Algorithms of Advanced Fuzzy Clustering: Development, Analysis, and Application to Rule-Based Modeling

by

Yinghua Shen

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software Engineering and Intelligent Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Fuzzy clustering is one of the most significant techniques used to explore the structure of data. With the development of information technology and new requirements arising from data analysis, new challenges have been raised for fuzzy clustering algorithms due to the emerging characteristics of the data, e.g., the data could be distributed, granular, big, or partially supervised by domain experts or data analysts. The effectiveness, efficiency, and even feasibility of the commonly encountered fuzzy clustering algorithms, e.g., Fuzzy C-Means (FCM), could no longer be guaranteed. This severe situation invokes the urgent needs for the more advanced clustering algorithms. Hence, in this dissertation, our main objective is to develop and analyze a series of fuzzy clustering algorithms to address the general issues mentioned above. Besides, since clustering plays a significant role in constructing the fuzzy rule-based model (FRBM), several of those proposed clustering algorithms are used to either expand the application scenarios of the FRBM or improve the performance of the FRBM. Identifying the major characteristics of data encountered nowadays, proposing the corresponding novel data structure exploration solutions, and applying these novel solutions to system identification, constitute the major originality of this dissertation.

The methods used to realize our main objective are briefly introduced as follows. To cluster distributed data, the horizontal collaborative fuzzy clustering (HCFC) algorithm is

refined. Specifically, a granular structure is formed as the global representative of all the distributed data. To cluster homogenous granular data, we propose a comprehensive framework to unify the processes of information granule formation, granular data clustering, and clustering results evaluation. To cluster heterogeneous granular data, we propose the approximation methods such that information granules could be transformed into the same form; afterwards, homogeneous granular clustering could be directly used. To cluster big data, we propose a hyperplane division-based method to get the subsets of the original data; then different clustering strategies are provided when different clustering requirements are sought (e.g., a large number of clusters is pursued). To make use of the knowledge (which is provided by domain experts or data analysts) about the data during the clustering process, we form two implementation methods of the knowledge tidbits. Furthermore, we specifically focus on applying two refined clustering algorithms to improving the FRBM. By using the HCFC algorithm, we make it possible to build the FRBM when input and output data are not allowed to be gathered together considering the data privacy. By using the supervision hints (knowledge tidbits) derived from the output space, we conduct a supervised clustering of the input space to improve the performance of FRBM in terms of the root-mean-square error (RMSE). Experiments on both synthetic and publicly available data are used to examine the effectiveness, efficiency, and feasibility of the proposed methods.

Preface

The research conducted in this thesis was performed by Yinghua Shen under the supervision of Professor Witold Pedrycz. Yinghua Shen was partially supported by the Alberta Innovates.

Chapter 3 of this thesis includes the material published as Y. Shen and W. Pedrycz, "Collaborative fuzzy clustering algorithm: some refinements," *Int. J. Approx. Reason.*, vol. 86, pp. 41–61, 2017. I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition.

Chapter 4 of this thesis includes the material published as Y. Shen, W. Pedrycz, and X. Wang, "Clustering homogeneous granular data: formation and evaluation," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1391–1402, 2019. I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. X. Wang was involved with manuscript composition.

Chapter 5 of this thesis includes the material accepted for publication as Y. Shen, W. Pedrycz, and X. Wang, "Approximation of fuzzy sets by interval type-2 trapezoidal fuzzy sets," *IEEE Trans. Cybern.*, pp. 1–13, 2019 (Early Access). I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. X. Wang was involved with manuscript composition.

Chapter 6 of this thesis includes the material submitted to *IEEE Trans. Fuzzy Syst.* as Y. Shen, W. Pedrycz, Y. Chen, X. Wang, and A. Gacek, "Hyperplane division in Fuzzy C-Means: clustering big data". I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and

was involved with the concept formation and manuscript composition. Y. Chen contributed to the formation of the idea. X. Wang and A. Gacek were involved with manuscript composition.

Chapter 7 of this thesis includes the material submitted to *IEEE Trans. Fuzzy Syst.* as Y. Shen, W. Pedrycz, X. Hu, X. Wang, and A. Gacek, "Identification of fuzzy rule-based models with collaborative fuzzy clustering". I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. X. Hu, X. Wang and A. Gacek were involved with manuscript composition.

Chapter 8 of this thesis includes the material submitted to *IEEE Trans. Fuzzy Syst.* as Y. Shen, W. Pedrycz, X. Jing, A. Gacek, and X. Wang, "Identification of fuzzy rule-based models with output space knowledge guidance". I was responsible for experiment development, data collection and analysis, and the manuscript composition. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition. X. Jing, A. Gacek and X. Wang were involved with manuscript composition.

Acknowledgements

It is my honor that I could be a PhD student of Professor Witold Pedrycz. I appreciate all the time and patience he spent to supervise my research. His immense knowledge, ideas, passion, and encouragement makes my PhD experience interesting and memorable. From him I learnt how to seriously conduct the academic research.

I would like to express my gratitude to all the professors who provided constructive comments and suggestions that improve the quality of the dissertation. They are Professor Emil Petriu, Professor Marek Reformat, Professor Petr Musilek, Professor Mojgan Daneshmand, Professor Hai Jiang, and Professor Aminah Robinson.

On top of research, I would like to give my thanks to Professor Hai Jiang (the course instructor of ECE 212) and to Wing Hoy (the lab technician). I gained a rich experience from them about how to interestingly deliver knowledge to undergraduate students.

I am grateful to all my fellow lab mates and friends encountered in University of Alberta. Discussing research problems and holding parties with them makes my life in Canada amusing and impressive.

I would like to give my special gratitude to my girlfriend Yuan Chen. How time flies, I could not believe that she has accompanied with me for seven years. I really appreciate her encouragement and love during my difficult time on the research, which makes me a better myself. I feel lucky and contended for sharing my happiness and sorrow with her.

Finally, my parents Yaming Shen and Yuelan Wang are deserved to be greatly appreciated. They raised me, educated me, and supported me. Their accompany, love, and tolerance has always been an important motive force for me to move forward.

Yinghua Shen

University of Alberta

September 2019

Table of Contents

List of Tables

List of Figures

List of Symbols

X	data set
$\boldsymbol{x}_k$	k-th data point in X
N	number of data points
n	number of features
$\boldsymbol{R}^n$	n dimensional space
U	partition matrix
u_{ik}	i-th row and k-th column in U
V	matrix of prototypes
$\boldsymbol{v}_i$	i-th prototype in V
σ_j	variance of the j-th feature
c	number of clusters
m	fuzzification coefficient
$X[ii]$	ii-th data site
$\alpha(ii, jj)$	collaboration strength between $X[ii]$ and $X[jj]$ from the perspective of $X[ii]$
cov	coverage
sp	specificity
G	granular data set
$\boldsymbol{g}_k$	k-th granular data point in G
$\tilde{U}$	granular partition matrix
$\tilde{u}_{ik}$	i-th row and k-th column in $\tilde{U}$
$\tilde{V}$	matrix of granular prototypes
$\tilde{\boldsymbol{v}}_i$	i-th granular prototype in $\tilde{V}$
$A(x)$	membership of type-1 fuzzy set
$\tilde{A}(x)$	membership of type-2 fuzzy set

List of Abbreviations

FCM	Fuzzy *C*-Means
FRBM	fuzzy rule-based model
CFC	collaborative fuzzy clustering
HCFC	horizontal collaborative fuzzy clustering
PJG	principle of justifiable granularity
RC	reconstruction criterion
TFS	trapezoidal fuzzy set
IT2 TFS	interval type-2 trapezoidal fuzzy set
FOU	footprint of uncertainty
PFCM	proximity-based Fuzzy *C*-Means
AFCM	augmented Fuzzy *C*-Means
RFCM	refined Fuzzy *C*-Means
LSE	least-square error
RMSE	root-mean-square error
PSO	particle swarm optimization
MISO	multiple-input single-output
MIMO	multiple-input multiple-output

Chapter 1 Introduction

Clustering is one of the most important techniques used to explore the structure of data. It intends to gather those data points close (in terms of distance, similarity, functionality, etc.) to each other into a group and distributes those far apart from each other into the different groups. Many different kinds of clustering concepts and algorithms have been proposed so far, which could be roughly classified into partition-based methods [1]–[3], graph-based methods [4], [5], hierarchy-based methods [6], [7], and density-based methods [8], [9]. Among these methods, the fuzzy partition-based methods, e.g., Fuzzy C-Means (FCM) [2], [3], which bring the concept of fuzzy set [10] into clustering, have seen a rapid development in both theory and real-world applications. By assigning the cluster membership degree, which is a value in interval [0, 1], to a certain data point, the structure of data could be described by some overlapped clusters which are more suitable to represent and handle the complex phenomena in real world.

1.1 Motivation

However, nowadays the real world poses more and more challenges for developing powerful clustering algorithms on the encountered data. Being different from the conventional scenario (where data is usually stored in one data site, with numeric values, small in sample size and low in feature dimensionality, or without the expert knowledge to describe its nature), many new characteristics of data have emerged which is attributed to either the development of the information technology or the emerging requirements of the data analysis. We list several most commonly encountered characteristics of data as follows. (a) Distributed. The interested data are distributed across different sites, and considering the constraints such as security, privacy, and network bandwidth, data sites are not allowed to communicate with each other by directly exchanging their original raw data.

(b) Granular. To model the complexity of real-world phenomena, granular data such as intervals, fuzzy numbers, linguistic terms, probability distributions, etc., could either be formed based on some numeric evidence or directly provided by human beings (e.g., people could directly circle an interval on a questionnaire to indicate his/her opinion about the reasonable price of a certain car make). Granular data delivers a high-level description or abstraction of the real-world phenomenon. (c) Big. The data we are interested could have either a very high feature dimensionality (e.g., pixels of an image) or a very large sample size (e.g., high-frequency data recorded by data sensors); or both of them. As a matter of fact, nowadays we live in a world where big data is ubiquitous. (d) Supervised. Normally, analysis of the data about a certain phenomenon is a standalone task, i.e., structure or information contained inside data is simply reflected by themselves. However, domain experts or data analysts may have their own knowledge about the data to be analyzed, e.g., during the clustering process domain experts may be very confident that several data points must be grouped together while some other points should never be grouped together. In other words, clustering of data could be partially supervised by domain experts or data analysts. Of course, these previously mentioned characteristics of data do not always stand alone, quite often they could overlap with each other. For example, it is not hard to find a data set that is both distributed across different data sites and also big in each data site. We may depict the relations of these characteristics in Figure 1.1.

These characteristics of data greatly challenge the conventional clustering algorithms including the fuzzy clustering algorithms. Although many methods have been proposed to cluster data with these characteristics, their shortcomings still exist (these methods would be reviewed and compared with the proposed methods in the corresponding chapters). Hence, it prompts us to construct more effective, robust, and efficient fuzzy clustering algorithms to tackle these new challenges.

Clustering alone is a major task of data analysis, however, its linkage to the fuzzy rule-based model (FRBM) [11], a system identification method, is obvious. The reason is that clustering serves as an efficient and effective way to partition the input space of the FRBM into subspaces inside which then local models are further constructed. Now that more advanced fuzzy clustering algorithms have been devised to face different characteristics of the interested data, a straightforward idea is that FRBMs could also benefit from them. In other words, we could eventually build more powerful FRBMs with the improved clustering methods. This forms the application side of the advanced fuzzy clustering algorithms proposed in this dissertation.

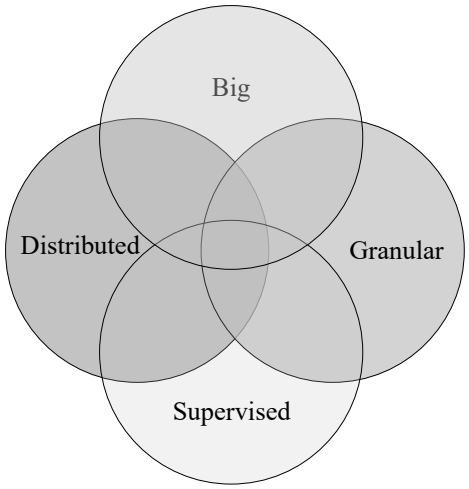


Figure 1.1. Characteristics of data of interest.

1.2 Objectives and Originality

To sum up, we intend to propose more advanced fuzzy clustering algorithms or refine some of the current fuzzy clustering algorithms to deal with the challenges derived from the emerging characteristics of data; and apply some of these clustering algorithms to building innovative FRBMs. A roadmap of the thesis is displayed in Fig. 1.2. The major objectives of this study are further listed as follows:

- Clustering distributed data. For the distributed data with the same instances but expressed in different feature spaces, we intend to refine the horizontal collaborative fuzzy clustering (HCFC) algorithm. Specifically, we will optimize the collaboration strength among the data sites; and build the granular partition matrix as the global data structure.
- Clustering homogenous granular data. For the homogenous granular data, we first justify how we could form the granular data from the numeric data. By stressing that different granular data may have different quality, we further propose a weighted granular clustering method. Finally, we propose the granular reconstruction criterion to make the evaluation of granular clustering methods possible.
- Clustering heterogenous granular data. We consider a more complicated case in the granular data clustering, i.e., different types of granules could be formed in different sources (e.g., data is both granular and distributed). We propose two approximation methods to transform heterogeneous granular data into homogenous ones. For a fuzzy set of arbitrary shape, it will either be transformed into a trapezoidal fuzzy set (TFS) or an interval type-2 trapezoidal fuzzy set (IT2 TFS). Clustering could then be applied to these homogenous granular data.
- Clustering big data. For the big data with a large sample size, we intend to propose an efficient hyperplane division method to split the original input space into many subspaces. Then, clustering is conducted in each subspace. Here, the divide-and-conquer strategy makes clustering the big data possible. Based on the clustering requirements, it is quite efficient to cluster the data into either a large number or a small number of clusters.
- Building FRBM with CFC algorithm. We consider a case where the input and output data could not be gathered together due to the privacy consideration but the FRBM (used to describe the relationship between input and output spaces) is still desired to be

constructed. By treating input and output spaces as two data sites, the HCFC algorithm is used to build this kind of FRBM.

- Building FRBM with knowledge guidance. The knowledge from the output space has been largely neglected when building the FRBM. With this knowledge tidbit, we could determine which data pair in the input space should not be put in the same group (in this way, clustering of the input space is partially supervised). Performance of the FRBM is expected to be improved by adopting this knowledge tidbit.

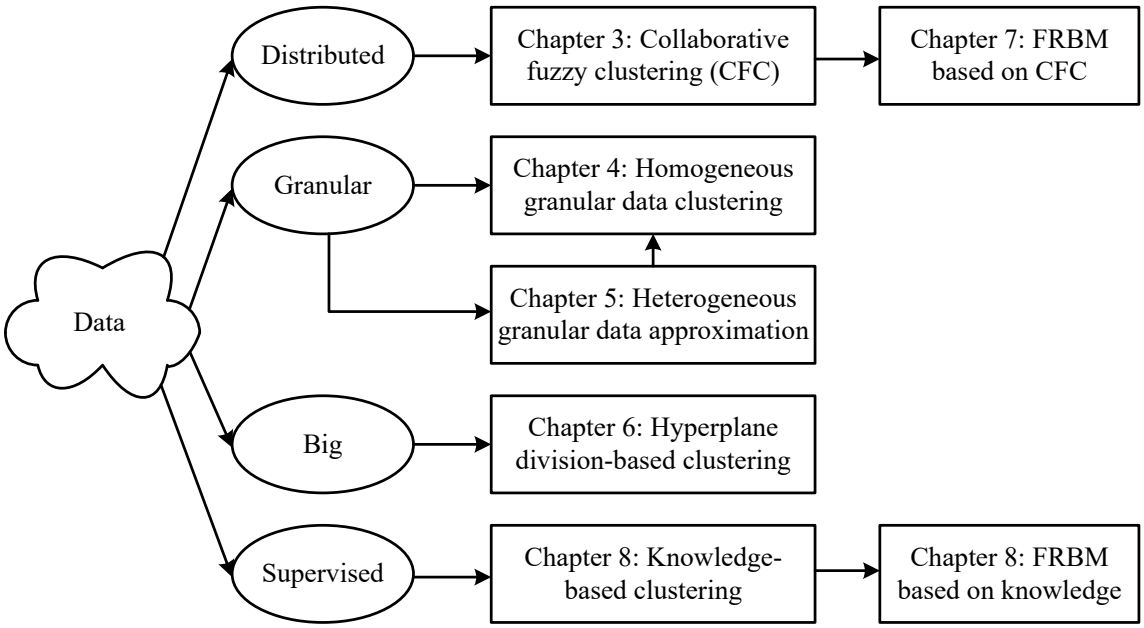


Figure 1.2. A general roadmap of the research.

The originality of this study can be briefly highlighted as follows:

- Improvement of the HCFC algorithm is shown.
- A comprehensive framework of granular clustering is proposed.
- New perspectives for fuzzy set approximation are illustrated.
- Systematic framework for big data clustering is shown.
- We expand the application scenario of FRBM.

- We highlight and implement a knowledge tidbit to improve the performance of FRBM.

1.3 Organization

By revolving around the major objectives, the dissertation is organized as follows:

In Chapter 2, we briefly review some methods that are fundamental to this research, including the FCM algorithm, horizontal collaborative fuzzy clustering (HCFC) algorithm, proximity-based FCM (PFCM) algorithm, Takagi-Sugeno (TS) FRBM, principle of justifiable granularity (PJG), type-2 fuzzy set, and particle swarm optimization (PSO).

In Chapter 3, we refine the current HCFC algorithm in terms of 1) analyzing the impact of the linkage strengths on the performance of the clustering algorithm and 2) forming the granular partition matrix as a global data structure for all the data sites.

In Chapter 4, the systematic framework of how the homogeneous granular data could be formed, clustered, and evaluated is provided. We would see that the PJG, weighted FCM, and the reconstruction criterion play important roles in the entire process.

Chapter 5 focuses on the task of heterogeneous granular data clustering. Intuitively, if the heterogeneous granular data are transformed into the homogenous ones, methods given in Chapter 4 could be used immediately. Hence, instead of proposing new clustering methods for heterogeneous granular data, Chapter 5 gives the methods to approximate any shape of fuzzy sets into TFS or IT2 TFS.

In Chapter 6, we propose the hyperplane division-based method to cluster the big data with a large number of samples. The strategies satisfying the different requirements (if a small or large number of clusters are pursued) of the clustering task are also provided.

Focusing on building the FRBM when the input and output data could not be owned by a user simultaneously, in Chapter 7 we treat input and output data as two different data sites and use the HCFC algorithm to make it possible to build FRBM under this situation.

In Chapter 8, we highlight that a knowledge tidbit, which describes the relationship between input and output data, could be very useful for the function approximation problem. We specifically apply this knowledge tidbit to the clustering process of the input space when forming the condition parts of the FRBM to improve the model performance.

Finally, in Chapter 9 we draw the conclusions from the current study and point out several interesting topics which deserve further studies.

Chapter 2 Preliminaries

In this chapter, we cover several methods or concepts which are fundamental to our research. Specifically, FCM as an important conventional fuzzy clustering algorithm (which is also the base of the proposed more advanced fuzzy clustering algorithms) is revisited in Section 2.1. The HCFC algorithm is introduced in Section 2.2 to facilitate the presentation in Chapter 3. A brief introduction of the proximity-based FCM (PFCM) is given in Section 2.3 which lays foundation for the discussion in Chapter 8. The commonly used Takagi-Sugeno FRBM is introduced in Section 2.4. Principle of justifiable granularity (PJG) regarded as a sound mechanism to form information granules for the given data evidence is introduced in Section 2.5. The concept of the type-2 fuzzy set is covered in Section 2.6. Finally, the PSO algorithm as an important population-based optimization method, which may be quite usefully when the objective function is non-differentiable (of course, more than this it could also be used to optimize the structure of FRBM) is given in Section 2.7.

2.1 Fuzzy C-Means

Suppose we have the data set as $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)^{\mathrm{T}}$, $\boldsymbol{x}_k$ is the k-th data point in the n dimensional space $\boldsymbol{R}^n$. The generic version of the FCM algorithm [3] minimizes the following objective function as

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^{m} \left\| \boldsymbol{x}_k - \boldsymbol{v}_i \right\|^2 \tag{2.1}$$

with the weighted Euclidean distance expressed as

$$\left\| \boldsymbol{x}_k - \boldsymbol{v}_i \right\|^2 = \sum_{j=1}^{n} \frac{(x_{kj} - v_{ij})^2}{\sigma_j^2} \tag{2.2}$$

where σ_j is a standard deviation of the j-th variable of the data and fuzzification coefficient m is usually greater than 1. Note that if not specified, the weighted Euclidean distance is used throughout this study. The data are partitioned into c clusters coming in the form of the partition matrix $U = [u_{ik}]_{c\times N}$, $i = 1, 2,\ldots, c$; $k = 1, 2,.., N$, and a collection of prototypes represented as $V = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_c)^{\mathrm{T}}$. The k-th data is described in terms of the k-th column membership grades in the partition matrix. By the alternating optimization (AO) algorithm [12], each element in the partition matrix is calculated as

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|\boldsymbol{x}_k - \boldsymbol{v}_i\|}{\|\boldsymbol{x}_k - \boldsymbol{v}_j\|}\right)^{2/(m-1)}} \tag{2.3}$$

and each entry in the prototype is obtained as follows,

$$v_{it} = \frac{\sum_{k=1}^{N} u_{ik}^{m} x_{kt}}{\sum_{k=1}^{N} u_{ik}^{m}} \tag{2.4}$$

where $t = 1, 2,\ldots, n$.

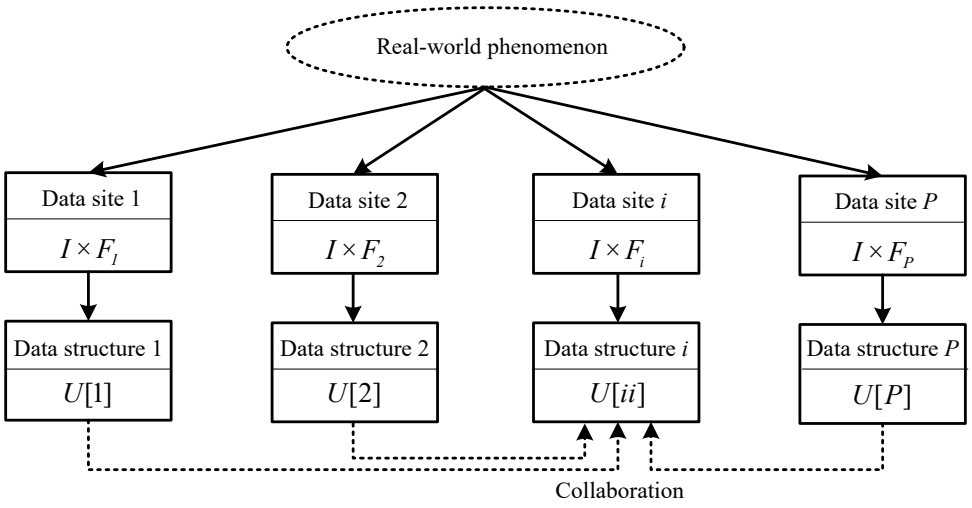


Figure 2.1. HCFC - an overview.

2.2 Horizontal Collaborative Fuzzy Clustering

HCFC algorithm [13] focuses on finding the data structure in the distributed data sites which have the same sample space (i.e., the samples in the same row of the different data sites correspond to the same entity, say a patient), but the different feature space (e.g., different medical indices). Its idea could be briefly visualized in Figure 2.1. Here P data sites participate in the collaboration, I and F_i stand for the sample and feature spaces associated with the corresponding data sites, respectively. More specifically, in the HCFC data structure of i-th data site is represented by the partition matrix $U[ii]$ constructed by running some clustering methods (e.g., FCM). As noted, through the collaboration data site i can modify its data structure by considering the structure information outside.

The formulation of the HCFC algorithm is given as follows. Suppose P data sites are denoted by $X[1]$, …, $X[ii]$, …, $X[P]$, all the data sites have the same instance spaces with N instances inside, but different feature spaces composed of $n[1]$,…, $n[ii]$,…, $n[P]$ features, respectively. Each instance in data site $X[ii]$ is represented by $\boldsymbol{x}_k[ii]$, $k = 1, 2,\ldots, N$, which is a data point in space $\boldsymbol{R}^{n[ii]}$. The feature spaces may overlap meaning that some common features may exist in different data sites. Since we are concerned with the collaboration among the data sites, $\alpha(ii, jj)$ is used to represent a collaboration strength between $X[ii]$ and $X[jj]$ from the perspective of $X[ii]$. As there is no collaboration between $X[ii]$ and itself, we set the value of $\alpha(ii,ii)$ to zero. Besides, we express the data structure by a partition matrix U or a prototype matrix V (each row stands for a cluster prototype), e.g., for data site $X[ii]$ we may express its structure by $U[ii]$ or $V[ii]$.

Given a specific data site $X[ii]$, we determine its structure (clusters) by not only considering the data available locally but also exploits the structure (conveyed by the partition matrices) from other data sites. This leads to the HCFC problem in which the local structure is optimized through the minimization of the following objective function

$$Q[ii] = \sum_{k=1}^{N}\sum_{i=1}^{c[ii]} u_{ik}^{m}[ii] d_{ik}^{2}[ii] + \sum_{jj=1}^{P} \alpha(ii, jj) \sum_{k=1}^{N}\sum_{i=1}^{c[ii]} (u_{ik}[ii] - u_{ik}[jj])^2 d_{ik}^{2}[ii] \tag{2.5}$$

where $c[ii]$ is the predetermined number of clusters for $X[ii]$. Here, $u_{ik}[ii]$ is the ik-th entry of the partition matrix $U[ii]$, and $d_{ik}^{2}[ii]$ is the squared Euclidean distance between the k-th data point $\boldsymbol{x}_k[ii]$ and i-th prototype $\boldsymbol{v}_i[ii]$, i.e., $d_{ik}^{2}[ii] = \left\| \boldsymbol{x}_k[ii] - \boldsymbol{v}_i[ii] \right\|^2 = \sum_{j=1}^{n[ii]} \left(\left(x_{kj}[ii] - v_{ij}[ii] \right) / \sigma_j[ii] \right)^2$, where $\sigma_j[ii]$ represents the standard deviation of the j-th attribute in $X[ii]$. This weighted distance is used to cope with situations where features exhibit different ranges. m is the fuzzification coefficient to control the shape of fuzzy clusters.

For each data site $X[ii]$, the optimization task is to minimize the objective function $Q[ii]$ under the assumption that $U[ii]$ forms a family of partition matrices, i.e., $U[ii] \in U = \left\{ u_{ik}[ii] \in [0,1] \middle| \sum_{i=1}^{c} u_{ik}[ii] = 1,\ 0 < \sum_{k=1}^{N} u_{ik}[ii] < N \right\}$. Here $U[ii]$ and $V[ii]$ are the collections of variables needed to be optimized, $U[jj]$ is the known partition matrix passed from the jj-th data site. The technique being used to solve this optimization task is similar to the one used in the "standard" FCM, which comes down to a Picard iterative process (i.e., the alternating optimization algorithm). Here we show the formulas to determine the partition matrix and prototypes for $X[ii]$ as

$$u_{sk}[ii] = \frac{1}{1 + \psi[ii]} \left(\varphi_{sk}[ii] + \frac{1}{\sum_{j=1}^{c[ii]} d_{sk}^{2}[ii] / d_{jk}^{2}[ii]} \right) \tag{2.6}$$

where $\varphi_{sk}[ii] = \sum_{jj=1}^{P} \alpha(ii, jj)\, u_{sk}[jj]$, $\psi[ii] = \sum_{jj=1}^{P} \alpha(ii, jj)$, s = 1, 2,…, $c[ii]$, k = 1, 2,…, N.

$$v_{st}[ii] = \frac{A_{st}[ii] + C_{st}[ii]}{B_s[ii] + D_s[ii]} \tag{2.7}$$

where $t = 1, 2, \ldots, n[ii]$, $A_{st}[ii] = \sum_{k=1}^{N} u_{sk}^2[ii] x_{kt}[ii]$, $B_s[ii] = \sum_{k=1}^{N} u_{sk}^2[ii]$,

$$C_{st}[ii] = \sum_{jj=1}^{P} \alpha(ii, jj) \sum_{k=1}^{N} \left(u_{sk}[ii] - u_{sk}[jj]\right)^2$$

$$D_s[ii] = \sum_{jj=1}^{P} \alpha(ii, jj) \sum_{k=1}^{N} \left(u_{sk}[ii] - u_{sk}[jj]\right)^2.$$

To measure the performance of HCFC, we take the structure revealed in $X[jj]$ (represented by $U[jj]$), $jj = 1, 2, \ldots, P$, and check how well it performs on $X[ii]$ by computing the following sum

$$W[ii \mid jj] = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^2[jj] \| \boldsymbol{x}_k[ii] - \boldsymbol{v}_i[ii \mid jj] \|^2 \tag{2.8}$$

where the prototype $\boldsymbol{v}[ii|jj]$ is induced (computed) on a basis of $U[jj]$ as follows

$$\boldsymbol{v}_i[ii \mid jj] = \frac{\sum_{k=1}^{N} u_{ik}^2[jj] \boldsymbol{x}_k[ii]}{\sum_{k=1}^{N} u_{ik}^2[jj]} \tag{2.9}$$

Then the global index for data site $X[ii]$ becomes

$$W[ii] = \sum_{jj=1, jj \neq ii}^{P} W[ii \mid jj] \tag{2.10}$$

Finally, the global performance index for all data sites reads as

$$W = \sum_{ii=1}^{P} W[ii] \tag{2.11}$$

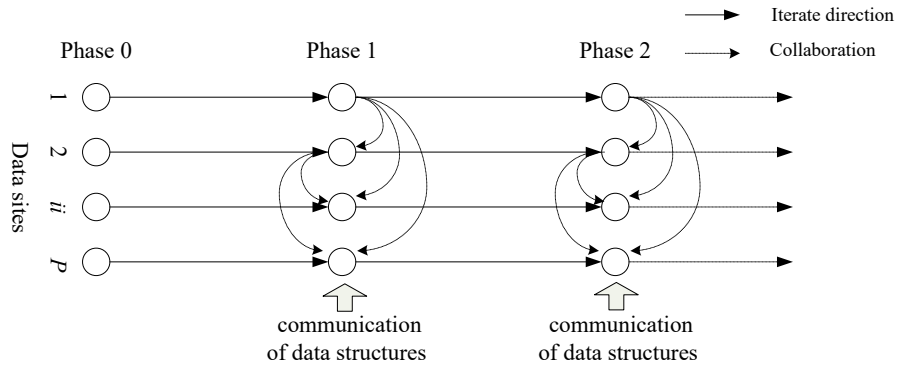


Figure 2.2. Graphic abstract of the HCFC algorithm.

The processing flow of the HCFC algorithm is further summarized as follows. At the initial phase (phase 0), no collaboration takes place, i.e., data sites form their own data structures based on the locally available data. Once this phase has been completed, in following phases, data sites start to exchange their local findings and the optimization is carried out so that both local data structure and those structures from other sites are taken into consideration. See Figure 2.2 for a detailed graphic abstract of this process.

2.3 Proximity-Based Fuzzy C-Means

As we mentioned, the domain experts or data analysts may have some knowledge about the closeness of two data points. For example, they are certain that points $\boldsymbol{x}_{k1}$ and $\boldsymbol{x}_{k2}$ are quite close to each other, hence a proximity value $P[k_1, k_2] = 0.9$ could be used to describe their closeness. However, from the data site, the closeness of two data points could be directly derived from the partition matrix U, which is derived after using certain clustering algorithms (say, FCM), that is,

$$\tilde{P}(k_1, k_2) = \sum_{i=1}^{c} (u_{ik_1} \wedge u_{ik_2}) \tag{2.12}$$

where u_{ik_1} and u_{ik_2} are, respectively the k_1th and k_2th elements in the ith row in partition matrix U, c is the number of clusters. Symbol $\wedge$ stands for the minimum operation, i.e., $u_{ik_1} \wedge u_{ik_2} = \min(u_{ik_1}, u_{ik_2})$. One may envision that if the data $\boldsymbol{x}_{k1}$ and $\boldsymbol{x}_{k2}$ have the similar membership values (to any of the prototypes), their proximity $\tilde{P}(k_1, k_2)$ is then close to one; on the contrary, it is close to zero. Hence, the defined proximity measure reflects the closeness between data points. Note that the proximity between two values satisfies the conditions that (i) symmetry, i.e., $\tilde{P}(k_1, k_2) = \tilde{P}(k_2, k_1)$; and (ii) reflexivity, i.e., $\tilde{P}(k_1, k_1) = 1$. These generic requirements of proximity also makes it practically relevant to quantify the relationship between two points [14].

The crux of the proximity-based FCM (PFCM) [14] is that the clustering process of the data could be guided or supervised by the knowledge by means of the proximity constraints among the data pairs. More specifically, we would minimize the difference between the proximity values of a data pair derived from the FCM algorithm and those values of the corresponding data pair provided by experts. The objective function is thus formulated as follows:

$$\min \quad J = \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} \left(\tilde{P}(k_1, k_2) - P(k_1, k_2) \right)^2 B(k_1, k_2) \tag{2.13}$$

where N is the number of data, $B(k_1, k_2)$ is a binary-valued entry, equaling either 0 or 1, to illustrate if the proximity guidance from the expert is provided (1 means that the guidance is provided).

Since index J is essentially a function of the membership degree $u_{st} \in U$, $s = 1, 2, \ldots, c$, $t = 1, 2, \ldots, N$, we determine the derivative of J with respect to u_{st}, that is

$$\frac{\partial J}{\partial u_{st}} = \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} \frac{\partial}{\partial u_{st}} \left[\left(\tilde{P}(k_1, k_2) - P(k_1, k_2) \right)^2 \right] B(k_1, k_2)$$

$$= 2 \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} [\left(\tilde{P}(k_1, k_2) - P(k_1, k_2) \right) B(k_1, k_2) \frac{\partial}{\partial u_{st}} \sum_{i=1}^{c} (u_{ik_1} \wedge u_{ik_2})] \tag{2.14}$$

where the inner derivative could be further determined as

$$\frac{\partial}{\partial u_{st}} \sum_{i=1}^{c} (u_{ik_1} \wedge u_{ik_2}) = \begin{cases} 1 & \text{if } t = k_1 \text{ and } u_{st} \le u_{ik_2}, \\ 1 & \text{if } t = k_2 \text{ and } u_{st} \le u_{ik_1}, \\ 0 & \text{otherwise}. \end{cases} \tag{2.15}$$

To keep index J decreasing, we update partition matrix U along the negative direction of the gradient in (2.14). That is, we have

$$u_{st}(iter + 1) = \left[u_{st}(iter) - \alpha \frac{\partial J}{\partial u_{st}(iter)} \right]_* \tag{2.16}$$

where α is the step size used to control changes of membership grades, $[\cdot]_*$ is the truncation operator to make sure the obtained membership value is positioned within the unit interval, and *iter* stands for the index of successive iterations. We summarize the PFCM algorithm in Figure 2.3.

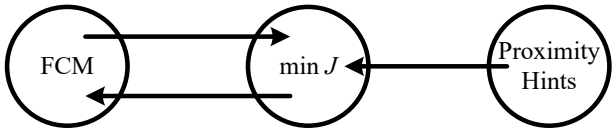


Figure 2.3. PFCM: a general flow of the optimization activities.

2.4 Takagi-Sugeno FRBM

Rule-based models [15] are very powerful models used to extract the knowledge or relationship contained in the data. It is normally represented in the form of

If *antecedent* then *consequent* (2.17)

When we bring the notion of fuzzy set into the rule, we come with the fuzzy rule-based model (FRBM) [11]. Although many different kinds of FRBMs have been proposed so far, it is the Mamdani [16] and the Takagi-Sugeno (TS) [17] ones gain a wide attention. When both *antecedent* and *consequent* are assigned with the fuzzy sets (or fuzzy relation), we have the Mamdani FRBM; while if only *antecedent* is fuzzified but leaves the *consequent* as a certain functional form, we have the TS FRBM. In this study, our focus is the latter one.

2.4.1 Architecture of the TS FRBM

The TS model is composed of a series of rules in the form

Rule i: If $\boldsymbol{x}$ is $A_i(\boldsymbol{x})$ then $y = f_i(\boldsymbol{x})$, $i = 1, 2, \ldots, c$. (2.18)

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^{\mathrm{T}}$ is a n-dimensional input variable vector whose values are located in space $\boldsymbol{R}^n$, here $x_1, x_2, \ldots, x_n$ are n input variables. In the condition part of the rule, $A_i(\boldsymbol{x})$ is

a membership function describing the membership degree of $\boldsymbol{x}$ belonging to the i-th rule (or to which degree *Rule i* becomes activated). In the conclusion part, function $y = f_i(\boldsymbol{x})$, used to describe the local behavior of the system, is the output of *Rule i* in the space $\boldsymbol{R}$ and is usually a polynomial function. The output of the model is the weighted aggregation of the c outputs of the rules described as

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}) f_i(\boldsymbol{x}) \tag{2.19}$$

Depending on the detailed form of the function $f_i(\boldsymbol{x})$, we could have either the zero-order TS model when the conclusion part is a constant function $f_i(\boldsymbol{x}) = w_i$ or the first-order TS model when $f_i(\boldsymbol{x}) = a_{i0} + a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n$. Due to the efficiency, interpretability, and sound performance, the zero-order TS model has been widely studied and used in many applications. In this dissertation, only the zero-order TS model is used to examine all the proposed methods.

2.4.2 Development of FRBM

Let us assume that the provided N input-output data pairs $(\boldsymbol{x}_k, y_k)$, $k = 1, 2, \ldots, N$ are organized as $(X, \boldsymbol{y})$, where $X = [\boldsymbol{x}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}}, ..., \boldsymbol{x}_N^{\mathrm{T}}]^{\mathrm{T}}$ and $\boldsymbol{y} = [y_1, y_2, ..., y_N]^{\mathrm{T}}$. The development of the FRBM focuses on estimating the parameters for the fixed architecture. For the condition part, the membership degree $A_i(\boldsymbol{x}_k)$ of a data point $\boldsymbol{x}_k$ to a cluster $\boldsymbol{v}_i$ could be obtained through partitioning the input space with standard FCM algorithm, or with the proposed more advanced clustering methods in later chapters. Now estimation of parameters lies in obtaining values of the coefficients of $f_i(\boldsymbol{x})$. We introduce two commonly used design strategies presented in the literature: (a) least-square error (LSE)-based strategy; and (b) cluster-centric-based strategy.

2.4.2.1 LSE-based Strategy

LSE-based strategy [18], [19] divides the input space without using any information from the output space. In this strategy, for the condition part, a fuzzy clustering algorithm, the FCM in particular, is applied to partition the input data X into clusters represented as $\{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_c\}$. $A_i(\boldsymbol{x})$ standing in (2.18) is determined as

$$A_i(\boldsymbol{x}) = 1 / \sum_{s=1}^{c} \left(\frac{\|\boldsymbol{v}_i - \boldsymbol{x}\|}{\|\boldsymbol{v}_s - \boldsymbol{x}\|} \right)^{2/(m-1)} \tag{2.20}$$

where m is the fuzzification coefficient with a value greater than 1, and $\|\boldsymbol{x} - \boldsymbol{v}_s\|^2$ is the squared weighted Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{v}_s$.

The output of the FRBM is obtained by aggregating the weighted c outputs coming from all the rules, which is denoted by

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}) w_i \tag{2.21}$$

where $A_i(\boldsymbol{x})$ is used as the weight of *Rule i*. Note that with the constraint of the FCM algorithm, $\sum_{i=1}^{c} A_i(\boldsymbol{x}) = 1$ is always satisfied. Based on (2.21), N estimated outputs for data X are organized as $\hat{\boldsymbol{y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_N]^{\mathrm{T}}$. With the least square method, we optimize w_i by minimizing the objective function Q expressed as

$$Q = (\hat{\boldsymbol{y}} - \boldsymbol{y})^{\mathrm{T}} (\hat{\boldsymbol{y}} - \boldsymbol{y}) = \sum_{k=1}^{N} (\hat{y}_k - y_k)^2 \tag{2.22}$$

Let us introduce the notation $\boldsymbol{w} = [w_1, w_2, ..., w_c]^{\mathrm{T}}$ and $P = [p_{ki}]_{N \times c}$, where $p_{ki} = A_i(\boldsymbol{x}_k)$. Then we have

$$\hat{\boldsymbol{y}} = P\boldsymbol{w} \tag{2.23}$$

Hence, the optimal value of $\boldsymbol{w}$ to minimize Q is obtained directly as

$$\boldsymbol{w}_{\mathrm{opt}} = (P^T P)^{-1} P^T \boldsymbol{y} \tag{2.24}$$

2.4.2.2 Cluster-centric-based Strategy

This kind of strategy [20] combines the input and output spaces to form a joint input-output (n+1)-dimensional space $\boldsymbol{R}^{n+1}$. The new formed data set is represented as $Z = [X, \boldsymbol{y}]$, which is a N by n+1 matrix concatenating X and $\boldsymbol{y}$. The augmented FCM (AFCM, for brief) algorithm is then applied to Z by incorporating the weight coefficient αn ($\geq$0) for the output data. Obviously, when α=1/n AFCM degenerates to the conventional FCM. The objective function for the AFCM algorithm is formulated as

$$J = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m}\left(\left\|\boldsymbol{v}_i - \boldsymbol{x}_k\right\|^2 + \alpha n\left\|w_i - y_k\right\|^2\right) \tag{2.25}$$

The AFCM algorithm is implemented by iteratively updating the following formulas (obtained through zeroing the derivatives of J with respect to $\boldsymbol{v}_i$, w_i, and u_{ik}).

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{N} u_{ik}^{m}\boldsymbol{x}_k}{\sum_{k=1}^{N} u_{ik}^{m}} \tag{2.26}$$

$$w_i = \frac{\sum_{k=1}^{N} u_{ik}^{m} y_k}{\sum_{k=1}^{N} u_{ik}^{m}} \tag{2.27}$$

$$u_{ik} = 1/\sum_{j=1}^{c}\left(\frac{\left\|\boldsymbol{v}_i - \boldsymbol{x}_k\right\|^2 + \alpha n\left\|w_i - y_k\right\|^2}{\left\|\boldsymbol{v}_j - \boldsymbol{x}_k\right\|^2 + \alpha n\left\|w_j - y_k\right\|^2}\right)^{1/(m-1)} \tag{2.28}$$

The finally formed c clusters are represented as $[\boldsymbol{v}_i^{\mathrm{T}}, w_i]^{\mathrm{T}}$, i = 1, 2,…, c. Hence, all the needed parameters for FRBM in (2.18) are obtained. With prototypes $\boldsymbol{v}_i$, the $A_i(\boldsymbol{x})$ in the condition part in (2.18) is obtained directly from (2.20). By representing $A_i(\boldsymbol{x}_k)$ as $\dot{u}_{ik}$ in this part, the partition matrix for N input data is obtained as $\dot{U} = [\dot{u}_{ik}]_{c\times N}$.

The output of the model is further obtained by minimizing the following objective function denoted as

$$F = \sum_{i=1}^{c}\sum_{k=1}^{N} \dot{u}_{ik}^{m}\left\|w_i - \hat{y}_k\right\|^2 \tag{2.29}$$

Here, we require that the estimated output $\hat{y}_k$ should be put in a position such that its summed weighted distance to the c constants w_i is minimized. By zeroing the gradient of F with respect to $\hat{y}_k$, we have

$$\hat{y}_k = \frac{\sum_{i=1}^{c} \dot{u}_{ik}^{m} w_i}{\sum_{i=1}^{c} \dot{u}_{ik}^{m}} \tag{2.30}$$

The performance of FRBM is measured by the widely acknowledged root-mean-square error (RMSE) index, which is represented as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(\hat{y}_k - y_k)^2} \tag{2.31}$$

2.5 Principle of Justifiable Granularity

The principle of justifiable granularity (PJG) proposed in [21] has been widely used as a vehicle to form the granules in the realm of Granular Computing. With which we intend to find a justifiable information granule (e.g., interval) such that as much as evidence (i.e., the data) could be included in this granule but also not to make this granule too general leading to a poor semantic explanation. To realize this, two functions (criteria) known as *coverage* and *specificity* are used to describe the two major considerations mentioned above, then they are maximized simultaneously. Taking the interval as the example of the information granule, normally *coverage* is a monotonically increasing function of the number of data points contained in the interval (cardinality of the interval), and *specificity* could be treated as the decreasing function of the interval length. It is obvious that these two objectives are in conflicts because the increase of the *coverage* would lead to the decrease of *specificity*, hence leading to a multi-objective optimization problem.

Pedrycz and Homenda [21] solved this problem by transferring the original problem to a single objective optimization by taking the product of these two functions. Moreover, they adopted the divide-and-conquer strategy to divide the interval into two partitions and

then apply the optimization technique to each part separately. For illustration, suppose we have a series of one-dimensional data denoted by X, the optimization problem could be divided into two sub-optimization tasks as follows

$$\begin{aligned} &\max_{b} \quad V^{+} = f_1\left(card\left\{x_k \,|\, median < x_k \le b\right\}\right) * f_2\left(\left|median - b\right|\right) \\ &\text{s.t.} \quad b \in \left[median, x_{\max}\right] \end{aligned} \tag{2.32}$$

$$\begin{aligned} &\max_{a} \quad V^{-} = f_1\left(card\left\{x_k \,|\, a \le x_k < median\right\}\right) * f_2\left(\left|median - a\right|\right) \\ &\text{s.t.} \quad a \in \left[x_{\min}, median\right] \end{aligned} \tag{2.33}$$

where functions f_1 (as an increasing function) and f_2 (as a decreasing function) correspond to *coverage* and *specificity* respectively, V^{+} and V^{-} are values of the objective functions, a and b denote the left and right boundaries of the interval needed to be optimized, *median* stands for the median of the data series, $x_{\min}$ and $x_{\max}$ respectively denote the minimum and maximum values of the data series. For simplicity we consider the two functions in the form $f_1(u) = u$ and $f_2(u) = 1 - u$.

2.6 Type-2 Fuzzy Set

A type-2 fuzzy set [22] serves as a viable model to capture the uncertainty of type-1 fuzzy set. The crux of any type-2 fuzzy set is that the membership degree of an element x in the universe of discourse X is not a single numeric value but rather an interval [all such intervals constitute a so-called footprint of uncertainty (FOU)]. Interval type-2 fuzzy sets (IT2 TFSs) [23], as a special case of type-2 fuzzy sets, have attracted attention due to their reduced computational cost. Let us denote the IT2 TFS by $\tilde{A}(x)$, whose FOU is bounded by TFSs $\tilde{A}^{+}(x)$ and $\tilde{A}^{-}(x)$ which are defined by (2.34) and (2.35), respectively. An illustration of the IT2 TFS is shown in Figure 2.4. As it has been noted, an IT2 TFS can be uniquely specified by the vector of parameters $\tilde{t} = (\tilde{t}_1, \tilde{t}_2, \tilde{t}_3, \tilde{t}_4, \tilde{t}_5, \tilde{t}_6, \tilde{t}_7, \tilde{t}_8, h)^{\mathrm{T}}$.

$$A^{-}(x)=\begin{cases} g_1^{-}(x)=h(x-\tilde{t}_1)/(\tilde{t}_2-\tilde{t}_1),\tilde{t}_1\le x<\tilde{t}_2 \\ h,\tilde{t}_2\le x<\tilde{t}_3 \\ g_2^{-}(x)=h(\tilde{t}_4-x)/(\tilde{t}_4-\tilde{t}_3),\tilde{t}_3\le x<\tilde{t}_4 \\ 0,\text{otherwise} \end{cases} \tag{2.34}$$

$$A^{+}(x)=\begin{cases} g_1^{+}(x)=(x-\tilde{t}_5)/(\tilde{t}_6-\tilde{t}_5),\tilde{t}_5\le x<\tilde{t}_6 \\ 1,\tilde{t}_6\le x<\tilde{t}_7 \\ g_2^{+}(x)=(\tilde{t}_8-x)/(\tilde{t}_8-\tilde{t}_7),\tilde{t}_7\le x<\tilde{t}_8 \\ 0,\text{otherwise} \end{cases} \tag{2.35}$$

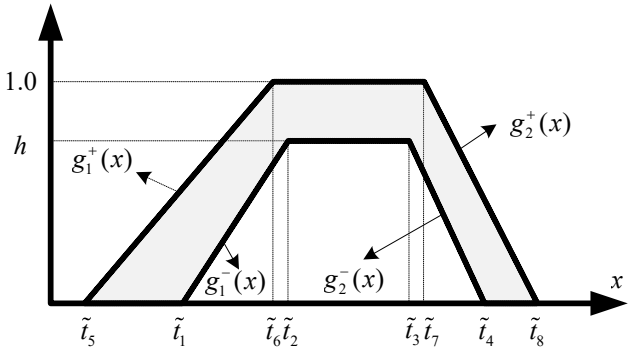


Figure 2.4. Illustration of IT2 TFS.

2.7 Particle Swarm Optimization

Population-based optimization methods are usually used when the objective function to be optimized is complex such as non-differentiable, non-continues, etc. We introduce the Particle Swarm Optimization (PSO) [24] method as one of the most commonly used population-based optimization methods in this section.

PSO has been widely used to solve global optimization problems due to advantages such as simplicity of implementation, few parameters, and high convergence rate. The main idea of PSO is sketched as follows. Suppose we have a swarm of particles of size N moving around in a search space. Each particle has its own position and velocity. A fitness function is used to describe the goodness of the position of the particle. For a given particle, it has the knowledge of the best position it has visited so far (*pbest*) and the best position within

the entire population (*gbest*). The velocity of each particle consists of three major parts: the inertial component, velocity from the current location to *pbest*, and that from the current location to *gbest*. Formulas used to update the location and velocity of the *j*th ($j = 1, 2,\ldots, N$) particle are expressed as

$$\boldsymbol{e}_j^{(k+1)} = \boldsymbol{e}_j^{(k)} + \boldsymbol{\theta}_j^{(k)} \tag{2.36}$$

$$\boldsymbol{\theta}_j^{(k+1)} = w\boldsymbol{\theta}_j^{(k)} + c_1\boldsymbol{r}_j^{(k)}(\boldsymbol{pbest}_j^{(k)} - \boldsymbol{e}_j^{(k)}) + c_2\boldsymbol{s}_j^{(k)}(\boldsymbol{gbest}^{(k)} - \boldsymbol{e}_j^{(k)}) \tag{2.37}$$

where $w \in [0,1]$ is the inertia constant, c_1 and c_2 represent the cognitive and social constants, respectively (both usually chosen to be 2), $\boldsymbol{r}_j$ and $\boldsymbol{s}_j$ are the random vectors with their components uniformly distributed in the interval [0, 1]. In this dissertation, the PSO is only used in Chapter 5 to find the optimal IT2 TFS which could be represented as 9 parameters; hence, as could be envisioned, we eventually have a 9-dimentinal search space $\boldsymbol{R}^9$.

2.8 Summary

In this chapter, we reviewed several fundamental concepts and methods which are quite significant to the contents introduced in later chapters. The FCM algorithm serves as the basic method for all the advanced clustering methods in this dissertation. The TS FRBM would be carefully studied in Chapter 7 and Chapter 8. Based on the HCFC, some related refinements would be given in Chapter 3. The PFCM lays the foundation for the knowledge-guided FRBM introduced in Chapter 8. The PJG delivers a mechanism to get more abstract structures (i.e., information granules) from the numeric data, which would be used in Chapters 3, 4, and 5. The PSO as a sound population-based optimization method would be used to find approximated IT2 TFS in Chapter 5.

Chapter 3 Horizontal Collaborative Fuzzy Clustering Method for Distributed Data[a]

Horizontal collaborative fuzzy clustering (HCFC) is a method to explore the data structure (clusters represented by the partition matrix or prototypes) of a given data site (e.g., patient database of a specific hospital) not only by using the locally available data but also by taking advantage of structural information from other data sites. Since each data site delivers a certain perspective at the overall system or phenomenon, HCFC also forms a conceptual and algorithmic setting to facilitate a construction of the holistic and unified view of the system under consideration.

The HCFC concept has attracted an increasing attention owing to several compelling and practically motivated reasons. First, quite often data tend to be distributed across different sites. Second, due to considerations of security, privacy, and network bandwidth, data sites are not allowed to directly communicate by exchanging original data. Hence, simply gathering all data and forming a single data repository is not feasible. The concept of HCFC is compatible with the aforementioned two real-world requirements. In fact, we could note that the techniques based on HCFC concept actually belong to the intersection of distributed data mining and abstraction-based data mining, which are two major data mining techniques summarized in [25] to harvest knowledge from large repositories of data.

It is obvious that the HCFC method is quite suitable for discovering the structures of the individual data sites or the global structure of all the data sites when the original raw data are not allowed to be exchanged with each other. In this chapter, to make the HCFC method for the distributed data more sound, we address the following three open questions:

[a] A version of this chapter has been published as [108].

(a) assessing the necessity of reordering partition matrices prior to invoking the collaboration process; (b) analyzing the impact of linkage strengths on the performance of the clustering algorithm; and (c) forming a representative global data structure with the use of the concept of information granules leading to the so-called granular partition matrix. The refinements of the HCFC model are illustrated in Figure 3.1.

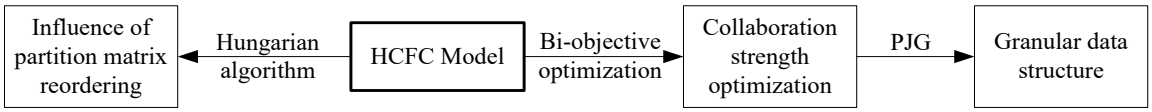


Figure 3.1. Refinements of the HCFC model.

3.1 Development of the HCFC: A Brief Review

Since the emergence of the concept of HCFC, a number of related ideas and algorithms have been proposed to realize the collaboration mechanism among the data sites. However, the main idea to solve this problem is nearly the same which finally comes down to a mathematical programming problem. According to the form of the objective function, these algorithms could be further divided into two categories: (a) global objective-based HCFC [26]–[31], and (b) local objective-based HCFC [32], [33], [42]–[45], [34]–[41]. The main difference between these two major categories is that, in the objective function, whether the structural information (represented by the partition matrix, prototypes, or proximity matrix) revealed in all data sites are treated simultaneously as the unknown variables or simply the structure information in a specific data site is required to be determined while leaving those from other data sites being known. The main advantage of the first category is that, the consistency of the identification of the clusters would be guaranteed because the clusters across the different data sites belonging to one group would be identified by the same index [30]. But the shortcoming is also straightforward, as the structure of a specific data site relies directly on the raw data coming from other data sites, which is

contrary to the original HCFC concept that is motivated by considering that in reality accessing all the data resource may be forbidden due to the confidentiality, security, or bandwidth limitations. Considering these aspects, in this chapter we are more interested in the HCFC algorithms located in the second category.

The local objective-based HCFC algorithms have experienced some development. Regarding to the application aspect, Loia et al. [34] applied the proximity based HCFC algorithm to the area of Intelligent Web, and used the method to focus on the reconciliation between two separated facets of web information and obtained a combination of results leading to a comprehensive data organization. Prasad et al. [35], Chou et al. [36], and Lin et al. [37], used the HCFC based methods to generate fuzzy rules for Mamdani and TSK fuzzy inference systems, by incorporating the mechanism of HCFC they tried to make the rule based system obtain the ability to solve the big data issue. Zhou et al. [29] utilized a simplified version of HCFC algorithm to the distributed network environments (whose topology is represented by a graph), where the collaboration process only happens between a vertex (stands for a specific data site) and its neighbor vertexes.

For the theoretical aspect, since the original concept of HCFC only contains a single collaboration phase, Pedrycz and Rai [33] further refined the concept by making the collaboration an iterative process where a specific data site would periodically utilize other data sites' structure information resulting from the collaboration process. This collaboration iterative process would not terminate until all data structure information remain steady. Falcon et al. [42], [43] specifically focused on the determination of the collaboration strength among the data sites, they proposed the rough set and PSO combined strategy to optimize these strength. Most of the HCFC algorithms mentioned above are constructed based on the FCM clustering algorithm, however Ghassany et al. [44], Rastin et al. [45], and Sublime et al. [46] realized the mechanism of HCFC through Kohonen's SOM, but the structure of objective function is similar to those discussed in [33].

3.2 Reordering Partition Matrix

In this section we propose a method to assess the necessity of partition matrix reordering in HCFC algorithm. The core idea is to utilize an accurate and efficient recording algorithm to make all the partition matrices consistent, whose influence on collaborative clustering would be compared to the original HCFC algorithm.

The motivation to reorder the partition matrix before the collaboration is due to the second term of the objective function shown in (2.5), where the structure difference between two data sites is represented by the subtraction of their corresponding partition matrices. The main concern here is that for a specific data site, it would adjust its data structure by considering all the structures outside, however even for the same row indexes in these partition matrices they may not all refer to the same cluster. This is not hard to understand. Suppose we have 5 data points (instances) which could be structured into 2 clusters. If we run FCM algorithm two times on this data set, it is likely to obtain a result (two partition matrices U_1 and U_2) as shown in Table 3.1. It is obvious the same cluster names refer to the different clusters, i.e., Cluster 1 reflected by U_1 is identical to Cluster 2 from U_2. Note that this phenomenon happens when running a certain clustering algorithm on the same data site two times. Now if we consider two or more data sites, the situation of inconsistency of the cluster label could become worse. Thus, for a specific data site, it may be confusing to reconcile its data structure with some misleading structures. Hence, one may envision that the collaboration could be more meaningful if all the structures are reordered so that the rows in the partition matrices are made consistent.

Based on Hungarian algorithm [47], we revise the HCFC algorithm, through which we guarantee that all the partition matrices are reordered according to the benchmark partition matrix such that the row consistency across the partition matrices is retained. Note that we are not saying the modified algorithm must outweigh the original one in [33], here we

merely want to examine the necessity of reordering, however, to make the results convincing a more robust reordering algorithm (Hungarian algorithm) is adopted.

Table 3.1. Clustering results of FCM.

		Inst. 1	Inst. 2	Inst. 3	Inst. 4	Inst. 5
U_1	Clus. 1	0.03	0.01	0.01	0.06	1.00
	Clus. 2	0.97	0.99	0.99	0.94	0.00
U_2	Clus. 1	0.97	0.99	0.99	0.94	0.00
	Clus. 2	0.03	0.01	0.01	0.06	1.00

3.3 Optimization of the Collaboration Strength

One of the most important research topics in HCFC is to establish the optimized collaboration strength among the different data sites as different strength levels may lead to distinct collaborative effect (and ensuring results). Here the collaborative effect during the collaboration process could be explained from two perspectives: (a) the new data structure of each data site tends to be away from the original one obtained without any collaboration; and (b) the new structures become more similar to each other. The collaborative effect could be reflected by indices constructed either from the data structure information (e.g., prototypes and partition matrices [13], [42], [43], [45]) or from the data structure performance (e.g., the objective function value when applying the data structure to a certain data site [33], [41]).

The main idea is to determine the collaboration strength such that the collaborative effect could be optimized. Pedrycz [13] treated two collaborative effect indices (constructed from the partition matrices) as the functions of the collaborative strength and studied its impact on the collaborative process. Pedrycz and Rai [33] further assumed that all data sites have the same collaboration strength and treated this strength as a variable of a function which was constructed based on the data structure performance. Falcon et al. [43] formulated the collaborative effect indices used in [13] and utilized the PSO algorithm

to optimize one of these indices to find the optimal strength between each pair of the data sites. Falcon et al. [42] further improved their optimization strategy by simultaneously taking into consideration the two indices stemming from [13] to make the final optimal strength more balanced.

These researches offer useful insights on how to optimize the collaborative strength, however, among which two main shortcomings exist: (a) Most of them do not offer a specific range limitation for the strength, or the only requirement is that the strength is greater than or equal to zero. Coletta et al. [41] limited this range to interval [0, 1] but without giving any explanation. We believe that collaborative strength should be limited to interval [0, 1] due to the intrinsic construct of the CFC model: we know that equation (2.5) is composed of two terms where the first one stands for the utilization of the local information and second term is that of the outside information, since the data site's own information has been weighted as one intuitively it is not reasonable to assign higher weights to other data sources. (b) Most of research believes that the pursuit of the collaboration is to find each data site's structure such that the collaboration effect is maximized, that is either to make the similarity between the structures maximized (let us call it *similarity*) or to maximize the distance between the reconciled structure and its original structure (let us call it *distance*). As a matter of fact, these two goals could be unified as one because large *similarity* would lead to large *distance*, we could envision that when the final reconciled structures get close to each other, they are moving far away from their original structures. However, during the strength optimization process, it becomes of interest that does extremely resembling the reconciled structures the only objective we expect? It is true that modifying the data structure as per the outer information could be beneficial, but is it necessary to sacrifice too much local information? We believe some balance could be identified. Here regarding *similarity* and *distance* as two trade-off objectives, our target is to determine a collaboration strength one should choose so that the

data structure similarity is maximized but meanwhile the structures would not departure too far from their original ones.

The performance index W derived from (2.8)-(2.11) is a sound indicator used to describe the structure *similarity* after the collaboration. To make a distinction let us denote it by W' in this part. The rationale behind this measure is that if the structure of $X[ii]$ is similar to that of $X[jj]$, then the structure should also obtain a good performance on $X[jj]$ (a more similar structure should lead to a lower value of $W[ii]$). For the structure *distance*, since after the collaboration, structures tend to depart from their original ones, we apply the reconciled structure of $X[ii]$ to its own data and get the performance value, and finally summarize these values arising from all data sites, which could be described in the following form

$$Q' = \sum_{ii=1}^{P} \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}[ii]^2 \, \| \boldsymbol{x}_k[ii] - \boldsymbol{v}_i[ii] \|^2 \tag{3.1}$$

We could imagine that in general if collaboration strength α is set to zero, W' would be the highest (as the structures substantially vary from each other) and when α increases, W' would have a tendency to decrease; however, Q' could experience an opposite trend, when α is zero Q' would be the lowest (as only the local structures are utilized), and Q' is going to increase when α arises (as the structures start to resemble each other). Hence, our target could be modeled in the form of a bi-objective optimization problem which is formally written as

$$\begin{aligned} &\min_{\alpha} \quad W' = \sum_{ii=1}^{P} \sum_{jj=1, jj \neq ii}^{P} \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}^2[jj] \, \| \boldsymbol{x}_k[ii] - \boldsymbol{v}_i[ii \mid jj] \|^2 \\ &\min_{\alpha} \quad Q' = \sum_{ii=1}^{P} \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik}[ii]^2 \, \| \boldsymbol{x}_k[ii] - \boldsymbol{v}_i[ii] \|^2 \\ &\text{s.t.} \quad \alpha \in [0,1] \end{aligned} \tag{3.2}$$

where both W' and Q' are the functions of α because $u_{ik}^2[ii]$ and $u_{ik}^2[jj]$ in (3.2) are determined by (2.6) where α is needed.

Note that the above formulated model is intrinsically distinct from those being discussed in the literature. We could imagine that when the collaboration strength range is set between zero and one, our method would always obtain a lower strength value because the two contradictory objectives make α difficult to reach the converged index level which could be easily derived from minimizing the index W' or maximizing the index Q'. However, this is exactly what could have been expected: we need to sacrifice some collaboration strength to achieve a sound balance between *similarity* (derived from the reconciled structures) and *distance* (derived from the original and reconciled structures).

The problem in (3.2) could be solved by a number of methods. Here we simply choose the traditional linear weighted method which would transfer a multi-objective optimization problem to the single-objective one. Since W' is much larger than Q' with the increasing data site number P, we normalize both indices to assume values in-between [0, 1] such that Q' would not be ignored. Hence we further formulate the model in the form

$$\begin{aligned} \min_{\alpha} \quad & S = \frac{W'}{W'_{\max}} + \frac{Q'}{Q'_{\max}} \\ \text{s.t.} \quad & \alpha \in [0,1] \end{aligned} \tag{3.3}$$

where $W'_{\max}$ and $Q'_{\max}$ correspond to the maximum values of W' and Q', respectively. These maxima are observed after we have examined all values of α.

3.4 Granular Partition Matrix

With the optimized collaboration strength α each data site obtains its final reconciled data structure, however, one of our interest is to seek the overall data structure without gathering all the raw data into one site. With all these local data structures (represented by

the partition matrices), the intuitive idea is to unify them into one single partition matrix. In fact a number of research has made their effort to realize this idea: Cleuziou et al. [30] proposed the *Assignment Rule* which used the geometric mean to merge all the partition matrices into a numeric one; Jiang et al. [26] improved this method by taking into consideration the weights of each individual data site derived from the maximum entropy method. However, the major limitation of these methods is that the global representative structure is a single-valued partition matrix (i.e., each entry is a single numeric number), which could make the global data structure dominated by some extreme local structures (i.e., outliers). Even if the outlier effect could be reduced by considering data site weights, the numeric representative is still short of comprehensively displaying the global landscape of the results. Hence, we believe a specifically designed information granule (interval) which covers most local "opinions" while ignoring those extreme cases could better capture the essence of the global structure. In case of P partition matrices $U[ii]$, $ii = 1, 2,\ldots,$ P, for each entry series $\{u_{ik}[1], u_{ik}[2],\ldots, u_{ik}[P]\}$ with the principle of justifiable granularity we could obtain the optimized interval $[u_{ik}^{-}, u_{ik}^{+}]$ and as such the final granular partition matrix $\tilde{U}$. Furthermore, for each entry of the granular partition matrix, we form the index

$$V_{ik} = V_{ik}^{-}\left(u_{ik}^{-}\right) + V_{ik}^{+}\left(u_{ik}^{+}\right) \tag{3.4}$$

to measure the quality of the constructed granule that describes the membership degree of i-th data to the k-th cluster. Generally, a high value of V_{ik} demonstrates that the memberships provided from P data sites tend to exhibit high consensus, otherwise more distributed opinions should be obtained.

3.5 Experimental Studies

In this section, we report the experiments results on both the low-dimensional synthetic data and the real-world data to demonstrate the performance of the developed approach.

3.5.1 Synthetic Data

We construct 5 two-dimensional data sites where each data site has 600 data points (instances) split into 3 to 5 clusters. These data clusters are generated according to the Gaussian distribution with the different mean vectors (cluster centers) and covariance matrices with the values listed in Table 3.2

Table 3.2. Synthetic data set - statistical characteristics.

Data sites	Cluster centers	Corresponding cluster covariance matrices
1	$\boldsymbol{v}1 = [4.5, 9.2]$ $\boldsymbol{v}2 = [11.0, 9.0]$ $\boldsymbol{v}3 = [5.0, 4.0]$ $\boldsymbol{v}4 = [12.0, 4.5]$	$\Sigma 1 = \begin{bmatrix} 2.0 & 0.8 \\ 0.8 & 1.2 \end{bmatrix}$ $\Sigma 2 = \begin{bmatrix} 2.2 & -0.9 \\ -0.9 & 1.2 \end{bmatrix}$ $\Sigma 3 = \begin{bmatrix} 2.0 & -0.5 \\ -0.5 & 1.2 \end{bmatrix}$ $\Sigma 4 = \begin{bmatrix} 1.0 & -0.2 \\ -0.2 & 1.2 \end{bmatrix}$
2	$\boldsymbol{v}1 = [4.0, 4.2]$ $\boldsymbol{v}2 = [6.0, 6.0]$ $\boldsymbol{v}3 = [4.0, 10.2]$ $\boldsymbol{v}4 = [11.5, 8.0]$	$\Sigma 1 = \begin{bmatrix} 3.0 & 0.3 \\ 0.3 & 1.3 \end{bmatrix}$ $\Sigma 2 = \begin{bmatrix} 2.5 & 0.7 \\ 0.7 & 1.9 \end{bmatrix}$ $\Sigma 3 = \begin{bmatrix} 3.0 & 0.1 \\ 0.1 & 1.5 \end{bmatrix}$ $\Sigma 4 = \begin{bmatrix} 4.0 & -0.2 \\ -0.2 & 4.0 \end{bmatrix}$
3	$\boldsymbol{v}1 = [4.5, 6.0]$ $\boldsymbol{v}2 = [7.0, 9.0]$ $\boldsymbol{v}3 = [10.0, 6.0]$	$\Sigma 1 = \begin{bmatrix} 5.0 & 0.6 \\ 0.6 & 2.4 \end{bmatrix}$ $\Sigma 2 = \begin{bmatrix} 3.0 & -0.5 \\ -0.5 & 1.8 \end{bmatrix}$ $\Sigma 3 = \begin{bmatrix} 2.1 & 0.5 \\ 0.5 & 5.0 \end{bmatrix}$
4	$\boldsymbol{v}1 = [6.0, 4.0]$ $\boldsymbol{v}2 = [6.0, 10.0]$ $\boldsymbol{v}3 = [10.0, 10.0]$	$\Sigma 1 = \begin{bmatrix} 3.0 & 0.9 \\ 0.9 & 4.0 \end{bmatrix}$ $\Sigma 2 = \begin{bmatrix} 1.9 & 0.1 \\ 0.1 & 3.2 \end{bmatrix}$ $\Sigma 3 = \begin{bmatrix} 1.8 & -0.7 \\ -0.7 & 1.2 \end{bmatrix}$
5	$\boldsymbol{v}1 = [4.3, 10.0]$ $\boldsymbol{v}2 = [11.0, 9.0]$ $\boldsymbol{v}3 = [5.0, 3.5]$ $\boldsymbol{v}4 = [11.2, 4.0]$ $\boldsymbol{v}5 = [2.0, 6.5]$	$\Sigma 1 = \begin{bmatrix} 2.0 & 0.5 \\ 0.5 & 3.0 \end{bmatrix}$ $\Sigma 2 = \begin{bmatrix} 2.5 & 0.5 \\ 0.5 & 1.2 \end{bmatrix}$ $\Sigma 3 = \begin{bmatrix} 1.7 & -0.5 \\ -0.5 & 3.1 \end{bmatrix}$ $\Sigma 4 = \begin{bmatrix} 4.0 & -0.2 \\ -0.2 & 4.0 \end{bmatrix}$ $\Sigma 5 = \begin{bmatrix} 4.0 & -0.5 \\ -0.5 & 5.0 \end{bmatrix}$

3.5.1.1 Partition Matrix Reordering

Since we want to examine the effect of partition matrix reordering on the original CFC algorithm in [33], both modified and original algorithms are applied to the 5 data sites. Here, we consider the collaboration strength between data sites $X[ii]$ and $X[jj]$ as one when $ii \neq jj$, otherwise $\alpha(ii, jj) = 0$. The number of clusters c ranges from 2 to 7, and 40 collaborative phases are used during the experiments. Specifically, the clustering results of both algorithms are depicted from two perspectives: (a) collaboration performance, which is quantified by index W in (2.11); and (b) the location of data centers (prototypes), where we show the trajectories of the produced data centers.

a. Perspective from the performance index

It is clear shown in Figure 3.2 that whatever number of clusters is predetermined the only difference between the two algorithms happens in the first collaboration phase (here the modified algorithm slightly gets a better performance), after which performance indices of both algorithms merge together and finally converge to the same level (we specifically show the results within 6 collaboration phases).

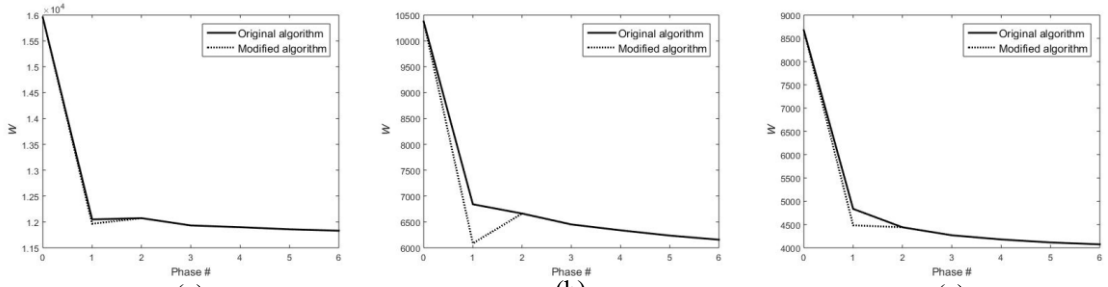


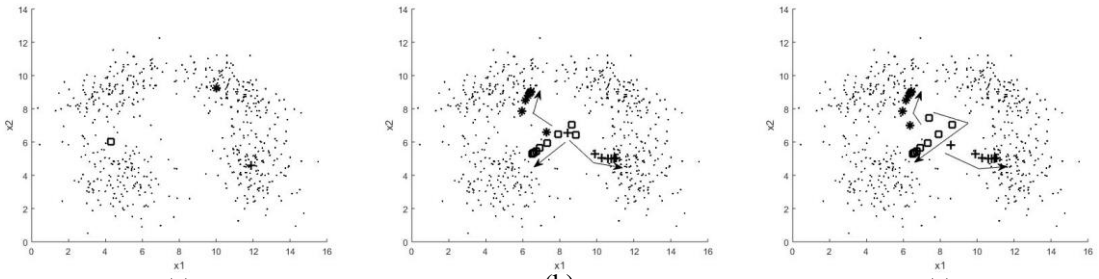
Figure 3.2. Synthetic data sites: performance; (a) c= 2, (b) c = 4, (c) c = 6.

b. Perspective from the prototype trajectories

The movement paths of the prototypes (when cluster number c is 3) with the collaboration continues are further depicted in Figure 3.3, where distinct symbols are used to represent different cluster prototypes, and the arrows express the movement trend of the

prototypes. As could be noted, although at the initial collaboration phases the inconsistency of the prototype trajectories presents, for both algorithms after 40 collaboration phases the distribution of the prototypes are much similar to each other, i.e., both algorithms converge to nearly the same prototypes.

Hence, from the above two perspectives we may conjecture that partition matrix reordering is not necessary in a repetitive collaboration process.



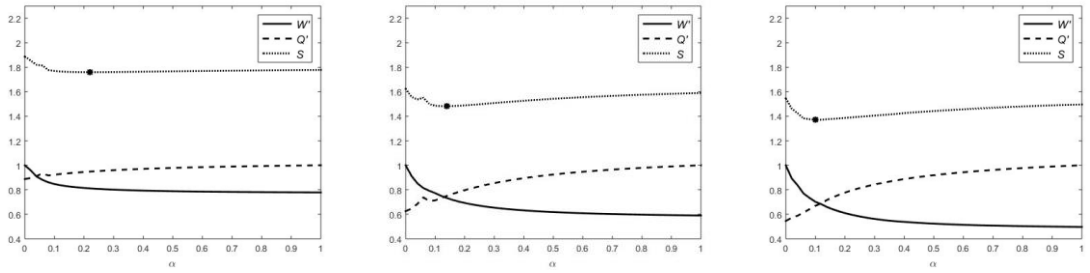
(a) (b) (c)

Figure 3.3. Synthetic Data site 1 with c = 3: prototype trajectories of the (a) FCM, (b) original HCFC, and (c) the modified HCFC.

3.5.1.2 Optimization of Collaboration Strength

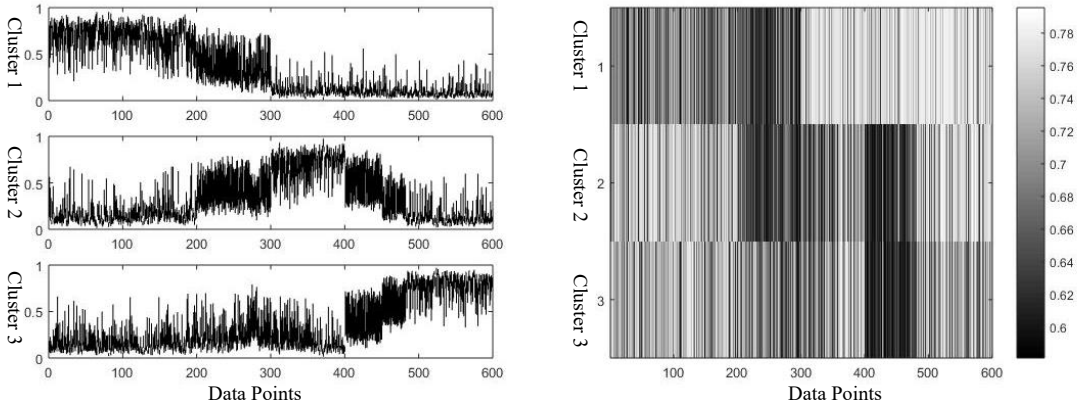
In this section, we focus on the original CFC algorithm in [33], and intend to seek the optimized collaboration strength among the data sites. We assume that $\alpha \in [0,1]$ and the increment step size for α is 0.02. We also consider the number of clusters c ranges from 2 to 7. For each data site, we depict the changing trends of the collaboration effect indexes (W' and Q') as well as that of the synthetic index S when the certain cluster number is prescribed. The results for the synthetic data sites are shown in Figures 3.4, from which we could see that in general, as we expect, the collaboration effect index W' (similarity among the data structures) would decrease along with the increasing α, while index Q' (distance from the original data structures) shows an opposite trend. The optimal value of the index S could always be found which is pointed by the asterisk. In this case, the

collaboration strength among the data sites is relatively small, and it is also obvious that distinct cluster numbers lead to varies collaboration strength, but they stay close to each other within an interval around 0.1 to 0.2.



(a) (b) (c)

Figure 3.4. Collaboration strength optimization for Synthetic data sites; (a) $c = 2$, (b) $c = 4$, (c) $c = 6$.



(a) (b)

Figure 3.5. Results of Synthetic data set when $c = 3$.

3.5.1.3 Granular Partition Matrix

After the optimal collaboration strength is found, the ensuing local data structures are obtained. Now we could use the principle of justifiable granularity to unify the "opinions" from individual sites. Taking for example when cluster number is predefined as 3, the experiment results are presented in Figure 3.5 where plot (a) corresponds to the visualized granule partition matrix and plot (b) relates to the granule quality index V_{ik} defined in (3.4).

Clearly shown in plot (a), now each membership of a data point to a certain cluster is represented by an interval, we can even roughly grasp the partition of the whole synthetic data set: most data points from #1 to #200 have high membership values to Cluster 1, those from #201 to #300 nearly get the even membership degrees to Cluster 1 and 2, Cluster 2 dominants the data points from #301 to #450 while those from #450 to #600 get high membership to Cluster 3.

3.5.2 Publicly Available Data

To further validate our methods, in this part we consider four real-world data sets which are Water Treatment Plant (Water), Breast Cancer Wisconsin (Breast Cancer), Mice Protein Expression (Mice Protein), and SPECTF Heart (Heart). We remove the instances with some missing values in Water and Mice Protein. Information of these data could be found in UCI machine learning data repository.

3.5.2.1 Partition Matrix Reordering

The experiment assumptions (i.e., the collaboration strength, the cluster number, and number of collaboration phases) are same as those in Section 3.5.1.1. Here, we only compare the original and modified algorithms based on the collaboration performance index W in (2.11). Besides, for each data set we report the results of 4 collaboration phases (phase 0, 1, 2, and 40) when cluster number is respectively 3, 5, and 7. The results are documented in Table 3.3.

Same as that observed in the Synthetic case, both original and modified algorithms merged together in phase 2 and finally converge to a certain level. Here the major difference from the Synthetic case is that there is no guarantee that the modified algorithm will obtain a better performance in phase 1. As been highlighted in Table 3.3, for Water, Mice Protein, and Heart the performance of the modified algorithm is even worse than the original CFC; and two algorithms nearly get the even performance on Breast Cancer.

Hence, from both synthetic and real-world data, we could observe that: (a) reordering the partition matrices is not always helpful even at the beginning of the collaboration phase, and (b) partition matrix reordering is not necessary because both algorithms finally converge to the same position.

Table 3.3. Collaboration performance of both algorithms on UCI data sets.

		c =3		c =5		c =7	
		Original	Modified	Original	Modified	Original	Modified
Water	Phase 0	5.20	5.20	4.05	4.05	3.54	3.54
	Phase 1	3.16	3.54	1.99	2.32	1.50	1.72
	Phase 2	3.08	3.08	1.91	1.91	1.42	1.42
	Phase 40	2.93	2.93	1.76	1.76	1.26	1.26
Breast Cancer	Phase 0	1.33	1.33	1.15	1.15	1.06	1.06
	Phase 1	0.80	0.79	0.51	0.52	0.39	0.40
	Phase 2	0.79	0.79	0.50	0.50	0.37	0.37
	Phase 40	0.71	0.71	0.43	0.43	0.32	0.32
Mice Protein	Phase 0	5.85	5.85	4.37	4.37	3.58	3.58
	Phase 1	3.48	3.60	2.15	2.26	1.57	1.67
	Phase 2	3.45	3.45	2.13	2.13	1.54	1.54
	Phase 40	3.33	3.33	2.00	2.00	1.43	1.43
Heart	Phase 0	1.48	1.48	1.17	1.17	1.01	1.01
	Phase 1	0.85	0.89	0.52	0.55	0.38	0.40
	Phase 2	0.84	0.84	0.52	0.52	0.38	0.38
	Phase 40	0.82	0.82	0.50	0.50	0.35	0.35

Note: For each data set, the performance index W has been normalized such that two decimal places are maintained.

3.5.2.2 Optimization of Collaboration Strength

With the same experiment set up in Section 3.5.1.2, we summarize all the optimal collaboration strength values (α_{opt}) for each real-world data set in Table 3.4. Generally, for

each data set α_{opt}'s tend to be close to each other which is consistent with that observed in the synthetic data. Moreover, the collaboration strength varies greatly across the different data sets: Water gets the most significant collaboration strength which roughly locates in interval [0.8, 1], Breast Cancer and Heart show the extremely minor collaboration strength under 0.1 (those for Heart are even less than 0.05), and finally similar to the synthetic data Mice Protein also gets a minor strength roughly between 0.1 and 0.15.

Table 3.4. Optimal collaboration strengths for UCI data sets.

	$c = 2$	$c = 3$	$c = 4$	$c = 5$	$c = 6$	$c = 7$
Water	0.78	0.94	0.88	0.98	0.96	0.92
Breast Cancer	0.06	0.06	0.08	0.08	0.08	0.08
Mice Protein	0.08	0.10	0.12	0.14	0.14	0.14
Heart	0.04	0.02	0.04	0.02	0.02	0.04

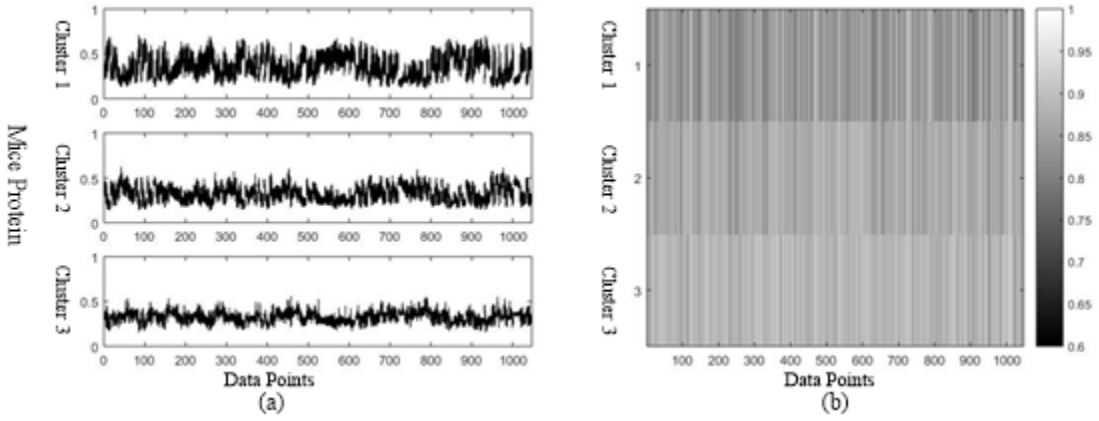


Figure 3.6. Granular partition matrices and the corresponding granule quality for Mice Protein when $c = 3$.

3.5.2.3 Granular Partition Matrix

As with Section 3.5.1.3, after the local data structures for each data set (a group of data sites) are formed, the granular representative of the global data structure as well as its

quality (for each entry) can be obtained. Here, for simplicity we merely list those results for Mice Protein data in Figure 3.6.

For Mice Protein, from the visualized granular partition matrix in plot (a) the granules of data point membership to Cluster 1 are much longer than those to Cluster 2 and 3, and Cluster 3 generally gets the shortest granules. This is consistent with the corresponding granule quality in plot (b), where when cluster index varies from 1 to 3 the color gradually becomes brighter (i.e., higher values of index V_{ik}).

3.6 Summary

Collaborative fuzzy clustering is a technique which is useful in a) refining the local data structures and b) determining the common global data structure, when the interested data are distributed and are not allowed to be gathered into one single data repository. We have focused on three research issues arising in the HCFC algorithm. We used the Hungarian algorithm to assure that the same rows in the partition matrices refer to the same cluster, and experimentally show that the partition matrix reordering is not necessary. We investigated the issue of collaboration strength optimization and here a new optimization mechanism aimed at the determination of collaboration strength was proposed. Finally, we proposed the use of PJG to construct the granular partition matrix to better reflect the global structure of the distributed data.

Chapter 4 Granular Clustering Method for Homogenous Granular Data[b]

Due to the evident complexity of real-world phenomena, granular (as opposed to numeric) data have been considered as a viable vehicle to support modeling pursuits [48]–[53]. Usually, granular data are represented (formalized) as intervals, fuzzy numbers, linguistic terms, probabilities, or hybrid constructs being built by invoking a synergy of several of them. Recently, clustering granular data has gained an increasing interest. Some of the ideas of clustering fuzzy data go back to pioneering studies by Hathaway et al. [54] who introduced a so-called parametric model of clustering to cluster heterogeneous fuzzy data (HFD) consisting of a mixture of numeric data, intervals, and LR-type fuzzy numbers. Since then numerous granular clustering algorithms have been proposed [55], [56], [65]–[69], [57]–[64] offering various developments and augmentations to the existing methods. Whereas they form interesting extensions and improvements, a fundamental question is still left unattended as to the origin of granular data. In their studies, Hung and Yang [58] as well as D'Urso and Giordani [62] used Chinese tea data and blood pressure data described in the form of LR-type fuzzy numbers but without mentioning how those granules were created. Coppi et al. [63] and Chan et al. [70] used questionnaires to provide linguistic terms, but they required respondents to give fuzzy numbers describing these linguistic terms; a process which to some degree, is difficult to implement in reality. Yang et al. [71], Auephanwiriyakul and Keller [66] directly fuzzified numeric data (regarded as the centers of the corresponding LR-type fuzzy numbers) by randomly generating boundaries of the fuzzy numbers; the setting of such boundaries is not supported by any sound estimation mechanism.

[b] A version of this chapter has been published as [126].

Normally, granular data do not manifest themselves directly, but they are, in one way or another, a result of an abstract perception and aggregation of numeric data by establishing some perspective at which one can perceive the phenomenon (system) of interest. The fundamental question is on how to construct granular data on a basis of available experimental numeric data. Only then one may proceed with the clustering pursuits of information granules and deliver a sound evaluation procedure of the obtained results.

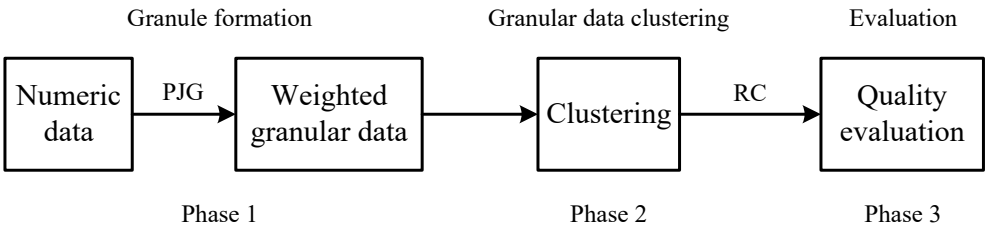


Figure 4.1. A systematic information processing procedure.

In this Chapter, we develop a comprehensive conceptual and algorithmic framework to cope with a problem of clustering homogeneous information granules. While there have been several approaches to coping with granular (viz. non-numeric) data, the origin of granular data considered there is somewhat unclear and, as a consequence, the results of clustering start lacking some full-fledged interpretation. In this study, we offer a holistic view at clustering information granules and an evaluation of the results of clustering. We start with a process of forming information granules with the use of the PJG. With this regard, we discuss a number of parameters used in this development of information granules as well as quantify the quality of the granules produced in this manner. In the sequel, FCM is applied to cluster the derived information granules, which are represented in a parametric manner and associated with weights resulting from the usage of the PJG. The quality of clustering results is evaluated through the use of the reconstruction criterion

(RC) which quantifies the concept of information granulation and degranulation. A systematic information processing procedure proposed in this section is illustrated in Figure 4.1.

4.1 Development of Granular Clustering: A Focused Review

In this section, we offer a brief review of the developments in the area of granular clustering, which is mainly developed from the perspective of used granule types, clustering algorithms, as well as the qualification methods of the clustering results.

The types of information granules can be arranged into three main categories: (a) Heterogeneous fuzzy data (HFD) were first proposed in Hathaway et al. [54], however it was required that the same type of membership functions should be provided for a certain feature. To alleviate this constraint, more flexible forms of membership functions (e.g., a feature which is composed of both Gaussian and trapezoidal fuzzy numbers) were considered in Pedrycz et al. [55]. (b) Interval data (hyperboxes) regarded as the most intuitively appealing granular data were used to formalize the concepts in Pedrycz and Bargiela [61] as well as Gacek and Pedrycz [65]. (c) LR-type fuzzy numbers are the most widely used granular data encountered in current research, including triangular [56]–[60], [62], trapezoidal [56], [63]–[69], [72], and Gaussian [56] fuzzy numbers.

Although different granule types have been covered, only Gacek and Pedrycz [65] gave a sound mechanism to derive intervals on a basis of numeric time series, in other studies it was assumed that granular data are given a priori.

A critical issue in clustering granular data is how to measure the distance between two granular data. Two major avenues are reported in the existing literature: (a) Some characteristics (parameters) forming a numeric vector are used as the representatives of the granular data, thus transforming the original granular data space to the numeric one such that the standard distance measures (used for numeric data) can be used. Most of the

granular clustering algorithms [54], [55], [64], [65], [56]–[63] fall within this category. (b) Overall granular information is utilized [66], [68], [69], [72]. In virtue of the representation theory, a one-dimensional fuzzy set information granule is represented by its α-cuts, the distance between two one-dimensional granules is obtained by integral of the distance of their corresponding α-cuts (derived from interval analysis). Distance between two higher dimensional granules is then derived as a sum of those distances obtained for each feature.

Furthermore, based on the used distance, different clustering algorithms were proposed, including those considering the FCM method [54], [55], [67]–[69], [56]–[60], [64]–[66], Possibilistic *k*-Means methods [62], [63] (where a penalty term was added to the FCM so that the fuzziness and compactness of the clusters were considered simultaneously), as well as hierarchical clustering algorithms [61], [68].

Regarding the evaluation of the quality of the clustering results, which should constitute a significant component of the overall clustering process, it is not well articulated. An alternative sought there concerns cluster validity indices [54], [56]. Pedrycz and Bargiela [61] designed the so-called compatibility index of the component granules (derived on a basis of the distance between granules and compactness of the constructed clusters) and used this measure to find the optimal cluster number (thus of best clustering quality) when a collection of hyperboxes are encountered. Gacek and Pedrycz [65] stressed that the cluster validity-based qualification methods may be inadequate as diverse indices may lead to the distinct results of clustering. To alleviate this shortcoming, a reconstruction criterion was proposed to evaluate the quality of the constructed clusters expressed in the form of hyperboxes.

The current research on granular data clustering is predominantly focused on the development of clustering algorithms. Especially, an attention is being paid to how to form the distance and how to choose a suitable generic clustering model (phase 2 shown in Figure 4.1). In contrast, inadequately treated are the phases of the formation of information

granules on a basis of numeric data and the follow-up assessment of the quality of the constructed clusters. They need to be carefully developed.

4.2 From Numeric Data to Information Granules

In this section, the PJG is used to form information granules constructed on a basis of numeric data. Table 4.1 lists the *coverage* and *specificity* formulas for both left-hand and right-hand sides of the interval and triangular fuzzy set, where θ is the median of the numeric data. This principle has been presented for one-dimensional data. It can be applied to multidimensional data by projecting the data on the individual coordinates (variables) and building information granules there and then forming a multidimensional construct by taking a Cartesian product of them.

Table 4.1. Coverage and Specificity formulas for intervals and triangular fuzzy sets.

		Interval	Triangular fuzzy set
Right-hand side	*Coverage*	$\sum_{x_i \in [\theta, b]} f(x_i) \Big/ N$	$\sum_{x_i \in [\theta, b]} f(x_i) \Big/ N$
	Specificity	$1-(b-\theta)/range$	$1-(b-\theta)/(2range)$
Left-hand side	*Coverage*	$\sum_{x_i \in [a, \theta]} f(x_i) \Big/ N$	$\sum_{x_i \in [a, \theta]} f(x_i) \Big/ N$
	Specificity	$1-(\theta-a)/range$	$1-(\theta-a)/(2range)$

Let us assume that a data set X is given, which is composed of N numeric data points $\boldsymbol{x}_k, k = 1, 2, ..., N,$ located in $\boldsymbol{R}^n$. Now we have the tool to form the granules, but the ensuing question is on how we select the numeric data (as the prototypes) based on which the PJG will be carried out. Usually numeric representatives of the data can be obtained by running some clustering method (e.g., FCM) on the data. However, selecting the appropriate cluster validity index to determine the optimal cluster number requires more attention because it happens that the prototype may locate exactly between the two clusters thus provides a

weak representative. Of course, one may increase the prototype number to offset the shortcoming brought by using inappropriate cluster validity index, however this will incur a huge computing overhead for clustering algorithms. To make the solution simple and efficient, we randomly select a certain percentage p (e.g., 20%) of points from the data set X, resulting a prototype set of size $M = pN$. Another appealing fact behind this random selection is that the dense part of a cluster contained in the numeric data set always gets a higher probability to be represented by the selected prototypes thus making the nature of data structure retained (i.e., essential part of the data will not be lost). Thereafter, around each randomly selected prototype, we build a hyperbox with the side length equals to a ratio r (e.g., 0.1) of the range of each feature, such that the PJG would be performed inside. Here, the range of the j-th feature is defined as

$$boundary_j = \max_k \left(x_{kj} \right) - \min_k \left(x_{kj} \right) \tag{4.1}$$

The ensuing side length of the hyperbox for the j-th feature is represented as

$$range_j = r \bullet boundary_j \tag{4.2}$$

Besides, we assume that the information granule constructed around the projection of a randomly selected prototype is formed inside the interval $[\theta_j - range_j/2, \theta_j + range_j/2]$, where θ_j represents the projection of that prototype on the j-th feature. Note that the role of the range in (4.2) is consistent with that defined in (2.32) and (2.33), i.e., we make the value of *specificity* to be confined to the unit interval. However, as it has been observed, the definition is slightly different. We deem that it is unnecessary and unreasonable to use the entire range of a feature [i.e., the $boundary_j$ in (4.1)] in the PJG such that all data points would be included for building an information granule around a certain selected point. Because if all data points were used, the points belonging to other granules and those outliers of the data set would have been included. Those are, however, noisy artifacts. We denote by N' the number of data points encapsulated in the certain hyperbox.

To gain a better insight into the overall process of generating information granules from the numeric data, we illustrate this process in Figure 4.2 when a two-dimensional data set is encountered. In Figure 4.2 (a), four randomly selected points (denoted by asterisks) are shown, a rectangle (the bottom left one) is given to limit the granules constructed around the prototype formed in this way, inside which we see $N' = 7$ points are confined. Once the points used for constructing the granules have been fixed, the PJG could be utilized for the data resulting from the projections on each feature. Figure 4.2 (b) and (c), respectively demonstrate the process of building the intervals and triangular fuzzy sets around these randomly selected points.

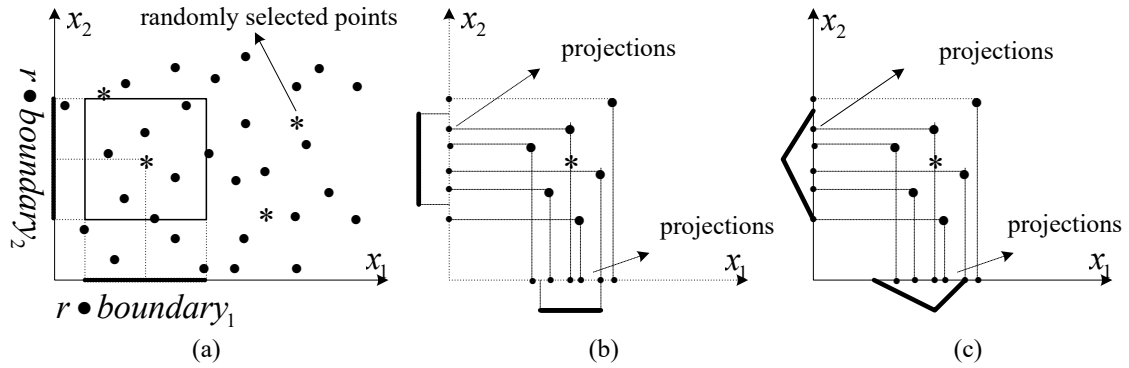


Figure 4.2. Illustration of building information granules around the randomly selected numeric representatives.

4.3 Weighted Granular FCM Clustering

Once a certain type of information granule has been decided upon, information granules are to be described by a collection of suitable parameters. For example, when considering intervals, say $[a, b]$, such granules are represented parametrically by their bounds (a and b). Triangular fuzzy numbers are characterized by triples (a, θ, b). As a result, having a numeric data set $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ where $\boldsymbol{x}_k$ is a vector in $\boldsymbol{R}^n$ and $k = 1, 2, \ldots, N$, these numeric data are finally represented as a collection of information granules

$G = \{\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_M\}$ where $\boldsymbol{g}_s$ is a vector of the parameters of information granule; the parameters are determined by running the PJG as described in the previous section. The dimensionality of $\boldsymbol{g}_s$ depends upon the assumed parametric representation of the granules. As noted above, in case of intervals, $\boldsymbol{g}_s$ is located in $\boldsymbol{R}^{2n}$, while for the triangular membership functions $\boldsymbol{g}_s$ is positioned in $\boldsymbol{R}^{3n}$.

Having the above parametric representation of information granules, we proceed with the details of the weighted FCM algorithm. The weights being associated with the granular data are reflective of the quality of the already constructed information granules. As anticipated, the weight of the *s*-th granular data $\boldsymbol{g}_s$ in *G* is derived based upon the concepts of *coverage* and *specificity*. However, being different from those defined in (2.32) and (2.33) for the one-dimensional data, in a multidimensional case the quantification of the concepts of *coverage* and *specificity* requires more attention. When interval information granule is considered, we require that the number of data (i.e., *P*) included there could serve as a meaningful indicator of *coverage*, however the *specificity* could be represented by taking an average of the individual values of *specificity* considered for *n* features. Hence, the quality of information granule $\boldsymbol{g}_s$ is expressed in the following form

$$Q_s = P\frac{\sum_{j=1}^{n} sp_{sj}}{n} \tag{4.3}$$

where sp_{sj} stands for the *specificity* of interval formed for the *j*-th feature of the *s*-th granular data, which could be obtained by adding the values of the *specificity* measures produced for the left-hand and right-hand side of the granule as outlined in Table 4.1. Moreover, when triangular fuzzy set is used, instead of using the number of data covered by the granule, we use the concept of σ-count to reflect the *coverage* delivered by the *s*-th granule $\boldsymbol{g}_s$. The quality of information granule is then represented as

$$Q_s = \sum_{i=1}^{P} \min_j \left[A_{sj}\left(x_{ij}\right) \right] \frac{\sum_{j=1}^{n} sp\left(A_{sj}(x)\right)}{n} \tag{4.4}$$

where $A_{sj}(x)$ is the triangular membership function of the *j*-th feature of the *s*-th granular data, x_{ij} is the projection of *i*-th numeric point on *j*-th feature, *P* is still the number of points covered by $\boldsymbol{g}_s$, and the minimum function acts as the *t*-norm operator to aggregate the different membership values from *n* features for each numeric value x_{ij}. As before, *specificity* of fuzzy set $A_{sj}(x)$ is obtained by adding its values obtained for the left-hand and right-hand side of *specificity* measures; refer to Table 4.1.

The weight associated with granular data $\boldsymbol{g}_s$ is taken in the form

$$w_s = \frac{Q_s}{\max(Q_s)} \tag{4.5}$$

where $\max(Q_s)$ serves as a normalization coefficient keeping the values of the weight confined to the unit interval. Hence, the better the quality of the constructed information granule, the higher its contribution to the clustering process.

Now considering the weighted granular (parametric) data *G*, the FCM clustering algorithm produces a solution by solving the following optimization problem with constraints

$$\begin{aligned} \min \quad & J(U,V) = \sum_{i=1}^{c} \sum_{s=1}^{M} u_{is}^{m} w_s \left\| \boldsymbol{g}_s - \boldsymbol{v}_i \right\|^2 \\ \text{s.t.} \quad & \sum_{i=1}^{c} u_{is} = 1, \; 0 < \sum_{s=1}^{M} u_{is} < M \end{aligned} \tag{4.6}$$

The minimization is completed over the partition matrix $U = [u_{is}]_{c \times M}$ and the family of prototypes $V = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_c)^{\mathrm{T}}$; w_s is the weight of the *s*-th granular data $\boldsymbol{g}_s$, *c* stands for

the number of clusters, and m (>1) is the fuzzification coefficient. $\|\boldsymbol{g}_s - \boldsymbol{v}_i\|^2$ is the squared Euclidean distance between $\boldsymbol{g}_s$ and the granular prototype $\boldsymbol{v}_i$ as

$$\|\boldsymbol{g}_s - \boldsymbol{v}_i\|^2 = \sum_{t=1}^{n'} (g_{st} - v_{it})^2 \tag{4.7}$$

where g_{st} and v_{it} are respectively the t-th coordinate of the information granule $\boldsymbol{g}_s$ and the granular prototype $\boldsymbol{v}_i$.

To reduce the influence caused by different ranges of the values of the individual features, the squared weighted Euclidean distance for two information granules is obtained as

$$\|\boldsymbol{g}_s - \boldsymbol{v}_i\|^2 = \sum_{t=1}^{n'} \frac{(g_{st} - v_{it})^2}{\sigma_t^2} \tag{4.8}$$

where σ_t is the standard deviation reported for the t-th variable (attribute).

The above minimization problem with constraints is solved by using a technique of Lagrange multipliers. This leads to an iterative process in which the partition matrix and the prototypes are computed (updated) as follows

$$u_{is} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{g}_s - \boldsymbol{v}_i\|}{\|\boldsymbol{g}_s - \boldsymbol{v}_j\|} \right)^{2/(m-1)}} \tag{4.9}$$

$$v_{it} = \frac{\sum_{s=1}^{M} u_{is}^m w_s g_{st}}{\sum_{s=1}^{M} u_{is}^m w_s} \tag{4.10}$$

where, i = 1, 2,…, c; s = 1, 2,…, M; t = 1, 2,…, n', m is the fuzzification coefficient, c is the number of clusters, M is the number of constructed information granules. Depending on the parametric form of information granules, the number of the parameters n' of each

granule $\boldsymbol{g}_s$ equals 2n (interval) or 3n (triangular fuzzy set). w_s and g_{st} are the weight and element of $\boldsymbol{g}_s$, respectively.

4.4 Granular Clustering Evaluation and Optimization

A reconstruction criterion (involving both granulation and degranulation mechanisms) has been proposed as a sound method to quantify the performance of the clustering algorithm in the presence of numeric data. Generally, it consists of three steps: (a) clustering numeric data to reveal the data structure (prototypes and partition matrices); (b) using the obtained structure to reconstruct the original data; and (c) quantifying differences between original and reconstructed data. Evidently, the lower these differences are, the better the performance of the clustering method in the sense of its ability to represent original data.

To be consistent with the already used notation, let us assume that granular data set G consists of M granular data $\boldsymbol{g}_s$, $s = 1, 2, \ldots, M$, and each data is represented either in the 2n-dimensional space (interval case) or the 3n-dimensional case (triangular fuzzy set). We apply the reconstruction criterion to the parametric representation of granular data G with the detailed steps outlined as follows:

(a) Granulation. The weighted FCM is used to cluster the data into c groups. The granules formed in this manner become represented in terms of the partition matrix U and prototypes as $V = (\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_c)^{\mathrm{T}}$, obtained by (4.9) and (4.10), respectively. Of course, other clustering algorithms can also be used here, as long as they represent the data structure through prototypes and partition matrix.

(b) Degranulation. This process determines a datum (estimate of the original information granule $\boldsymbol{g}_s$) by minimizing the following distance

$$\min \sum_{i=1}^{c} u_{is}^{m} \left\| \hat{\boldsymbol{g}}_s - \boldsymbol{v}_i \right\|^2 \tag{4.11}$$

where m is the fuzzification coefficient, u_{is} stands for the element of U describing the membership of s-th information granule $\boldsymbol{g}_s$ to i-th prototype $\boldsymbol{v}_i$, both V and U are derived from the granulation process. As before either a generic or weighted Euclidean distance is used. The result of optimization of (4.11) is given in the form

$$\hat{\boldsymbol{g}}_s = \frac{\sum_{i=1}^{c} u_{is}^{m} \boldsymbol{v}_i}{\sum_{i=1}^{c} u_{is}^{m}} \tag{4.12}$$

which could be regarded as a linear combination of the c prototypes $\boldsymbol{v}_i$, $i = 1, 2, \ldots, c$. We obtain the reconstructed granular data set, which is denoted by $\hat{G} = \{ \hat{\boldsymbol{g}}_1, \hat{\boldsymbol{g}}_2, \ldots, \hat{\boldsymbol{g}}_M \}$.

(c) Comparison. Here we express the reconstruction error between the original granular data G and the reconstructed ones $\hat{G}$, by summing up differences for each original-reconstructed granular data pairs $(\boldsymbol{g}_s, \hat{\boldsymbol{g}}_s)$. Since the information granule $\boldsymbol{g}_s$ is intrinsically an $\boldsymbol{R}^{n'}$ dimensional numeric vector, so is its reconstructed version $\hat{\boldsymbol{g}}_s$. We denote the difference between the two data sets G and $\hat{G}$ as

$$E = \sum_{s=1}^{M} \left\| \boldsymbol{g}_s - \hat{\boldsymbol{g}}_s \right\|^2 \tag{4.13}$$

with the weighted Euclidean distance is being used.

Note that the value of the reconstruction error depends on a certain fuzzification coefficient m. This makes the optimization of m possible: when varying the values of m in a certain range, one determines a minimal value of the reconstruction error.

4.5 Experimental studies

To demonstrate the usefulness of the proposed approach and quantify its performance, a series of experiments has been completed. We report results obtained for both synthetic and publicly available data.

4.5.1 Synthetic Data

The 2D synthetic data are composed of 10 clusters, which are normally distributed with centers (mean vectors) reported in Table 4.2 and the same covariance matrices $\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$. Each cluster is composed of 100 data points. The data are illustrated in Figure 4.3 (a).

Table 4.2. 10 Centers of synthetic data.

Cluster #	Cluster center	Cluster #	Cluster center
1	[2.0, 2.0]	6	[3.5, 5.0]
2	[5.0, 2.0]	7	[0.0, 5.0]
3	[8.0, 2.0]	8	[2.0, 8.0]
4	[10.0, 5.0]	9	[5.0, 8.0]
5	[6.5, 5.0]	10	[8.0, 8.0]

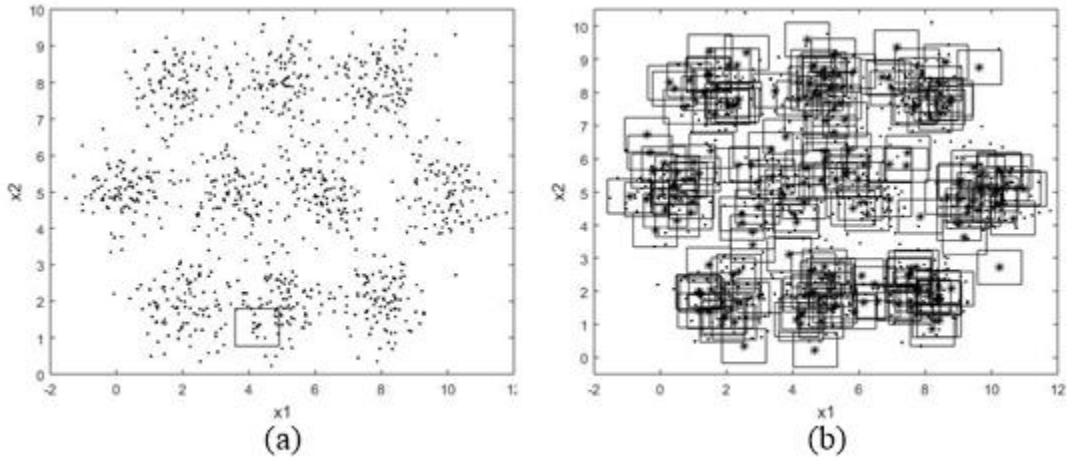
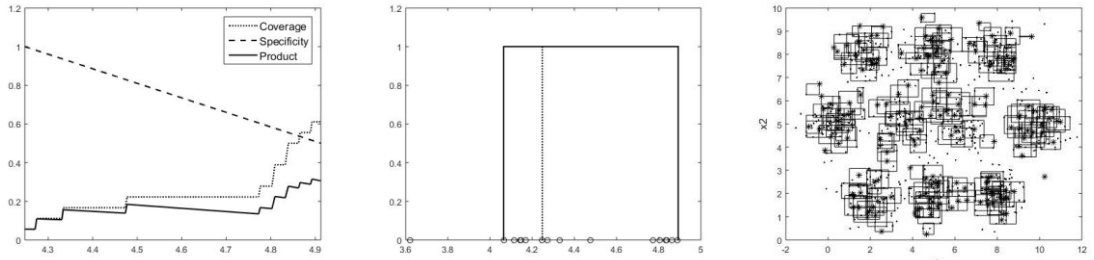


Figure 4.3. Synthetic data, randomly selected points, and boundaries of information granules.

4.5.1.1 Formation of Information Granules

To construct information granules from numeric data, we first randomly pick $p = 0.2$ data from the synthetic data, resulting in $M = 200$ centers (prototypes) which are shown as the asterisks in Figure 4.3 (b). By forming the boundaries of each feature (variable) from (3) and setting the ratio r as 0.1, the ranges for center of each granule are derived from (4) and shown as a collection of hyperboxes (rectangles) in Figure 4.3 (b).



(a) (b) (c)

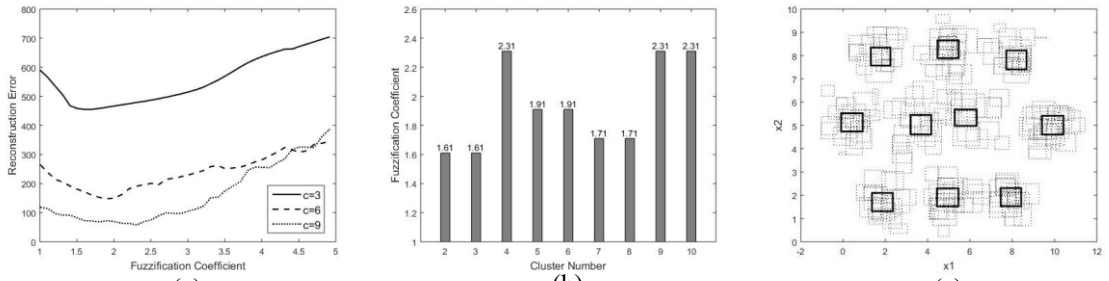
Figure 4.4. (a) Optimization process, (b) optimal granules for the one-dimensional data from a sample patch, and (c) optimal granules for entire synthetic data.

We illustrate the optimization process of PJG for a selected one-dimensional data set, i.e., data of the feature x_1 confined to the rectangle shown in Figure 4.3 (a). Here the range of feature x_1 is denoted by $boundary_1 = 13.30$, as such we get the ensuing range $range_1 = 1.33$ for the selected point $\theta = 4.25$. All data included in the interval $[\theta - range_1/2, \theta + range_1/2] = [3.59, 4.91]$ are used during the PJG process. We list these 18 one-dimensional data points $X = \{4.15, 4.27, 4.17, 4.86, 4.81, 4.25, 4.84, 4.89, 4.12, 4.48, 4.77, 4.25, 4.83, 4.80, 4.14, 4.33, 4.07, 3.62\}$. The data are displayed in Figure 4.4 (b). Taking the optimization of the right-hand side, we range values of b from θ to $\theta + range_1/2$ with a step size of $range_1/2/200$. The optimal value is obtained by maximizing the product of *coverage* and *specificity*. For this one-dimensional data, the optimization process when the interval is selected shown in Figure 4.4 (a), where we find that the product is maximized nearly at

the end of $\theta + range_1/2$. Taking the left-hand side into consideration, we finally obtain the complete optimal interval granules shown in Figure 4.4 (b). For the formed information granule, a few points being positioned far away from θ are excluded once the optimization process has been completed. By applying the PJG on the projected data in each rectangle, we finally obtain the interval granules in Figure 4.4 (c). When comparing these granular data with their original rectangular ranges, the optimized granules are shrunk and become more separated.

4.5.1.2 Clustering Information Granules

Once information granules have been formed, and the weights were determined based on (4.5), one proceeds with the clustering procedure. As the reconstruction criterion serves as a mechanism to optimize the fuzzification coefficient m, we first determine the optimal value of m for different numbers of clusters.



(a) (b) (c)

Figure 4.5. (a) Fuzzification coefficient optimization for weighted FCM, (b) optimized fuzzification coefficients, and (c) locations of prototypes obtained for $c = 10$, $m = 2.31$.

We range the values of m from 1.01 to 5 with a step size of 0.1 and determine the relationship between the reconstruction error and m for different number of clusters c (ranging it from 2 to 10). The 10-fold experiment is conducted to avoid any possible bias. The threshold ε used in the stopping criteria for the weighted FCM is set as 0.00001, and the weighted Euclidean distance in (4.8) is used as the distance computed for the granular

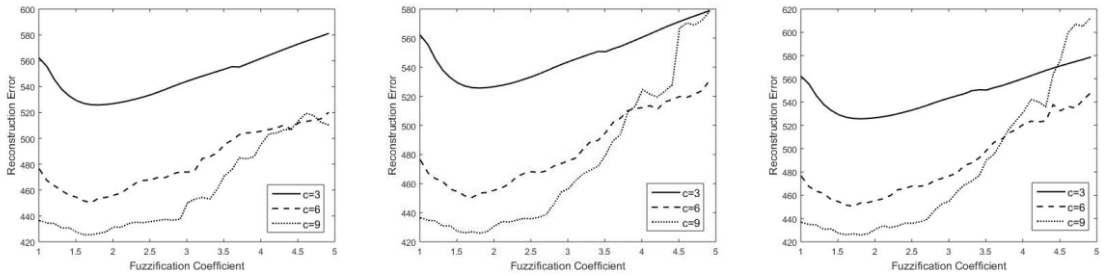
data. For interval granules, Figure 4.5 (a) shows that the reconstruction error varies when different values of m are used; the optimal m could be clearly determined. Also Figure 4.5 (b) shows the differences among the optimal values of m's obtained for different number of clusters.

With the optimized value of m, we cluster the granular data. Here we report the clustering results (granular prototypes) for intervals when c assumes values equal to 10. It is seen from Figure 4.5 (c) that the obtained interval prototypes capture the essence of the corresponding granular data. They reflect the clusters dominating the granular data.

4.5.1.3 Comparative analysis with other methods

We compare different granular clustering algorithms from the perspective of the reconstruction criterion. The proposed weighted FCM algorithm is compared with the clustering methods introduced by Effati et al. [68] as well as Zarandi et al. [69], which are two representative algorithms of granular clustering. However, to make these methods comparable, we make the following assumptions: (i) As it has been examined in Hathaway et al. [54], different parameterization of information granules may lead to different clustering results. This entails that we have to use the same parametric representation of information granules. For instance, the triangular information granule is represented as the triple (a, θ, b) in the weighted FCM, however it is described as (a_1, a_2, a_3) in [68] and [69], where a_1 is the central point of the core of the fuzzy sets, a_2 and a_3 stand for the length of the left and right part support separated by a_1. In this case, we have the relationships $a_1 = \theta$, $a_2 = \theta - a$, and $a_3 = b - \theta$. Hence, when carrying out comparisons, the weighted FCM algorithm operates in the parametric data described in the form of (a_1, a_2, a_3). (ii) Since the weights of granular data have been considered in the weighted FCM, we also include these weights in the two other methods used in the comparative study. (iii) The update formulas in [68] and [69] make the incorporation of the standard deviation of each feature difficult. To achieve the consistency, here the weighted FCM uses the standard Euclidean distance

instead of its weighted version. Some other experimental settings such as the structures of granules, range of values of cluster number c and fuzzification coefficient m, as well as the step sizes are the same as those used in the previous section.



(a) Weighted FCM (b) Effati et al. (c) Zarandi et al.

Figure 4.6. Relationship between fuzzification coefficient and reconstruction error for three methods.

Table 4.3. Reconstruction error for three methods: comparative analysis (triangular fuzzy sets as information granules).

c	Weighted FCM		Effati et al.		Zaranti et al.	
	m	Error	m	Error	m	Error
2	1.61	**1048.82 ± 0.00**	1.61	1050.99 ± 0.01	1.51	1049.26 ± 0.03
3	1.71	**928.59 ± 0.02**	1.81	932.96 ± 0.24	1.81	929.20 ± 0.28
4	1.91	886.05 ± 8.02	2.01	889.21 ± 5.88	2.01	**883.00 ± 7.44**
5	1.91	860.56 ± 0.01	1.91	867.09 ± 0.08	1.91	**859.49 ± 0.08**
6	1.71	**850.90 ± 2.87**	1.81	858.07 ± 3.18	1.81	851.04 ± 3.46
7	1.51	**837.79 ± 4.16**	1.61	846.21 ± 1.01	1.61	838.10 ± 1.61
8	1.51	830.28 ± 0.26	1.51	837.80 ± 0.85	1.51	**830.16 ± 1.91**
9	1.51	824.25 ± 2.43	1.51	831.40 ± 2.29	1.51	**822.08 ± 0.06**
10	1.21	821.82 ± 5.17	1.41	829.66 ± 4.29	1.41	**821.47 ± 3.67**

Note: the entities in boldface represent the best reconstruction performance obtained for three methods.

Figure 4.6 illustrates the optimization process of m when c is set as 3, 6, and 9 for intervals. The optimal value of m is clearly visible. In fact, for both intervals and triangular fuzzy sets, different methods show a similar dependence between m and the reconstruction error for the given number of clusters.

Table 4.3 reports the values of the reconstruction error (shown are the mean values and standard deviations of the 10-fold experiment) when the triangular fuzzy set is selected. The weighted FCM exhibits the best performance when c assumes values 2, 3, 6, and 7, while the method presented in [69] exceeds the other two methods for the remaining cluster numbers. However, the difference among these performance values is not that large.

4.5.2 Publicly Available Data

Here the UCI data Yeast, Banknote, and CCPP are used to test the proposed method. Specifically, here we only focus on granular clustering algorithms in terms of their optimization, clustering, and evaluation, while the process of the formation of information granules is not reported here. To construct the granular data, percentage of randomly selected points is set as follows: $p = 0.2$ for Yeast and Banknote, and $p = 0.1$ for CCPP. In all data sets we use the same range ratio as $r = 0.15$. Without any doubt, other values of p and r may be used, however we tend to maintain these values relatively small because the formed granules could capture the essence of the data structure and the computation overhead is acceptable. During the clustering and optimization process, the values of the fuzzification coefficient m varies from 1.01 to 3 with a step size of 0.1; other settings are the same as those reported in Section 4.5.1.3.

We report the optimization process of the three methods on the UCI data when the triangular information granules are used, see Figure 4.7, as in the case of the interval information granules we obtain similar results. For a given data set, similar to what we have observed in the case of synthetic data, all three methods get the similar relationships between m and the reconstruction error for the same cluster number. However, when we

focus on a certain algorithm, its performance depends on the data being used. Smooth curves describing relationship between value of *m* and the corresponding error are exhibited for the CCPP data, while in the case of the Yeast and Banknote data we observe more oscillations. As in the case of the synthetic data, the optimal value of *m* tends to be located in the interval of [1.01, 2].

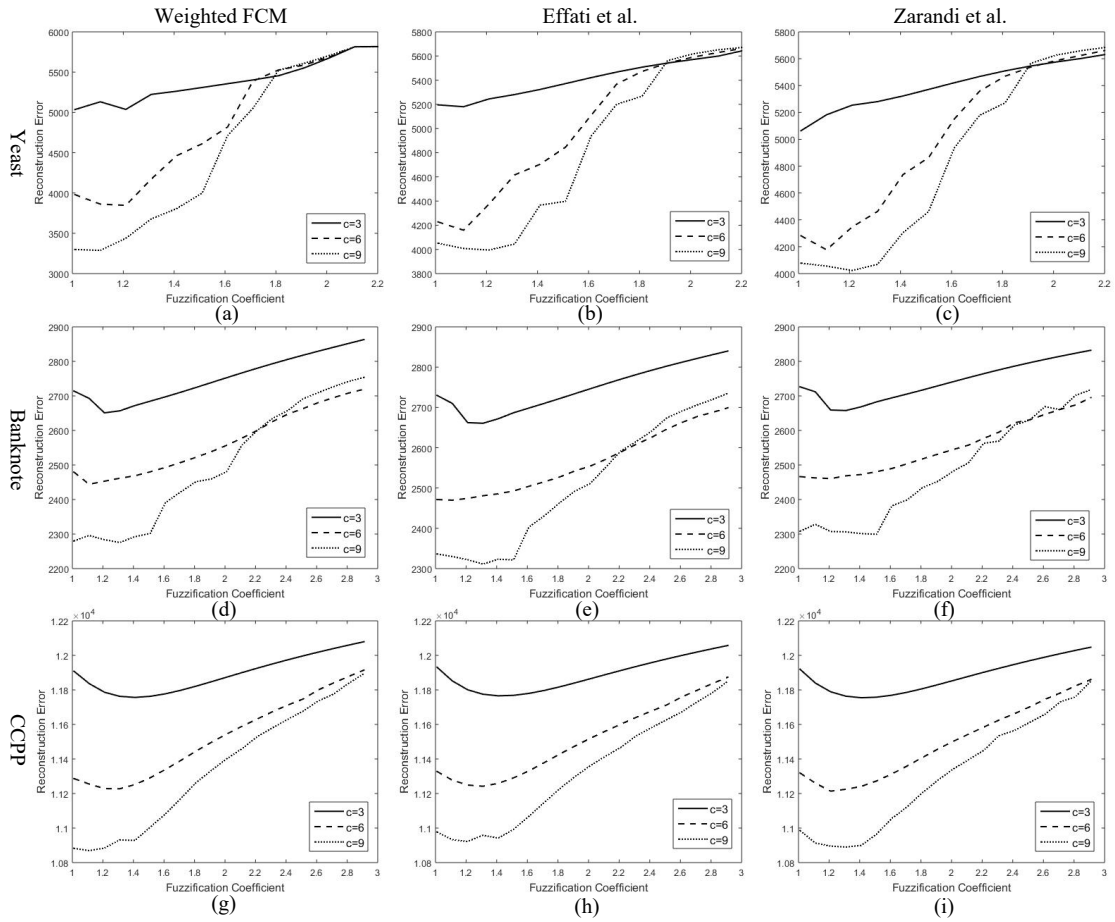


Figure 4.7. Relationship between fuzzification coefficient and reconstruction error for three methods on UCI data.

Given the optimal value of *m*, we run the three clustering algorithms considering different numbers of clusters. The results are reported for selected values of *c*, namely *c* set

to 3, 6, and 9, see Table 4.4. Surprisingly, only when there are three clusters of triangular fuzzy sets, the method by Zaranti et al. [69] yields the best performance. The weighted FCM performs the best in all remaining scenarios however the differences are not large.

4.6 Summary

This Chapter is aimed at formulating and solving essential issues encountered in clustering granular data: (a) formation of granular data on a basis of numeric experimental evidence; (b) weighting granular data to reflect their quality and abilities to describe numeric data they originated from; (c) optimization of the fuzzification coefficient along with the way of evaluating the performance of different granular clustering algorithms. To address these problems, the PJG-based granular data generating method, the weighted FCM granular clustering algorithm, as well as the granular reconstruction criterion were proposed and studied. Some observations of a general nature can be made. (a) The PJG forms a sound and algorithmically viable way to construct information granules on a basis of numeric data. (b) It is beneficial to take the quality of the formed information granule into consideration, such that the final obtained clustering results (e.g., granular prototypes) would not be greatly influenced by those of less significance. (c) From the granular reconstruction criterion, the overall representation of information granule-based clustering algorithms failed to show the advantage over the representative-based one (i.e., the proposed weighted FCM method), although they have utilized all information of the information granule.

Table 4.4. Reconstruction error for three methods on UCI data: comparative analysis.

	Methods		Weighted FCM		Effati et al.		Zaranti et al.	
Data sets	Granule type	Cluster #	*m*	Error	*m*	Error	*m*	Error
Yeast	Interval	3	1.01	**3543.41±24.67**	1.01	3556.29±12.60	1.11	3552.07±0.00
		6	1.11	**2697.97±170.94**	1.01	2836.81±50.54	1.01	2803.96±144.00
		9	1.11	**2314.00±134.10**	1.11	2504.32±112.72	1.11	2522.93 ± 18.18
	Triangular	3	1.01	**4934.42±285.99**	1.11	5226.00 ± 48.04	1.01	5195.86±161.21
	fuzzy set	6	1.21	**3847.31 ± 35.10**	1.11	4153.91 ± 15.81	1.11	4174.65 ± 13.02
		9	1.11	**3256.40±233.36**	1.21	3998.09 ± 40.81	1.21	4038.94 ± 41.93
Banknote	Interval	3	1.21	**1601.32±24.31**	1.21	1610.07 ± 24.31	1.21	1612.02 ± 24.32
		6	1.11	**1320.58±30.97**	1.11	1334.62 ± 36.10	1.11	1352.06 ± 39.53
		9	1.01	**1194.90 ± 53.98**	1.01	1211.65 ± 43.71	1.01	1223.60 ± 42.51
	Triangular	3	1.21	**2651.00 ± 0.00**	1.31	2660.68 ± 0.08	1.31	2657.77 ± 0.00
	fuzzy set	6	1.11	**2448.86 ± 24.39**	1.11	2468.61 ± 18.26	1.21	2461.53 ± 19.15
		9	1.31	**2289.53 ± 24.11**	1.31	2330.77 ± 29.39	1.51	2311.09 ± 28.25
CCPP	Interval	3	1.41	**6992.08 ± 0.00**	1.41	6993.40 ± 0.00	1.51	6993.65 ± 0.00
		6	1.21	**6426.60 ± 39.58**	1.21	6433.69 ± 36.46	1.21	6435.29 ± 35.84
		9	1.31	**6033.22 ± 17.98**	1.31	6056.12 ± 17.36	1.31	6060.51 ± 18.54
	Triangular	3	1.41	11756.64 ± 0.07	1.41	11765.80 ± 0.07	1.41	**11754.91 ± 0.03**
	fuzzy set	6	1.31	**11227.33 ± 8.28**	1.31	11242.05 ± 8.47	1.21	11227.03±23.64
		9	1.11	**10882.15±45.27**	1.21	10903.84±41.97	1.31	10889.13±39.97

Note: the entities in boldface represent the best reconstruction performance obtained for three methods.

Chapter 5 Granular Clustering for Heterogenous Granular Data based on Approximation of Fuzzy Sets[c]

In the former section, the clustering method for homogenous granular data is given. However, it happens that in real life heterogeneous granular data may be obtained from the different data sources. To make clustering on this kind of data possible, in this chapter, we propose two methods to transform the heterogenous information granules into the homogenous ones. Specifically, we assume that all the encountered heterogenous information granules could be represented by homogenous fuzzy sets.

Methods for retaining major characteristics of fuzzy sets of non-regular shape are briefly reviewed below. Pedrycz et al. [55] proposed the possibility-necessity (PN) model and the piecewise linear spline (PLS) representation to describe a fuzzy set as a set of parameters to be used in further processing. Similar to the PLS scheme, the idea of piecewise linear approximation is also presented in Coroianu et al. [73]. Wang and Li [74] proposed approximating a fuzzy set by a step-like membership function. Grzegorzewski and Mrowka [75] used trapezoidal fuzzy sets (TFSs) to approximate fuzzy sets, where the distance between two fuzzy sets was described by the Euclidean distance between their α-cuts. They proposed the closed-form trapezoidal approximation operator to calculate the optimal TFS. However, since the parameters derived from [75] may either not constitute a fuzzy number [76]–[79] or fail to have a sound interpretation for some skewed fuzzy sets [80], many contributions have attempted to fill these gaps [79]–[84]. Examples of non-approximation-based methods in [85], [86] provide alternatives for preserving the characteristics of fuzzy sets.

It is apparent that the piecewise linear approximation of any membership function

[c] A version of this chapter has been published as [127].

yields an inevitable approximation error. Based on its value, one can judge whether this approximation is acceptable vis-à-vis the interpretation gained and proceed accordingly: if the benefits are justifiable, the linear approximation is deemed relevant. The main point to be made here is the following: if we wish to realize a piecewise linear approximation of a fuzzy set, its description can be regarded as a type-2 fuzzy set (in particular, interval-valued fuzzy set) with the elevation of the type of the fuzzy set being inevitably associated with the approximation of the shape of the membership function. Put it differently: a simpler functional form of membership function entails its elevated type. Following this line of thought, we propose a gradient-based method to approximate a fuzzy set through a trapezoidal fuzzy set (TFS). By adding some constraints in the formulated optimization problem, the major characteristics of the fuzzy set such as the core, the major part of the support, and the shape could be preserved; also the form of the optimized result as a TFS is guaranteed. We regard the optimized TFS as the "skeleton" of the original fuzzy set. Based on this "skeleton", we further extend the TFS to higher type, i.e., an interval type-2 trapezoidal fuzzy set (IT2 TFS), so that more information of the original fuzzy set could be captured but the number of the parameters used to describe the original fuzzy set is still controlled (nine parameters are required for an IT2 TFS). The PJG is used to assure that the formed type-2 granule has a sound interpretation. Synthetic fuzzy sets have been used to demonstrate the usefulness of the proposed approximation methods. The overall process of this two-phase approximation method is illustrated in Figure 5.1.

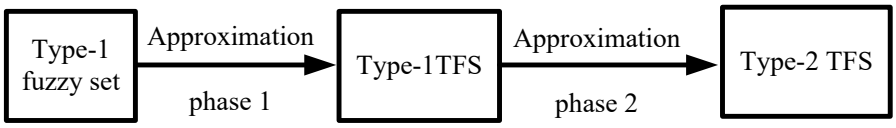


Figure 5.1. Scheme of the two-phase approximation of fuzzy set: type-1 approximation followed by the type-2 elevation process.

5.1 Focused Family of Fuzzy Sets

Various shapes of fuzzy sets are derived from different sources and operations. Here, we note the three basic features of the fuzzy set which are essential to the proposed approximation algorithms in this study.

1) Finite support. The support of the fuzzy set needs to be a finite subset over a space of real numbers.
2) Interval core. The core of the fuzzy set is either a number or an interval.
3) Continuity. The membership function of the fuzzy set is continuous over real numbers.

Examples of major categories of fuzzy sets presented in this study are shown in Fig. 5. Here fuzzy sets in Figure 5.2 (b) and Figure 5.2 (c) are truncated versions of the general form in Figure 5.2 (a). Most of fuzzy sets are qualified with the listed three features. By using some preprocessing techniques more fuzzy sets can be included. For instance, (a) an unbounded fuzzy set could be truncated to preserve the major part of the membership function; (b) a non-normalized membership function can be normalized by its height (the final optimized fuzzy set can be transformed back through multiplying its membership function by this height); and (c) a non-continuous membership function can be made continuous by connecting the disconnected parts of the graph of the membership function.

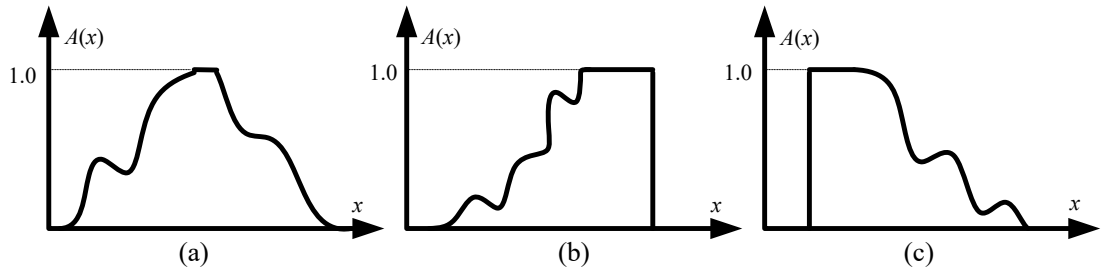


Figure 5.2. Categories of focused fuzzy sets. (a) General fuzzy set; (b) Right-hand side truncated fuzzy set; (c) Left-hand side truncated fuzzy set.

5.2 Approximation of Fuzzy Set by TFS

In this section, we first depict the type-1 approximation method for the general fuzzy set given in Figure 5.2. Considering that the membership function of the fuzzy set is denoted by $A(x)$ defined on X; n_1 and n_2 are respectively the left- and right-hand side bounds of the core ($n_1 = n_2$ when the core only contains a single element), while a and b are the left- and the right-hand side bounds of the support of the fuzzy set. A TFS, with the membership function $A_{\mathrm{TFS}}(x)$, is uniquely represented by the vector of parameters $\boldsymbol{t} = (t_1, t_2, t_3, t_4)^{\mathrm{T}}$. Approximation of the original fuzzy set by the TFS is realized such that the Euclidean distance between $A(x)$ and $A_{\mathrm{TFS}}(x)$, refer to (5.1), is minimized. Their membership functions are shown in Figure 5.3.

$$A(x)=\begin{cases} f_1(x), a \le x < n_1 \\ 1, n_1 \le x < n_2 \\ f_2(x), n_2 \le x < b \\ 0, \text{otherwise} \end{cases}, A_{\mathrm{TFS}}(x)=\begin{cases} g_1(x) = (x - t_1)/(t_2 - t_1), t_1 \le x < t_2 \\ 1, t_2 \le x < t_3 \\ g_2(x) = (t_4 - x)/(t_4 - t_3), t_3 \le x < t_4 \\ 0, \text{otherwise} \end{cases} \tag{5.1}$$

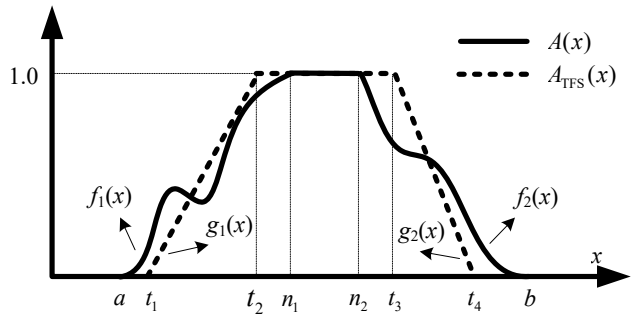


Figure 5.3. Approximation of fuzzy set by TFS.

The approximation problem is expressed as the following constrained optimization task

$$\min\ Q(\boldsymbol{t}) = \int_a^b \left(A(x) - A_{\mathrm{TFS}}(x) \right)^2 dx \tag{5.2}$$

$$\text{s.t.}\ a - t_1 \le 0, t_1 - t_2 \le 0, t_2 - n_1 \le 0, n_2 - t_3 \le 0, t_3 - t_4 \le 0, t_4 - b \le 0 \tag{5.3}$$

where Q is the minimized objective function of $\boldsymbol{t}$. Constraints $a \le t_1$ and $t_4 \le b$ are provided to make the objective function tractable (in terms of getting the gradient with respect to $\boldsymbol{t}$). $t_2 \le n_1$ and $n_2 \le t_3$ are used to retain the core of the original fuzzy set.

In light of the series of constraints, we resort to the method of Lagrange multipliers; refer to Chapter 23 in [87]. Let us introduce the vector as

$$\boldsymbol{h}(\boldsymbol{t}) = \left(h_1(\boldsymbol{t}),\ldots,h_6(\boldsymbol{t})\right)^{\mathrm{T}} = (a - t_1, t_1 - t_2, t_2 - n_1, n_2 - t_3, t_3 - t_4, t_4 - b)^{\mathrm{T}} \tag{5.4}$$

Then we form the augmented objective function by accommodating a vector of Lagrange multipliers capturing the constraints

$$L(\boldsymbol{t}, \boldsymbol{\mu}) = Q(\boldsymbol{t}) + \boldsymbol{\mu}^{\mathrm{T}} \boldsymbol{h}(\boldsymbol{t}) \tag{5.5}$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6)^{\mathrm{T}}$ is the vector of multipliers. The optimization problem is solved in an iterative fashion as

$$\boldsymbol{t}^{(k+1)} = \left\lfloor \boldsymbol{t}^{(k)} - \alpha^{(k)} \left(\nabla Q(\boldsymbol{t}^{(k)}) + \boldsymbol{\mu}^{\mathrm{T}(k)} \mathrm{D}\left(\boldsymbol{h}(\boldsymbol{t}^{(k)}) \right) \right) \right\rfloor_* \tag{5.6}$$

$$\boldsymbol{\mu}^{(k+1)} = [\boldsymbol{\mu}^{(k)} + \beta^{(k)} \boldsymbol{h}(\boldsymbol{t}^{(k)})]_+ \tag{5.7}$$

where k stands for the kth iteration of the algorithm, ∇ is the gradient operator, D(·) is the first order derivative operator, α and β are positive step sizes. To make sure that the core of the fuzzy set $A(x)$ is always included in $[t_2, t_3]$, when $t_2^{(k+1)} > n_1$ we set $t_2^{(k+1)} = n_1$; when $t_3^{(k+1)} < n_2$, we set $t_3^{(k+1)} = n_2$. To guarantee the $\boldsymbol{t}^{(k+1)}$ is resulted as a TFS, when $t_1^{(k+1)} > t_2^{(k+1)}$ we set $t_1^{(k+1)} = t_2^{(k+1)} - \xi$; when $t_4^{(k+1)} < t_3^{(k+1)}$ we set $t_4^{(k+1)} = t_3^{(k+1)} + \xi$; where ξ is a small positive number used to make the calculation of $\nabla Q(\boldsymbol{t})$ feasible. In (5.6), the operator $[\cdot]_*$ indicates the operations mentioned above. In (5.7), the truncation operator $[\cdot]_+ = \max\{\cdot, 0\}$ applied component wise is used to ensure that the elements of $\boldsymbol{\mu}$ satisfy the Kuhn-Tucker condition.

It is straightforward to determine $\mathrm{D}(\boldsymbol{h}(\boldsymbol{t}))$ in (5.6) as

$$\mathrm{D}(\boldsymbol{h}(\boldsymbol{t})) = \left(\mathrm{D}(h_1(\boldsymbol{t})),\ldots,\mathrm{D}(h_6(\boldsymbol{t}))\right)^{\mathrm{T}} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.8}$$

However, getting $\nabla Q(\boldsymbol{t})$ requires more attention. For simplicity, we summarize the final formulas as follows.

$$\nabla Q(\boldsymbol{t}) = (\partial Q/\partial t_1, \partial Q/\partial t_2, \partial Q/\partial t_3, \partial Q/\partial t_4)^{\mathrm{T}}$$

$$= \begin{pmatrix} \dfrac{2t_2}{(t_2-t_1)^2}\displaystyle\int_{t_1}^{t_2} f_1(x)dx - \dfrac{2}{(t_2-t_1)^2}\int_{t_1}^{t_2} xf_1(x)dx - \dfrac{1}{3} \\ \dfrac{2}{(t_2-t_1)^2}\displaystyle\int_{t_1}^{t_2} xf_1(x)dx - \dfrac{2t_1}{(t_2-t_1)^2}\int_{t_1}^{t_2} f_1(x)dx - \dfrac{2}{3} \\ \dfrac{2}{(t_4-t_3)^2}\displaystyle\int_{t_3}^{t_4} xf_2(x)dx - \dfrac{2t_4}{(t_4-t_3)^2}\int_{t_3}^{t_4} f_2(x)dx + \dfrac{2}{3} \\ \dfrac{2t_3}{(t_4-t_3)^2}\displaystyle\int_{t_3}^{t_4} f_2(x)dx - \dfrac{2}{(t_4-t_3)^2}\int_{t_3}^{t_4} xf_2(x)dx + \dfrac{1}{3} \end{pmatrix} \tag{5.9}$$

It is obvious that if functions $f_1(x)$, $xf_1(x)$, $f_2(x)$, and $xf_2(x)$ have antiderivatives, we could produce the exact closed-form formulas for $\nabla Q(\boldsymbol{t})$. However, when the antiderivative of the function, represented by $\eta(x)$, is difficult or impossible to be obtained, we have to resort to numerical integration to produce an approximate value of $\nabla Q(\boldsymbol{t})$. In this chapter, we use

$$\int_L^U \eta(x)dx \approx \frac{U-L}{N}\left(\frac{\eta(L)}{2} + \sum_{k=1}^{N-1}\eta\left(L + k\frac{U-L}{N}\right) + \frac{\eta(U)}{2}\right) \tag{5.10}$$

where L and U are the lower and upper bounds of the integral, N is the number of the subintervals that divide the integral range; the higher the value of N the better the resulted approximation.

5.3 Approximation of Fuzzy Set by IT2 TFS

In this section we intend to capture the major characteristics of the fuzzy set by an IT2 TFS. We anticipate that the properly formed FOU of $\tilde{A}(x)$ could capture the main characteristics of $A(x)$, illustrated in Figure 5.4.

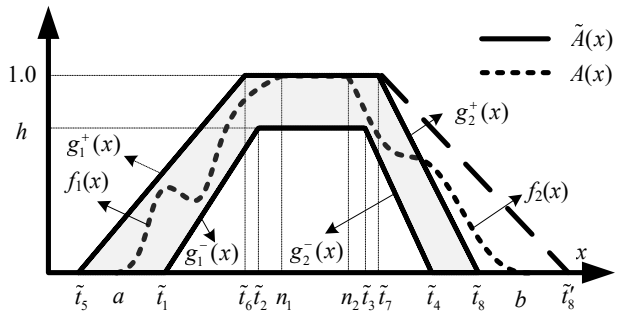


Figure 5.4. Approximation of fuzzy set by IT2 TFS.

Intuitively, we hope that the formed FOU (grey area) could "cover" a larger portion of the membership function $A(x)$, but at the same time it has to be as specific as possible. Since essentially FOU is an information granule, the principle of justifiable granularity is behind the formulation of the proposed optimization problem, where the criteria of *coverage* and *specificity* are used to model the expectation mentioned above. We use the following expressions to reflect the essence of the *coverage* and *specificity* concepts for the FOU.

$$coverage_{\tilde{A}} = \int_{\Omega} A(x)dx \Big/ \int_{X} A(x)dx \tag{5.11}$$

$$specificity_{\tilde{A}} = (S - S_{\text{FOU}})/S \tag{5.12}$$

where $\Omega = \{x | A^-(x) \le A(x) \le A^+(x)\}$ is a set on which $A(x)$ is covered by $\tilde{A}(x)$, by using $A(x)$ instead of the number valued one, we anticipate that higher values of $A(x)$ have the priority over the lower ones to be covered; $\int_X A(x)dx$ is a factor to normalize the *coverage*

to [0, 1], as expected $coverage_{\tilde{A}} = 1$ when $A(x)$ is entirely contained in $\tilde{A}(x)$; S_{FOU} is the area of FOU, S is the area of rectangle with length equals the range of X and width equals to one, which is the maximum area the FOU could cover.

Let us note that *coverage* is used to measure how much experimental evidence (data) has been captured by the information granule. Obviously, the larger the *coverage*, the better support from the evidence. *Specificity*, on the other hand, supports a sound semantics of the formed information granule. The higher the *specificity*, the better interpretability of the granule. A more specific information granule helps us grasp the essence of a concept. However, it is apparent that these two criteria are in conflict: increasing the *coverage* is realized at the expense of *specificity*, which becomes lower. By maximizing the product, a sound compromise between the two criteria could be obtained. Hence, we formulate the objective function as follows,

$$\max \quad J(\tilde{\boldsymbol{t}}) = coverage_{\tilde{A}} \cdot specificity_{\tilde{A}} \tag{5.13}$$

where $\tilde{\boldsymbol{t}}$ is the vector of the parameters of the IT2 TFS.

We further add the constraints as

$$\tilde{t}_1 \le \tilde{t}_2 \le \tilde{t}_3 \le \tilde{t}_4, \tilde{t}_5 \le \tilde{t}_6 \le \tilde{t}_7 \le \tilde{t}_8, 0 \le h \le 1 \tag{5.14}$$

$$\begin{aligned} &\tilde{t}_1 \ge t_1, (t_2 + t_3)/2 \ge \tilde{t}_2 \ge t_2, (t_2 + t_3)/2 \le \tilde{t}_3 \le t_3, \tilde{t}_4 \le t_4, \\ &\tilde{t}_5 \le t_1, \tilde{t}_6 \le t_2, \tilde{t}_7 \ge t_3, \tilde{t}_8 \ge t_4 \end{aligned} \tag{5.15}$$

where (5.15) guarantees the skeleton be covered by FOU of the IT2 TFS and prevents $A^{-}(x)$ from inclining greatly to one side of the skeleton. Note that derivative relations among parameters can be obtained from (5.15), e.g., $\tilde{t}_1 \ge t_1$ and $\tilde{t}_5 \le t_1$ entails $\tilde{t}_1 \ge \tilde{t}_5$. They together with (5.14) make $\tilde{\boldsymbol{t}}$ a valid IT2 TFS.

Obviously, parameters (exclude h) in constraints of the strategy show a high degree of dependence, which complicates the optimization process. We come up with the idea of

transformation to replace the original optimization model with a new one where only independent and bounded variables are needed. Suppose $\boldsymbol{e}_i$, $i = 1, 2,\ldots, 8$, is a non-negative number, by using a series of transformation steps we guarantee the constraints are preserved.

Suppose we have $\boldsymbol{e} = (e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, h)^{\mathrm{T}}$, a value of $\boldsymbol{e}$ determines that of $\tilde{\boldsymbol{t}}$ through the transformation. By adjusting $\boldsymbol{e}$, we minimize the objective function in (5.13). The new optimization model is given as

$$\max \quad J(\boldsymbol{e}) = coverage_{\tilde{A}} \cdot specificity_{\tilde{A}} \tag{5.16}$$

$$\text{s.t.} \quad 0 \le e_i \le (b-a)/2, i = 1, 2, ..., 8; 0 \le h \le 1 \tag{5.17}$$

where a and b are the left- and right-hand side bounds of the support of $A(x)$.

5.4 Experimental Studies

The proposed type-1 and type-2 approximation algorithms are comprehensively validated in terms of their effectiveness and stability on both synthetic membership functions and those generated from the FCM algorithm.

5.4.1 Synthetic Data

Here we consider the simple fuzzy set with half (separated by the single-valued core) of its membership function $A(x)$ as the parabolic, square root, or Gaussian (with finite support) membership function. Although we only consider three simple functions, since both right- and left-hand sides of $A(x)$ have three candidate types, the fuzzy set exhibits a high level of flexibility (i.e., we could have 3×3 = 9 different shapes). The formulas governing these membership functions are as follows.

1) Parabolic function: $f(x) = 1 - \left((x-n)/\sigma\right)^2$
2) Square root function: $f(x) = 1 - \sqrt{|(x-n)/\sigma|}$

3) Gaussian function: $f(x)=\exp\left(-\left((x-n)/\sigma\right)^2\right)$

where n is the modal value of the membership function, and σ is used to control the spread of the membership function. By assigning certain types of membership function given above to the left- and right-hand sides of $A(x)$, we build up a fuzzy set. We show the built nine fuzzy sets in Table 5.1. The universe of discourse is given as X = [0, 10]. We carry out the two approximation methods to determine the TFS and IT2 TFS representations of these fuzzy sets. To evaluate the stability of these algorithms, we implement the 10-fold experiment.

Table 5.1. Summary of nine fuzzy sets.

Fuzzy set #	Left- and right-hand sides of $A(x)$	$f_1(x)$		$f_2(x)$	
	$[f_1(x), f_2(x)]$	n	σ	n	σ
1	[Parabolic, Parabolic]	4	3	4	4
2	[Parabolic, Root]	4	3	4	4
3	[Parabolic, Gaussian]	4	3	4	1.5
4	[Root, Root]	5	-3	5	4
5	[Root, Parabolic]	5	-3	5	4
6	[Root, Gaussian]	5	-3	5	1.5
7	[Gaussian, Gaussian]	5	2	5	1.5
8	[Gaussian, Parabolic]	5	2	5	4
9	[Gaussian, Root]	5	2	5	4

5.4.1.1 Type-1 Approximation Method

The detailed steps for the type-1 approximation method are given as follows.

Step 1: Initialization. Instead of randomly generating four points positioned between bounds a and b of $A(x)$, we set initial values of the four parameters of the TFS as $t_1^{(0)}=a$, $t_2^{(0)}=a+(n-a)/2$, $t_3^{(0)}=n+(b-n)/2$, $t_4^{(0)}=b$. Initial value of the vector of Lagrange

multipliers is set as $\boldsymbol{\mu} = (1,1,1,1,1,1)^{\mathrm{T}} \cdot (b-a)$, the iteration step sizes of $\alpha^{(0)}$ and $\beta^{(0)}$ are set as (b–a)/100. Initializing these parameters by considering information of the core and bounds of support of the fuzzy set is experimentally beneficial. Besides, the threshold ε and the maximum iteration number I used in the step of termination condition checking are set as 1e-5 and 5000, respectively; and the small positive number ξ in the step of parameters update is set as 1e-5.

Table 5.2. Approximated TFSs of synthetic fuzzy sets.

Fuzzy set #	t_1	t_2	t_3	t_4
1	1.00 ± 0.00	3.00 ± 0.00	5.33 ± 0.00	8.00 ± 0.00
2	1.00 ± 0.00	3.00 ± 0.00	4.00 ± 0.00	6.78 ± 0.00
3	1.00 ± 0.00	3.00 ± 0.00	4.23 ± 0.00	6.37 ± 0.00
4	2.92 ± 0.00	5.00 ± 0.00	5.00 ± 0.00	7.78 ± 0.00
5	2.92 ± 0.00	5.00 ± 0.00	6.33 ± 0.00	9.00 ± 0.00
6	2.92 ± 0.00	5.00 ± 0.00	5.23 ± 0.00	7.37 ± 0.00
7	1.84 ± 0.00	4.70 ± 0.00	5.23 ± 0.00	7.37 ± 0.00
8	1.84 ± 0.00	4.70 ± 0.00	6.33 ± 0.00	9.00 ± 0.00
9	1.84 ± 0.00	4.70 ± 0.00	5.00 ± 0.00	7.78 ± 0.00

Step 2: Gradient Determination. With the given membership function $A(x)$ composed of functions $f_1(x)$ and $f_2(x)$, the exact formulas for $\nabla Q(\boldsymbol{t})$ could be derived based on (5.9). For clarity, we avoid presenting the exact $\nabla Q(\boldsymbol{t})$ for the nine fuzzy sets here.

Step 3: Parameters update. All the information needed for (5.6) and (5.7) is well formed, hence the new iteration values of $\boldsymbol{t}^{(k+1)}$ and $\boldsymbol{\mu}^{(k+1)}$ are consequently obtained.

Step 4: Termination condition check. Here two criteria are used: (a) $\max\{\boldsymbol{t}^{(\mathrm{k}+1)} - \boldsymbol{t}^{(\mathrm{k})}\} < \varepsilon$, and $\max\{\cdot\}$ is the operator to find the maximal entry of the vector; and (b) I = 5000.

If either of these criteria is satisfied, the iteration process is stopped, otherwise go to *Step 3*.

We list the mean and standard deviation values for each parameter of the approximated TFS for each fuzzy set in Table 5.2. Overall, the proposed type-1 algorithm produces a sound approximation where the main characteristics of the original fuzzy sets have been retained. The nearly zero values of the standard deviation demonstrate a high degree of stability of this algorithm.

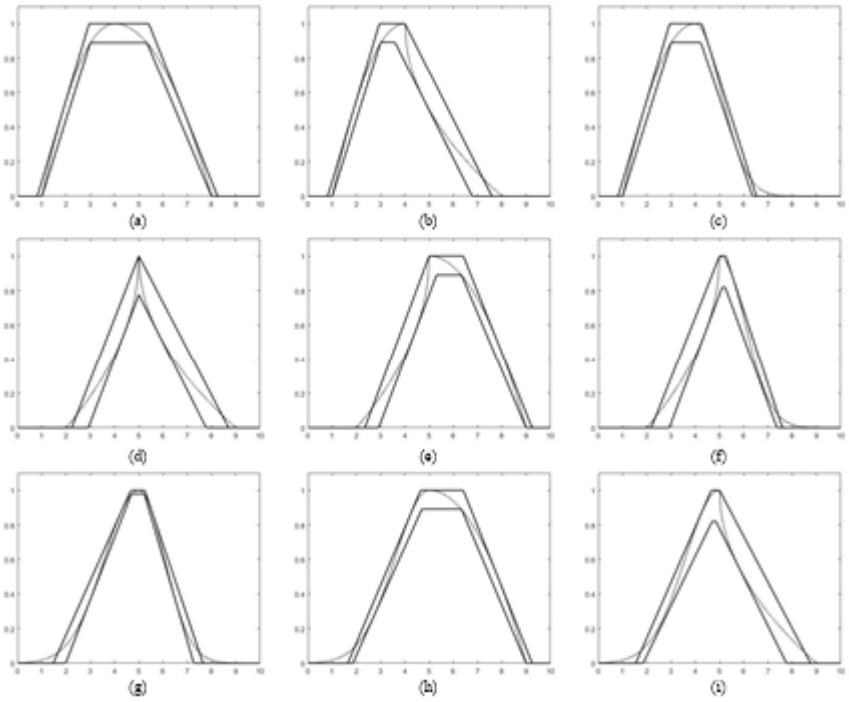


Figure 5.5. Approximated IT2 TFSs: (a) to (i) correspond to the nine fuzzy sets.

5.4.1.2 Type-2 Approximation Method

The detailed steps for the IT2 TFS-based approximation method are given as follows.

Step 1: Initialization. Parameters such as swarm size M, inertia coefficient w, cognitive constant c_1, social constant c_2, and the number of iterations I are critical to the performance of the algorithm. As illustrated in [24], when $c_1 = c_2 = 2$ PSO produces good performance, so the same setting for c_1 and c_2 is maintained across all experiments. Besides, we set $w =$

0.1, $M = 500$, and $I = 500$ to yield a good performance. The initial population, personal and global best locations could be generated accordingly.

Step 2: Parameters update. For each of the 500 particles, (2.35), (2.36), and the transformation serving in the fitness function evaluation are used to update the location and velocity. In the sequel, we find the global best location.

Step 3: Termination condition check. When the maximum iteration number I is reached, terminate the program and return the IT2 TFS obtained for the selected strategy.

The formed IT2 TFSs are shown in Figure 5.5.

5.4.2 FCM-Constructed Fuzzy Sets

In what follows, let us consider the membership functions derived from the FCM algorithm on a data set X in d-dimensional feature space. By projecting on the individual coordinates for each obtained prototype we form fuzzy sets defined in the one-dimensional space. We arrange the obtained c one-dimensional prototypes in an increasing order, i.e., $v_1 \le v_2, \ldots, \le v_c$. Then the membership value of x is expressed as

$$A_i'(x) = 1 \Big/ \sum_{j=1}^{c} \left(\frac{x - v_i}{x - v_j} \right)^{2/(m-1)}, i = 1, 2, \ldots, c. \tag{5.18}$$

where m is the fuzzification coefficient controlling the shape of the formed fuzzy set.

Here, preprocessing of fuzzy sets derived from FCM should be taken before any approximation method is applied, an example is given to show the reason. Suppose we obtain three prototypes as $v_1 = 1$, $v_2 = 5$, and $v_3 = 9$, when $m = 2$ and $c = 3$. Three membership functions are obtained from (5.18). Taking $A_1'(x)$ for example, shown in Figure 5.6 (a), the rippling effect is observed between v_2 and v_3; when x departures from v_2 and v_3 towards negative and positive infinite respectively, membership value converges to $1/c$. These observations greatly reduce the interpretability of the formed fuzzy sets. We preprocess the membership functions by (5.19)-(5.21) to avoid these issues, where the new formed fuzzy sets $A_1(x)$ and $A_c(x)$ become the left- and right-hand side truncated fuzzy sets, while $A_i(x)$,

$i = 2, 3, \ldots, c–1$, is the general fuzzy set; x^L and x^U are the lower and upper bounds of the universe of discourse X. Preprocessed fuzzy sets are shown in Figure 5.6 (b).

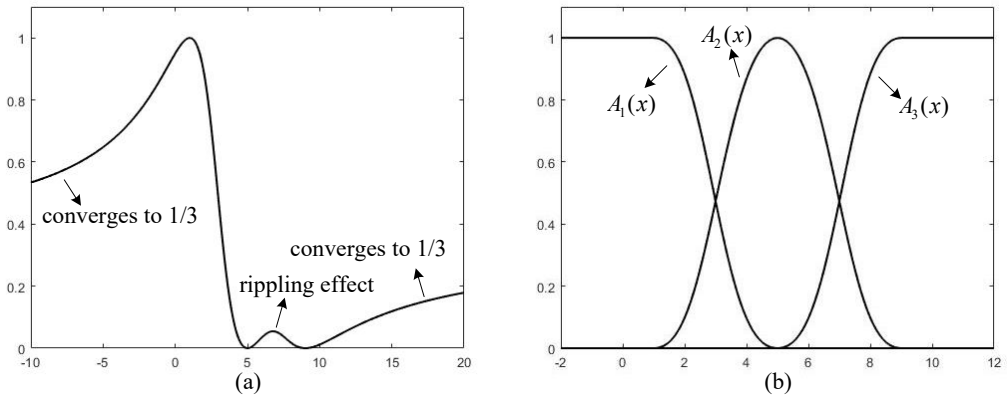


Figure 5.6. (a) Rippling effect and convergence; and (b) preprocessed membership functions.

$$A_1(x) = \begin{cases} 1, x \in [x^L, v_1] \\ 1 \Big/ \left[1 + \sum_{j=2}^{c} \left(\frac{x - v_1}{x - v_j} \right)^{2/(m-1)} \right], x \in (v_1, v_2] \\ 0, \text{otherwise} \end{cases} \tag{5.19}$$

$$A_c(x) = \begin{cases} 1, x \in [v_c, x^U] \\ 1 \Big/ \left[1 + \sum_{j=1}^{c-1} \left(\frac{x - v_c}{x - v_j} \right)^{2/(m-1)} \right], x \in [v_{c-1}, v_c) \\ 0, \text{otherwise} \end{cases} \tag{5.20}$$

$$A_i(x) = \begin{cases} 1 \Big/ \left[1 + \sum_{j=1, j \neq i}^{c} \left(\frac{x - v_i}{x - v_j} \right)^{2/(m-1)} \right], x \in [v_{i-1}, v_{i+1}] \\ 0, \text{otherwise} \end{cases} \tag{5.21}$$

With this preprocessing in mind, we approximate the fuzzy sets derived from the FCM performed on a real-world data set, Banknote (1372 instances and 4 features, URL: https://archive.ics.uci.edu/ml/datasets/banknote+authentication). Four features (F_1 variance, F_2 skewness, F_3 kurtosis of the Wavelet transformed image, F_4 entropy of imagine) have been

adopted. For illustration, we set the cluster number as $c = 4$, m is assigned with 2 or 3, respectively such that fuzzy sets with both regular and more complex shapes are explored. Since the performance of all three algorithms are stable, instead of the 10-fold experiment only one-fold is implemented. The skeleton-based type-2 method is selected due to its better performance in terms of stability. Besides, the closed-form of the antiderivative of the membership function is difficult to obtain considering the different values of m, hence the numerical integration in (5.10) is used to obtain the gradient vector in (5.9). For simplicity, we only report part of the formed TFSs and IT2 TFSs in Figure 5.7. Generally, both of the algorithms perform well on the formed fuzzy sets.

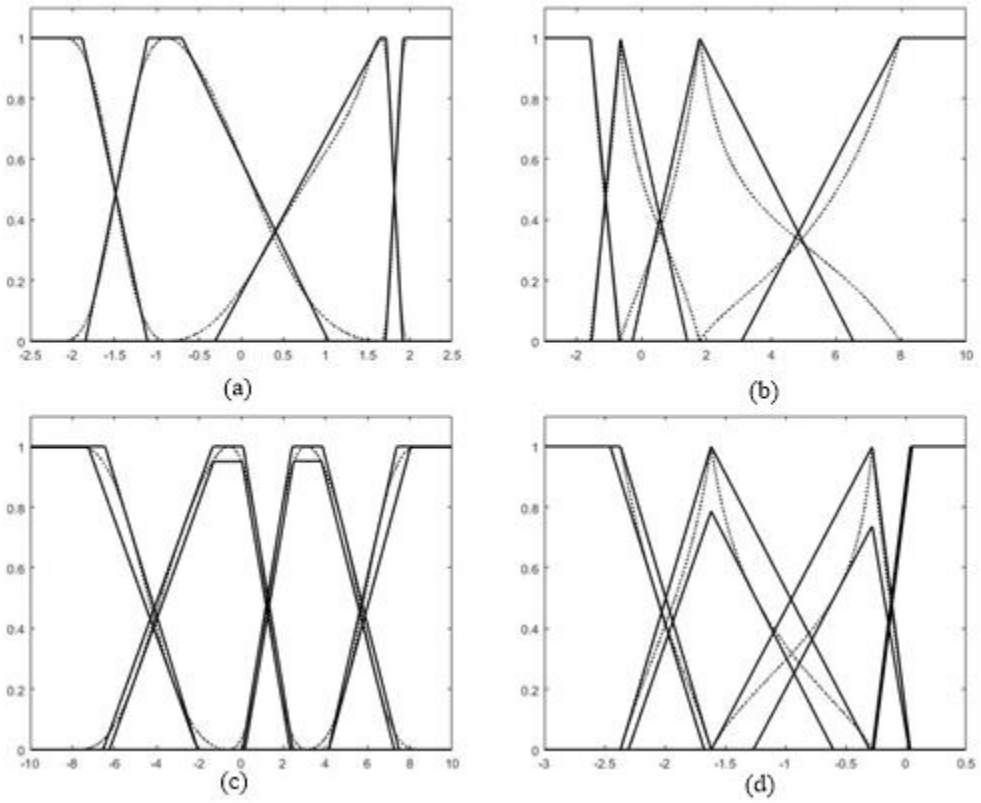


Figure 5.7. Approximated TFSs (a) for F1, $m = 2$; (b) for F3, $m = 3$; and IT2 TFSs (c) for F2, $m = 2$; (d) for F4, $m = 3$.

5.5 Summary

Trapezoidal fuzzy set (TFS) has been treated as a sound structure to capture the main characteristics of the general fuzzy set, such that further analysis and modeling based on general fuzzy sets could be simplified. The proposed gradient-based approximation method efficiently finds the optimal TFS by minimizing its distance to the fuzzy set such that the core, the major part of the support, and the major shape of the original fuzzy set could be preserved. By increasing the number of parameters from four to nine, we extend the TFS to a higher type, i.e., interval type-2 trapezoidal fuzzy set (IT2 TFS), to grasp more features of the original fuzzy set. With the principle of justifiable granularity, information granules built in this manner exhibit good coverage of the data without sacrificing sound interpretation capabilities.

Both type-1 and type-2 approximation algorithms have been comprehensively validated to show their effectiveness, robustness, and efficiency in approximating fuzzy sets. Since the heterogenous information granules (i.e., a fuzzy set of any shape) could be approximated by either the TFS or IT2 TFS, we finally obtain a data set that is composed of homogenous information granules. Now, the homogenous granular data clustering and evaluation method proposed in the Chapter 4 could be directly applied to solve the clustering problem for heterogenous granular data.

Chapter 6 Hyperplane Division-Based Clustering Method for Big Data

In the previous chapter, we have studied clustering algorithms for distributed or granular data. In this chapter, we focus on clustering big data characterized by a large sample size. As a matter of fact, there are many methods proposed to handle the big data clustering problem. However, we would see later in the literature review section that the proposed methods could be ineffective sometimes. And the clustering requirements (especially when many clusters are pursued) has not been seriously considered in the current research. With these two considerations, the objectives of this chapter are formulated as follows: (a) Propose a new method (i.e., the hyperplane division method) to form subsets of data such that the data subspaces, supported or spanned by the data points in each subset, do not overlap each other. (b) Design customized strategies to meet different purposes or requirements of the clustering task. When a small number of clusters is considered, we obtain representatives of the entire data based on representatives of prototypes from the subsets; otherwise, the aggregated prototypes all together are used as representatives of the entire data.

6.1 Clustering of Big Data with Large Sample Size: A Focused Review

To determine structures of such data, two major categories of methods are encountered in literature: (a) representative-based methods and (b) entire data-based methods. For methods falling in the former category, a small part of the data set [88]–[91] or a transformed data set with a far lower sample size [92] is used as the representative of the original data. By applying fuzzy clustering algorithms to this representative data set, the data structure (represented as the partition matrix or a series of prototypes) is obtained, which is further treated as that of the original data. To produce a part of the data, either

random sampling methods [88], [89] or progressive sampling methods [90], [91] are commonly used; and to form the transformed data, the bit-reduced method [92] partitions data into many bins, each bin is then merged into a single data point by averaging all the data points located in this bin.

Although representative-based methods make it possible to cluster data with large sample size, their clustering performance largely depends on the effectiveness of the sampling strategy (i.e., how representative the formed samples are). In fact, the obtained data structure formed by these methods could be easily biased (e.g., results coming from two sampling trials completed with a certain sampling strategy could be quite distinct). To handle this problem, entire data-based methods take into consideration all data points in the clustering process. Either incremental methods [93]–[97] or distributed methods [93]–[95] are used to partition the entire data into subsets of data. In incremental methods, clustering algorithms are applied to the subsets sequentially, the obtained data structure (prototypes) of the current subset is used as initial structure for the following subset. In distributed methods, clustering is performed on the individual subsets simultaneously, a final clustering on the aggregated prototypes is used to get prototypes of the entire data. These distributed algorithms could be parallelized by big data analysis frameworks like Apache Spark and Hadoop, one could refer to [98]–[102] for examples of parallelized fuzzy clustering algorithms.

By sacrificing some computational efficiency, entire data-based methods normally produce better and more stable cluster quality. However, they are also faced with several limitations. We point out two facts which are normally neglected in literature: (a) similarity existing among a large portion of the obtained subsets, which may influence cluster quality; and (b) impact of the cluster number on the efficiency of clustering algorithms. We illustrate these facts as follows.

In the entire data-based methods, two major methods are used to obtain subsets from the entire data set. (a) Sampling. Either random or progressive sampling is used to sample the data indexes ranging from 1 to N (the sample size). Since an index corresponds to a data point, the subset of data is formed as per the selected indexes. (b) Data division based on the so-called context variable. This kind of methods is encountered in the Conditional Fuzzy C-Means (CFCM) algorithms [103]–[105], where the output variable is used as the context variable to guide the division of the entire data. For example, observations of the context variable could be divided into different groups, then those data whose values of context variable belong to the same group form a subset of the entire data. Although the aforementioned methods have been widely used in tasks of clustering big data, it happens that the obtained subsets are quite similar to each other. We illustrate this effect by performing the random sampling and context variable-based division methods, respectively on two real-world data (only the first two features of each data set are considered so that the resulted subsets could be easily visualized; the output of data set CASP is used as the context variable in CFCM algorithm) where data are usually clustered together. The obtained subsets from the two methods are respectively shown in Figure 6.1 (a) and (b). Obviously, for either methods, subsets from the specific data set are quite similar to each other. For instance, in Figure 6.1 (a) nine subsets of data set HTRU obtained from the random sampling method show similar distributions across the two-dimensional feature space.

With regard to the second aspect, the design of clustering algorithms given full consideration to the number of clusters has not been well articulated and elaborated on. As a matter of fact, most of the time clustering algorithms assume that a relatively small number of clusters is considered. A question arises when we need to cluster the data into many clusters, say thousands of clusters. For real-world data, such as images or web pages, it is common that they could be grouped into a large number of classes (e.g., different

topics). As another example, we may need to form prototypes as the abstraction of big data (not necessarily aim at finding the intrinsic or optimal clusters contained in data), based on which some other information processing systems (say, fuzzy rule-based systems) could be constructed. Here, a large number of prototypes are beneficial to improve the accuracy of the constructed system (assuming that the accuracy criterion is the major design objective). Considering these scenarios and formulated requirements, even clustering of subsets of the entire data makes the available clustering methods computationally demanding.

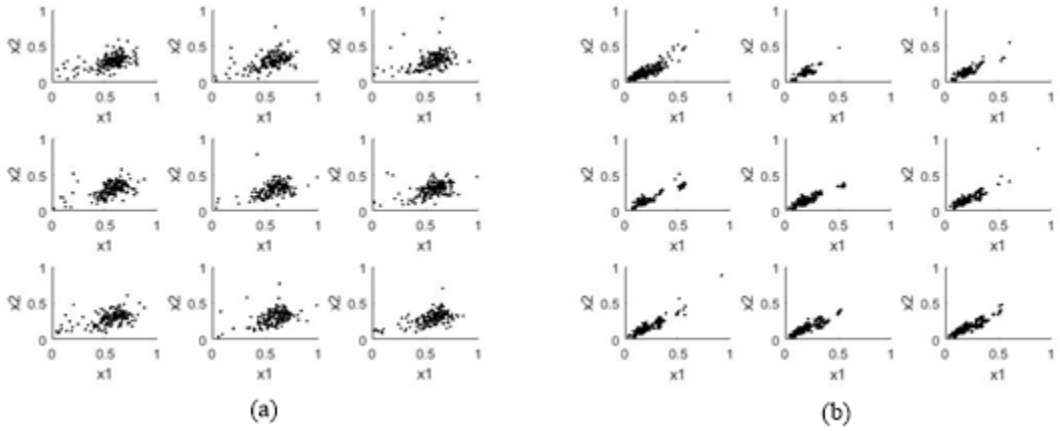


Figure 6.1. Similar subsets derived from each of the two entire data-based methods. (a) Nine subsets of data set HTRU when random sampling rate is set as $g = 0.01$; (b) Nine subsets of data set CASP when values of output variable are split into $P = 100$ intervals with each interval having the same number of data.

6.2 Data Set Division with Hyperplanes

Suppose a data set is denoted by $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where the kth data point $\boldsymbol{x}_k$ is a vector in a n-dimensional space $\boldsymbol{R}^n$ spanned over n features $x_1, x_2, \ldots, x_n$ (We use the non-italic and non-bold face lowercase letter to represent a feature. Also note that data sets with categorical or nominal features are not the focus of this study, we only consider those with numeric features). To obtain P disjoint subsets $X[1], X[2], \ldots, X[P]$ of X such that each of

them only focuses on a local region of the entire data space, we construct a collection of hyperplanes to divide the data set.

Taking an example of a one-dimensional data set, we sort the data in an increasing order. To construct the disjoint subsets, several one-dimensional thresholds are selected to divide the data. Specifically, to obtain P subsets where each of them consists of M data points (i.e., $M = N/P$), the 1st to the Mth ordered data could be obtained as the 1st subset by treating the M+1th point as a threshold; the M+1th to the 2Mth data is treated as the 2nd subset with the 2M+1th point as another threshold, etc.

This idea can be extended to high-dimensional data by constructing a one-dimensional index variable y for the entire data. These index thresholds could be determined by equally splitting the data (i.e., to obtain subsets with the same number of points). The value of index for a given data $\boldsymbol{x}_k$ ($k = 1, 2,\ldots, N$) is described as follows

$$y_k = \varphi(x_{k1}, x_{k2}, ..., x_{kn}) \tag{6.1}$$

where x_{ki} is the value of the ith feature x_i ($i = 1, 2,\ldots, n$).

Assume that the N index values have been ordered in an increasing order, namely $o_1 \leq o_2 \leq \ldots \leq o_N$. As before, the thresholds of these index values are formed so that in each subset there is the same number of data. Note that sorting values of the one-dimensional index of a big data is generally easier and more efficient than clustering this data set.

There could be many forms of the function φ, as long as we guarantee that data points could be divided into disjoint subsets by the ranging values of the designed function. For instance, we could take φ as the weighted sum or the positively weighted squared sum of all the features. For simplicity, in this study we set φ as

$$y_k = \sum_{i=1}^{n} x_{ki} \tag{6.2}$$

For illustrative purposes, let us consider a two-dimensional data. Suppose we have eight data $\boldsymbol{x}_1 = [1.0, 1.3]$, $\boldsymbol{x}_2 = [3.0, 0.9]$, $\boldsymbol{x}_3 = [2.5, 3.0]$, $\boldsymbol{x}_4 = [4.0, 4.0]$, $\boldsymbol{x}_5 = [-1.0, -1.0]$, $\boldsymbol{x}_6$

= [-4.0, -2.0], $\boldsymbol{x}_7$ = [-2.0, -2.5], $\boldsymbol{x}_8$ = [6.0, 1.0]; they are shown as the asterisks in Figure 6.2. Based on (6.2), we calculate the index value for each point as $y_1 = 2.3$, $y_2 = 3.9$, $y_3 = 5.5$, $y_4 = 8.0$, $y_5 = -2.0$, $y_6 = -6.0$, $y_7 = -4.5$, $y_8 = 6.5$. Hence, we have the order in the form $o_1=y_6 \le o_2=y_7 \le o_3=y_5 \le o_4=y_1 \le o_5=y_2 \le o_6=y_3 \le o_7=y_8 \le o_8=y_4$. If we intend to divide these data into two subsets with four points in each part, we may choose $o_5 = y_2$ as the threshold, thus we form two subsets as $X[1] = \{\boldsymbol{x}_1, \boldsymbol{x}_5, \boldsymbol{x}_6, \boldsymbol{x}_7\}$ and $X[2] = \{\boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_8\}$.

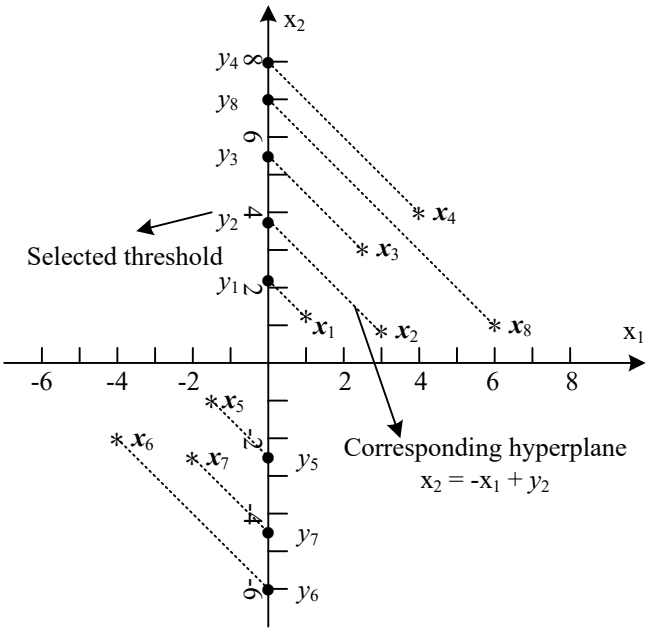


Figure 6.2. Development of subsets of data.

With the real-world data mentioned in the previous section, we obtain the subsets derived from the hyperplane division method and show them in Figure 6.3 when $P = 9$. Obviously, comparing to the subsets in Figure 6.1 (a) and (b), now each subset only focuses on a much reduced (confined) space, and the superposition of all subsets constitutes the entire data. Interestingly, note that the formed index in (6.2) could be regarded as a special case of the context variable used in the CFCM algorithm. Here instead of using the output variable we use the aggregated features as the context variable.

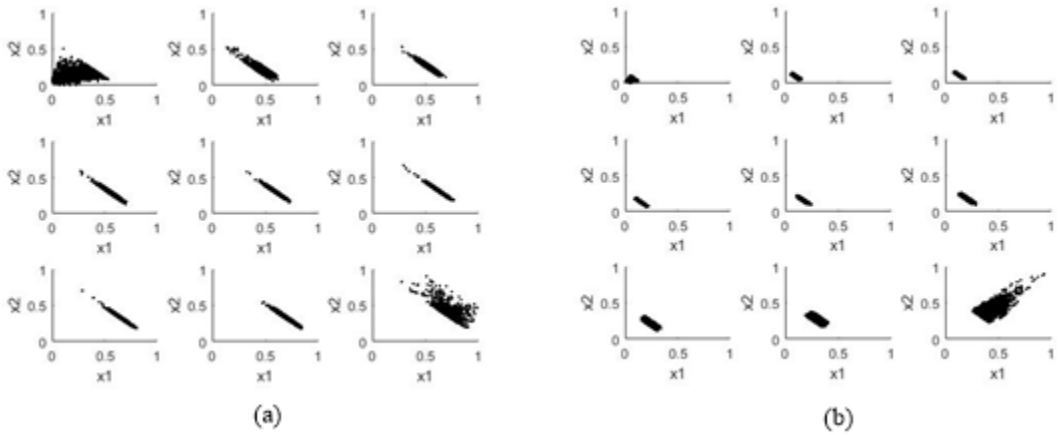


Figure 6.3. Subsets obtained for (a) HTRU and (b) CASP with the hyperplane division method.

6.3 Two Development Strategies for Big Data Clustering

Since the FCM clustering algorithm is widely regarded as one of the most important representatives of the fuzzy clustering algorithms, we use it to illustrate the proposed strategies for clustering big data.

Strategy A: The purpose in this scenario is to cluster the entire data into many clusters. For example, thousands of topics or themes may exist in application areas such as text and image mining. Or the accuracy criterion is more preferred in some information processing systems which are built based on the constructed clusters; hence, with more clusters to abstract the data, accuracy of these systems could be greatly improved. However, clustering the entire data into thousands of clusters is not always tolerable considering the induced high time complexity (detailed analysis is provided in the next section). To solve this problem, we first split the entire data into P subsets by the hyperplane division method. Second, we apply FCM to clustering each obtained subset into a much smaller number of clusters, say d cluster. Without any doubt, there could be some optimal number of clusters in each obtained subset, just similar to the case where an optimal value of c could be obtained according to a selected cluster validity index. This is feasible but more

computational effort is incurred. For simplicity, we may simply specify $d = c/P$ where c is the number of clusters for the entire data set, and P is the number of subsets. In other words, we intend to find the d (instead of the optimal number of) representatives in each subset. Hence, to sum up although c is very large, by increasing the value of P, the cluster number d for each subset remains relatively low. With the FCM algorithm, d prototypes (representatives) are formed for each subset, and these Pd prototypes considered together are used to represent the structure of the entire data. The strategy is illustrated in Figure 6.4 (a).

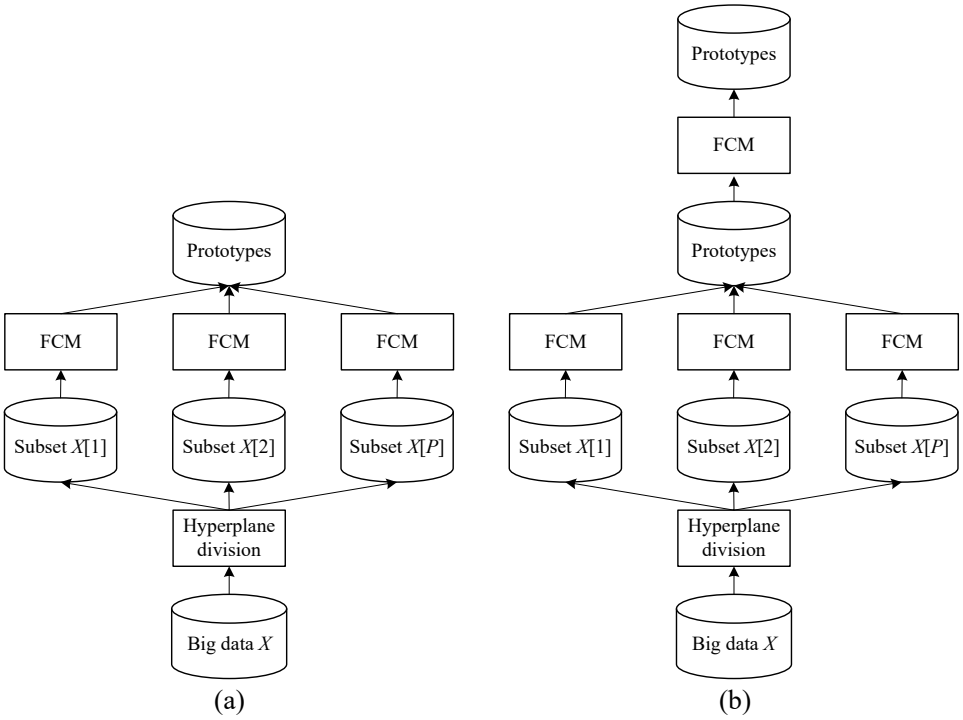


Figure 6.4. Big data clustering: (a) Strategy A; and (b) Strategy B.

Strategy B: The purpose in this scenario is to cluster the entire data into a relatively small number of clusters, which is a focus of most of the current study on big data clustering. This purpose exists because people may only need to understand the data at a very high level of abstraction. Or instead of the accuracy the criterion of interpretability of some information systems is of interest. Although the number of clusters c of the entire

data is small, the large sample size makes the computing demanding. We can complete the clustering task in three steps. The first two steps (obtain and cluster subsets) are the same as the ones used in *Strategy A*, through which we obtain P subsets and d prototypes for each subset, respectively. These Pd prototypes are treated as the representatives of the entire data. Note that now Pd is far smaller than N. At the third step, we perform the FCM algorithm on these prototypes and form c clusters. The obtained c centers of the Pd prototypes represent the structure of the entire data. This proposed strategy is illustrated in Figure 6.4 (b).

6.4 Computational Complexity of Proposed Clustering Strategies

In this section, we complete some theoretical computational complexity analysis of the two proposed strategies and show their advantages over the method used to directly cluster the entire data. Since hyperplane division method is involved in both strategies, plus it is very efficient compared with the clustering algorithm, its computational complexity is not considered in this analysis. The time complexity of the FCM algorithm is $O(Inc^2N)$, where I is the number of iterations, n is the number of features, c is the number of clusters for the entire data, and N is the number of data.

For *Strategy A*, the time complexity of applying FCM to each obtained subset is $O(Ind^2N/P)$, where $d = c/P$ is the number of clusters in each subset and P is the number of subsets. By substituting c/P for d, we get the time complexity to be $O(Inc^2N/P^3)$. Considering that we have P subsets, the time complexity of applying FCM to all the subsets is determined to be $O(PInc^2N/P^3) = O(Inc^2N/P^2)$. Hence, we note that in the asymptotic estimate, the time complexity of *Strategy A* is reduced by P^2 times compared with directly clustering the entire data. For example, by dividing the entire data into $P = 10$ subsets we make the clustering process 100 times faster, which is highly encouraging.

For *Strategy B*, the time complexity is composed of two parts. The first part is derived from clustering each of the P subsets into d clusters, that is $O(PInd^2N/P) = O(Ind^2N)$. For the second part, we further cluster the Pd prototypes into c clusters, which leads to a time complexity of $O(Inc^2Pd)$. To sum up, the time complexity for *Strategy B* is $O(Ind^2N + Inc^2Pd) \approx O(Ind^2N)$ considering that Pd is far lower than N. Note that for *Strategy B*, we do not require the relation $d = c/P$ holds because c is assumed relatively small. However, as long as $d < c$, we can still benefit from the proposed method. For example, when $d/c = 10/50 = 1/5$ the proposed strategy is 25 times faster than the method of direct clustering the entire data, which is still compelling.

6.5 Evaluation of Clustering Algorithms

In this study, three indices are used to evaluate the clustering algorithms: (a) running time, (b) reconstruction criterion [106], and (c) classification error [107].

6.5.1 Reconstruction Criterion

The crux of the reconstruction criterion is that after the structure (prototypes and the partition matrix) of data set is obtained from FCM algorithm, each data point will be reconstructed based on this structure. If the obtained structure is more relevant to the data, then the estimated data reconstructed from this structure should be closer to the original data. By quantifying the difference between the reconstructed data and the original ones, the quality of the formed clusters is measured. With the obtained structure resulting from (2.3) and (2.4), we use the following performance index.

$$\min \quad \sum_{i=1}^{c} u_{ik}^{m} \left\| \hat{\boldsymbol{x}}_k - \boldsymbol{v}_i \right\|^2 \tag{6.3}$$

Here, we require to put the reconstructed data point in such a position that is close to all the found prototypes. By zeroing the gradient of this objective function with respect to $\hat{\boldsymbol{x}}_k$, we get the optimal reconstructed data point as

$$\hat{\boldsymbol{x}}_k = \sum_{i=1}^{c} u_{ik}^m \boldsymbol{v}_i \Big/ \sum_{i=1}^{c} u_{ik}^m \tag{6.4}$$

The difference (i.e., the reconstruction error) between original data X and the reconstructed one $\hat{X}$ is denoted by

$$E = \frac{1}{N} \sum_{k=1}^{N} \left\| \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k \right\| \tag{6.5}$$

6.5.2 Classification Error

Based on the obtained partition matrix, we can form clusters $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, ..., \boldsymbol{\omega}_c$ in the following way:

$$\boldsymbol{\omega}_i = \left\{ \boldsymbol{x}_k \left| u_{ik} > \max_{j=1,2,...,c; j \neq i} u_{jk} \right. \right\} \tag{6.6}$$

With the provided label of each data point, the dominant class (with the largest number of data points) in each formed cluster is obtained in a direct manner. Suppose the number of the dominant class in $\boldsymbol{\omega}_i$ is N_i, then the classification error CE of a certain clustering algorithm is denoted by

$$CE = 1 - \frac{1}{N} \sum_{i=1}^{c} N_i \tag{6.7}$$

6.6 Experimental Studies

In this section, both synthetic and publicly available data sets are used to illustrate the proposed hyperplane division method as well as the two clustering strategies proposed based on this method.

6.6.1 Synthetic Data

We construct a 2D synthetic data which is composed of four Gaussian clusters whose spreads (covariance matrices) $\boldsymbol{\Sigma}_i$ and centers $\boldsymbol{v}_i$ are summarized in Table 6.1, and each

cluster is composed of 200 data points making the sample size to be $N = 800$. The formed data points are shown as the small dots in Figure 6.5 (a), and dashed lines are used to show the axes passing through the origin. As for the setting of the experiment, we divide the data set into $P = 10$ subsets (with $M = 80$ samples located in each subset). For *Strategy A*, the number of clusters d in each subset is set as 6, 9, and 12, then the number of clusters c of the entire data is directly obtained by Pd as 60, 90, and 120, respectively. For *Strategy B*, (remember that there is no such relation as $d = c/P$ between d and c) we fix d at 9 and set c as 10, 15 and 20. The fuzzification coefficient m is set as 1.8, 2.0, and 2.2. Finally, the 10-fold experiment is used to avoid any experimental bias.

Table 6.1. Centers and covariance matrices of four clusters.

Cluster #	Center	Covariance matrix
Cluster 1	$\boldsymbol{v}_1 = [-3, 2]$	$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Cluster 2	$\boldsymbol{v}_2 = [2, 7]$	$\boldsymbol{\Sigma}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$
Cluster 3	$\boldsymbol{v}_3 = [-5, -5]$	$\boldsymbol{\Sigma}_3 = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$
Cluster 4	$\boldsymbol{v}_3 = [5, 0]$	$\boldsymbol{\Sigma}_4 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$

First, for either of the strategies, with the hyperplane division method we split this 2D data into 10 subsets, the formed P-1 = 9 hyperplanes (lines in this case) are shown as the solid oblique lines in Figure 6.5(a). Second, we implement the proposed two clustering strategies on these subsets. Since the clustering results (prototypes) in the 10-fold experiment remain steady with the given values of parameters d, c, and m, we only show

these prototypes from one attempt of the 10-fold experiment. Specifically, for *Strategy A* when $d = 9$, $c = 90$, and $m = 2.0$, nine prototypes are formed for each subset forming 90 prototypes as the structure of the entire data [see the boldface circles in Figure 6.5(a)]. For *Strategy B*, when $d = 9$, $c = 10$, and $m = 2.0$, 90 prototypes [shown as plain circles in Figure 6.5(b)] are first obtained; then by clustering these prototypes into 10 clusters, we get the prototypes [shown as bold circles in Figure 6.5(b)] as the abstraction of the original data. For comparison, we also show the prototypes derived from direct clustering the entire data (the benchmark method) into 90 clusters and 10 clusters, which are respectively shown as boldface circles in Figure 6.6(a) and (b). In general, prototypes from both strategies could summarize the original data points nicely. *Strategy A* tends to offer more even distributed prototypes among the Gaussian clusters than the benchmark method [e.g., too many prototypes in the top-left cluster are found in Figure 6.6(a)]. Distribution of the prototypes from *Strategy B* is also greatly distinct from that of the benchmark method, except for the top cluster the abstraction of other three Gaussian clusters are dissimilar for different methods.

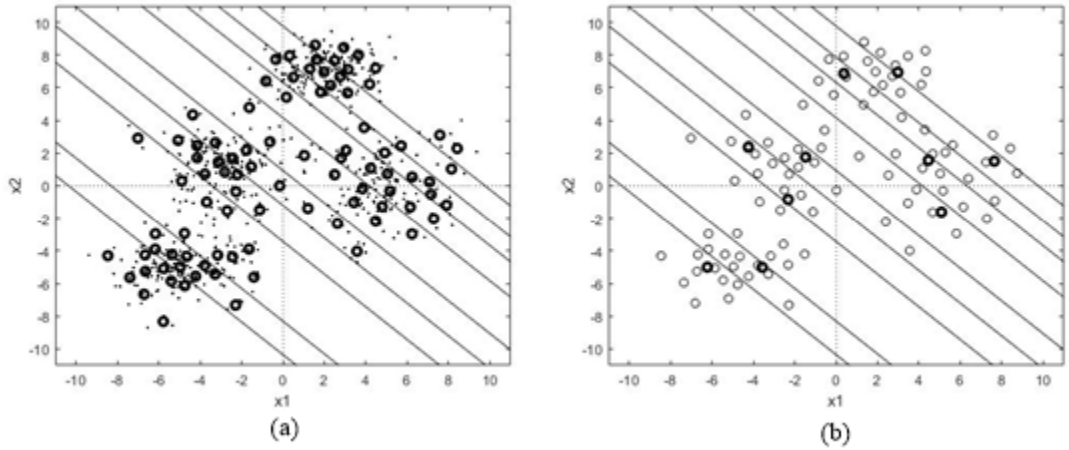


Figure 6.5. Clustering results of the proposed clustering methods: (a) Strategy A, $d = 9$, $c = 90$, $m = 2.0$; and (b) Strategy B, $d = 9$, $c = 10$, $m = 2.0$.

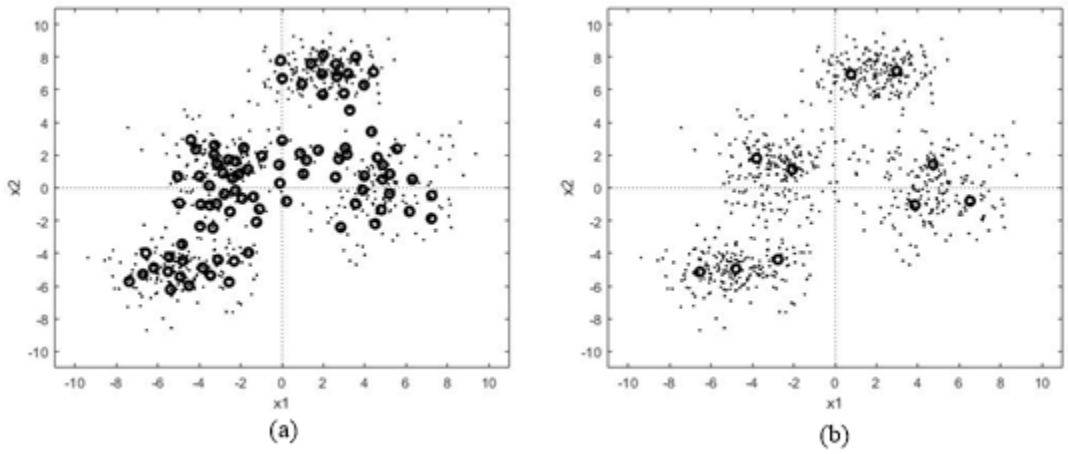


Figure 6.6. Clustering the entire data into (a) $c = 90$ clusters; and (b) $c = 10$ clusters.

As has been shown in Figure 6.1, quite often subsets obtained from sampling-based methods and conventional context variable-based methods are similar to each other. Focusing on the provided synthetic data, in this part, we show the negative impact of similar subsets on the clustering results. The number of the formed subsets is still set as $P = 10$, the number of clusters in each subset is $d = 4$, the fuzzification coefficient is $m = 2.0$. We repeat sampling the entire data 10 times with a sampling rate of $g = 1/P = 0.1$ to form these similar subsets (note that each subset is sampled from the entire data set, and for each subset each data point is sampled without replacement). The obtained $Pd = 40$ prototypes for *Strategy A* (without using the hyperplane division method) are shown in Figure 6.7(a). Obviously, these formed prototypes tend to be clustered with each other compared with those in Figure 6.7(b) derived from the benchmark method. This result is predictable, since we have similar subsets from sampling and each subset is clustered into four clusters (which is the intrinsic cluster number of Data set 1), the formed four centers for each subset are naturally close to each other. With (6.4) and (6.5), the reconstruction error with these prototypes is obtained around 0.32 which is much larger than 0.13 derived from the benchmark method and 0.12 from *Strategy A* with the hyperplane division method. Since

prototypes are clustered, the new centers formed on them based on *Strategy B* (without using the hyperplane division method) are also constrained in a limited space. Although this will not cause major issues when cluster number c is consistent with the intrinsic cluster number (i.e., $c = 4$), for other values of c, say $c = 8$, the clustering results [see Figure 6.7(c)] are greatly distinct from those derived from the benchmark method [see Figure 6.7(d)].

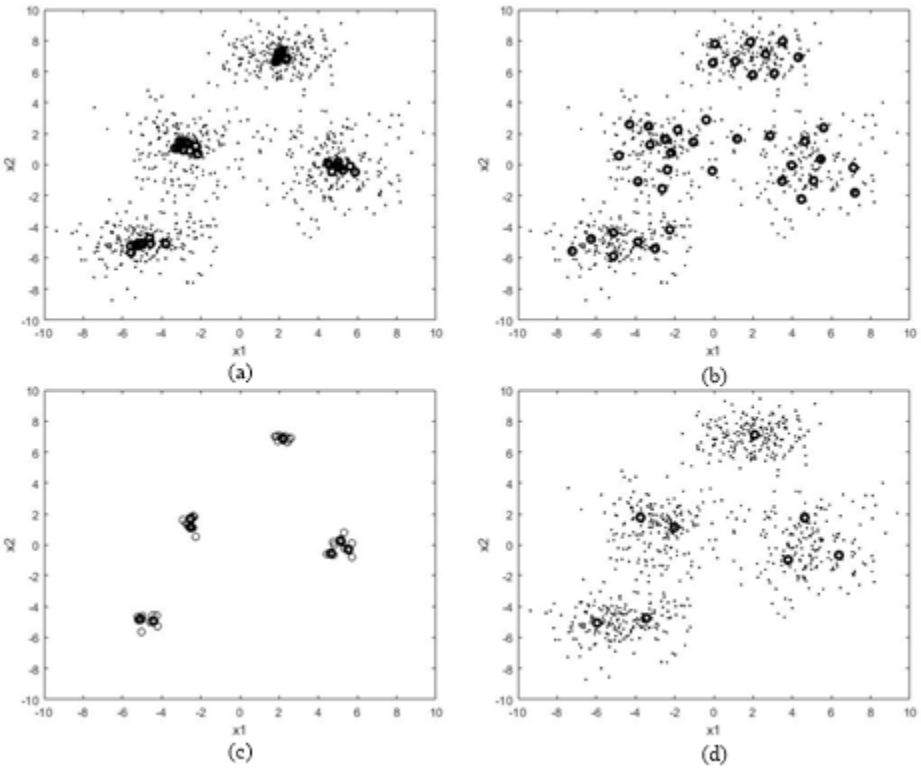


Figure 6.7. Disadvantage of the sampling-based subsets formation method. Prototypes produced by (a) Strategy A without using the hyperplane division method when $c = 40$; (b) the benchmark method when $c = 40$; (c) Strategy B without using the hyperplane division method when $c = 8$; (d) clustering entire data when $c = 8$.

6.6.2 Publicly Available Data

In this part, we comprehensively evaluate the proposed two development strategies (by exploring more combinations of the parameters c, d, and m) on 14 real-world data sets.

6.6.2.1 Experiments for Strategy A

We divide the data into $P = 10$ subsets and range the number of clusters d in each subset from 1 to 20 with a step size of 1; hence, the corresponding cluster number c of the entire data ranges from 10 to 200 with a step size of 10. Fuzzification coefficient m is still set as 1.8, 2.0, and 2.2, respectively. When a certain combination of these parameters is fixed, 10-fold experiment is used to get the performance of the clustering methods in terms of both reconstruction criterion and running time.

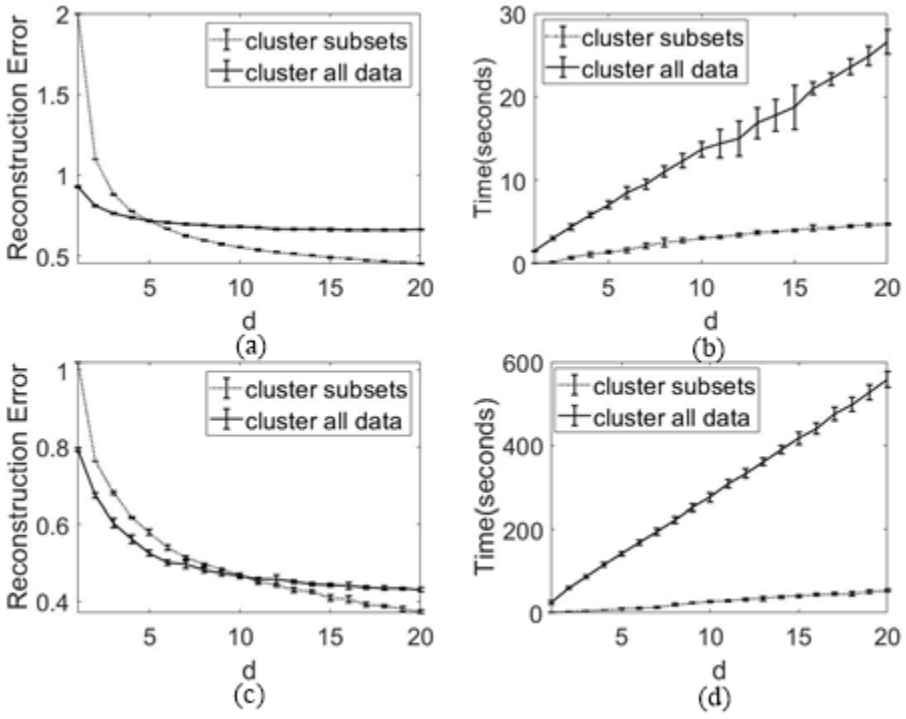


Figure 6.8. Clustering performance of Strategy A and the benchmark method on CCPP: (a) reconstruction error; (b) running time; and on Video: (c) reconstruction error; (d) running time.

We show the detailed clustering performance of *Strategy A* and the benchmark method for data sets CCPP and Video in Figure 6.8 when $m = 2.0$. Both mean and standard deviation (shown as half of the vertical segment) of either the reconstruction error and the running time have been illustrated.

From Figure 6.8(b) and (d), we see that for each data set the mean value of the running time of *Strategy A* is always much less than that of the benchmark method, and the gap of running time between two methods keeps widening with the increasing value of d. Less variation of the running time is also found for *Strategy A*. As for the reconstruction error in Figure 6.8(a) and (c), there exists a point where *Strategy A* starts to obtain better performance ($d = 6$ for CCPP and $d = 10$ for Video) than the benchmark method. Besides, this advantage of *Strategy A* keeps growing with the increasing value of d. This observation is extremely inspiring, because when c is a large number, say $c = 200$, clustering with *Strategy A* not only greatly improves the computational efficiency, but also leads to more effective clustering results.

We further summarize the performance of *Strategy A* for different data sets in Table 6.2. Specifically, column d^* specifies the point where *Strategy A* starts to have lower reconstruction error than the benchmark method. Column *PE* means that how much reconstruction error has been reduced when a certain value of d larger than d^* is chosen (we set $d = 20$), which is defined in (6.8). And similarly, *PT* means that how much time has been reduced when $d = 20$, which is defined in (6.9). Obviously, the larger the values of *PE* and *PT*, the better the performance of *Strategy A*. From Table 6.2, we see that d^* could always be found such that reconstruction error of *Strategy A* is lower. Moreover, we observe that the higher the value of fuzzification coefficient m, the smaller the value of d^*. Besides, except for three cases (shown as boldface numbers in Table 6.2) where moderate improvement of cluster quality is observed, large amount of reduction of the reconstruction

error is obtained with *Strategy A*. As for the efficiency, generally more than 80% of the computational time will be reduced with *Strategy A*.

$$PE = (E_{\text{cluster all data}} - E_{Strategy})/E_{\text{cluster all data}} \tag{6.8}$$

$$PT = (Time_{\text{cluster all data}} - Time_{Strategy})/Time_{\text{cluster all data}} \tag{6.9}$$

Table 6.2. Clustering performance of Strategy A.

	$m = 1.8$			$m = 2.0$			$m = 2.2$		
	d^*	*PE* (%)	*PT* (%)	d^*	*PE* (%)	*PT* (%)	d^*	*PE* (%)	*PT* (%)
CASP	5	15.41	89.98	4	25.64	91.10	2	33.20	89.00
CCPP	9	13.46	83.44	5	31.56	82.23	4	39.12	81.83
Appliance	2	18.44	91.20	1	21.15	92.21	1	25.02	90.87
CBM	14	**0.80**	87.16	6	12.02	89.02	4	29.91	87.16
Video	17	**3.68**	90.14	11	13.20	90.38	4	23.59	89.81
GPU	10	43.91	89.84	8	58.11	89.62	8	93.12	90.70
PM2.5	8	18.81	89.86	5	26.12	90.41	4	26.61	89.41
HTRU	18	**0.50**	86.07	9	12.74	88.07	4	23.37	87.02
Shuttle	4	22.75	90.08	4	32.34	91.72	4	38.45	90.80
AReM	4	10.68	89.48	3	9.85	90.34	2	13.15	89.81
Avila	2	19.71	88.30	1	16.84	89.66	1	19.44	88.01
Letter	1	16.84	89.66	1	12.30	58.15	1	11.40	70.61
MAGIC	3	17.18	87.38	2	17.51	89.61	2	15.04	88.59
MoCap	1	9.79	66.87	1	9.28	68.45	1	8.88	71.28

6.6.2.2 Experiments for Strategy B

In this part, we set $P = 100$, range d from 1 to 10, and control c at a low level from 2 to 50. We illustrate the clustering performance of *Strategy B* and the benchmark method for two data sets in Figure 6.9 when fuzzification coefficient m is fixed at 2.0. Focusing on the reconstruction error, we see that in Figure 6.9(a) and (c), for most values of c *Strategy*

B obtains a much better performance when *d* exceeds a small number (4 for CASP and 5 for PM2.5) compared with the benchmark method. For the computational time, different from the observation in *Strategy A*, *Strategy B* is not always the winner. This is consistent with its theoretical analysis in Section 6.4, i.e., as long as $c > d$ *Strategy B* is more efficient. This observation is clearly identified in Figure 6.9(b) and (d).

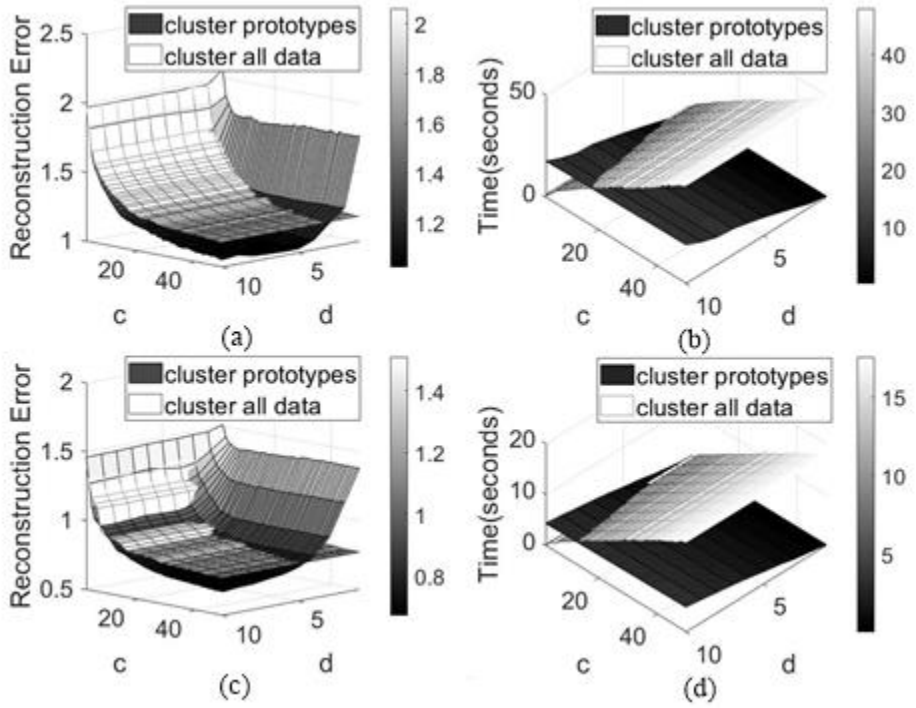


Figure 6.9. Clustering performance of Strategy B and the benchmark method on CASP: (a) reconstruction error; (b) running time; and on PM2.5: (c) reconstruction error; (d) running time.

Table 6.3. Clustering performance of Strategy B.

	m = 1.8				*m* = 2.0				*m* = 2.2			
	c = 20		*c* = 35		*c* = 20		*c* = 35		*c* = 20		*c* = 35	
	PE (%)	*PT* (%)	*PE* (%)	*PT* (%)	*PE* (%)	*PT* (%)	*PE* (%)	*PT* (%)	*PE* (%)	*PT* (%)	*PE* (%)	*PT* (%)
CASP	3.46	32.26	7.21	59.23	5.31	11.18	9.80	44.21	14.35	29.28	15.25	56.75
CCPP	2.02	18.75	1.19	46.53	4.25	16.00	4.85	43.50	7.69	15.69	7.83	45.60
Appliance	10.88	27.38	10.78	55.50	14.44	25.87	14.10	53.10	17.29	31.23	18.51	56.70
CBM	0.84	34.61	**-0.24**	57.45	1.44	42.48	**-1.53**	63.18	0.68	35.50	**-3.25**	56.70
Video	**-2.85**	47.36	**-8.73**	68.46	2.77	45.20	**-1.99**	66.61	1.01	38.94	**-1.19**	64.34
GPU	11.97	37.00	13.12	63.85	15.05	46.14	23.99	68.20	11.98	51.23	2.29	72.41
PM2.5	3.68	38.58	5.42	63.68	7.33	37.11	11.81	62.50	9.91	39.39	14.71	63.82
HTRU	0.15	12.16	**-0.30**	44.47	4.06	11.49	3.56	43.34	7.23	17.45	8.34	46.90
Shuttle	**-0.03**	31.56	0.00	59.49	**-0.01**	35.75	**-0.04**	59.05	**-0.01**	29.14	**-0.03**	57.21
AReM	2.99	19.51	3.76	50.80	2.32	23.98	3.53	55.06	11.70	20.64	11.00	52.96
Avila	11.54	9.69	14.66	43.98	11.22	20.23	11.61	51.26	15.93	20.75	15.37	52.08
Letter	9.53	44.51	9.43	66.92	11.14	19.61	11.14	30.45	10.31	-12.60	10.24	15.74
MAGIC	15.20	17.38	16.00	49.21	16.72	18.32	17.50	52.28	13.47	21.29	14.34	50.81
MoCap	10.06	82.61	9.97	76.59	9.42	32.91	9.23	38.68	8.89	22.80	8.66	39.12

We further report the clustering performance in Table 6.3 for all the data sets when d is fixed at 10, c equals to 20 and 35, and m equals to 1.8, 2.0, and 2.2. Slightly larger value of d is used because of the better performance it acquires than that from small values, say 2 or 3. *PE* and *PT* have the same definitions as in (6.8) and (6.9). For most combinations of parameters m and c, compared with the benchmark method, 0.15% to 23.99% of the reconstruction error and 9.46% to 72.41% of the running time are reduced by *Strategy B*. That is, there exists many cases where cluster quality and clustering efficiency are improved simultaneously. However, we also see cases where efficiency is greatly improved but the cluster quality is slightly reduced (shown as the boldface negative numbers in Table 6.3). Although these cases are not our expectation, in practice it may be tolerable, because by sacrificing a small amount of accuracy or cluster quality we obtain a great deal of elevation of the efficiency.

6.6.2.3 Performance of Proposed Methods with Classification Error

So far, clustering performance of the proposed methods are evaluated without the need to know the ground truth class information. Now with the provided class labels for the last 7 data sets, we further evaluate the proposed methods by classification error in (6.7). All the experimental settings are the same as those in Section 6.6.2.1 and Section 6.6.2.2.

For the classification error of *Strategy A*, we show its trend with the increasing value of d (the cluster number used in each subset) as the dashed lines in Figure 6.10(a) for AReM. Compared to *Strategy A*, the trend of the classification error of the benchmark method is shown as solid lines in these plots. We see that *Strategy A* obtains a better performance after d exceeds a certain value (3 in this case). This observation is consistent with that obtained from the reconstruction criterion, for ease of comparison we show the corresponding reconstruction errors in Figure 6.10(b) for AReM. For *Strategy B*, we calculate the classification error with different combinations of values of d, c, and m on different data sets. When m is fixed at 2.0, we show the classification error of *Strategy B*

and the benchmark method for MAGIC in Figure 6.11(a). Its counterpart when reconstruction criterion is used is shown in Figure 6.11(b).

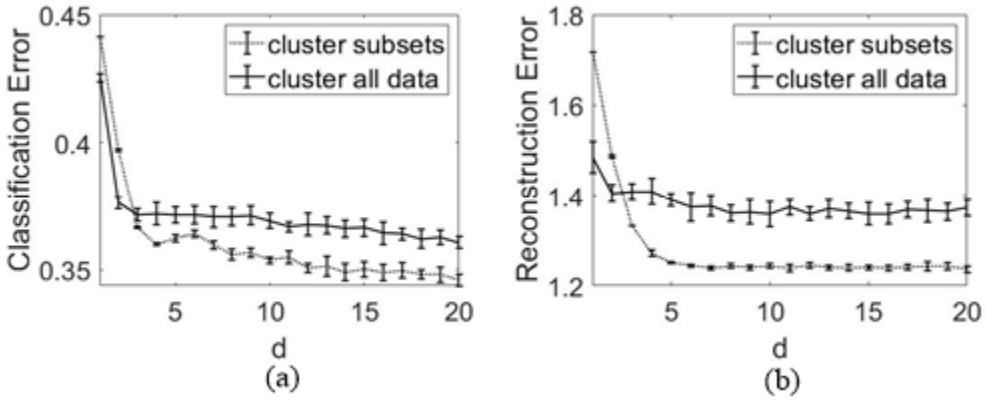


Figure 6.10. Clustering performance in terms of (a) Classification error, (b) reconstruction error for AReM from Strategy A with $m = 2.0$.

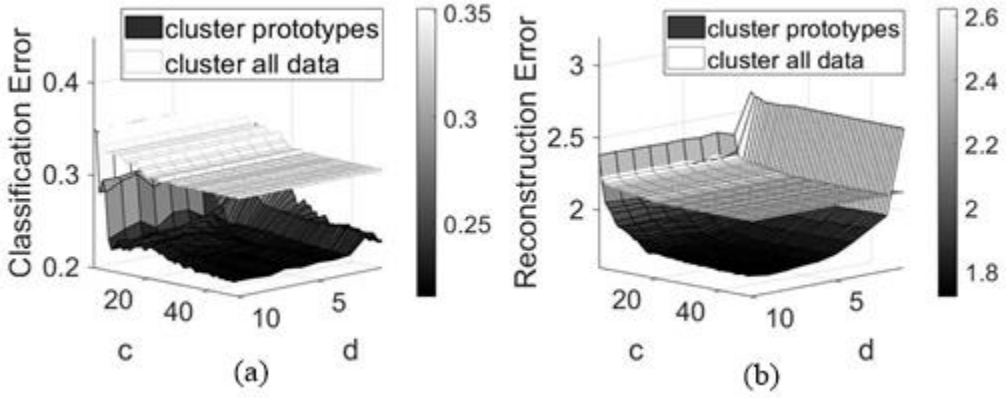


Figure 6.11. Clustering performance in terms of (a) Classification error, (b) reconstruction error for MAGIC from Strategy B with $m = 2.0$.

6.7 Summary

In this chapter, we proposed a comprehensive and systematic framework to handle the big data clustering task. Specifically, the proposed hyperplane division method is a new and efficient tool used to decompose the data into small subsets. It is quite different from the conventional subset formation methods such as the widely used sampling techniques. This division method builds the non-overlapping local regions and makes the clustering algorithm later used focus exactly on a local region of the entire data space. This division method is the primary cause for the improved performance derived from the proposed two development strategies. Besides, these two strategies fully reflect the fact that different number of clusters may be considered in different applications. When the number of clusters of the entire data to be clustered is large, by choosing *Strategy A* we could greatly improve the clustering performance in terms of both the cluster quality and the clustering efficiency. When this cluster number is small, we can choose *Strategy B*; here although absolute better cluster quality is not guaranteed compared with direct clustering the entire data, efficiency of the clustering algorithm will be greatly improved. Note that *Strategy B* could serve as a sound method when the intrinsic clusters (normally a small number of clusters) are of interest.

Chapter 7 Identification of Fuzzy Rule-Based Models with Collaborative Fuzzy Clustering

A major concern in the identification of the FRBM is how to form subspaces of the input space of the system. In earlier works, each input variable of the system is partitioned into a group of overlapping fuzzy sets provided by domain experts; then the entire space is partitioned by taking the Cartesian product of these fuzzy sets. However, with the increasing dimensionality of the input space, along with a lack of knowledge coming from domain experts, partitioning the input space in this manner becomes intolerable (curse of the dimensionality) and infeasible. This makes clustering algorithms a very attractive and compelling alternative, because they often imply the efficiency of the overall design.

Although numerous research have been done (reviewed in the next section), several limitations are still present. First, all the current methods assume that the original data (both input and output data) are available to those who consider modeling the system. Second, most of these methods focus on modeling multiple-input single-output (MISO) systems, while less attention is paid to multiple-input multiple-output (MIMO) systems. However, there are situations when the input data are available to one of the users (as modelers of systems) and the corresponding output data (one- or multi-dimensional) are available to another user, while in light of some constraints (say, confidentiality) input and output data cannot be put together.

We illustrate the situations mentioned above by two users (systems) seeking for collaboration with each other, as shown in Figure 7.1. The main point is that no original data are allowed to be shared between the two users. In Figure 7.1(a), System A has its own collected input data (shown by solid lines), where x_i^{A} is the i-th (i = 1, 2,…, n) input (independent) variable, but lacks the output data it is interested in. Those output data can

be provided (not given to System A thus shown by dashed lines) by System B, with the *j*-th ($j = 1, 2,\ldots, h$) output (dependent) variable represented as y_j^{B}. Of course, both participants agree to collaborate with each other is based on the assumption that this collaboration is beneficial to both parts. Hence, System A may also agree to provide System B with the information with which System B is interested in building a prediction model. We show a similar case in Figure 7.1(b).

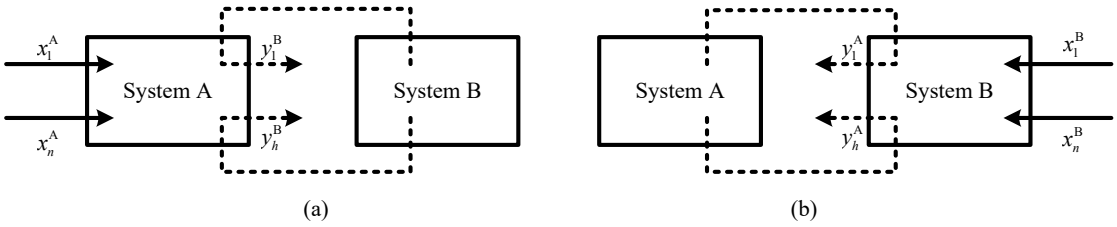


Figure 7.1. Collaboration between Systems A and B to build prediction models for (a) System A and (b) System B.

Since collaborative fuzzy clustering (CFC) [13], [108] is efficient in identifying the structure of one data set by considering the structural information of other data sets but without requiring the raw data, the major objective of this study is to apply this mechanism to the process of construction of FRBMs (for either MISO or MIMO systems) to address the limitations of the current research mentioned previously.

7.1 Clustering for Building the FRBM: A Brief Review

Here, two major categories of methods are observed: (a) Clustering is performed in the input space of the system without considering any information from the output space [18], [19], [109]–[112]. (b) Subspaces of the input space are obtained by considering information from both input and output spaces [18], [20], [120]–[125], [105], [113]–[119]. Obviously, revealing the relationships between input and output spaces becomes an essential aspect of

system modeling. By exploiting the structural information from the output space, those relationships are considered in the process of division of the input space, which makes the unsupervised learning (i.e., clustering the input space) kind of supervised to better model the system. Hence, we are more interested in those clustering algorithms in Category (b). Depending on how the output information is utilized, three groups of methods are observed:

- The output space is first divided into subspaces, which gives rise to an equal number of subspaces of the overall input space; then clustering is performed in each subspace of the input space to obtain more refined clusters (subspaces). The subspaces of the output space are formed directly when it is discrete [122]–[124], otherwise clustering algorithms are used to divide the continuous space [105], [116].
- The input and output spaces are concatenated to form a joint input-output space in which the clustering is performed. Due to its effectiveness (obtained subspaces of the input space are more relevant compared with those without using the output space) and efficiency (only one more dimension is added compared with directly clustering the input space), this is the commonly used strategy to make input and output spaces "know" each other in the modeling process.
- The output information is not limited to the data therein, it could be either the output data associated with their partition information (i.e., a partition matrix) [119] or weighted version of the output data [20], [117]. The method presented in [119] assigns a higher weight to the output space to compensate the unequal dimensionality of the input and output spaces. While methods in [20], [117] allow for a dynamic usage of the structural information in the output space (an optimal weight of output data is determined to maximize the performance of the FRBM), making them better in revealing the relationships between input and output spaces.

7.2 CFC-Based Strategy for FRBM

In this part, we propose another form of cluster-centric-based strategy which is based on the concept and algorithm of CFC. We propose an innovative collaborative mechanism to form the $[\boldsymbol{v}_i^{\mathrm{T}}, w_i]^{\mathrm{T}}$ pursued in Section 2.4.2.2. However, we assume that the input data X and the output data $\boldsymbol{y}$ are only available locally to the two different users (systems), i.e., they could not be gathered together by a single user. The idea is shown in Figure 7.2, where User (System) A has data X while User (System) B has data $\boldsymbol{y}$. FCM algorithm is applied to X and $\boldsymbol{y}$ in a collaborative way to form the same number of clusters (c clusters). Since X and $\boldsymbol{y}$ are in the distinct spaces, the partition matrices are used as the exchanged data structures between them. λ reflects the intensity of modifying the structure of X according to the structure of $\boldsymbol{y}$, while μ is that of updating the structure of $\boldsymbol{y}$ based on the structure of X. Both λ and μ are nonnegative numbers.

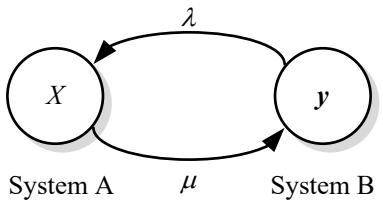


Figure 7.2. Collaboration between input data X and output data $\boldsymbol{y}$: a schematic view.

To facilitate our presentation and emphasize the collaboration aspect of the CFC algorithm, suppose that these two data sets are represented as $D[1]$ and $D[2]$, once $D[1] = X$ then $D[2] = \boldsymbol{y}$**;** similarly, if $D[1] = \boldsymbol{y}$ then $D[2] = X$. Then the idea of collaboration among two users can be formulated as an optimization problem minimizing the objective function defined as

$$J = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^2[1]\left\|\boldsymbol{v}_i[1] - \boldsymbol{x}_k[1]\right\|^2 + \beta(1,2)\sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik}[1] - u_{ik}[2])^2\left\|\boldsymbol{v}_i[1] - \boldsymbol{x}_k[1]\right\|^2 \qquad (7.1)$$

Index 1/2 specifies the information related to data set $D[1]$/$D[2]$. Specifically, $u_{ik}[1]$ represents the membership of data point $\boldsymbol{x}_k[1]$ in $D[1]$ to the prototype $\boldsymbol{v}_i[1]$; by replacing 1 by 2, we get the corresponding explanation for $u_{ik}[2]$. $\beta(1,2)$ represents the intensity of modifying the structure of $D[1]$ according to that of $D[2]$. By minimizing the first term in (7.1), the structure in $D[1]$ is explored, note that this term is also used as the objective function in the FCM algorithm. By minimizing the second term, the structure of $D[2]$ is exploit to modify that of $D[1]$. By minimizing the two terms simultaneously, a trade-off between the exploration of structure in $D[1]$ and the exploitation of structure of $D[2]$ is pursued.

The constraints used for (7.1) is similar to those used in the FCM algorithm, i.e., $\left\{u_{ik}[1]\in[0,1]\middle|\ \sum_{i=1}^{c}u_{ik}[1]=1,\ 0<\sum_{k=1}^{N}u_{ik}[1]<N\right\}$. This optimization problem can be solved by the alternating optimization algorithm. By setting the gradient of J to zero with respect to $u_{ik}[1]$ and $\boldsymbol{v}_i[1]$, we show formulas used to update the partition matrix and prototypes for $D[1]$ as

$$u_{sk}[1]=\frac{1}{1+\beta(1,2)}\left(\beta(1,2)u_{sk}[2]+1/\sum_{j=1}^{c}\frac{\left\|\boldsymbol{v}_s[1]-\boldsymbol{x}_k[1]\right\|^2}{\left\|\boldsymbol{v}_j[1]-\boldsymbol{x}_k[1]\right\|^2}\right) \tag{7.2}$$

$$v_{st}[1]=\frac{\sum_{k=1}^{N}u_{sk}^2[1]x_{kt}[1]+\beta(1,2)\sum_{k=1}^{N}\left(u_{sk}[1]-u_{sk}[2]\right)^2x_{kt}[1]}{\sum_{k=1}^{N}u_{sk}^2[1]+\beta(1,2)\sum_{k=1}^{N}\left(u_{sk}[1]-u_{sk}[2]\right)^2} \tag{7.3}$$

where s = 1, 2,…, c, k = 1, 2,…, N, and t = 1, 2,…, n. The detailed process of derivation of (7.2) and (7.3) can be found in [108], which is not reported here.

The processing flow of the CFC algorithm is summarized as follows. At the initial phase (phase 0), no collaboration takes place between $D[1]$ and $D[2]$, i.e., data structure of each data set is formed based on the locally available data. In following phases, local findings of data sets are exchanged, and the optimization is carried out so that for a certain

data set both its local data structure and the one from the other data set are taken into consideration. See Figure 7.3 for a graphic abstract of this process.

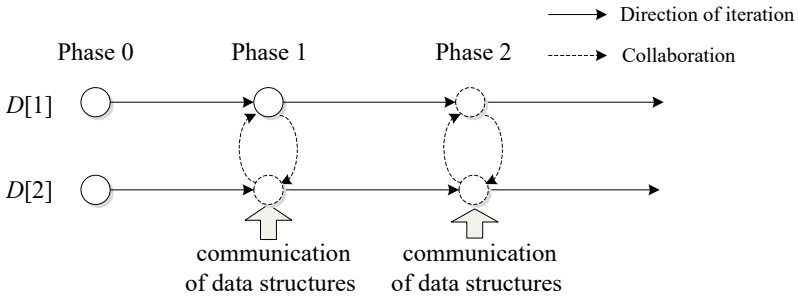


Figure 7.3. Illustration of applying the CFC algorithm to data sets D[1] and D[2].

7.3 FRBM for the MIMO System

In this part, we introduce the FRBM for the MIMO system and elaborate how the proposed CFC-based strategy could be extended to estimate parameters and outputs of the FRBM for MIMO system. For comparative studies, we also adapt the LSE- and AFCM-based strategies introduced in Section 2.4.2 for the MIMO system.

FRBM for the MIMO system could be represented as

$$\textit{Rule } i\text{: If } \boldsymbol{x} \text{ is } A_i(\boldsymbol{x}) \text{ then } \tilde{\boldsymbol{y}} = \tilde{\boldsymbol{w}}_i,\ i = 1, 2, \ldots, c. \tag{7.4}$$

The only difference between models in (7.4) and the zero-order TS model for the MIMO system is that now the output $\tilde{\boldsymbol{y}} = \tilde{\boldsymbol{w}}_i$ of each rule becomes a vector in a h-dimensional output space $\boldsymbol{R}^h$, specifically we have $\tilde{\boldsymbol{y}} = [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_h]^{\mathrm{T}}$ and $\tilde{\boldsymbol{w}}_i = [w_{i1}, w_{i2}, ..., w_{ih}]^{\mathrm{T}}$. Suppose that the provided input-output data pairs ($\boldsymbol{x}_k$, $\tilde{y}_k$), k = 1, 2,…, N, are organized as (X, Y), X = $[\boldsymbol{x}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}}, ..., \boldsymbol{x}_N^{\mathrm{T}}]^{\mathrm{T}}$ and $Y = [\tilde{\boldsymbol{y}}_1^{\mathrm{T}}, \tilde{\boldsymbol{y}}_2^{\mathrm{T}}, ..., \tilde{\boldsymbol{y}}_N^{\mathrm{T}}]^{\mathrm{T}} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_h]$ where $\tilde{\boldsymbol{y}}_k = [\tilde{y}_{k1}, \tilde{y}_{k2}, ..., \tilde{y}_{kh}]^{\mathrm{T}}$ and $\boldsymbol{y}_j = [y_{1j}, y_{2j}, ..., y_{Nj}]^{\mathrm{T}}$, we introduce the extended three strategies as follows.

7.3.1 CFC-Based Strategy

The proposed CFC-based strategy in Section 7.2 could be directly extended to design the FRBM for MIMO system. For this extended strategy we similarly assume that the input data X and output data Y are separately owned by User (System) A and User (System) B. However, there are two evident options to realize the extension. Option 1: we use the input variables to predict the entire output variables. In other words, the structure learnt in the input space is shared among all the output variables. The topology of this kind of collaboration is shown in Figure 7.4(a). Option 2: each output variable is predicted individually by the input variables. That is, different structures in input space are learnt for different output variables. This kind of collaboration is illustrated in Figure 7.4(b). Obviously, for Option 1, only one group of optimal collaboration strength pair could be found, while for Option 2, h (i.e., the number of output variables) groups of such pairs are pursued.

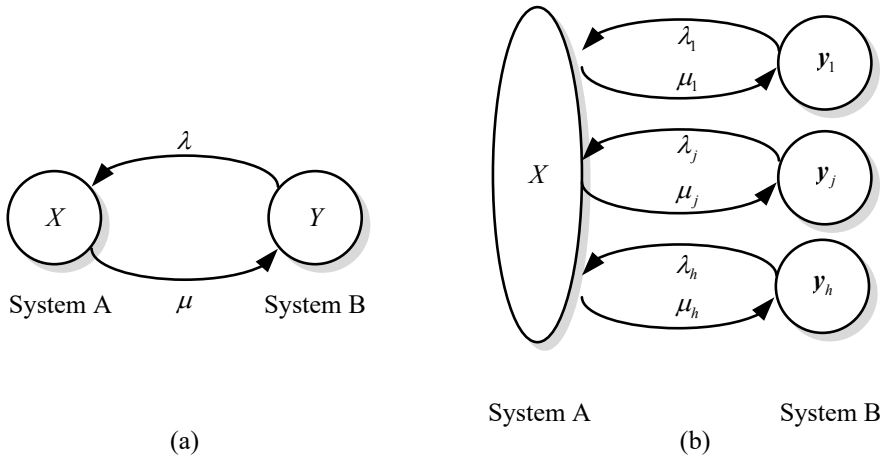


Figure 7.4. Collaboration between input data X and (a) entire output data Y; (b) output data of each output variable.

Regarding to the implementation of Option 1, we only need to replace the output data $\boldsymbol{y}$ in Section 7.2 by Y. The formulas and process to realize the collaboration between input

and output data are exactly the same as those in Section 7.2. To estimate the h-dimensional output $\hat{\tilde{\boldsymbol{y}}}_k$, (2.29) and (2.30) are modified to their multi-dimensional versions as follows

$$F = \sum_{i=1}^{c}\sum_{k=1}^{N} \tilde{u}_{ik}^{m} \left\| \tilde{\boldsymbol{w}}_i - \hat{\tilde{\boldsymbol{y}}}_k \right\|^2 \tag{7.5}$$

$$\hat{\tilde{\boldsymbol{y}}}_k = \frac{\sum_{i=1}^{c} \tilde{u}_{ik}^{m} \tilde{\boldsymbol{w}}_i}{\sum_{i=1}^{c} \tilde{u}_{ik}^{m}} \tag{7.6}$$

To evaluate the performance of the extended strategy, the RMSE index in (2.31) is changed accordingly as

$$\text{RMSE} = \sqrt{\frac{1}{Nh}\sum_{k=1}^{N} \left\| \hat{\tilde{\boldsymbol{y}}}_k - \tilde{\boldsymbol{y}}_k \right\|^2} \tag{7.7}$$

where the used distance is the standard Euclidean distance. The optimal output could be estimated by tuning collaboration strength (λ, μ) to minimize (7.7).

As for the implementation of Option 2, it could be treated as a series of implementation of the proposed strategy in Section 7.2. The optimal predicted output data for each variable are still obtained based on tuning the collaboration strength (λ_j, μ_j) to minimize the index in (2.32). However, to make the comparison among different options and strategies possible, these data will be concatenated as $\hat{\tilde{\boldsymbol{y}}}_k$ in $\boldsymbol{R}^h$, then (7.7) is used as the performance index for Option 2.

7.3.2 LSE-Based Strategy

The LSE-based strategy used for identification of the FRBM for MIMO system is composed of a series of application of the LSE-based strategy introduced in Section 2.4.2.1 to each output variable $\tilde{y}_j, j = 1, 2, ..., h,$ in (7.4). After the h-dimensional output $\hat{\tilde{\boldsymbol{y}}}_k$ is concatenated from h individually estimated one-dimensional output data, (7.7) is used to measure the performance of this extended strategy.

7.3.3 AFCM-Based Strategy

In this strategy, the input and output data are concatenated into a new data set as $Z = [X, Y]$, which is a N by $n+h$ matrix. Similar to the extended CFC-based strategy in Section 7.3.1, we also have two options here. Option 1: All the output variables share the same structure of the input space. To perform the AFCM algorithm on Z, y_k and w_i in (2.26), (2.28), (2.29) are replaced by $\tilde{\boldsymbol{y}}_k$ and $\tilde{\boldsymbol{w}}_i$, and the coefficient αn in (2.26) and (2.29) is changed to $\alpha n/h$. Then the h-dimensional output $\hat{\tilde{\boldsymbol{y}}}_k$ is obtained from (7.6), with the performance measured by (7.7). Optimal $\hat{\tilde{\boldsymbol{y}}}_k$ is obtained when (7.7) is minimized with the optimal α. Option 2: output variables are predicted individually with the strategy in Section 2.4.2.2. The optimal output data of each variable are obtained by tuning α to minimize (2.32), then concatenated to form $\hat{\tilde{\boldsymbol{y}}}_k$. Performance of Option 2 in this strategy is also measured by (7.7).

7.4 Experimental Studies

In this part, we examine the performance of the proposed CFC-based strategy for FRBMs for MISO system on both synthetic and publicly available data, its performance is also compared with LSE- and AFCM-based strategies. Moreover, a group of data used for multi-target regression are used to check the performance of the extended strategies for FRBMs for the MIMO system.

7.4.1 Synthetic Data

We generate a group of three-dimensional synthetic data (two-dimensional input space and one-dimensional output space) in a way used in [20]. For the input data, 200 data points are generated uniformly over a two-dimensional space [0,10] × [0,10], and 5 cluster centers $\boldsymbol{v}_i$ are randomly given along with their corresponding output w_i in Table 7.1. For the output data, (2.20) is first used to obtain the membership of each input data point to each cluster

center, then (2.30) is used to get the corresponding output by setting the fuzzification coefficient m as 2. The generated input data (shown as asterisks) with the specified cluster centers (shown as diamonds), and the mapping between the input and output data are illustrated in Figure 7.5(a) and (b), respectively. Based on this data set, we further generate 10 data sets by injecting additive noise of different intensity (standard deviation) into the output data. Hence, the output for the 11 data sets could be expressed as $y_k = y_k + f_k$, $k = 1, 2,\ldots, 200$, where $f_k \sim N(0, \sigma^2)$, $\sigma = 0, 0.1, 0.2,\ldots, 1$.

Table 7.1. Prototypes in input and output data.

i	$\boldsymbol{v}_i$	w_i
1	[1.5, 0.5]	1.0
2	[1.0, 4.0]	0.5
3	[4.0, 8.0]	3.0
4	[6.0, 2.5]	-2.0
5	[8.0, 6.5]	5.0

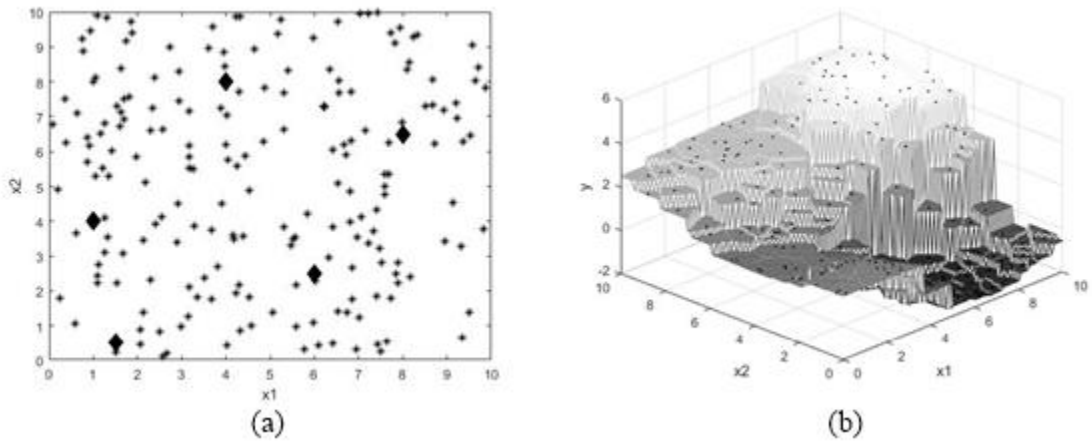


Figure 7.5. Illustration of (a) Input data; and (b) input-output mapping.

As for the experimental settings, the cluster number c is arbitrarily set as 5, 8, and 11, respectively; the fuzzification coefficient m is set as 2; 60% of the data are used as training data and the rest are used for testing; 20-fold experiment is used to determine the mean and standard deviation of the performance of each strategy. For the proposed CFC-based strategy, we range the collaboration strength λ and μ from 0 to 5 with a step size of 0.2. Maximum iteration number in each collaboration phase is set as 20, and 5 collaboration phases are executed. For the AFCM-based strategy, the weight coefficient α is ranged from 0 to 5 with a step size of 0.1. We report in Table 7.2 the average and standard deviation of performance index (i.e., RMSE in (2.31)) of the LSE-, AFCM-, and CFC-based strategies (note that boldface entries represent the best performance among the strategies). The optimal values of the collaboration strength obtained for the CFC-based strategies are also reported in Table 7.3.

Table 7.2. Mean and standard deviation of RMSE for three strategies (m = 2).

σ	Strategies	c = 5		c = 8		c = 11	
		Training	Testing	Training	Testing	Training	Testing
0	LSE	0.91±0.11	0.92±0.13	0.87±0.14	0.94±0.17	0.78±0.11	0.89±0.13
	AFCM	0.56±0.06	**0.56±0.12**	0.64±0.64	0.69±0.09	0.62±0.07	0.72±0.07
	CFC	**0.23±0.01**	0.61±0.14	**0.15±0.15**	**0.68±0.10**	**0.11±0.01**	**0.71±0.06**
0.1	LSE	0.93±0.12	0.95±0.12	0.86±0.09	0.96±0.18	0.78±0.08	0.87±0.09
	AFCM	0.57±0.06	**0.57±0.07**	0.61±0.06	0.72±0.10	0.60±0.08	**0.69±0.10**
	CFC	**0.24±0.01**	0.62±0.08	**0.15±0.01**	**0.69±0.10**	**0.11±0.01**	0.70±0.06
0.2	LSE	0.94±0.09	1.00±0.15	0.85±0.14	0.94±0.13	0.76±0.10	0.90±0.15
	AFCM	0.59±0.07	**0.62±0.10**	0.65±0.06	0.72±0.09	0.59±0.07	**0.72±0.10**
	CFC	**0.26±0.01**	0.67±0.08	**0.17±0.01**	**0.68±0.09**	**0.12±0.01**	0.72±0.09
0.3	LSE	0.96±0.13	1.01±0.12	0.90±0.10	0.99±0.12	0.82±0.10	0.88±0.10
	AFCM	0.60±0.04	0.63±0.07	0.64±0.05	0.73±0.07	0.63±0.05	0.68±0.05
	CFC	**0.28±0.01**	**0.62±0.07**	**0.18±0.01**	**0.69±0.08**	**0.13±0.01**	**0.65±0.06**
0.4	LSE	1.02±0.09	1.11±0.14	0.97±0.12	1.09±0.13	0.87±0.10	0.96±0.13
	AFCM	0.67±0.05	**0.70±0.07**	0.69±0.08	**0.77±0.09**	0.66±0.08	0.77±0.13
	CFC	**0.29±0.01**	0.76±0.09	**0.19±0.01**	0.79±0.10	**0.14±0.01**	**0.77±0.11**
0.5	LSE	0.97±0.09	1.06±0.11	0.97±0.10	1.07±0.15	0.88±0.11	1.02±0.12

	AFCM	0.72±0.05	**0.76±0.05**	0.71±0.05	**0.79±0.10**	0.71±0.07	0.81±0.11
	CFC	**0.32±0.01**	0.78±0.05	**0.19±0.01**	0.80±0.08	**0.14±0.01**	**0.78±0.09**
0.6	LSE	1.15±0.05	1.18±0.14	1.11±0.10	1.21±0.15	1.05±0.10	1.18±0.11
	AFCM	0.88±0.04	**0.92±0.07**	0.87±0.05	**0.94±0.08**	0.84±0.05	**0.96±0.07**
	CFC	**0.38±0.03**	0.93±0.07	**0.25±0.02**	0.97±0.08	**0.18±0.04**	0.96±0.10
0.7	LSE	1.12±0.10	1.15±0.12	1.02±0.09	1.11±0.11	0.96±0.06	1.04±0.10
	AFCM	0.89±0.04	**0.91±0.08**	0.82±0.04	**0.89±0.08**	0.80±0.05	**0.87±0.08**
	CFC	**0.39±0.02**	0.94±0.10	**0.22±0.01**	0.90±0.09	**0.17±0.03**	0.91±0.05
0.8	LSE	1.21±0.07	1.27±0.11	1.13±0.09	1.27±0.12	1.08±0.10	1.18±0.10
	AFCM	0.97±0.06	**1.05±0.07**	0.95±0.06	**1.08±0.11**	0.95±0.06	**1.03±0.09**
	CFC	**0.35±0.02**	1.17±0.11	**0.22±0.02**	1.17±0.10	**0.17±0.02**	1.09±0.13
0.9	LSE	1.32±0.11	1.39±0.13	1.24±0.11	1.38±0.17	1.25±0.09	1.35±0.10
	AFCM	1.08±0.05	**1.11±0.10**	1.07±0.07	**1.17±0.11**	1.08±0.07	**1.20±0.11**
	CFC	**0.43±0.04**	1.14±0.10	**0.26±0.03**	1.20±0.11	**0.20±0.02**	1.20±0.11
1.0	LSE	1.40±0.11	1.39±0.09	1.28±0.08	1.41±0.10	1.26±0.07	1.42±0.11
	AFCM	1.18±0.06	**1.19±0.10**	1.09±0.04	**1.25±0.09**	1.12±0.07	**1.27±0.11**
	CFC	**0.42±0.02**	1.21±0.11	**0.25±0.02**	1.31±0.10	**0.19±0.07**	1.32±0.10

From Table 7.2, we see that for all the cases (with different values of cluster number c, different intensity of noise σ, for either the training or testing data), the AFCM- and CFC-based strategies show a much better performance than the LSE-based strategy. This highlights the benefit of the cluster-centric fuzzy modeling when the data structure of output data is considered during the design process of the model. Focusing on the training data, for all the cases (different values of c and σ) the proposed CFC-based strategy obtains a superb performance than other two strategies; in many cases its RMSE value is less than one third of that derived from the AFCM-based strategy. This great improvement should attribute to the dynamic interaction between the input and output data (they exchange structures with each other and modify their own structures with the different collaboration strength such that a better RMSE value is observed). For the testing data, the CFC-based strategy still obtains a sound performance although it could not reach the superb performance appeared in the training case. The proposed strategy obtains the best

performance for $\sigma = 0.3$ when $c = 5$, for $\sigma = 0$, 0.1, 0.2, and 0.3 when $c = 8$, and $\sigma = 0$, 0.3, 0.4, and 0.5 when $c = 11$, for other cases it obtains slightly lower but similar performance to the AFCM-based strategy. These observations are extremely compelling, because although the input and output data are not allowed to be put together, with the proposed approach much better model performance on the training data and the comparative (sometimes even better) model performance on the testing data could be obtained compared with models built on the complete input-output data. However, not surprisingly, for all the strategies, their performance decreases with the increasing intensity of the noise.

Table 7.3. Optimal values of the collaboration strength ($m = 2$).

σ	$c = 5$				$c = 8$				$c = 11$			
	Training		Testing		Training		Testing		Training		Testing	
	λ_{opt}	μ_{opt}	λ_{opt}	μ_{opt}	λ_{opt}	μ_{opt}	λ_{opt}	μ_{opt}	λ_{opt}	μ_{opt}	λ_{opt}	μ_{opt}
0	2.8	0.0	3.8	0.0	3.0	0.0	4.4	0.0	4.0	0.0	3.2	0.0
0.1	2.6	0.0	4.2	0.0	2.8	0.0	2.8	0.0	3.8	0.0	4.2	0.0
0.2	3.0	0.0	3.2	0.0	3.2	0.0	5.0	0.0	5.0	0.0	3.8	0.0
0.3	3.0	0.0	5.0	0.0	4.2	0.0	3.4	0.0	4.6	0.0	4.6	0.2
0.4	5.0	0.0	4.8	0.0	3.4	0.0	4.2	0.0	4.2	0.0	4.8	0.0
0.5	4.6	0.0	3.8	0.0	5.0	0.0	4.6	0.0	4.6	0.0	4.4	0.0
0.6	2.6	0.0	4.4	0.0	4.8	0.0	3.4	0.0	3.4	0.0	4.8	0.0
0.7	3.4	0.0	5.0	0.0	4.6	0.0	5.0	0.0	4.4	0.0	4.2	0.0
0.8	4.2	0.0	3.0	0.0	4.4	0.0	4.8	0.0	4.6	0.0	4.6	0.0
0.9	4.6	0.0	5.0	0.0	4.0	0.0	4.4	0.0	4.8	0.0	3.4	0.0
1.0	4.6	0.0	4.6	0.0	4.4	0.0	2.6	0.0	4.4	0.0	3.6	0.0

From Table 7.3, we see that in most of the cases we obtain a large value of λ and a zero-valued μ, indicating the imbalanced collaboration between the input and output data. Considering the meaning of λ, experimental results show that modifying the structure of input data based on that from output data is more beneficial in improving the model performance. The finding of the imbalanced collaboration is very useful for practical applications, because instead of searching optimal collaboration strength in a two-

dimensional space, we could search in the space of one variable (most of the time, λ). For illustration, we show the performance with respect to different values of λ and μ for $\sigma = 0$, 0.5, and 1.0 when $c = 11$ in Figure 7.6. For comparison, we also add the optimal performance of LSE- and AFCM-based strategies for the specified values of σ and c. Since their performance are not related to λ and μ, they remain at the same level with the changing collaboration strength. From all the plots, we see that the collaboration between the input and output data is beneficial only when the value of either λ or μ is maintained around 0 while leaving the other variable much larger than 0; otherwise, we have much worse performance (e.g., see the case when both λ and μ equal 0). Moreover, generally setting μ around 0 is more helpful. Besides, to see how the collaboration process affects the parameter estimation of the FRBM, given the input and output data generated when $\sigma = 0$, 0.5, and 1.0, we cluster data into $c = 5$ clusters when three different pairs of collaboration strength are used. The prototypes for the input data (shown as diamonds for $(\lambda, \mu) = (0, 0)$, circles for optimal (λ, μ), and squares for $(\lambda, \mu) = (5, 5)$) are illustrated in Figure 7.7. Obviously, with different values of (λ, μ), the locations of the prototypes vary dramatically.

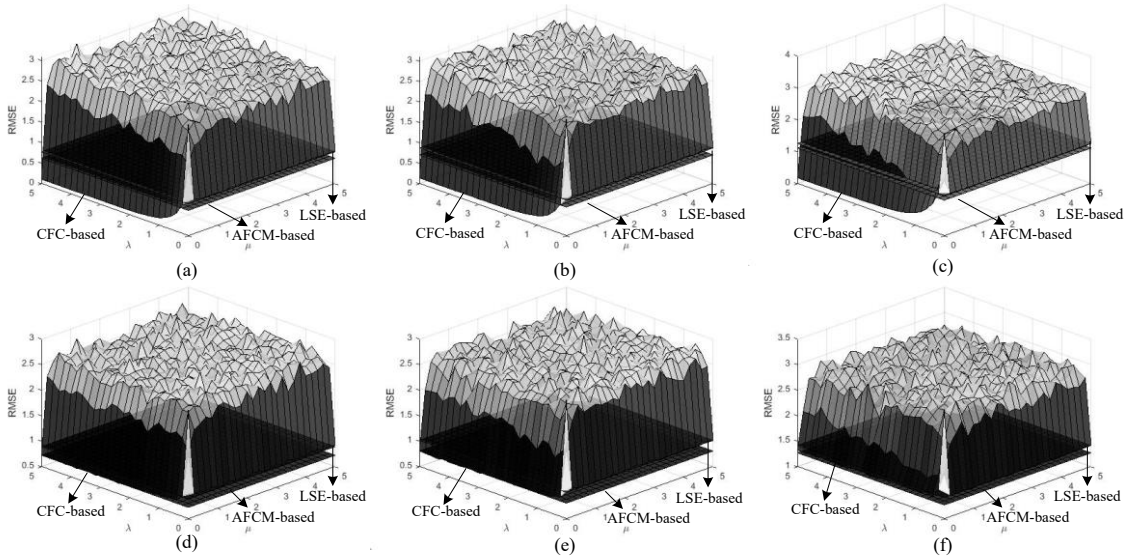


Figure 7.6. Performance of the three strategies on training data when (a) $\sigma = 0$; (b) $\sigma = 0.5$; (c) $\sigma = 1$; and that on testing data when (d) $\sigma = 0$; (e) $\sigma = 0.5$; (f) $\sigma = 1$.

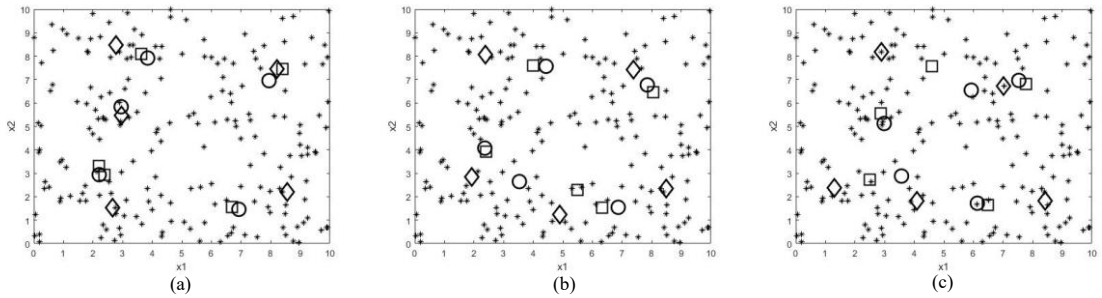


Figure 7.7. Distribution of prototypes for $(\lambda, \mu) = (0, 0)$, optimal values of (λ, μ), and $(\lambda, \mu) = (5, 5)$ for selected data sets with different noise level: (a) $\sigma = 0$; (b) $\sigma = 0.5$; (c) $\sigma = 1$.

7.4.2 Publicly Available Data

In this part, we use 14 publicly available data to test the performance of the proposed CFC-based strategy and compare it with that derived from the LSE- and AFCM-based strategies. These data sets are obtained either from the UCI machine learning repository or the KEEL-data set repository.

The experimental settings are the same as those in the synthetic case with the exception that the number of rules c for FRBM is arbitrarily set as 5, 7, or 9, respectively; and the fuzzification coefficient m is set to 1.5, 2, or 3, respectively. We only report in Table 7.4 the mean and standard deviation of the performance of three strategies when $c = 7$. However, analysis of the results when $c = 5$ and 9 are also provided.

From Table 7.4, for all the cases (either training or testing data, any value of m, any data set), the cluster-centric-based strategies always produce better performance than the LSE-based strategy. For all the cases in the training data, the proposed CFC-based strategy greatly outperforms other strategies. These two findings are consistent with what we have observed in the synthetic case. For the testing data, performance of the CFC-based strategy is still similar to that of the AFCM-based strategy, however, we see more cases (than in synthetic scenarios) where the former strategy obtains the best performance. Specifically,

when m = 1.5 the CFC-based strategy beats other strategies on Concrete, Ele2, Friedman, Concrete2, and Noise; when m = 2 it outperforms others on Autoprice, Dee, Ele2, Friedman, Concrete 2, Noise, PM10, and Wankara; when m = 3 except Concrete and Machine, it obtains the best performance on the remaining 12 data sets.

For cases when c = 5 and 9, the proposed strategy still obtains the superb performance than others on the training data. For the testing data, it performs the best on 6, 10, and 14 data sets when m equals 1.5, 2, and 3, respectively for c = 5; and the best on 5, 7, and 13 data sets when m equals 1.5, 2, and 3, respectively for c = 9.

To sum up, all these experimental results indicate a promising prospect of using the proposed CFC-based strategy to design the FRBM to address the challenge when the input and output data are not allowed to be shared directly with each other. The model accuracy based on the proposed strategy is comparable to the AFCM-based strategy (which is the state-of-the-art design strategy that comprehensively uses the output information during the modeling process). Interestingly, we see that although the proposed strategy could not always be the winner for the performance in terms of the RMSE, it does the best job in many cases.

Table 7.4. Mean and standard deviation of RMSE obtained for three strategies (c = 7).

Data sets	Strategies	m = 1.5		m = 2		m = 3	
		Training	Testing	Training	Testing	Training	Testing
Autompg8	LSE	3.71±0.18	3.82±0.27	3.79±0.14	3.92±0.29	4.07±0.26	4.18±0.36
	AFCM	3.33±0.16	**3.57±0.25**	3.48±0.16	**3.62±0.28**	3.75±0.17	3.82±0.30
	CFC	**1.04±0.11**	3.83±0.25	**1.08±0.14**	3.69±0.29	**1.12±0.09**	**3.79±0.28**
Autoprice	LSE	3206.44±353.75	3369.83±563.40	3655.83±418.97	3734.76±673.28	3601.32±248.01	4237.73±401.83
	AFCM	2239.01±227.45	**2745.04±414.29**	2449.40±264.43	2854.61±507.53	2811.57±352.51	3651.23±546.76
	CFC	**706.19±133.87**	2790.59±427.68	**716.33±243.12**	**2823.88±531.44**	**722.05±140.76**	**3489.33±434.47**
Building	LSE	98.62±10.98	105.12±15.22	101.93±10.39	109.25±14.98	107.33±8.44	110.97±11.25
	AFCM	77.25±9.95	**93.32±15.88**	86.58±9.64	**97.33±15.78**	97.49±8.94	101.88±12.51
	CFC	**24.72±9.09**	101.06±20.89	**27.16±8.37**	99.29±18.81	**30.68±5.01**	**98.22±12.16**
Concrete	LSE	14.02±0.80	14.48±0.96	14.19±1.07	16.05±1.53	16.08±0.72	16.77±1.07
	AFCM	12.74±0.86	13.99±1.03	12.37±1.00	**14.78±1.43**	13.43±0.71	**14.64±1.12**
	CFC	**1.79±0.42**	**13.80±1.29**	**1.66±0.21**	14.84±1.36	**1.67±0.25**	14.90±1.05
Dee	LSE	0.46±0.02	0.47±0.03	0.47±0.02	0.48±0.03	0.54±0.02	0.55±0.04
	AFCM	0.43±0.02	**0.44±0.04**	0.45±0.02	0.46±0.03	0.50±0.02	0.50±0.03
	CFC	**0.13±0.02**	0.44±0.04	**0.13±0.02**	**0.46±0.03**	**0.13±0.01**	**0.48±0.03**
Ele2	LSE	447.15±21.93	461.91±23.10	439.63±21.59	446.34±31.66	527.74±28.04	536.62±48.73
	AFCM	368.64±7.60	390.16±20.32	370.92±13.26	388.52±28.07	430.01±25.00	450.12±38.51
	CFC	**178.81±8.50**	**387.32±18.58**	**185.61±8.81**	**387.09±24.78**	**206.34±6.65**	**423.51±33.91**
Energy	LSE	3.87±0.32	3.94±0.24	4.65±0.34	4.77±0.39	5.23±0.19	5.29±0.27
	AFCM	2.93±0.11	**3.06±0.12**	3.54±0.11	**3.66±0.14**	4.46±0.06	4.49±0.15
	CFC	**1.18±0.12**	3.16±0.16	**1.10±0.08**	3.69±0.27	**1.13±0.05**	**4.43±0.19**
Friedman	LSE	3.31±0.19	3.30±0.23	3.30±0.13	3.36±0.19	3.23±0.09	3.62±0.13

	AFCM	2.72±0.09	2.78±0.09	3.12±0.04	3.15±0.10	3.55±0.05	3.67±0.12
	CFC	**0.83±0.05**	**2.77±0.06**	**0.79±0.04**	**3.04±0.10**	**0.78±0.03**	**3.37±0.11**
Machine	LSE	61.92±23.60	70.02±32.54	90.73±19.07	101.97±36.74	112.68±17.05	109.21±30.84
	AFCM	29.94±8.83	**52.81±29.77**	36.00±12.22	**73.64±31.44**	68.80±17.04	**74.75±33.85**
	CFC	**16.35±8.08**	57.51±29.14	**21.39±6.75**	76.43±31.51	**22.42±7.28**	78.80±22.00
Concrete2	LSE	13.46±0.26	13.67±0.44	14.99±0.41	15.25±0.72	15.86±0.75	15.96±0.79
	AFCM	11.58±0.23	11.95±0.39	12.25±0.28	12.27±0.44	13.04±0.19	13.19±0.37
	CFC	**2.27±0.10**	**11.90±0.43**	**2.17±0.07**	**12.01±0.52**	**2.26±0.08**	**12.91±0.37**
Noise	LSE	6.02±0.10	6.10±0.15	6.15±0.13	6.20±0.13	6.57±0.13	6.64±0.18
	AFCM	5.07±0.09	5.13±0.15	5.21±0.10	5.33±0.12	5.70±0.08	5.75±0.18
	CFC	**1.11±0.05**	**5.01±0.15**	**1.01±0.03**	**5.29±0.13**	**1.03±0.03**	**5.49±0.17**
PM10	LSE	0.83±0.02	0.85±0.04	0.86±0.02	0.86±0.03	0.86±0.02	0.86±0.04
	AFCM	0.76±0.03	**0.80±0.04**	0.83±0.02	0.84±0.02	0.84±0.02	0.84±0.04
	CFC	**0.13±0.01**	0.85±0.03	**0.13±0.01**	**0.83±0.02**	**0.14±0.02**	**0.84±0.04**
Wankara	LSE	5.31±0.20	5.54±0.32	6.10±0.24	6.05±0.47	7.36±0.29	7.50±0.47
	AFCM	4.64±0.20	**5.18±0.42**	5.37±0.18	5.55±0.55	6.46±0.23	6.88±0.53
	CFC	**1.81±0.16**	5.33±0.44	**1.79±0.14**	**5.38±0.56**	**1.79±0.09**	**6.21±0.52**
Wizmir	LSE	4.79±0.17	4.87±0.16	5.79±0.10	5.77±0.16	7.10±0.22	7.28±0.24
	AFCM	4.31±0.06	**4.42±0.14**	5.15±0.06	**5.09±0.18**	6.22±0.08	6.34±0.18
	CFC	**1.55±0.04**	4.61±0.18	**1.53±0.04**	5.20±0.16	**1.59±0.05**	**6.19±0.15**

7.4.3 Experiments with FRBM for MIMO

In this part, we test the performance of the extended three strategies on 12 multi-target regression data sets which are obtained from the Mulan website (http://mulan.sourceforge.net/datasets-mtr.html).

The experimental settings are same as those in Section 7.4.2 except that number of rules c is set arbitrarily as 3, 6, or 9, respectively. We have observed the superb performance of the CFC-based strategy for the MISO system on the training data. This also holds in the MIMO case. However, recall that we have two options (Option 1 and Option 2) in the extended CFC- and AFCM-based strategies. We find that both strategies implemented in Option 2 (predict the output variable individually) obtain a much better performance than that in Option 1. This finding is as expected because in Option 2 more parameters in the condition part of the FRBM have been involved in the prediction task. In what follows, let us focus on the model performance on the testing data after all this indicates how the model performs on the unseen data which could be a more important aspect. We report these results of three extended strategies in Table 7.5 when $c = 6$ (similar observations are found for cases when $c = 3$ and $c = 9$, which are not reported here).

To sum up, in Option 1 there is no absolute winner among the three extend strategies. When the number of output variables is large (say, large than 15), the LSE-based strategy could be a sound choice, otherwise the cluster-centric-based strategies are more suitable. In Option 2, the cluster-centric-based strategies dominate the model performance. In both options, AFCM- and CFC-based strategies show similar performance, however, when m is large (say, around 3) the CFC-based strategy usually performs better. Besides, selection of the option is a problem of the trade-off between the accuracy and computational efficiency of the model. Obviously, although performance of strategies in Option 2 are better, more time is needed for the training process of model identification.

Table 7.5. Mean and standard deviation of RMSE of extended strategies on testing Data (c = 6).

Data set	Strategy	m = 1.5		m = 2		m = 3	
		Option 1	Option 2	Option 1	Option 2	Option 1	Option 2
Andro	LSE	8.65±1.06	8.35±1.65	8.86±1.68	8.76±1.11	8.91±1.23	8.83±0.93
	AFCM	**7.50±1.88**	**6.52±1.67**	**7.90±1.92**	**6.83±1.56**	8.44±1.64	**7.54±1.36**
	CFC	8.05±1.88	7.48±1.39	8.34±1.82	7.91±1.12	**7.83±1.53**	7.82±1.20
Atp1d	LSE	99.74±61.48	85.61±4.52	98.55±8.12	93.39±8.75	105.60±42.89	100.15±10.92
	AFCM	91.60±34.56	**79.39±4.42**	**90.32±6.97**	**83.15±5.74**	99.90±16.44	92.82±5.22
	CFC	**91.49±23.87**	81.51±4.53	90.85±6.19	84.24±5.53	**97.69±17.35**	**90.83±4.67**
Atp7d	LSE	76.64±6.09	108.47±97.37	141.43±168.96	203.12±223.27	96.46±18.29	99.75±19.18
	AFCM	**71.97±5.96**	93.18±73.47	109.20±85.15	116.98±81.49	100.32±11.90	79.58±15.42
	CFC	74.35±6.94	**73.02±11.42**	**101.57±44.23**	**82.62±20.97**	**95.14±14.59**	**77.85±14.53**
Edm	LSE	0.44±0.02	0.44±0.03	0.44±0.01	0.44±0.02	0.45±0.02	0.45±0.02
	AFCM	**0.42±0.02**	**0.42±0.02**	0.43±0.02	0.42±0.02	0.44±0.03	0.43±0.02
	CFC	0.44±0.02	0.42±0.02	**0.43±0.03**	**0.41±0.02**	**0.42±0.02**	**0.41±0.02**
Enb	LSE	3.88±0.22	3.81±0.16	4.79±0.40	4.59±0.31	5.16±0.43	5.12±0.28
	AFCM	**3.22±0.12**	**3.13±0.10**	**3.65±0.12**	**3.60±0.17**	**4.41±0.18**	4.37±0.14
	CFC	3.35±0.12	3.30±0.12	3.84±0.12	3.78±0.14	4.51±0.17	**4.36±0.16**
Jura	LSE	12.56±1.51	12.46±1.47	12.55±1.44	12.17±1.26	11.63±1.81	12.54±1.67
	AFCM	**9.75±1.19**	**9.09±1.49**	**10.93±1.76**	**9.39±1.09**	12.09±1.71	**11.02±1.59**
	CFC	10.18±1.48	10.12±1.37	10.95±1.75	10.26±1.20	**10.61±1.94**	11.67±1.27
Oes10	LSE	855.83±381.47	667.27±181.29	**965.28±395.97**	919.65±365.54	**1045.03±320.13**	1015.09±395.01
	AFCM	**837.07±377.76**	**604.81±159.89**	984.05±366.32	886.09±349.55	1211.89±310.40	1057.85±378.59
	CFC	883.16±365.23	614.62±151.88	983.03±362.76	**880.31±362.13**	1071.33±296.45	**969.14±349.24**
Oes97	LSE	1068.12±293.66	1005.47±323.82	**1203.52±259.94**	1209.55±413.91	**1116.32±388.77**	1434.49±414.71

	AFCM	**1044.98±272.74**	926.37±282.52	1266.03±272.21	1141.48±387.39	1355.91±442.15	1508.68±382.11
	CFC	1079.16±296.88	**908.23±274.42**	1278.95±295.52	**1128.64±392.33**	1203.98±389.04	**1317.86±353.89**
Rf1	LSE	**16.64±0.28**	16.59±0.36	**16.20±0.29**	16.09±0.32	19.20±0.31	19.25±0.28
	AFCM	16.93±0.30	15.21±0.40	16.74±0.34	**14.54±0.29**	21.09±0.64	15.51±0.22
	CFC	16.91±0.35	**15.00±0.41**	16.72±0.32	14.79±0.28	**17.01±0.35**	**15.20±0.23**
Scm20d	LSE	227.86±2.11	226.71±1.66	233.07±1.63	233.04±2.23	235.71±2.19	235.75±3.20
	AFCM	**218.96±2.01**	**208.49±1.39**	**228.33±1.57**	**213.78±2.12**	237.84±1.59	223.60±2.04
	CFC	224.61±1.80	211.52±1.07	228.67±1.56	214.69±1.87	**233.23±1.45**	**220.74±1.82**
Slump	LSE	10.71±0.65	10.82±0.82	11.93±2.21	12.18±2.99	11.94±0.87	11.78±1.04
	AFCM	**10.03±0.70**	9.46±0.89	**10.14±0.81**	10.04±0.82	10.71±0.62	10.30±1.07
	CFC	10.17±0.88	**9.31±0.77**	10.16±0.86	**9.82±0.69**	**10.54±0.69**	**10.04±0.90**
Wq	LSE	**1.23±0.01**	1.25±0.01	**1.26±0.01**	1.26±0.01	**1.28±0.02**	1.28±0.02
	AFCM	1.25±0.01	**1.24±0.01**	1.28±0.02	**1.23±0.01**	1.30±0.02	**1.25±0.02**
	CFC	1.28±0.02	1.26±0.01	1.28±0.02	1.25±0.01	1.29±0.02	1.27±0.02

7.5 Summary

Although it is significant and meaningful, the requirements of building prediction models when the input and output data could not be shared among users (systems) have not been carefully identified and studied yet. In this chapter, we highlighted and illustrated this kind of requirement raised in real world due to the major consideration of the privacy of data. To meet this challenge, an innovative CFC-based strategy has been proposed to design the FRBM for either the MISO or MIMO system. The collaboration between two different users (owners of data) by exchanging the structures (i.e., partition matrices) of data with each other makes it possible to build FRBM without gathering the input and output data together. To our surprise, in most cases (with different data sets and different number of rules), the model performance (accuracy) produced with the CFC-based strategy is better than that obtained from the most commonly used LSE-based strategy and comparative to the state-of-the-art AFCM-based design strategy. We also observed many cases where the proposed strategy produces even better performance than the AFCM-based strategy. Note that LSE- and AFCM-based strategies assumes that both input and output data are accessible to the user who wish to build the model. Hence, we believe that the CFC-based strategy serves as a sound solution to design FRBM when input and output data are not allowed to be shared. Considering the similar performance between the CFC- and AFCM-based strategies, the collaborative mechanism could also be regarded as an alternative way (rather than giving the weight to the output space used in the AFCM-based strategy) to find more relevant structures in the input space.

Chapter 8 Identification of Fuzzy Rule-Based Models with Output Space Knowledge Guidance

So far, three characteristics of the interested data have been covered, i.e., distributed, granular, and big. In this chapter, we would cover the last one, that is, supervised (i.e., data analysis is supervised by knowledge of experts). We will see how this knowledge tidbit could be helpful for clustering of the input space when building the FRBM, which finally turned out to be beneficial to improving the performance of the FRBM.

We point out a domain modeling knowledge, which could be potentially useful but was not articulated and incorporated into the existing design practices. This knowledge tidbit can be articulated as follows *when two output values are far apart in the output space, their corresponding input values should be allocated to different clusters*. As it could be envisioned, it is not reasonable to map similar inputs to very distinct outputs. If such situation occurs in data, those similar inputs are better to be positioned in different clusters. However, when two output values are similar, we do not have to impose any restriction on the location of the corresponding inputs because they could be located in the same cluster or in different clusters. Among the current studies identified so far, only the context variable-based methods have considered this type of knowledge tidbits (although this knowledge tidbit is only partially manifested in [105]). However, note that this knowledge tidbits has been utilized only at a coarse level (or at the cluster level), because once a cluster (context) in the output space has been determined, no further information of the output values of the input values belonging to this context is utilized. Hence, the main objective of this study is to find some other alternatives to implement this modeling knowledge tidbit when building the FRBM. In our opinion, this domain knowledge could be reflected at a finer level (or the data level), that is, more information of the output space could be used to guide the clustering process of the input data.

8.1 Knowledge Tidbit Derived from Output Space

The essential knowledge-oriented concept is the closeness of a pair of data points. We introduce the concept of proximity, which serves as a sound closeness measure. Suppose that the output values Y located in the output space are clustered into c groups, resulting the partition matrix UY with c rows and N columns. We generate a proximity matrix PY (a $N \times N$ matrix) based on UY, whose entry $PY(k_1, k_2)$ describes the closeness (proximity) between output values y_{k_1} and y_{k_2}, which is defined as

$$PY(k_1, k_2) = \sum_{i=1}^{c} (u_{ik_1}^{y} \wedge u_{ik_2}^{y}) \tag{8.1}$$

where $k_1, k_2 = 1, 2, \ldots, N$, $u_{ik_1}^{y}$ and $u_{ik_2}^{y}$ are, respectively the k_1th and k_2th elements in the ith row in partition matrix UY. Symbol $\wedge$ stands for the minimum operation between two entries, i.e., $u_{ik_1}^{y} \wedge u_{ik_2}^{y} = \min(u_{ik_1}^{y}, u_{ik_2}^{y})$. One may envision that if y_{k_1} and y_{k_2} have the similar membership values (to any of the prototypes), their proximity $PY(k_1, k_2)$ is then close to one; on the contrary, $PY(k_1, k_2)$ is close to zero. Hence, the defined proximity measure reflects the closeness of the data points.

Clearly, UY provides with information on an extent to which each output value belongs to a certain cluster (hence a cluster level knowledge shown in context variable-based methods), while PY gives more detailed information about the relationship (proximity) between any pair of output values (hence a data level knowledge proposed and highlighted in this study).

Based on the proximity matrix PY, we further form a matrix BY with binary-valued entries defined as

$$BY(k_1, k_2) = \begin{cases} 1 & \text{if } PY(k_1, k_2) < \tau \\ 0 & \text{otherwise.} \end{cases} \tag{8.2}$$

where τ is a threshold given in advance stating that two output values are regarded to be far apart from each other. The choice of the value of this threshold could bring some difficulties; usually it should assume values close to zero. Here, if $BY(k_1, k_2) = 1$, this means that since output values y_{k_1} and y_{k_2} are far apart, placing their corresponding input values in distinct clusters is required. Hence, the final knowledge tidbit derived from the output space is preserved in the proximity matrix PY along with this binary matrix BY (with specific attention being paid to those entries in BY with values equal to one).

8.2 Splitting Input Space with Knowledge Derived from Output Space

With the obtained knowledge tidbit from the output space, the ensuing question is on how to apply it to the process of identification of FRBM. Since the partition of input data X in input space could be finally represented as a partition matrix UX, we need to guarantee that UX is formed in such a way that the knowledge tidbit derived from the output space has been fully considered. A simple diagram to show our objective is illustrated in Figure 8.1. In this part, we form two methods to obtain the final UX of the input space.

Method A: The partition matrix UX of the input space is first formed in a usual way by using the standard FCM algorithm. Based on UX, the proximity of any pair of input values could be derived and represented as the proximity matrix PX. Since the knowledge tidbit derived from the output space has been partially preserved in BY, from which an intuitive conclusion is that when $BY(k_1, k_2) = 1$ the difference between $PX(k_1, k_2)$ and $PY(k_1, k_2)$ should be minimized. That is, we hope to directly update the entries in the partition matrix UX to make the value of $PX(k_1, k_2)$ small. This method used to identify FRBM could be summarized as the proximity fuzzy clustering (PFCM)-based strategy.

Method B: The partition matrix UX is formed based on a new distance measure. If we know two input values should not be placed in the same cluster, a new distance could be devised to make sure that they should have the distinct membership degrees to the same

cluster prototype. Then the partition matrix *UX* is updated based on the newly formed distance measure and the cluster prototypes, which is similar to an iterative process in the standard FCM algorithm. Obviously, this method gives an indirect way to update the partition matrix *UX* compared with *Method A*. We may summarize this method as the Refined FCM (RFCM)-based strategy.

In what follows, we introduce these two methods in detail.

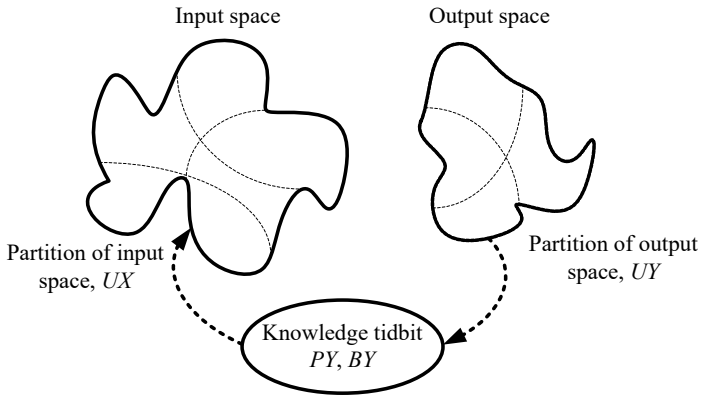


Figure 8.1. Partition of input space based on knowledge tidbit derived from output space.

8.2.1 PFCM-based Strategy to Divide the Input Space

With the provided proximity matrix *PX* for the input space and the proximity and binary matrices *PY* and *BY* (i.e., the knowledge tidbits) derived from output space, we form the optimization problem as follows

$$\min \quad J = \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} \left(PX(k_1,k_2) - PY(k_1,k_2) \right)^2 BY(k_1,k_2) \tag{8.3}$$

where $PX(k_1,k_2) = \sum_{i=1}^{c} (u_{ik_1}^x \wedge u_{ik_2}^x)$, $u_{ik_1}^x$ and $u_{ik_2}^x$ are, respectively the k_1th and k_2th elements in the ith row in partition matrix *UX*. By minimizing *J*, we hope to maintain the proximity of each pair of input values (whose corresponding output values are far apart) in a low level.

Since index J is essentially a function of the membership degree $u_{st}^{x} \in UX$, $s = 1, 2,\ldots,$ c, $t = 1, 2,\ldots, N$, we determine the derivative of J with respect to u_{st}^{x}, that is

$$\frac{\partial J}{\partial u_{st}^{x}} = \sum_{k_1=1}^{N}\sum_{k_2=1}^{N}\frac{\partial}{\partial u_{st}^{x}}\Big[\big(PX(k_1,k_2) - PY(k_1,k_2)\big)^2\Big]BY(k_1,k_2)$$

$$= 2\sum_{k_1=1}^{N}\sum_{k_2=1}^{N}[\big(PX(k_1,k_2) - PY(k_1,k_2)\big)BY(k_1,k_2)\frac{\partial}{\partial u_{st}^{x}}\sum\nolimits_{i=1}^{c}(u_{ik_1}^{x} \wedge u_{ik_2}^{x})] \quad (8.4)$$

where the inner derivative could be further determined as

$$\frac{\partial}{\partial u_{st}^{x}}\sum\nolimits_{i=1}^{c}(u_{ik_1}^{x} \wedge u_{ik_2}^{x}) = \begin{cases} 1 & \text{if } t = k_1 \text{ and } u_{st}^{x} \le u_{sk_2}^{x}, \\ 1 & \text{if } t = k_2 \text{ and } u_{st}^{x} \le u_{sk_1}^{x}, \\ 0 & \text{otherwise}. \end{cases} \quad (8.5)$$

To keep index J decreasing, we update partition matrix UX along the negative direction of the gradient in (8.4). That is, we have

$$u_{st}^{x}(iter+1) = \left[u_{st}^{x}(iter) - \alpha\frac{\partial V}{\partial u_{st}^{x}(iter)}\right]_{*} \quad (8.6)$$

where α is the step size used to control changes of membership grades, $[\cdot]_{*}$ is the truncation operator to make sure the obtained membership value is positioned within the unit interval, and *iter* stands for the index of successive iterations.

The partition of the input space based on the PFCM-based strategy could be summarized as two nested loops. In the external loop, the input space is partitioned based on formulas used to get the prototypes and partition matrix from the standard FCM algorithm. Once the partition matrix UX of the input space is obtained, we go into the inner loop in which we update UX according to (8.6) to minimize index J. When some termination condition (e.g., no significant changes happens to J) of the inner loop reaches, we return the updated UX to the external loop to get the prototypes. This process continues until the external loop reaches a termination condition (e.g., maximum number of

iterations). Let us denote by Q the objective function of the standard FCM, which is induced by the updated UX from the internal loop.

8.2.2 RFCM-based Strategies to Divide the Input Space

In this part, we form an optimization problem similar to the one in the standard FCM algorithm to explore the input data structure. The difference is that here we introduce a penalty coefficient β_{ik} to adjust the distance between a data point $\boldsymbol{x}_k$, $k = 1, 2,\ldots, N$, to the specific prototype $\boldsymbol{v}_i$, $i = 1, 2,\ldots, c$. The optimization problem is formed as

$$\min \quad Q = \sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik}^{x})^{m}\beta_{ik}\left\|\boldsymbol{x}_k - \boldsymbol{v}_i\right\|^2 \tag{8.7}$$

where m is the fuzzification coefficient, $\|\cdot\|$ is the standard Euclidean distance, and penalty coefficient β_{ik} is defined as follows

$$\beta_{ik} = \begin{cases} \dfrac{1}{\sum_{h=1}^{\mathrm{Card}(S_i)} PY(y_k, y_{ih}) / \mathrm{Card}(S_i)} & \text{if } \mathrm{Card}(S_i) \geq \mathrm{Card}(S_j) \\ 1 \quad \text{otherwise}. \end{cases} \tag{8.8}$$

where S_i, $i = 1, 2,\ldots, c$, represents a set of elements, each element of which has the shortest distances (in terms of the standard Euclidean distance) to $\boldsymbol{v}_i$ (rather to other prototypes) but is prohibited to be put in the same cluster with $\boldsymbol{x}_k$; $\boldsymbol{x}_{ih} \in S_i$, and Card($S_i$) is the number of elements in S_i; $PY(y_k, y_{ih})$ is the proximity between y_k and y_{ih}.

We give an example to explain (8.8). Suppose we intend to partition the input space into two parts, and we are provided with two prototypes $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ shown as two diamonds in Figure 8.2. Given the data point $\boldsymbol{x}_k$ represented as a square in Figure 8.2, we need to get its distance to $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, respectively. From the knowledge tidbit obtained from the output space, we have already known which data points should not be placed in the same cluster with $\boldsymbol{x}_k$ (focusing on the kth row of the binary matrix BY, these points are indexed by the columns with entries valued one). By calculating the standard Euclidean distance between

these points with $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, we can easily assign them to the right clusters. Assume that we have five points incompatible with (i.e., should be placed far apart from) $\boldsymbol{x}_k$ which are shown as black dots in Figure 8.2, three of them are assigned to $\boldsymbol{v}_1$ and the other two are given to $\boldsymbol{v}_2$ as shown in Figure 8.2. Then we think that $\boldsymbol{x}_k$ is more incompatible with $\boldsymbol{v}_1$, hence a penalty should be added to the standard Euclidean distance between $\boldsymbol{x}_k$ and $\boldsymbol{v}_1$ (i.e., $\|\boldsymbol{x}_k - \boldsymbol{v}_1\|$); while no penalty is needed for $\|\boldsymbol{x}_k - \boldsymbol{v}_2\|$. Since the value of proximity $PY(y_k, y_{ih})$ is always a value less than τ, a threshold close to zero, its reciprocal leads to a large penalty to the standard Euclidean distance.

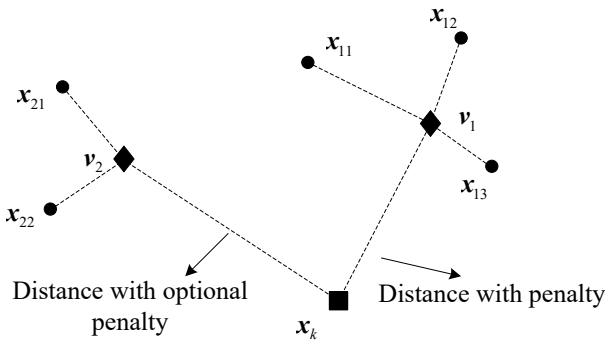


Figure 8.2. Illustration of the penalty of a standard Euclidean distance.

Or as another choice, we could consider the penalty of a distance as long as the incompatible points with $\boldsymbol{x}_k$ are found in a cluster. In this case, both distances $\|\boldsymbol{x}_k - \boldsymbol{v}_1\|$ and $\|\boldsymbol{x}_k - \boldsymbol{v}_2\|$ should be penalized but with the different degree. We may slightly modify (8.8) to (8.9) to realize this. In (8.9), Card(S_i) in the numerator is used to stress that if more incompatible points with $\boldsymbol{x}_k$ are find in $\boldsymbol{v}_i$ then distance $\|\boldsymbol{x}_k - \boldsymbol{v}_i\|$ should be assigned with a larger value of penalty coefficient. Let us name RFCM-based strategies, respectively using (8.8) and (8.9) as RFCM(a)- and RFCM(b)-based strategies. We will see in the experiments that both strategies could show better model performance in some scenarios.

$$\beta_{ik} = \begin{cases} \dfrac{\mathrm{Card}(S_i)}{\sum_{h=1}^{\mathrm{Card}(S_i)} PY(y_k, y_{ih}) / \mathrm{Card}(S_i)} & \text{if } \mathrm{Card}(S_i) \neq \varnothing, \\ 1 \quad \text{otherwise}. \end{cases} \tag{8.9}$$

The constraints for (8.7) is the same as those for the standard FCM algorithm, that is $0 \le u_{ik}^x \le 1$ and $\sum_{i=1}^{c} u_{ik}^x = 1$. With the Lagrangian method, we could obtain formulas for partition matrix UX (or for prototypes $\boldsymbol{v}_i$) to minimize Q when prototypes (or partition matrix) are fixed, which are given as follows

$$u_{ik}^x = 1 / \sum_{s=1}^{c} \left(\frac{\beta_{ik} \left\| \boldsymbol{v}_i - \boldsymbol{x}_k \right\|^2}{\beta_{sk} \left\| \boldsymbol{v}_s - \boldsymbol{x}_k \right\|^2} \right)^{1/(m-1)} \tag{8.10}$$

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{N} (u_{ik}^x)^m \beta_{ik} \boldsymbol{x}_k}{\sum_{k=1}^{N} (u_{ik}^x)^m \beta_{ik}} \tag{8.11}$$

Hence, with the Picard iteration used in the standard FCM algorithm, we dynamically change the positions of the prototypes and the membership values of each point to these prototypes in their own way to minimize index Q.

8.3 Output Determination with the Developed FRBM

In this part, we show how to use the constructed TS model (which has already considered the knowledge tidbit residing in the output space) to get the output when a new input data $\boldsymbol{x}_{\text{new}}$ is encountered. Basically, as long as we know its membership value $A_i(\boldsymbol{x}_{\text{new}})$ to each cluster $\boldsymbol{v}_i$, then with the optimal parameters determined in (2.24) for zero-order TS model the output for $\boldsymbol{x}_{\text{new}}$ can be obtained directly through (2.21). However, we would note that the way we obtain $A_i(\boldsymbol{x}_{\text{new}})$ is different from the conventional method that is based on calculating the standard Euclidean distance between $\boldsymbol{x}_{\text{new}}$ and each of the cluster prototypes $\boldsymbol{v}_i$, we would get $A_i(\boldsymbol{x}_{\text{new}})$ based on the partition matrix UX obtained in the model training stage. The reason is rooted in the mechanisms of two proposed methods used to divide the input space.

For the PFCM-based strategy, from the inner loop we could obtain an adjusted partition matrix *UX* based on the knowledge tidbit derived from the output space, based on which then new prototypes are obtained in the external loop. However, note that new partition matrix of the input space has to be obtained based on these new prototypes in the external loop, which is usually significantly distinct from the adjusted *UX* obtained from the inner loop. It is the *UX* from the inner loop that preserves more knowledge derived from the output space. For RFCM-based strategies, although in the model training stage we know how to penalize the distance between a data point and a cluster prototype according to either (8.8) or (8.9), we do not know how to penalize that for $\boldsymbol{x}_{\text{new}}$ because we do not have information of how many points are incompatible with $\boldsymbol{x}_{\text{new}}$ considering that we know nothing about the real output of $\boldsymbol{x}_{\text{new}}$. To sum up, for PFCM-based strategy getting the $A_i(\boldsymbol{x}_{\text{new}})$ based on the prototypes $\boldsymbol{v}_i$ obtained in the phase of development (i.e., training) of TS model is not reliable; while for RFCM-based strategies this could be infeasible. Hence, we would get the $A_i(\boldsymbol{x}_{\text{new}})$ based on the partition matrix *UX* instead of prototypes.

The idea is as follows. In input data *X* we find *K* nearest data to $\boldsymbol{x}_{\text{new}}$ in terms of the standard Euclidean distance, which are represented as $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_K$. From the phase of input space partition, we have already obtained their membership values to a certain cluster $\boldsymbol{v}_i$, which are obtained as $A_i(\tilde{\boldsymbol{x}}_1),\ A_i(\tilde{\boldsymbol{x}}_2), ..., A_i(\tilde{\boldsymbol{x}}_K)$. Then we represent the membership of $\boldsymbol{x}_{\text{new}}$ to $\boldsymbol{v}_i$ as the aggregation of the weighted memberships of $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_K$ to $\boldsymbol{v}_i$ as

$$A_i(\boldsymbol{x}_{\text{new}}) = \sum_{j=1}^{K} \omega_j A_i(\tilde{\boldsymbol{x}}_j) \tag{8.12}$$

where ω_j describes the contribution made by $A_i(\tilde{\boldsymbol{x}}_j)$, that is determined as

$$\omega_j = 1 / \sum_{s=1}^{K} \frac{\left\| \boldsymbol{x}_{\text{new}} - \tilde{\boldsymbol{x}}_j \right\|}{\left\| \boldsymbol{x}_{\text{new}} - \tilde{\boldsymbol{x}}_s \right\|} \tag{8.13}$$

By using the reciprocal of the standard Euclidean distance, we reflect the fact that the larger the distance between $\boldsymbol{x}_{\text{new}}$ and $\tilde{\boldsymbol{x}}_j$ then the lower similarity between them hence the lower contribution should be made by $A_i(\tilde{\boldsymbol{x}}_j)$.

8.4 Experimental Studies

In this section we conduct comprehensive experimental studies to show the usefulness of the proposed methods in building FRBMs. Comparison with current methods where output information is either ignored or partially considered is also illustrated.

8.4.1 Synthetic Data

In this part, we show the advantage of the proposed methods in terms of improvement of the performance (accuracy) of the FRBM based on 2D synthetic data derived from the function

$$y = 0.6\sin(\pi x) + 0.3\sin(3\pi x) + 0.1\sin(5\pi x)$$

where $x \in [-1,1]$. We randomly generate 200 input data uniformly distributed in the domain, whose corresponding outputs are obtained directly from the function. The generated 2D data are shown as circles in Figure 8.3.

As for the experiment setting, 100 data are used for training and the remaining 100 data are used for testing. Fuzzification coefficient *m* is always fixed at 2. The input data would be clustered into *c* clusters ranging from 2 to 36, and the output data are clustered into *p* clusters ranging from 2 to 5. Note that the size of the proximity matrix *PY* is always equal to the number of training data, however, the value of each entry could be affected by the cluster number *p* of the output data (this is why we range the value of *p*). The threshold τ used in (8.2) is set as 0.1. Finally, 10-fold experiment is used to get mean and standard deviation of the performance of the proposed methods.

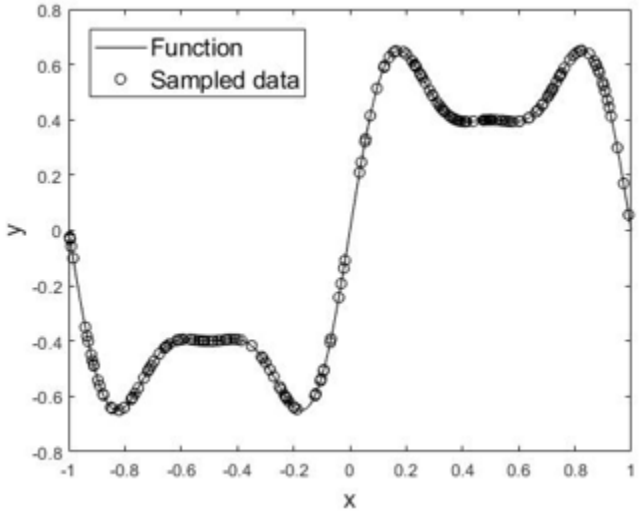


Figure 8.3. One-input one-output function and the generated 2D data.

We first show how each of the proposed strategies performs under the different combinations of c and p from Table 8.1 to Table 8.6, the best performance in each table is highlighted as the boldface. For both training and testing data, it is not surprising that generally when c, the cluster number in input space, is larger, then better performance of the FRBM is achieved. However, this is not the case for p, the cluster number in output space. From Tables 8.1 and 8.2, we see that for the PFCM-based strategy the best performance is achieved when $p = 2$ for both training and testing data. From Tables 8.3 and 8.4 for the RFCM(a)-based strategy, the best performance for training data is obtained when $p = 3$, while that for testing data happens when $p = 2$. Finally, from Tables 8.5 and 8.6 for the RFCM(b)-based strategy, both training and testing data exhibit the best performance when $p = 4$.

Table 8.1. RMSE for training data: PFCM-based strategy.

c \ *p*	2	3	4	5
2	0.112±0.005	0.139±0.017	0.135±0.015	0.130±0.009
6	0.115±0.015	0.100±0.010	0.103±0.028	0.071±0.018
12	0.097±0.010	0.091±0.014	0.097±0.023	0.090±0.029
24	0.069±0.014	0.080±0.020	0.100±0.036	0.098±0.040
36	**0.050±0.013**	0.073±0.026	0.071±0.017	0.096±0.028

Table 8.2. RMSE for testing data: PFCM-based strategy.

c \ *p*	2	3	4	5
2	0.125±0.005	0.135±0.012	0.136±0.014	0.131±0.014
6	0.115±0.015	0.111±0.015	0.114±0.032	0.085±0.021
12	0.102±0.010	0.105±0.024	0.097±0.021	0.081±0.022
24	0.088±0.015	0.083±0.030	0.095±0.026	0.086±0.018
36	**0.062±0.019**	0.075±0.022	0.074±0.021	0.084±0.022

Table 8.3. RMSE for training data: RFCM(a)-based strategy.

c \ *p*	2	3	4	5
2	0.117±0.006	0.113±0.008	0.125±0.007	0.126±0.006
6	0.118±0.021	0.116±0.010	0.114±0.014	0.123±0.030
12	0.091±0.011	0.091±0.006	0.091±0.011	0.089±0.012
24	0.069±0.007	0.053±0.017	0.058±0.012	0.064±0.013
36	0.047±0.016	**0.045±0.014**	0.047±0.011	0.046±0.016

Table 8.4. RMSE for testing data: RFCM(a)-based strategy.

c \ *p*	2	3	4	5
2	0.116±0.006	0.123±0.011	0.118±0.008	0.121±0.007
6	0.122±0.021	0.114±0.014	0.120±0.032	0.133±0.025
12	0.104±0.009	0.097±0.011	0.108±0.023	0.094±0.014
24	0.094±0.018	0.087±0.026	0.092±0.022	0.088±0.016
36	**0.059±0.010**	0.064±0.017	0.073±0.022	0.061±0.019

Table 8.5. RMSE for training data: RFCM(b)-based strategy.

c \ *p*	2	3	4	5
2	0.121±0.019	0.128±0.014	0.129±0.014	0.125±0.010
6	0.112±0.013	0.099±0.014	0.101±0.008	0.098±0.014
12	0.088±0.011	0.064±0.013	0.066±0.014	0.056±0.009
24	0.062±0.010	0.043±0.014	0.042±0.010	0.035±0.009
36	0.044±0.015	0.034±0.012	**0.029±0.008**	0.030±0.008

Table 8.6. RMSE for testing data: RFCM(b)-based strategy.

c \ *p*	2	3	4	5
2	0.114±0.011	0.118±0.017	0.125±0.011	0.124±0.009
6	0.109±0.012	0.100±0.007	0.108±0.011	0.109±0.019
12	0.095±0.013	0.081±0.022	0.082±0.016	0.063±0.020
24	0.076±0.025	0.062±0.019	0.070±0.024	0.058±0.020
36	0.063±0.024	0.061±0.015	**0.048±0.014**	0.055±0.018

To have a better insight on how each proposed strategy performs, from Figure 8.4 to Figure 8.6 we show determined outputs for testing data when the optimal combination of c and p (for testing data) is chosen. The figures show that the proposed strategies could nicely estimate the outputs, and the difference among the results of different strategies is not significant although minor advantage of the RFCM(b)-bases strategy could still be observed (we could see that more estimated outputs are overlapped with their corresponding real outputs).

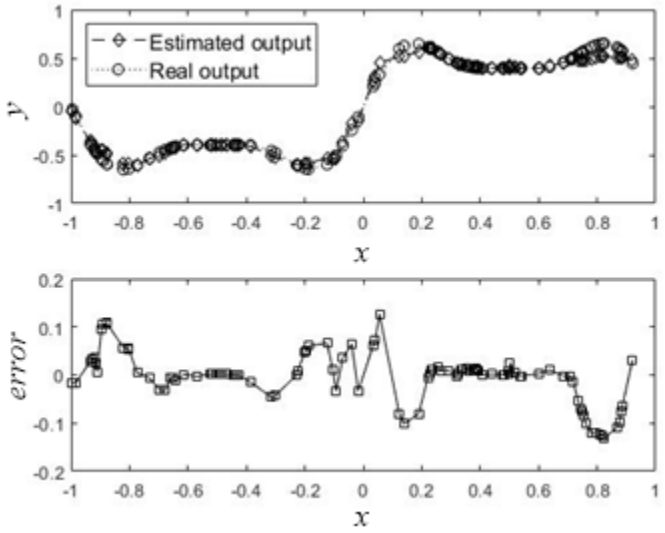


Figure 8.4. Determined outputs and actual outputs for testing data by PFCM-based strategy when c = 36 and p = 2.

To show the optimization process of the proposed methods, in Figure 8.7 we present how their objective functions change with the increasing number of iterations for the specific given combination of c and p. For the PFCM-based strategy, the value of objective function Q obtained in the external loop is used; while for RFCM-based strategies, the objective function Q in (8.7) is used. From Figure 8.7, all the methods converge after several iterations, both PFCM- and RFCM(b)-based strategies converge around four iterations, RFCM(a)-based strategy converges around two iterations.

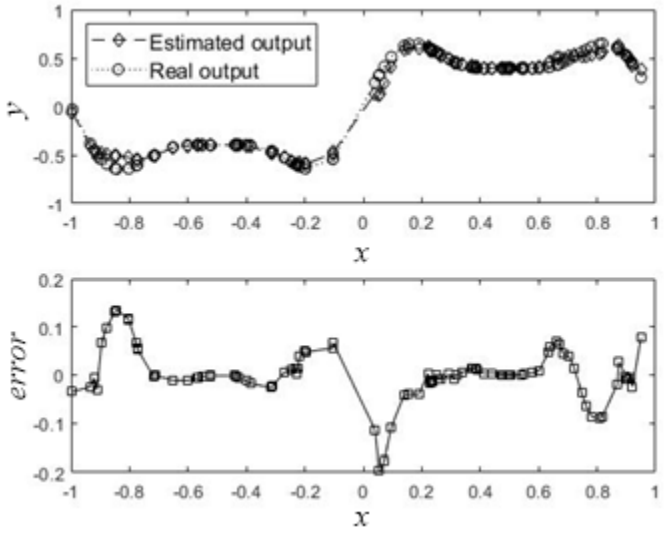


Figure 8.5. Determined outputs and actual outputs for testing data by RFCM(a)-based strategy when $c = 36$ and $p = 3$.

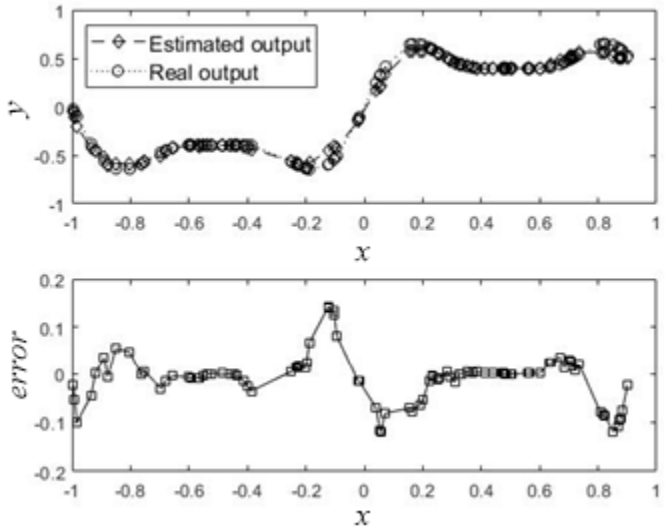


Figure 8.6. Determined outputs and actual outputs for testing data by RFCM(b)-based strategy when $c = 36$ and $p = 4$.

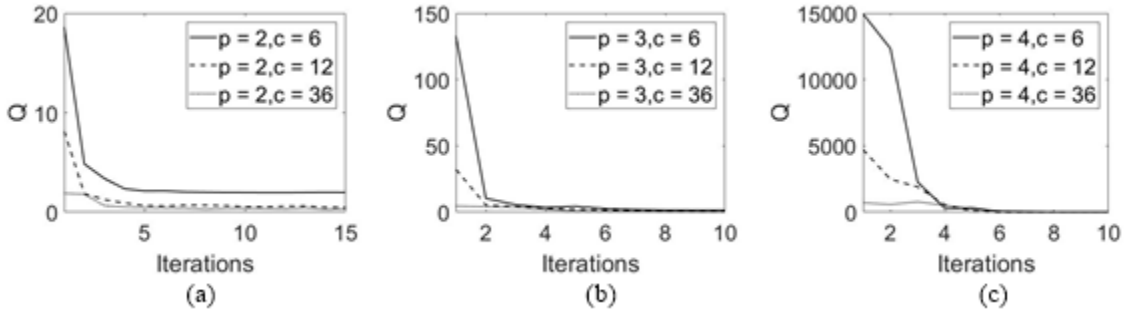


Figure 8.7. Changes of values of objective functions with the increasing iteration of the proposed methods. (a) PFCM-based strategy; (b) RFCM(a)-based strategy; (c) RFCM(b)-based strategy.

Table 8.7. Performance comparison with other methods.

	Number of rules	RMSE (Training)	RMSE (Testing)
RBF NN [116]	36	0.063±0.024	1.147±0.099
RBF NN + context-free clustering [116]	36	0.061±0.015	0.072±0.022
Linguistic modeling (c = 6, p = 6) [116]	36	0.055±0.006	0.063±0.007
Output-constrained method (c = 5, p = 1;1;2;2;2) [105]	8	**0.024**	0.049
PFCM-based strategy (c =36, p = 2)	36	0.050±0.013	0.062±0.019
RFCM-based strategy (a) (c =36, p = 2)	36	0.047±0.016	0.059±0.010
RFCM-based strategy (b) (c =36, p = 4)	36	0.029±0.008	**0.048±0.014**

It is of interest to compare the performance of proposed methods with some other methods where impact of the output space on input space partition has been considered. We show the results in Table 8.7. We observe that all the proposed methods obtain a better performance than those methods mentioned in [116] for both training and testing data under the same number of rules. As an improved version of the linguistic model in [116], the

output-constrained method [105] gets the better performance than the proposed methods on the training data but loses the advantage on the testing data over the RFCM(b)-based strategy.

8.4.2 Publicly Available Data

In this part, we examine the performance of the proposed PFCM- and RFCM-based strategies on 16 publicly available data sets which are obtained either from the UCI machine learning repository or the KEEL data set repository. We also compare the performance of these strategies with that of the LSE-based strategy, a naive Conditional FCM (CFCM)-based strategy (i.e., without any optimization) [116], and the Augmented FCM (AFCM)-based strategy [20]. Since LSE-based strategy does not use any output information when dividing the input space, it would be used as the benchmark method for other methods to compare with. For example, the RMSE of a certain strategy could be represented as RMSE_X, that of the LSE-based strategy is RMSE_{LSE}, then performance improvement of this certain strategy is calculated as

$$\%P = (\text{RMSE}_X - \text{RMSE}_{LSE})/ \text{RMSE}_{LSE} \tag{8.14}$$

As for the experimental setting, 60% of each data set are used for training while the rest for testing. The cluster number c in the input space is set as 4, 6, and 8; for simplicity, the cluster number p in the output space remains the same as c. The fuzzification coefficient m is not limited to a specific value anymore, we range its value from 1.1 to 3.0 with a step size of 0.1. For each specific value of m, the 10-fold experiment is used to get the mean and standard deviation of the performance. In other words, each strategy will use its best performance to compare with others. For the CFCM-based strategy, 2, 3, and 4 contexts are respectively used, and input data in each context is clustered into two clusters (in this way, we keep the cluster number in input space as 4, 6, and 8). For the AFCM-based strategy, the weight of the output space ranges from 0 to 5 with a step size of 0.1.

The obtained results for all the six strategies are documented in Table 8.8. The lowest RMSE value among the six strategies for either the training or testing data is highlighted as boldface. And we use the statistical *t*-testing to see if this highlighted best performance is statistically different (with a significance level of 0.05) from the performance produced by the LSE-based strategy. An asterisk is used to show that the difference of performance between two methods are significant. Since we may be more interested in the results of testing, the improvement of performance of each method on the testing data (compared with the LSE-based strategy) is especially reported, and the entry with the greatest improvement is also highlighted. Finally, the fuzzification coefficients under which the best performance of each method is obtained are also illustrated.

Interestingly, for all the data sets, the best performance for either the training or testing data is always captured by one of the three proposed strategies. The PFCM- and RFCM(b)-based strategies have more chances to obtain the best performance than the RFCM(a)-based strategy. Specifically, when $c = 4$ PFCM-, RFCM(a)-, and RFCM(b)-based strategies respectively, get 5, 2, and 9 times best performance on the training data among the 16 data sets; and they have 3, 2, and 11 times best performance on the testing data. When $c = 6$, these strategies respectively, get 6, 0, and 10 times best performance on training data; while have 5, 0, and 11 times on testing data. Finally, when $c = 8$, these numbers become 8, 0, and 8 for training and 7, 0, and 9 for testing. Note however, although the RFCM(a)-based strategy has the least chance to have the best performance, most of the time it still obtains the third best among the six strategies.

The *t*-testing shows that all the best performance is significantly different from that obtained from the LSE-based strategy. In terms of the testing data, the improvement of the best performance (compared with that derived from the benchmark strategy) ranges between 5.1% and 40.0% for different data sets when $c = 4$; it ranges from 6.3% to 43.0% when $c = 6$; and ranges from 6.6% to 39.5% when $c = 8$. These huge improvement

highlights the advantage when the knowledge tidbit derived from the output space is considered with the proposed methods.

Interestingly, since intensive exploration of the weight of output space is pursued, the AFCM-based strategy obtains many times the fourth best and sometimes the third best performance among all the six strategies. As for the CFCM-based strategy, since no optimization has been used to build the model, it is not surprising that many times it could not even beat the benchmark strategy where output information has never been used when building the FRBM.

8.5 Summary

In this chapter, we highlighted a knowledge tidbit that could be beneficial to building the FRBM, that is, *if two outputs are far apart then their corresponding inputs should be put in the different clusters*. We proposed two different methods including PFCM-, RFCM(a), and RFCM(b)-based strategies to implement this knowledge tidbit when building the FRBM. From the experimental results, we observed the advantage of the proposed methods over some of those methods where either the information of the output space has not been considered (i.e., the LSE-based strategy) or that of the output space is biasedly (i.e., the AFCM-based strategy) or inadequately (i.e., the CFC-based strategy) considered. The proposed methods are complementary with each other, and each of them could show better performance than others in some scenarios.

Table 8.8. Comparison of RMSE for different strategies (only results of the first 8 data sets are reported).

Data #	Models	$c = 4$				$c = 6$				$c = 8$			
		Training	Testing	%P	m_{opt}	Training	Testing	%P	m_{opt}	Training	Testing	%P	m_{opt}
1	LSE	4.20±0.17	4.19±0.28	—	2.2	3.71±0.15	3.70±0.26	—	1.6	3.68±0.21	3.80±0.36	—	1.7
	AFCM	3.55±0.16	3.58±0.17	14.5	1.6	3.51±0.11	3.51±0.12	4.9	1.3	3.34±0.19	3.47±0.18	8.7	1.4
	CFCM	4.93±0.33	4.88±0.41	-16.6	1.6	4.23±0.30	4.10±0.29	-11.1	1.3	3.80±0.25	4.01±0.49	-5.6	1.5
	PFCM	**2.81±0.15***	3.39±0.26	19.0	1.7	**2.30±0.25***	**3.07±0.16***	**16.9**	1.3	**2.08±0.13***	**3.09±0.11***	**18.6**	1.4
	RFCM(a)	3.13±0.15	3.49±0.42	16.7	1.2	3.37±0.17	3.54±0.13	4.3	2.7	3.50±0.30	3.53±0.16	7.0	3.0
	RFCM(b)	2.96±0.50	**3.33±0.31***	**20.6**	1.1	2.57±0.11	3.17±0.17	14.2	1.3	2.39±0.28	3.20±0.13	15.8	1.4
2	LSE	0.49±0.03	0.50±0.03	—	1.4	0.46±0.02	0.46±0.04	—	1.7	0.46±0.02	0.46±0.02	—	1.5
	AFCM	0.44±0.01	0.44±0.02	12.1	1.5	0.44±0.02	0.44±0.04	4.2	1.3	0.44±0.02	0.44±0.02	5.3	1.5
	CFCM	0.51±0.02	0.51±0.03	-1.8	1.5	0.48±0.02	0.48±0.04	-4.1	1.7	0.45±0.02	0.44±0.02	3.4	1.5
	PFCM	0.32±0.03	0.43±0.02	14.7	1.5	0.28±0.03	0.42±0.03	8.6	1.7	**0.24±0.02***	**0.41±0.03***	**10.6**	1.5
	RFCM(a)	0.33±0.02	0.43±0.04	13.5	1.9	0.37±0.02	0.44±0.03	4.2	2.8	0.36±0.01	0.44±0.04	4.7	2.5
	RFCM(b)	**0.28±0.01***	**0.42±0.03***	**15.5**	1.9	**0.26±0.04***	**0.42±0.03***	**9.1**	1.3	0.26±0.04	0.42±0.03	9.6	1.5
3	LSE	3.71±0.42	3.79±0.39	—	1.1	3.67±0.18	3.76±0.26	—	1.4	3.67±0.11	3.73±0.14	—	1.3
	AFCM	3.27±0.09	3.34±0.12	11.9	1.2	2.89±0.10	3.00±0.14	20.0	1.3	2.75±0.14	2.83±0.18	24.3	1.4
	CFCM	4.76±0.11	4.73±0.28	-24.9	1.2	3.77±0.10	3.91±0.15	-4.2	1.3	3.38±0.06	3.48±0.08	6.7	1.3
	PFCM	2.52±0.09	3.23±0.19	14.8	1.8	1.96±0.55	3.01±0.26	19.9	1.4	1.97±0.53	2.89±0.29	22.5	1.6
	RFCM(a)	3.93±0.70	3.87±0.56	-2.2	1.8	3.30±0.25	3.47±0.17	7.5	1.7	3.11±0.54	3.36±0.30	10.1	1.4
	RFCM(b)	**2.34±0.09***	**3.12±0.20***	**17.7**	1.6	**1.69±0.18***	**2.84±0.16***	**24.5**	1.8	**1.68±0.20***	**2.81±0.17***	**24.7**	1.6
4	LSE	9.88±0.55	9.10±0.73	—	1.8	9.44±0.60	8.58±0.89	—	1.4	9.02±0.96	8.89±1.15	—	1.5
	AFCM	9.33±0.42	8.51±0.63	6.4	1.8	9.21±0.88	8.46±1.05	1.4	1.7	9.43±0.26	8.06±0.40	9.3	2.2
	CFCM	15.16±6.66	15.96±6.20	-75.4	2.0	15.45±7.69	16.66±6.51	-94.1	2.4	14.71±5.71	16.23±5.28	-82.6	1.3
	PFCM	8.14±0.78	8.12±0.67	10.8	1.8	7.14±0.87	**7.74±0.89***	**9.8**	1.2	7.82±0.90	8.42±0.82	5.3	1.1
	RFCM(a)	8.06±0.86	8.19±0.61	10.0	1.8	7.75±1.54	8.32±0.97	3.1	1.4	8.10±0.49	7.76±0.51	12.7	2.2

	RFCM(b)	**7.33±0.72***	**8.02±0.57***	**11.9**	1.8	**6.73±0.86***	7.83±0.90	8.7	1.4	**6.37±0.50***	**7.34±0.40***	**17.4**	2.2
5	LSE	3.31±0.18	3.36±0.27	—	1.7	3.29±0.11	3.40±0.14	—	2.3	3.21±0.10	3.29±0.14	—	1.6
	AFCM	2.70±0.06	2.76±0.08	18.0	1.2	2.56±0.14	2.67±0.16	21.7	1.3	2.47±0.10	2.57±0.15	21.9	1.3
	CFCM	3.04±0.18	3.07±0.18	8.7	1.4	2.70±0.09	2.70±0.08	20.8	1.5	2.65±0.04	2.74±0.12	16.9	1.5
	PFCM	2.18±0.12	2.29±0.11	31.9	1.6	**1.54±0.06***	**2.04±0.07***	**40.0**	1.5	1.54±0.13	2.08±0.10	36.8	1.3
	RFCM(a)	2.23±0.23	2.39±0.12	29.0	1.6	2.26±0.10	2.38±0.10	30.2	2.3	2.27±0.17	2.34±0.07	28.9	2.8
	RFCM(b)	**1.67±0.14***	**2.14±0.09***	**36.2**	1.4	1.55±0.11	2.08±0.07	38.8	1.5	**1.37±0.17***	**2.06±0.09***	**37.4**	1.9
6	LSE	6.76±0.61	7.09±0.87	—	1.4	6.41±0.61	6.04±0.51	—	1.4	5.82±0.41	5.65±0.39	—	1.4
	AFCM	5.70±0.27	5.27±0.59	25.7	1.9	5.57±0.52	5.08±0.64	16.0	1.4	5.22±0.44	4.92±0.49	12.9	1.4
	CFCM	6.59±0.28	6.85±0.44	3.3	1.4	5.73±0.33	6.04±0.50	0.1	1.3	5.31±0.56	5.81±0.54	-2.8	1.3
	PFCM	4.64±0.56	4.75±0.88	32.9	1.9	4.14±0.72	4.44±0.51	26.6	1.4	3.96±0.58	4.47±0.61	20.8	1.4
	RFCM(a)	5.21±0.82	5.21±0.52	26.5	1.1	5.10±0.51	5.11±0.70	15.5	1.4	4.96±0.50	5.16±0.57	8.6	1.8
	RFCM(b)	**4.14±0.49***	**4.55±0.53***	**35.8**	1.9	**3.41±0.39***	**4.22±0.61***	**30.2**	1.4	**2.96±0.51***	**4.23±0.52***	**25.0**	1.4
7	LSE	28.78±0.85	28.47±1.94	—	1.6	26.16±2.06	26.12±1.14	—	1.3	22.15±1.04	22.99±1.37	—	1.6
	AFCM	24.06±0.76	23.84±1.46	16.3	1.6	21.39±0.60	21.15±1.10	19.0	1.5	18.56±0.81	19.70±1.58	14.3	1.7
	CFCM	43.37±2.24	42.78±2.21	-50.2	1.5	31.98±2.08	30.66±1.76	-17.4	1.4	27.49±2.46	26.85±1.97	-16.8	1.3
	PFCM	19.99±1.50	19.24±2.25	32.5	2.2	16.14±0.94	17.08±2.56	34.6	1.5	12.50±1.13	**14.68±1.44***	**36.2**	1.6
	RFCM(a)	25.12±0.97	23.36±1.82	18.0	2.4	23.16±2.05	22.53±1.42	13.7	2.1	22.20±2.62	21.48±3.73	6.6	1.3
	RFCM(b)	**16.52±0.64***	**17.08±1.63***	**40.0**	1.6	**14.46±1.69***	**15.97±1.78***	**38.9**	1.5	**12.34±1.78***	15.06±1.77	34.5	2.1
8	LSE	1.08±0.05	1.09±0.11	—	1.9	0.81±0.05	0.81±0.06	—	1.9	0.59±0.03	0.60±0.03	—	1.9
	AFCM	0.98±0.12	0.99±0.10	8.9	1.5	0.74±0.02	0.72±0.04	11.9	2.6	0.60±0.03	0.61±0.05	-2.6	1.7
	CFCM	1.81±0.03	1.82±0.07	-67.2	1.7	1.11±0.03	1.13±0.03	-38.5	1.7	0.85±0.01	0.84±0.03	-40.8	1.4
	PFCM	0.86±0.07	0.78±0.11	28.6	1.5	0.61±0.03	0.51±0.04	37.2	1.9	0.50±0.04	0.47±0.03	21.4	1.5
	RFCM(a)	0.96±0.05	0.96±0.08	11.8	2.1	0.72±0.04	0.72±0.05	11.3	2.2	0.60±0.05	0.60±0.05	-0.9	1.7
	RFCM(b)	**0.77±0.09***	**0.75±0.12***	**31.5**	2.3	**0.51±0.03***	**0.46±0.05***	**43.0**	2.6	**0.46±0.03***	**0.45±0.05***	**24.3**	2.5

Chapter 9 Conclusions and Future Studies

To better explore the structure of data featured with emerging characteristics (distributed, granular, big, and supervised are considered in this dissertation), we have proposed or refined several more advanced fuzzy clustering algorithms. Some of these clustering algorithms have been used to partition the input space of the fuzzy rule-based model (FRBM), and these algorithms turned out to be helpful for 1) expanding the application scenarios of the FRBM, and 2) improving the performance of the FRBM. In this chapter, we briefly summarize the major contributions and point out what could be some interesting research topics for the future studies.

9.1 Major Contributions

(1) We experimentally verified the long-questioned problem that whether it is necessary to reorder the data structures (partition matrices) during the collaboration phases in HCFC. Then for each data site, by striking a balance between exploiting structure information from other data sites and exploring its own local information, we optimized the collaboration strength among data sites. Finally, to form a stable and representative global structure of the distributed data, the granular partition matrix was formed based on the locally revealed data structures.

(2) We proposed a systematic process for granular data clustering. First, since most research did not give clear descriptions about the origin of the granular data, we gave the principle of justifiable granularity (PJG)-based approach to highlight this point. By considering the quality of the formed information granules, we proposed the weighted granular clustering method based on the standard FCM; here instead of using the entire granule information, parameters of each granule were used in clustering. Finally, to evaluate the performance of the clustering algorithm, a granular reconstruction

criterion was proposed; this criterion could also be used to optimize the fuzzification coefficient and the cluster number.

(3) When the heterogenous information granules are encountered, we proposed two approximation methods to transform the irregular-shape information granules (fuzzy sets) into the simple-shape ones. With gradient-based method, we optimized a trapezoidal fuzzy set (TFS) such that its distance to the original fuzzy set was minimized. To capture more information present in the original fuzzy set, we used the interval type-2 trapezoidal fuzzy set (IT2 TFS) to cover the original fuzzy set as much as possible but still maintain itself with a sound semantic explanation.

(4) We proposed the hyperplane-based method to divide the big data into different subsets. This method turned out to be a much effective method than other methods (e.g., the sampling method) in finding out the structures contained in the data. Besides, we also highlighted that clustering algorithms should be implemented with a consideration of the requirements in reality. We gave two clustering strategies when either a large number or a small number of clusters are needed for the clustering task.

(5) We applied the horizontal collaborative fuzzy clustering (HCFC) to building the FRBM. The HCFC algorithm improves the FRBMs in terms of both expanding its application scenarios and increasing its performance. It enables that FRBMs could be constructed without gathering the input and output data into the same site. And this scenario happens when data privacy is a major concern between two data sites seeking for the collaboration. Also, the HCFC algorithm serves as a new mechanism to divide the input space considering the information from the output space.

(6) We pointed out that the knowledge tidbit, that is *when two output values are far apart their corresponding input values should not be put in a same cluster*, could be very helpful when building the FRBMs. Specifically, we proposed two methods to implement this piece of knowledge when dividing the input space of the FRBM. It

turned out that the proposed methods are very helpful for improving the model performance in terms of accuracy.

9.2 Future Studies

Although many interesting and important topics have been investigated so far, we point out that there are still many ideas which are worth to be further investigated. We list several directions which are of interest to be explored in our future studies.

(1) *Big data clustering: a hybrid approach of collaborative and granular clustering*

The big data considered so far is characterized with a large sample size. But what if the encountered data is big in terms of both sample size and feature dimensionality? The task in this topic is to cluster this kind of big data. As a matter of fact, finding the data structure of such a big data is not sufficiently studied in the literature; and finding data structure in data with a high feature dimensionality is quite difficult. Hence, in this topic we try to solve this problem in three major steps. First, we reduce the feature space through the HCFC algorithm; second, information granules are formed on this reduced data set; finally, granular clustering is used to find the granular data structure.

(2) *Vertical collaborative fuzzy clustering in the high dimensional feature space*

We have studied and refined the HCFC algorithm to cluster the distributed data; but there the focused data are characterized with the same observations and generally different features. What if we encounter data with different observations but the same features? In this topic we try to improve the vertical collaborative fuzzy clustering (VCFC) such that it is better used to cluster this kind of data in a high dimensional feature space. The issue here is that the data structure tends to be overwhelmed in the high dimensional feature space (a similar problem in the big data clustering with high dimensionality). To solve this problem, we still use the HCFC to reduce the feature dimensionality in each distributed data set. But here, the granular features will be formed after the collaborative clustering. VCFC could

be used to cluster the data in a much smaller feature space now, and the finally reconciled data structures are granular prototypes.

(3) *Fuzzy rule-based model in high dimensional feature space*

The motivation of this topic is to make the clusters (fuzzy relations) formed in the condition part of the FRBM more meaningful when the data used to build the model are located in a high dimensional feature. Note that when facing high dimensionality it is possible to get all the membership degrees similar to each other, which may negatively impact the performance of the TS model. Two methods are considered to solve this problem. Method (a): the feature clustering-based feature selection is proposed to select the suitable features. Data clustering is then performed on this reduced feature space to form the condition parts of the TS model. Method (b): bi-clustering (i.e., samples and feature are clustered simultaneously) could be used to form condition parts of the model.

(4) *Collaborative development of fuzzy models*

We have studied clustering of the distributed data which are not allowed to be gathered together due to some constraints, what if we would like to build a global TS model on these distributed data? For each data site of the distributed data, the local TS model could be constructed. To build a global TS model, the collaborative clustering algorithm could be used to generate the collaborated condition parts of each local TS model. The PJG could be used to make the output of the TS models in a form of interval.

Bibliography

[1] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[2] J. C. Dunn, "Well-separated clusters and optimal fuzzy partitions," *J. Cybern.*, vol. 4, no. 1, pp. 95–104, 1974.

[3] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, no. 2–3, pp. 191–203, 1984.

[4] N. Päivinen, "Clustering with a minimum spanning tree of scale-free-like structure," *Pattern Recognit. Lett.*, vol. 26, no. 7, pp. 921–930, 2005.

[5] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, no. 11, pp. 1101–1113, 1993.

[6] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *Comput. J.*, vol. 26, no. 4, pp. 354–359, 1983.

[7] G. Karypis, E.-H. S. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer (Long. Beach. Calif).*, no. 8, pp. 68–75, 1999.

[8] H. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 1, no. 3, pp. 231–240, 2011.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *Kdd*, 1996, vol. 96, no. 34, pp. 226–231.

[10] L. A. Zadeh, "Fuzzy Sets-Information and Control-1965," *Inf. Control*, 1965.

[11] R. R. Yager and D. P. Filev, "Essentials of fuzzy modeling and control," *New York*, vol. 388, 1994.

[12] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013.

[13] W. Pedrycz, “Collaborative fuzzy clustering,” *Pattern Recognit. Lett.*, vol. 23, no. 14, pp. 1675–1686, 2002.

[14] W. Pedrycz, V. Loia, and S. Senatore, “P-FCM: a proximity—based fuzzy clustering,” *Fuzzy Sets Syst.*, vol. 148, no. 1, pp. 21–41, 2004.

[15] M. Setnes, R. Babuska, and H. B. Verbruggen, “Rule-based modeling: Precision and transparency,” *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.*, vol. 28, no. 1, pp. 165–169, 1998.

[16] E. H. Mamdani, “Application of fuzzy algorithms for control of simple dynamic plant,” in *Proceedings of the institution of electrical engineers*, 1974, vol. 121, no. 12, pp. 1585–1588.

[17] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” in *Readings in fuzzy sets for intelligent systems*, Elsevier, 1993, pp. 387–403.

[18] X. Zhu, W. Pedrycz, and Z. Li, “Granular Models and Granular Outliers,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 6, pp. 3835–3846, 2018.

[19] X. Zhu, W. Pedrycz, and Z. Li, “A Design of Granular Takagi-Sugeno Fuzzy Model Through the Synergy of Fuzzy Subspace Clustering and Optimal Allocation of Information Granularity,” *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 2499–2509, 2018.

[20] W. Pedrycz and H. Izakian, “Cluster-Centric Fuzzy Modeling,” *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 6, pp. 1585–1597, 2014.

[21] W. Pedrycz and W. Homenda, “Building the fundamentals of granular computing: A principle of justifiable granularity,” *Appl. Soft Comput. J.*, vol. 13, no. 10, pp. 4209–4218, 2013.

[22] L. A. Zadeh, “The concept of a linguistic variable and its application to approximate reasoning—I,” *Inf. Sci. (Ny).*, vol. 8, no. 3, pp. 199–249, 1975.

[23] J. M. Mendel, “Uncertain rule-based fuzzy logic systems: Introduction and new,” *Dir. Ed. USA Prentice Hall*, pp. 25–200, 2000.

[24] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95 - International Conference on Neural Networks*, 1995, vol. 4, pp. 1942–1948.

[25] X. Zhang, W. K. Cheung, and Y. Ye, “Mining from distributed and abstracted data,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 6, no. 5, pp. 167–176, 2016.

[26] Y. Jiang, F.-L. Chung, S. Wang, Z. Deng, J. Wang, and P. Qian, “Collaborative fuzzy clustering from multiple weighted views,” *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 688–701, 2014.

[27] F. D. A. T. De Carvalho, Y. Lechevallier, and F. M. De Melo, “Relational partitioning fuzzy clustering algorithms based on multiple dissimilarity matrices,” *Fuzzy Sets Syst.*, vol. 215, pp. 1–28, 2013.

[28] F. de A. T. de Carvalho, F. M. de Melo, and Y. Lechevallier, “A multi-view relational fuzzy c-medoid vectors clustering algorithm,” *Neurocomputing*, vol. 163, pp. 115–123, 2015.

[29] J. Zhou, C. L. P. Chen, L. Chen, and H.-X. Li, “A collaborative fuzzy clustering algorithm in distributed network environments,” *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 6, pp. 1443–1456, 2013.

[30] G. Cleuziou, M. Exbrayat, L. Martin, and J.-H. Sublemontier, “CoFKM: A centralized method for multiple-view clustering,” in *2009 Ninth IEEE International Conference on Data Mining*, 2009, pp. 752–757.

[31] M. Zarinbal, M. H. F. Zarandi, and I. B. Turksen, “Relative entropy collaborative fuzzy clustering method,” *Pattern Recognit.*, vol. 48, no. 3, pp. 933–940, 2015.

[32] M. Prasad, L. Siana, D.-L. Li, C.-T. Lin, Y. T. Liu, and A. Saxena, “A preprocessed induced partition matrix based collaborative fuzzy clustering for data analysis,” in

2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2014, pp. 1553–1558.

[33] W. Pedrycz and P. Rai, "A multifaceted perspective at data analysis: A study in collaborative intelligent agents," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 38, no. 4, pp. 1062–1072, 2008.

[34] V. Loia, W. Pedrycz, and S. Senatore, "Semantic web content analysis: A study in proximity-based collaborative clustering," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 6, pp. 1294–1312, 2007.

[35] M. Prasad, K.-P. Chou, A. Saxena, O. P. Kawrtiya, D.-L. Li, and C.-T. Lin, "Collaborative fuzzy rule learning for Mamdani type fuzzy inference system with mapping of cluster centers," in *2014 IEEE Symposium on Computational Intelligence in Control and Automation (CICA)*, 2014, pp. 1–6.

[36] K.-P. Chou, M. Prasad, Y. Y. Lin, S. Joshi, C.-T. Lin, and J. Y. Chang, "Takagi-Sugeno-Kang type collaborative fuzzy rule based system," in *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, 2014, pp. 315–320.

[37] C.-T. Lin, M. Prasad, and J.-Y. Chang, "Designing mamdani type fuzzy rule using a collaborative FCM scheme," in *2013 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*, 2013, pp. 279–282.

[38] M. Prasad, Y. Y. Lin, C.-T. Lin, M. J. Er, and O. K. Prasad, "A new data-driven neural fuzzy system with collaborative fuzzy clustering mechanism," *Neurocomputing*, vol. 167, pp. 558–568, 2015.

[39] Z. Han, J. Zhao, Q. Liu, and W. Wang, "Granular-computing based hybrid collaborative fuzzy clustering for long-term prediction of multiple gas holders levels," *Inf. Sci. (Ny).*, vol. 330, pp. 175–185, 2016.

[40] W. Pedrycz and P. Rai, "Collaborative clustering with the use of Fuzzy C-Means

and its quantification," *Fuzzy Sets Syst.*, vol. 159, no. 18, pp. 2399–2427, 2008.

[41] L. F. S. Coletta, L. Vendramin, E. R. Hruschka, R. J. G. B. Campello, and W. Pedrycz, "Collaborative fuzzy clustering algorithms: Some refinements and design guidelines," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 3, pp. 444–462, 2011.

[42] R. Falcón, B. Depaire, K. Vanhoof, and A. Abraham, "Towards a suitable reconciliation of the findings in collaborative fuzzy clustering," in *2008 Eighth International Conference on Intelligent Systems Design and Applications*, 2008, vol. 3, pp. 652–657.

[43] R. Falcon, G. Jeon, R. Bello, and J. Jeong, "Learning collaboration links in a collaborative fuzzy clustering environment," in *Mexican International Conference on Artificial Intelligence*, 2007, pp. 483–495.

[44] M. Ghassany, N. Grozavu, and Y. Bennani, "Collaborative clustering using prototype-based techniques," *Int. J. Comput. Intell. Appl.*, vol. 11, no. 03, p. 1250017, 2012.

[45] P. Rastin, G. Cabanes, N. Grozavu, and Y. Bennani, "Collaborative clustering: How to select the optimal collaborators?," in *2015 IEEE Symposium Series on Computational Intelligence*, 2015, pp. 787–794.

[46] J. Sublime, N. Grozavu, Y. Bennani, and A. Cornuéjols, "Vertical collaborative clustering using generative topographic maps," in *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2015, pp. 199–204.

[47] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logist.*, vol. 52, no. 1, pp. 7–21, 2005.

[48] W. Pedrycz and A. Bargiela, "An optimization of allocation of information granularity in the interpretation of data structures: toward granular fuzzy clustering," *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 42, no. 3, pp. 582–590, 2011.

[49] J. T. Yao, A. V Vasilakos, and W. Pedrycz, “Granular computing: perspectives and challenges,” *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1977–1989, 2013.

[50] J. Zhao, Z. Han, W. Pedrycz, and W. Wang, “Granular model of long-term prediction for energy system in steel industry,” *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 388–400, 2015.

[51] W. Pedrycz and X. Wang, “Designing fuzzy sets with the use of the parametric principle of justifiable granularity,” *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 2, pp. 489–496, 2015.

[52] S. Wang, J. Watada, and W. Pedrycz, “Granular robust mean-CVaR feedstock flow planning for waste-to-energy systems under integrated uncertainty,” *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1846–1857, 2014.

[53] W. Pedrycz, R. Al-Hmouz, A. Morfeq, and A. Balamash, “The design of free structure granular mappings: the use of the principle of justifiable granularity,” *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2105–2113, 2013.

[54] R. J. Hathaway, J. C. Bezdek, and W. Pedrycz, “A parametric model for fusing heterogeneous fuzzy data,” *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 3, pp. 270–281, 1996.

[55] W. Pedrycz, J. C. Bezdek, R. J. Hathaway, and G. W. Rogers, “Two nonparametric models for fusing heterogeneous fuzzy data,” *IEEE Trans. Fuzzy Syst.*, vol. 6, no. 3, pp. 411–425, 1998.

[56] M.-S. Yang and C.-H. Ko, “On a class of fuzzy c-numbers clustering procedures for fuzzy data,” *Fuzzy sets Syst.*, vol. 84, no. 1, pp. 49–60, 1996.

[57] M.-S. Yang and H.-H. Liu, “Fuzzy clustering procedures for conical fuzzy vector data,” *Fuzzy Sets Syst.*, vol. 106, no. 2, pp. 189–200, 1999.

[58] W.-L. Hung and M.-S. Yang, “Fuzzy clustering on LR-type fuzzy numbers with an application in Taiwanese tea evaluation,” *Fuzzy sets Syst.*, vol. 150, no. 3, pp. 561–

577, 2005.

[59] W.-L. Hung, M.-S. Yang, and E. S. Lee, “A robust clustering procedure for fuzzy data,” *Comput. Math. with Appl.*, vol. 60, no. 1, pp. 151–165, 2010.

[60] M. B. Ferraro and P. Giordani, “On possibilistic clustering with repulsion constraints for imprecise data,” *Inf. Sci. (Ny).*, vol. 245, pp. 63–75, 2013.

[61] W. Pedrycz and A. Bargiela, “Granular clustering: a granular signature of data,” *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 32, no. 2, pp. 212–224, 2002.

[62] P. D’Urso and P. Giordani, “A weighted fuzzy c-means clustering model for fuzzy data,” *Comput. Stat. Data Anal.*, vol. 50, no. 6, pp. 1496–1523, 2006.

[63] R. Coppi, P. D’Urso, and P. Giordani, “Fuzzy and possibilistic clustering for fuzzy data,” *Comput. Stat. Data Anal.*, vol. 56, no. 4, pp. 915–927, 2012.

[64] P. D’Urso and L. De Giovanni, “Robust clustering of imprecise data,” *Chemom. Intell. Lab. Syst.*, vol. 136, pp. 58–80, 2014.

[65] A. Gacek and W. Pedrycz, “Clustering granular data and their characterization with information granules of higher type,” *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 4, pp. 850–860, 2014.

[66] S. Auephanwiriyakul and J. M. Keller, “Analysis and efficient implementation of a linguistic fuzzy c-means,” *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 5, pp. 563–582, 2002.

[67] P. Grzegorzewski, “Metrics and orders in space of fuzzy numbers,” *Fuzzy sets Syst.*, vol. 97, no. 1, pp. 83–94, 1998.

[68] S. Effati, H. S. Yazdi, and A. J. Sharahi, “Fuzzy clustering algorithm for fuzzy data based on α-cuts,” *J. Intell. Fuzzy Syst.*, vol. 24, no. 3, pp. 511–519, 2013.

[69] M. H. F. Zarandi and Z. S. Razaee, “A fuzzy clustering model for fuzzy data with outliers,” *Int. J. Fuzzy Syst. Appl.*, vol. 1, no. 2, pp. 29–42, 2011.

[70] K. Y. Chan, C. K. Kwong, and B. Q. Hu, “Market segmentation and ideal point

identification for new product design using fuzzy data compression and fuzzy clustering methods," *Appl. Soft Comput.*, vol. 12, no. 4, pp. 1371–1378, 2012.

[71] R. Yang, Z. Wang, P.-A. Heng, and K.-S. Leung, "Classification of heterogeneous fuzzy data by Choquet integral with fuzzy-valued integrand," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 5, pp. 931–942, 2007.

[72] H. S. Yazdi, M. GhasemiGol, S. Effati, A. Jiriani, and R. Monsefi, "Hierarchical tree clustering of fuzzy number," *J. Intell. Fuzzy Syst.*, vol. 26, no. 2, pp. 541–550, 2014.

[73] L. Coroianu, M. Gagolewski, and P. Grzegorzewski, "Nearest piecewise linear approximation of fuzzy numbers," *Fuzzy Sets Syst.*, vol. 233, pp. 26–51, 2013.

[74] G. Wang and J. Li, "Approximations of fuzzy numbers by step type fuzzy numbers," *Fuzzy Sets Syst.*, vol. 310, pp. 47–59, 2017.

[75] P. Grzegorzewski and E. Mrówka, "Trapezoidal approximations of fuzzy numbers," *Fuzzy Sets Syst.*, vol. 153, no. 1, pp. 115–135, 2005.

[76] T. Allahviranloo and M. A. Firozja, "Note on 'Trapezoidal approximation of fuzzy numbers,'" *Fuzzy Sets Syst.*, vol. 158, no. 7, pp. 755–756, 2007.

[77] C.-T. Yeh, "A note on trapezoidal approximations of fuzzy numbers," *Fuzzy Sets Syst.*, vol. 158, no. 7, pp. 747–754, 2007.

[78] A. Ban, "Approximation of fuzzy numbers by trapezoidal fuzzy numbers preserving the expected interval," *Fuzzy Sets Syst.*, vol. 159, no. 11, pp. 1327–1344, 2008.

[79] C.-T. Yeh, "Trapezoidal and triangular approximations preserving the expected interval," *Fuzzy Sets Syst.*, vol. 159, no. 11, pp. 1345–1353, 2008.

[80] P. Grzegorzewski and K. Pasternak-Winiarska, "Natural trapezoidal approximations of fuzzy numbers," *Fuzzy Sets Syst.*, vol. 250, pp. 90–109, 2014.

[81] P. Grzegorzewski and E. Mrówka, "Trapezoidal approximations of fuzzy numbers—revisited," *Fuzzy Sets Syst.*, vol. 158, no. 7, pp. 757–768, 2007.

[82] P. Grzegorzewski, “Trapezoidal approximations of fuzzy numbers preserving the expected interval—algorithms and properties,” *Fuzzy Sets Syst.*, vol. 159, no. 11, pp. 1354–1364, 2008.

[83] C.-T. Yeh and H.-M. Chu, “Approximations by LR-type fuzzy numbers,” *Fuzzy Sets Syst.*, vol. 257, pp. 23–40, 2014.

[84] C.-T. Yeh, “On improving trapezoidal and triangular approximations of fuzzy numbers,” *Int. J. Approx. Reason.*, vol. 48, no. 1, pp. 297–313, 2008.

[85] F. Herrera and L. Martínez, “A 2-tuple fuzzy linguistic representation model for computing with words,” *IEEE Trans. fuzzy Syst.*, vol. 8, no. 6, pp. 746–752, 2000.

[86] D. Wu and J. M. Mendel, “Perceptual reasoning for perceptual computing: A similarity-based approach,” *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 6, pp. 1397–1411, 2009.

[87] E. K. P. Chong and S. H. Zak, *An introduction to optimization*, vol. 76. John Wiley & Sons, 2013.

[88] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, “Fuzzy c-means algorithms for very large data,” *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 6, pp. 1130–1146, Dec. 2012.

[89] D. Mullick, A. Garg, A. Bajaj, A. Garg, and S. A. B, “Ant colony based fuzzy c-means clustering for very large data,” in *Advances in Fuzzy Logic and Technology 2017*, vol. 641, Springer, 2017, pp. 578–591.

[90] R. J. Hathaway and J. C. Bezdek, “Extending fuzzy and probabilistic clustering to very large data sets,” *Comput. Stat. Data Anal.*, vol. 51, no. 1, pp. 215–234, 2006.

[91] J. K. Parker and L. O. Hall, “Accelerating fuzzy-c means using an estimated subsample size,” *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 5, pp. 1229–1244, 2013.

[92] S. Eschrich, L. O. Hall, and D. B. Goldgof, “Fast accurate fuzzy clustering through data reduction,” *IEEE Trans. Fuzzy Syst.*, 2003.

[93] P. Hore, L. O. Hall, D. B. Goldgof, Y. Gu, A. A. Maudsley, and A. Darkazanli, "A scalable framework for segmenting magnetic resonance images," *J. Signal Process. Syst.*, vol. 54, no. 1–3, pp. 183–203, 2009.

[94] Y. Wang, L. Chen, and J. P. Mei, "Incremental fuzzy clustering with multiple medoids for large data," *IEEE Trans. Fuzzy Syst.*, vol. 22, no. 6, pp. 1557–1568, Dec. 2014.

[95] N. Labroche, "New incremental fuzzy C medoids clustering algorithms," in *Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS*, 2010.

[96] P. Hore, L. O. Hall, and D. B. Goldgof, "Single pass fuzzy c means," *IEEE Int. Conf. Fuzzy Syst.*, 2007.

[97] N. Bharill and A. Tiwari, "Handling Big Data with Fuzzy Based Classification Approach," in *Advance Trends in Soft Computing*, Springer, 2014, pp. 219–227.

[98] J. Wu, Z. Wu, J. Cao, H. Liu, G. Chen, and Y. Zhang, "Fuzzy Consensus Clustering with Applications on Big Data," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1430–1445, 2017.

[99] N. Bharill, A. Tiwari, and A. Malviya, "Fuzzy based scalable clustering algorithms for handling big data using apache spark," *IEEE Trans. Big Data*, vol. 2, no. 4, pp. 339–352, 2016.

[100] N. Ghadiri, M. Ghaffari, and M. A. Nikbakht, "BigFCM: Fast, precise and scalable FCM on hadoop," *Futur. Gener. Comput. Syst.*, vol. 77, pp. 29–39, 2017.

[101] X. Li, J. Song, F. Zhang, X. Ouyang, and S. U. Khan, "MapReduce-based fast fuzzy c-means algorithm for large-scale underwater image segmentation," *Futur. Gener. Comput. Syst.*, vol. 65, pp. 90–101, 2016.

[102] B. Hosseini and K. Kiani, "FWCMR: A scalable and robust fuzzy weighted clustering based on MapReduce with application to microarray gene expression,"

Expert Syst. Appl., vol. 91, pp. 198–210, 2018.

[103] W. Pedrycz and A. V. Vasilakos, "Linguistic models and linguistic modeling," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 29, no. 6, pp. 745–757, 1999.

[104] W. Pedrycz, "Conditional fuzzy C-means," *Pattern Recognit. Lett.*, vol. 17, no. 6, pp. 625–631, 1996.

[105] D. Wang, X. J. Zeng, and J. A. Keane, "An output-constrained clustering approach for the identification of fuzzy systems and fuzzy granular systems," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 6, pp. 1127–1140, 2011.

[106] W. Pedrycz and J. V. de Oliveira, "A development of fuzzy encoding and decoding through fuzzy clustering," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 4, pp. 829–837, 2008.

[107] X. Hu, W. Pedrycz, and X. Wang, "Fuzzy classifiers with information granules in feature space and logic-based computing," *Pattern Recognit.*, vol. 80, pp. 156–167, 2018.

[108] Y. Shen and W. Pedrycz, "Collaborative fuzzy clustering algorithm: Some refinements," *Int. J. Approx. Reason.*, vol. 86, pp. 41–61, 2017.

[109] Q. Zhang and M. Mahfouf, "A hierarchical Mamdani-type fuzzy modelling approach with new training data selection and multi-objective optimisation mechanisms: A special application for the prediction of mechanical properties of alloy steels," in *Applied Soft Computing Journal*, 2011, vol. 11, no. 2, pp. 2419–2443.

[110] B. Hartmann, O. Bänfer, O. Nelles, A. Sodja, L. Teslić, and I. Škrjanc, "Supervised hierarchical clustering in fuzzy model identification," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 6, pp. 1163–1176, 2011.

[111] S. Askari, "A novel and fast MIMO fuzzy inference system based on a class of fuzzy clustering algorithms with interpretability and complexity analysis," *Expert Syst.*

Appl., vol. 84, pp. 301–322, 2017.

[112] X. P. Xie, D. Yue, and S. L. Hu, "Fuzzy control design of nonlinear systems under unreliable communication links: A systematic homogenous polynomial approach," *Inf. Sci. (Ny).*, vol. 370–371, pp. 763–771, 2016.

[113] X. Hu, W. Pedrycz, and X. Wang, "Granular fuzzy rule-based models: A study in a comprehensive evaluation and construction of fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 5, pp. 1342–1355, 2016.

[114] C. Li, J. Zhou, L. Chang, Z. Huang, and Y. Zhang, "T–S fuzzy model identification based on a novel hyperplane-shaped membership function," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 5, pp. 1364–1370, 2016.

[115] C. Li, W. Zou, N. Zhang, and X. Lai, "An evolving T–S fuzzy model identification approach based on a special membership function and its application on pump-turbine governing system," *Eng. Appl. Artif. Intell.*, vol. 69, pp. 93–103, 2018.

[116] W. Pedrycz and K.-C. Kwak, "Linguistic models as a framework of user-centric system modeling," *IEEE Trans. Syst. Man, Cybern. A Syst. Humans*, vol. 36, no. 4, pp. 727–745, 2006.

[117] J. Li, W. Pedrycz, and X. Wang, "A rule-based development of incremental models," *Int. J. Approx. Reason.*, vol. 64, pp. 20–38, 2015.

[118] C. Li, J. Zhou, B. Fu, P. Kou, and J. Xiao, "T–S fuzzy model identification with a gravitational search-based hyperplane clustering algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 2, pp. 305–317, 2011.

[119] M. Sugeno and T. Yasukawa, "A Fuzzy-Logic-Based Approach to Qualitative Modeling," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 1, pp. 7–31, 1993.

[120] E. Kim, M. Park, S. Ji, and M. Park, "A new approach to fuzzy modeling," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 3, pp. 328–337, 1997.

[121] X. Hu, W. Pedrycz, and X. Wang, "Optimal allocation of information granularity in

system modeling through the maximization of information specificity: A development of granular input space," *Appl. Soft Comput. J.*, vol. 42, pp. 410–422, 2016.

[122] J. M. Leski, "Fuzzy (c+p)-means clustering and its application to a fuzzy rule-based classifier: Toward good generalization and good interpretability," *IEEE Trans. fuzzy Syst.*, vol. 23, no. 4, pp. 802–812, 2014.

[123] A. D. Torshizi, L. Petzold, and M. Cohen, "Multivariate soft repulsive system identification for constructing rule-based classification systems: Application to trauma clinical data," *Neurocomputing*, vol. 245, pp. 77–85, 2017.

[124] X. Gu, F.-L. Chung, H. Ishibuchi, and S. Wang, "Imbalanced TSK fuzzy classifier by cross-class Bayesian fuzzy clustering and imbalance learning," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 47, no. 8, pp. 2005–2020, 2016.

[125] X. Hu, W. Pedrycz, and X. Wang, "Development of granular models through the design of a granular output spaces," *Knowledge-Based Syst.*, vol. 134, pp. 159–171, 2017.

[126] Y. Shen, W. Pedrycz, and X. Wang, "Clustering Homogeneous Granular Data: Formation and Evaluation," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1391–1402, 2019.

[127] Y. Shen, W. Pedrycz, and X. Wang, "Approximation of Fuzzy Sets by Interval Type-2 Trapezoidal Fuzzy Sets," *IEEE Trans. Cybern.*, pp. 1–13, 2019.