time for detecting boundaries of ε_1 and ε_2 by automatic clustering for $\eta = 0.25$ after checking “Timer” item is shown.

In addition to the information given in Figure 7.24, if “Auto-Boundary” button is pressed after checking “ShowMeans” item in the main form, the number of elements in each cluster and means of elements in that cluster for each dimension are also shown for the optimal parameter values (Figure 7.25).



Noise Distance: ○ From Prototypes ● From Cores

Points: ○ Digits ● Rectangles

☑ Visual Noises

☐ Show Means

☐ Show BIS stages

K= 20

BIS-Stage No= 11

eps1= 0.99

eps2= 0.4

Cluster Number = 45

☑ Timer

Begin : 39.12

End : 39.23

Elapsed Time : 0

Normalized	N_S3F12BI
2418300654	0.356
0392156863	0.332
8366013072	0.328
6339869281	0.314
8431372549	0.327
2287581699	0.315
0261437909	0.323
8235294118	0.332
6209150327	0.323
4183006536	0.337
2156862745	0.289
0130718954	0.373
8104575163	0.356
6078431373	0.363
4052287582	0.333
2026143791	0.342
1	0.356

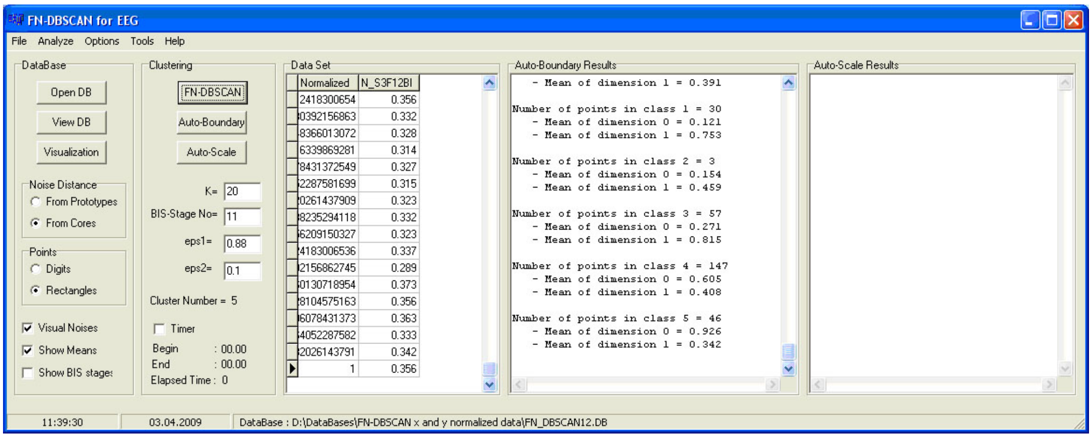
```
294    43    0.327    0.329      1
295    44    0.315    0.315      1
296    45    0.323    0.323      1
297    -1    0.332    0.000   -399
298    45    0.323    0.323      1
299    -1    0.337    0.000   -399
300    -1    0.289    0.000   -399
301    -1    0.373    0.000   -399
302    -1    0.356    0.000   -399
303    -1    0.363    0.000   -399
304    -1    0.333    0.000   -399
305    -1    0.342    0.000   -399
306    -1    0.356    0.000   -399

k=45   eps2=0.40    eps1=0.99   bd=1386.2

----------------------------------------
 Minimal BIS distance= 4.008
 eps1= 0.88
 eps2= 0.1
 OptK= 6
```

11:32:59 | 03.04.2009 | DataBase : D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.24 Auto-Boundary results.



Noise Distance: ○ From Prototypes ● From Cores

Points: ○ Digits ● Rectangles

☑ Visual Noises

☑ Show Means

☐ Show BIS stages

K= 20

BIS-Stage No= 11

eps1= 0.88

eps2= 0.1

Cluster Number = 5

☐ Timer

Begin : 00.00

End : 00.00

Elapsed Time : 0

Normalized	N_S3F12BI
2418300654	0.356
0392156863	0.332
8366013072	0.328
6339869281	0.314
8431372549	0.327
2287581699	0.315
0261437909	0.323
8235294118	0.332
6209150327	0.323
4183006536	0.337
2156862745	0.289
0130718954	0.373
8104575163	0.356
6078431373	0.363
4052287582	0.333
2026143791	0.342
1	0.356

```
   - Mean of dimension 1 = 0.391

Number of points in class 1 = 30
   - Mean of dimension 0 = 0.121
   - Mean of dimension 1 = 0.753

Number of points in class 2 = 3
   - Mean of dimension 0 = 0.154
   - Mean of dimension 1 = 0.459

Number of points in class 3 = 57
   - Mean of dimension 0 = 0.271
   - Mean of dimension 1 = 0.815

Number of points in class 4 = 147
   - Mean of dimension 0 = 0.605
   - Mean of dimension 1 = 0.408

Number of points in class 5 = 46
   - Mean of dimension 0 = 0.926
   - Mean of dimension 1 = 0.342
```

11:39:30 | 03.04.2009 | DataBase : D:\DataBases\FN-DBSCAN x and y normalized data\FN_DBSCAN12.DB

Figure 7.25 Auto-Boundary results with means of each cluster in each dimension.

7.3.2 *Functional Modules*

The functions of the buttons “OpenDB”, “ViewDB”, “Visualization” and “FN-DBSCAN” are the same as given in FN-DBSCAN programme.

Auto-Boundary - Runs FN-DBSCAN algorithm for each value of ε_1 and ε_2 parameters within range defined in Figure 7.20.

AutoScale - The scales of X and Y variables are set up in Figure 7.20. The scale of Y variable are increased as much as defined in “ScaleStep”. The optimal results for each scale and at the end the results for the optimal scale are found and written on the Memo given in the right hand side of form given in Figure 7.21.

7.3.3 Informative Components

In this programme, the functions of the informative components are the same as the ones given in FN-DBSCAN program except the followings:

ShowMeans - If this item is checked before pressing on the "Auto-Boundary" button, the number of elements in each cluster and cluster mean for each dimension for the optimal clustering structure are shown.

Show BIS stage - If this item is checked before pressing on the "Visualization" button, besides clustering results, the stage values both given by the expert (black lines) and obtained by the FN-DBSCAN method (colored lines) are shown.

BIS-Stage No - In order to compare the results found by the method with the results given by the expert and to find the nearest structure to the expert's opinion, it is necessary to enter a number in this part. Data sets are numbered from 1 to 21. Also, for each data set the stage values given by the expert are defined in the programme code. The comparison is made according to the number given into this part.

Elapsed Time - Shows the elapsed time for clustering in seconds if "Timer" item is checked.

Note that "The FJP Family" software system presented in this chapter is produced in the scope of the dissertation work. Consequently, this software system is used in order to evaluate the performances of NRFJP, FN-DBSCAN and FN-DBSCAN for EEG algorithms.

CHAPTER EIGHT
CONCLUSION

In this work, fundamentally two algorithms have been proposed. The first one is the NRFJP algorithm which is a robust version of the known fuzzy neighborhood based FJP clustering algorithm. The second one is the FN-DBSCAN algorithm which is a mixture of FJP and density-based DBSCAN algorithms. Moreover, in Borland C++ Builder 6.0 SDK system, the codes of both algorithms have been developed. The results of the work have been published in articles (Nasibov & Ulutagay, 2006c, 2007a,b,c, 2008a,b, 2009; Nasibov et al., 2008; Ulutagay & Nasibov, 2008a,b; Nasibov et al., 2009).

The fundamental idea of the FJP method is to compute the fuzzy relation matrix based on the distance between points. For this aim, for certain $\alpha \in (0,1]$, α-level sets and equivalence classes are constructed. At the same time, these α-degree equivalence classes determine each α-level set of the fuzzy clusters. Also note that, these α-level sets are not computed for all degrees, instead they are computed only for α-levels in which the number of clusters are affected. Then, the final level set is computed based on the maximal change interval of the α's. In other words, the α-level degree that reflects the cluster structure optimally and α-level set appropriate for this level are found simultaneously.

However, the FJP algorithm is not robust through noises. In order to eliminate such a disadvantage, in this work NRFJP algorithm has been suggested. In the NRFJP algorithm each point for which certain ε_1 fuzzy neighborhood cardinality is smaller than certain ε_2 threshold is perceived as noise. Note that, by changing the ε_1 and ε_2 parameters, it is possible to change the sensitivity of the NRFJP algorithm through noises. Moreover, if $\varepsilon_2 = 0$, the sensitivity of the NRFJP through noises is turned off, consequently NRFJP algorithm transforms into FJP algorithm.

The other proposed algorithm, FN-DBSCAN, is based on fuzzy neighborhood function. In the study, the effects of fuzzy neighborhood relation in density-based clustering have also been investigated. Besides being a more general algorithm, the FN-DBSCAN algorithm transforms into the well-known DBSCAN algorithm when the crisp neighborhood function is used. Experiments with various shapes and densities show that FN-DBSCAN algorithm is more robust than the DBSCAN algorithm is. On the other hand, FN-DBSCAN algorithm runs faster than the fuzzy neighborhood relation-based algorithms FJP and NRFJP. Thus, FN-DBSCAN algorithm combines the speed of DBSCAN and robustness of FJP algorithms. Computational experiments with parameter-based linear and exponential neighborhood functions have been performed. After experiments with several values of the parameters, the parameters that give better results have been obtained.

In summary, in this thesis;

1. Fuzzy neighborhood-based clustering method (FJP) has been handled and a *new NRFJP algorithm* which is FJP's noise-robust variant *has been suggested.*

2. To determine the optimal number of clusters in fuzzy neighborhood-based clustering, *a novel cluster validity criteria V_{FJP} has been proposed.*

3. The analysis of clustering structure has been investigated and *the sufficient condition for the optimal clustering structure has been formulated and proved as a theorem.*

4. In order to obtain the optimal values of the parameters of the NRFJP algorithm which determine the sensitivity through noise points, *an entropy-based method has been suggested.*

5. On the basis of a crisp density-based algorithm, DBSCAN, *a novel fuzzy neighborhood based FN-DBSCAN clustering algorithm has been proposed.* As a result of computational experiments, it has been shown that using fuzzy neighborhood relations in FN-DBSCAN algorithm *provides more robustness to noise points than using classical neighborhood relations as in the DBSCAN algorithm.*

6. *A modified version of the FN-DBSCAN algorithm has been developed* so as to apply cluster analysis to EEG data. Moreover, as *a real-world application*, this modified algorithm is applied to BIS sleep data which is recorded by EEG in the Department of Biophysics, Faculty of Medicine, Dokuz Eylül University. The experimental results showed that FN-DBSCAN based approach obtains closer results to the expert's opinion than the well-known and widely used FCM clustering based approach.

7. *The codes for all of the proposed algorithms have been developed* in Borland C++ Builder SDK and they have been designed as an integrated software system.

Improving the time and memory complexities of the proposed clustering algorithms will be the basis of our future investigations. Furthermore, it is aimed to investigate the behaviour of the algorithms in real-world applications from various fields.

REFERENCES

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD98)*, Seattle, WA, 94–105.

Akay, M. (2000). *Nonlinear Biomedical Signal Processing*. IEEE Press Series on Biomedical Engineering.

Ankerst, M., Breunig, M., Kriegel, H. P., & Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. *ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD99),*, Philedelphia, PA, 49–60.

Babuska, B. (1998). *Fuzzy Modeling for Control*. Kluwer Academic Publishers, Boston.

Ben-Hur, A., Horn, D., Siegelmann, H. T., & Vapnik, V. (2002). Support vector clustering. *Journal of Machine Learning Research*, 2, 125–137.

Bensaid, A. M., Hall, L. O., Bezdek, J. C., Clarke, L. P., Silbiger, M. L., Arrington, J. A., & Murtagh, R. F. (1996). Validity-guided (re)clustering with applications to image segmentation. *IEEE Transactions of Fuzzy Systems*, 4 (2), 112–123.

Bezdek, J. C. (1973). *Fuzzy Mathematics in Pattern Classification*. PhD thesis, Applied Mathematics Center, Cornell University, Ithaca.

Bezdek, J. C. (1974). Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3, 58–73.

Bezdek, J. C. (1975). Mathematical models for systematics and taxonomy. *8th Int. Conf. On Numerical Taxonomy*, San Fransisco, USA, 143–166.

Brecheisen, S., Kriegel, H., & Pfeifle, M. (2003). Efficient density-based clustering of complex objects. *4th IEEE International Conference on Data Mining*, United Kingdom, 43–50.

Cheeseman, P. & Stutz, J. (1996). *Advances in Knowledge Discovery and Data Mining*, chapter Bayesian Classification (AutoClass): Theory and Results, 153–180. AAAI/MIT Press, Cambridge, MA.

Daszykowski, M., Walczak, B., & Massart, D. (2004). Density-based clustering for exploration of analytical data. *Bioanal Chemistry*, 370–372.

Dong, Y., Zhuang, Y., Chen, K., & Tai, X. (2006). A hierarchical clustering algorithm based on fuzzy graph connectedness. *Fuzzy Sets and Systems*, 157, 1760–1774.

Dumitrescu, D., Lazzerini, B., & Jain, L. (2000). *Fuzzy Sets and Their Application to Clustering and Training*. CRC Press LLC.

Dunn, J. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3.

Dunn, J. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4, 95–104.

Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A density based algorithm for discovering clusters in large spatial datasets with noise. *2nd Int. Conf. KDDD*, Portland, Oregon, 1232–1239.

Fisher, D. (1987). Knowledge acquisition via conceptual clustering. *Machine Learning*, 2, 139–172.

Fukuyamo, Y. & Sugeno, M. (1989). A new method for choosing the number of clusters for the fuzzy c-means method. *5th Fuzzy System Symp.*, Japan, 247–250.

Gennari, J., Langley, P., & Fisher, D. (1989). Models for incremental concept formation. *Artificial Intelligence*, 40, 11–61.

Grabmaier, J. & Rudolph, A. (2002). Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery*, 6, 303–360.

Guha, S., Rastogi, R., & Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. *ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD98)*, Seattle, WA, 73–84.

Han, J. & Kamber, M. (2001). *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers.

Hartigan, J. (1975). *Clustering Algorithms*. John Wiley & Sons, New York.

Hartley, R. V. L. (1928). Transmission of information. *The Bell Systems Technical Journal*, 7, 535–563.

Höppner, F., Klawonn, F., Kruse, R., & Runkler, T. (1999). *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. John Wiley & Sons.

Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (SIGMOD-DMKD97)*, 55–58.

Johansen, J. W. & Sebel, P. S. (2000). Development and clinical application of electroencephalographic bispectrum monitoring. *Anesthesiology*, 93 (5), 1336–1344.

Karypis, G., Han, E. H., & Kumar, V. (1999). CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *Computer*, 32, 68–75.

Kaufmann, L. & Rousseeuw, P. J. (1990). *Finding Groups in Data*. John Wiley & Sons.

King, B. (1967). Step-wise clustering procedures. *Journal of the American Statistical Association*, 69, 86–101.

Klir, G. & Folger, T. (1988). *Fuzzy Sets, Uncertainty, and Information*. Prentice Hall, Englewood Cliffs, NJ.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 59–69.

Kung, C. C. & Su, J. Y. (2007). Affine takagi-sugeno fuzzy modelling algorithm by fuzzy c-regression models clustering with a novel cluster validity criterion. *IET Control Theory Applications*, 1 (5), 1255–1265.

Kwon, S. (1998). Cluster validity index for fuzzy clustering. *Electronic Letters*, 34 (22), 2176–2177.

Larose, D. (2005). *Discovering Knowledge in Data.* Wiley-Interscience.

Lukasiewicz, J. (1970). *Studies in Logic and the Foundations of Mathematics.* North Holland.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *5th Berkeley Symp. Mathemtaical Statistics & Probability*, 1, 281–297.

Murtagh, F. (1984). Structure of hierarchic clusterings: implications for information retrieval and for multivariate data analysis. *Information Processing and Management*, 20 (5), 611–617.

Nasibov, E. (2007). Analysis of fuzzy neighborhood relation in density-based clustering. *1st Int. Conf. On Soft Computing Technologies in Economy*, Baku, Azerbaijan, 113–123.

Nasibov, E. & Ulutagay, G. (2005a). A new fuzzy joint points method for fuzzy clustering. *Automatic Control and Computer Sciences*, 39 (6), 8–17.

Nasibov, E. & Ulutagay, G. (2005b). FJP: A new hierarchical method for fuzzy clustering. *3th Int. Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*, Antalya, Turkey, 212–220.

Nasibov, E. & Ulutagay, G. (2006a). A new fuzzy joint points criteria for cluster validity. *Int. Conf. on Modeling and Simulation (AMSE)*, Konya, Turkey, 625–629.

Nasibov, E. & Ulutagay, G. (2006b). On the fuzzy joint points method for fuzzy clustering problem. *Automatic Control and Computer Sciences*, 40 (5), 33–44.

Nasibov, E. & Ulutagay, G. (2006c). Optimal adjustment of the parameters of noise-resistant fuzzy joint points clustering method. *Int. Conf. On Application of Fuzzy Systems and Soft Computing (ICAFS)*, Siegen, Germany, 80–86.

Nasibov, E. & Ulutagay, G. (2007a). A new unsupervised approach for fuzzy clustering. *Fuzzy Sets and Systems*, 158, 2118–2133.

Nasibov, E. & Ulutagay, G. (2007b). On cluster analysis based on fuzzy relations between spatial data. *5th EUSFLAT Conference*, 2, Ostrava, Czech Republic, 59–62.

Nasibov, E. & Ulutagay, G. (2007c). Uzaysal veri analizinde bulanık mantık yaklaşımı. *Yöneylem Araştırması Endüstri Mühendisliği Kongresi (YA/EM'07)*, İzmir, Turkey, 1225–1230.

Nasibov, E. & Ulutagay, G. (2008a). A new clustering analysis for Bispectral Index Measurements. *2nd Int. Conf. On Control and Optimization with Industrial Applications*, Baku, Azerbaijan, 141.

Nasibov, E. & Ulutagay, G. (2008b). FN-DBSCAN: A novel density-based clustering method with fuzzy neighborhood relations. *8th Int. Conference on Application of Fuzzy Systems and Soft Computing (ICAFS'08)*, Helsinki, Finland, 101–110.

Nasibov, E. & Ulutagay, G. (2009). Robustness of density-based clustering methods with various neighborhood relations. *Fuzzy Sets and Systems*, doi:10.1016/j.fss.2009.06.012.

Nasibov, E., Özgören, M., Ulutagay, G., Öniz, A., & Kocaaslan, S. (2008). BIS monitoring analysis of sleep data via clustering. *Genç Bilim İnsanları ile Beyin Biyofiziği II. Çalıştayı*, İzmir, Turkey, 56.

Nasibov, E., Özgören, M., Ulutagay, G., Öniz, A., & Kocaaslan, S. (2009). *Uygulamalı Beyin Biyofiziği ve Multidisipliner Yaklaşım.* BIS verilerinin analizinde FCM ve FN-DBSCAN kümeleme yöntemlerinin karşılaştırılması, 233-250, ISBN 978-975-441-259-8.

Pedrycz, W. & Gomide, F. (1998). *An Introduction to Fuzzy Sets.* Massachusetts Institute.

Peirce, C. S. (1931). *Collected Papers of Charles Sanders Peirce.* Harvard University Press.

Rangayyan, R. M. (2002). *Biomedical Signal Analysis*. IEEE Press Series on Biomedical Engineering.

Rechtschaffen, A. & Kales, A. (1968). *A Manual of Standardized Terminology, Techniques, and Scoring System for Sleep Stages of Human Subjects*. Brain Information Service/Brain Research Institute, University of California, Los Angeles.

Russell, B. (1923). Vagueness. *Australian Journal of Philosophy*, 1.

Sander, J., Ester, M., Kriegel, H., & Xu, X. (1998). Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2, 169–194.

Shannon, C. E. (1948). The mathematical theory of communication. *The Bell System Technical Journal*, 27, 379–423, 623–656.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). WaveCluster: A multiresolution clustering approach for very large spatial databases. *Int. Conf. on Very Large Data Bases (VLDB98)*, New York, USA, 428–439.

Sneath, P. H. A. & Sokal, R. R. (1973). *Numerical Taxonomy*. Freeman.

Ulutagay, G. & Nasibov, E. (2008a). Density-based clustering methods with EEG data. *Genç Bilim İnsanları ile Beyin Biyofiziği II. Çalıştayı*, İzmir, Turkey, 55.

Ulutagay, G. & Nasibov, E. (2008b). A new clustering approach to EEG data for BIS measurements. *8th Int. Conference on Application of Fuzzy Systems and Soft Computing (ICAFS'08)*, Helsinki, Finland, 111–118.

Ulutagay, G. & Nasibov, E. (2009). Bulanık kümeleme yaklaşımı ile bis safha seviyelerinin belirlenmesi. *14. Biyomedikal Mühendisliği Ulusal Toplantısı (BİYOMUT'09)*, İzmir, Turkey, 246–249.

Velthuizen, R., Hall, L., Clarke, L., & Silbiger, M. (1997). An investigation of mountain method clustering for large data sets. *Pattern Recognition*, 30 (7), 1121–1135.

Wang, W., Yang, J., & Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining. *Int. Conf. on Very Large Data Bases (VLDB97)*, Athens, Greece, 186–195.

Ward, J. H. (1963). Hierachical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58, 236–244.

Xie, X. & Beni, G. (1991). A validity for fuzzy clustering. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 3, 841–846.

Yager, R. & Filev, D. (1994). Approximate clustering via the mountain method. *IEEE Transactions On Systems, Man and Cybernetics*, 24 (8), 1279–1284.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8, 338–353.

Zadeh, L. A. (1987). *Fuzzy Sets and Applications: Selected Papers*. Wiley-Interscience.

Zahid, N., Abouelala, O., Limouri, M., & Essaid, A. (2001). Fuzzy clustering based on k-nearest-neighbors rule. *Fuzzy Sets and Systems*, 120, 239–247.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD96)*, Montreal, Canada, 103–114.