Granular Fuzzy Models: Construction, Analysis, and Design

by

Orion Fausto Reyes-Galaviz

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

Building abstract concepts is essential to humans when acquiring knowledge, realizing processing (reasoning), and communicating findings. Abstraction comes hand in hand with information granules and information granulation. When analyzing digital images, we form groups (clusters) of pixels, colors, and textures that constitute familiar objects. This paramount ability of forming groups of objects (information granules), manipulating them, and producing sound conclusions, is realized in an almost subconscious manner. Information granules are critical when representing and processing knowledge. They become instrumental when solving everyday problems. The objective of this dissertation is to design, analyze, and optimize granular fuzzy models in which information granules play a pivotal role. In the presented considerations, fuzzy models serve as a vehicle for the construction of granular fuzzy models, offering a convenient and efficient way to describe complex and nonlinear systems. Fuzzy models produce numeric results. In contrast, granular fuzzy models produce results in the form of information granules. In this study, we develop a novel approach to the design and optimization of fuzzy relational structures. Subsequently, we transform them into their granular counterpart by experimenting with different strategies to optimally allocate information granularity. This direct generalization (abstraction) of the fuzzy model comes as a sound alternative to solve a system of relational equations, and to assess the quality of the original model. Granular fuzzy models are also developed by exploiting a concept of information granularity. We propose an innovative model coming as a network of associations among information granules, which form the backbone of the overall construct. Interval information granules positioned in the output space induce information granules in the input space, and we develop different strategies to optimize these intervals. Further optimization of the model is proposed by optimally re-distributing the induced information granules. The performance of granular models is assessed by considering criteria of coverage and information specificity (information granularity). The associations are further exploited to construct a granular model that is fundamentally structured around information granules regarded as hyperboxes. We develop two novel methods to construct a family of hyperboxes in the input space; one by realizing some constrictions, and the other one by engaging an optimization mechanism. We also propose a new approach to optimally eliminate or reduce possible overlap among hyperboxes. The resulting information granules are compared and assessed in terms of their coverage, and the data captured by those is evaluated by proposing a three-step verification process. Furthermore, in this research we propose a novel approach to refine information granules produced by the Fuzzy C-Means algorithm, and optimize their representation and classification abilities. We develop different strategies to adjust a location of the prototypes so that a certain performance index becomes optimized. The granular fuzzy models are optimized by population-based algorithms, such as particle swarm optimization and differential evolution. The experimental studies involve synthetic data and benchmark datasets from publicly-available repositories.

Preface

The research conducted in this thesis was performed under the supervision of Dr. Witold Pedrycz, and it was supported by the National Council of Science and Technology (CONACYT) Mexico under grant no. 213501/308562 and by the Autonomous University of Tlaxcala (UAT) Mexico. Most of the experiments presented in this thesis were completed by using the general Linux cluster for research use, provided and maintained by the Academic Information and Communication Technologies (AICT) department at the University of Alberta.

Chapter 4 of this thesis has been published as O. F. Reyes-Galaviz, and W. Pedrycz, "Fuzzy relational structures: Learning alternatives for fuzzy modeling," *Joint IFSA World Congress and NAFIPS Annual Meeting* (*IFSA/NAFIPS*), Edmonton AB, Canada, pp. 374-379, 2013, IEEE Press. I was responsible for the idea development, data collection and analysis, as well as the manuscript composition. Dr. W. Pedrycz was the supervisory author and was involved in the concept formation and the manuscript composition.

Chapter 5 of this thesis has been published as O. F. Reyes-Galaviz, and W. Pedrycz, "Granular Fuzzy Models: Analysis, Design, and Evaluation," *International Journal for Approximate Reasoning*, vol. 64, pp. 1-19, 2015. I was responsible for the idea development, data collection and analysis, as well as the manuscript composition. Dr. W. Pedrycz was the supervisory author and was involved in the concept formation and the manuscript composition

Chapter 6 of this thesis has been published as O. F. Reyes-Galaviz, and W. Pedrycz, "Granular Fuzzy Modeling with Evolving Hyperboxes in Multi-Dimensional Space of Numerical Data," *Neurocomputing*, vol. 168, Issue 30, pp. 240-253, 2015. I was responsible for the idea development, data collection, and analysis, as well as the manuscript composition. Dr. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition.

Chapter 7 of this thesis has been submitted to "*Fuzzy Sets and Systems*" as O. F. Reyes-Galaviz, and W. Pedrycz, "Enhancement of the Classification and Reconstruction Performance of Fuzzy C-Means with Refinements of Prototypes." I was responsible for the idea development, data collection, and analysis, as well as the manuscript composition. Dr. W. Pedrycz was the supervisory author and was involved with the concept formation and manuscript composition.

Acknowledgments

First and foremost I would like to thank my supervisor Professor Witold Pedrycz, to whom I express my deepest respect and gratitude. I profoundly appreciate all your contributions of ideas, innovations, and time as well as your patience, enthusiasm, and support (both moral and financial), which made my PhD experience more productive and stimulating. Your enjoyment and passion for research was (and still is) contagious and motivational for me, even during difficult times throughout my PhD pursuit. It has been an honor being your PhD student.

I would also like to thank the members of my supervisory committee, Professor Petr Musilek, Professor Marek Reformat, and my external examiner Professor Fernando Gomide for their brilliant comments and suggestions. I deeply appreciate all your contributions to my formation as a PhD student during my time at the University, and for the excellent example you provide as successful researchers and professors.

I fully acknowledge the supporting grant from the National Council of Science and Technology (CONACYT) Mexico under grant no. 213501/308562, and the financial assistance provided by the Autonomous University of Tlaxcala (UAT), Mexico.

Additionally, I would like to thank my parents Carlos A. Reyes Garcia and Adriana Galaviz Diaz, and to my siblings Rigel, Adrian, and Rogelio for their moral support. Dad, thank you for raising me with a love for knowledge, and for supporting me on all my pursuits, you are an exceptional father, mentor, and an incomparable role model who helped me become a researcher just like you. Your guidance, stories of your PhD quest, experiences, and wonderful history motivated me to follow a similar path. I look forward to writing more research papers with you. Mom, you are an outstanding mother, and an excellent computer engineer. You pursued your studies while taking care of us and showed us that all that is needed is determination and persistence to successfully finish your studies. I treasure the memories of us (me and my sister) playing at your University while you were studying in your classroom. Thank you for establishing an example to complete everything we set our minds on. Sister and brothers, thank you for being there for me, either on the phone, video call, or back home. You always had a warm smile and welcoming arms, which helped me get through hard times during my PhD pursuit.

I would also like to thank my girlfriend Sandra Duran for her support, encouragement, and patience. You started your PhD pursuit earlier than me, and now you are a successful woman scientist and a researcher, thank you for being an outstanding example for the determination and persistence needed to effectively achieve your goals.

Furthermore, I'm forever be thankful to the staff at the Computer Engineering department for always being so helpful and friendly, especially Pinder Bains and Kathleen Forbes. You were genuinely nice and always willing to help, and I'm grateful to have interacted with you throughout all these years.

And last but not least, I would like to thank my family and friends for all their support, love, and encouragement, and for helping me make my stay in Canada a wonderful and enjoyable experience. If I have forgotten anyone, I apologize.

Orion Fausto Reyes Galaviz
University of Alberta
January, 2016

Table of Contents

1. Introduction 1
  1.1. Research objectives and originality 4
  1.2. Construction and Optimization of the Granular Fuzzy Models: a General Structure. ... 5
    1.2.1. Overall Scheme 6
  1.3. Dissertation Organization 7
2. State of the Art 10
  2.1. Fuzzy Relational Models 10
  2.2. Granular Fuzzy Relational Equations 12
  2.3. Granular Fuzzy Models 14
  2.4. Granular Modeling with Hyperboxes 15
  2.5. Classification Techniques with the use of Clusters 18
  2.6. Summary 20
3. Background Knowledge 22
  3.1. Fuzzy Relational Models 22
    3.1.1. Fuzzy Relational Equations 22
  3.2. Fuzzy Clustering and Performance Assessment 25
    3.2.1. Fuzzy C-Means 25
    3.2.2. Reconstruction Criterion 27
    3.2.3. Classification Rate 28
  3.3. Construction of Granular Fuzzy Models 29
    3.3.1. Granular Fuzzy Relational Equations 30
  3.4. Granular Fuzzy Modeling through Associations among Information Granules 34
    3.4.1. Interval-Based Information Granules 34
    3.4.2. Conditional Fuzzy Clustering 35
  3.5. Granular Modeling with Hyperboxes 37
  3.6. Particle Swarm Optimization 40
  3.7. Differential Evolution 41
  3.8. Summary 43
4. Granular Fuzzy Relational Structures 44

4.1. Problem Formulation 44
4.2. Fuzzy Relational Models 45
4.2.1. Experimental Studies 47
4.3. Granular Fuzzy Relational Models 50
4.3.1. Experimental Studies 51
4.4. Summary 52
5. Granular Fuzzy Models 54
5.1. Problem Formulation 54
5.2. General Architecture of the Model and its Underlying Processing 55
5.2.1. Performance Evaluation: Coverage and Specificity Criteria 59
5.2.2. Experimental Studies 60
5.3. Further Refinements of the Model 64
5.3.1. Optimization of Output Information Granules 64
5.3.2. Experimental Studies 66
5.4. Distribution of the Number of Clusters (Information Granules) 68
5.4.1. Experimental Studies 70
5.5. Summary 73
6. Granular Fuzzy Modeling with Hyperboxes 75
6.1. Problem Formulation 75
6.2. Hyperbox Construction and their Underlying Processing 76
6.2.1. Detailed Description of the Granular Model 77
6.2.2. Development of the Model 78
6.2.3. Performance Evaluation: Dataset Coverage 79
6.3. Hyperbox Construction with a Numerical Constraint 81
6.3.1. Experimental Studies 82
6.4. Further Refinements of the Model 83
6.4.1. Optimal Construction of the Hyperboxes 84
6.4.2. Optimal Overlap Elimination 86
6.4.3. Detailed Evaluation of the Hyperboxes 88
6.4.4. Experimental Studies 89
6.5. Summary 94

7. Refinements of Prototypes to Enhance the Classification and Reconstruction Performance of FCM 95
7.1. Problem Formulation 95
7.2. Architecture and Main Phases of Processing 96
7.3. Optimization of Information Granules 99
7.3.1. Migration of Prototypes 100
7.3.2. The use of Differential Evolution 102
7.3.3. Experimental Studies 103
7.4. Summary 115
8. Conclusions and Future Studies 116
8.1. Possible Limitations of the Research 117
8.2. Future Research Directions 118
Bibliography 120

List of Tables

Table 3.1. Selected fuzzy relational operations. 23

Table 4.1. RMSE values obtained for learning scenarios and various combination operators for the training and testing datasets. 49

Table 4.2. RMSE values obtained for learning scenarios and various combination operators. ... 50

Table 4.3.Area under the curve for each protocol and for the training and testing data.............. 52

Table 5.1. Worst and best overall results obtained for the Boston housing dataset and by using EP with selected values of the parameters. 62

Table 5.2. Worst and best overall results obtained for the delta elevators dataset and by using EP with selected values of the parameters... 62

Table 5.3. Worst and best overall results obtained for the delta ailerons dataset and by using EP with selected values of the parameters... 63

Table 5.4 Worst and best results obtained for the Boston housing dataset when using DE with selected values of the parameters. 66

Table 5.5. Worst and best results obtained for the delta elevators dataset with the use of DE with selected values of the parameters. 67

Table 5.6. Worst and best results obtained for the delta ailerons dataset when using DE with selected values of the parameters. 67

Table 5.7. Worst and best results obtained for the Boston housing dataset using DE to distribute the number of clusters, with selected values of the parameters. 70

Table 5.8. Worst and best results obtained for the delta elevators dataset with the use of DE to distribute the number of clusters, with selected values of the parameters. 71

Table 5.9. Worst and best results obtained with the delta ailerons dataset with the use of DE to distribute the number of clusters, with selected values of the parameters. 71

Table 5.10. Impact of different incremental values of λ on the specificity and coverage of the output data, where *lgt, tr*, and *te* are, length, training set and testing set, respectively. 72

Table 6.1. Results obtained with Simpson's algorithm and the Differential Evolution Hyperbox (DEH) construction method, the *Hyper.* column shows how many hyperboxes are obtained at the end. 92

Table 6.2. Detailed contents of each Hyperbox, including number of data being covered, mean, and standard deviation (Std). 93
Table 7.1. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for the training, testing, and the reconstruction error for the synthetic dataset..... 105
Table 7.2. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for the training and testing data, for the minimum reconstruction error, and maximum classification rate for the synthetic dataset. 108
Table 7.3. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for all the datasets. The maximum classification rate and reconstruction error for all sets are shown. 110
Table 7.4. Comparison of the original prototypes found by the FCM algorithm, their new position after the optimization process, and the minimum and maximum values for the thyroid dataset, for $c = 3$, and $\varepsilon = 0.1$.......... 112
Table 7.5. Reconstruction errors obtained with the initial prototypes (FCM) and with all the protocols for all the datasets. The minimum reconstruction error and classification rate for all sets are shown. 114

List of Figures

Figure 1.1. Overall scheme for the design and construction of the granular fuzzy models........... 6

Figure 3.1. Performance index as a function of the level of granularity *ε* with an area under the curve (AUC)... 32

Figure 3.2. Allocation of information granularity protocols P_1 to P_5, and an illustration of the resulting granulation of the fuzzy relations. 32

Figure 3.3.Calculation of the new partition matrix by using the prototypes obtained in the clustering stage by applying (3.21) and (3.22). 36

Figure 3.4. (*a*) Different degrees of membership obtained with different values of *δ*. (*b*) Two dimensional fuzzy set has been defined with the use of (3.23) with the lower corner at (0.5, 0.3), the upper corner at (0.7, 0.6), and $\delta = 4$.. 37

Figure 4.1. Topology of a Logic Processor in an AND-OR mode. 46

Figure 4.2. Best approximations obtained for the MPG for the (*a*) training and (*b*) testing datasets.. 48

Figure 4.3. Best approximations for obtained the Auto-price for the (*a*) training and (*b*) testing sets... 49

Figure 4.4. Increasing values of *ε* vs coverage for protocols 1 to 6 by using the MPG (*a*) training and (*b*) testing sets. 51

Figure 4.5. Increasing values of *ε* vs coverage for protocols 1 to 6 by using the auto-price (*a*) training and (*b*) testing sets. 53

Figure 5.1. General architecture of the granular fuzzy model, where a collection of induced information granules c_h is formed by engaging a context-based clustering mechanism, giving rise to several membership degrees *υhik*, which are summed at each context *h* to trigger the corresponding activation level *zh*(***xk***). 56

Figure 5.2. Output information granules and their influence on the clusters induced in the input space. The intervals placed in the output space (*b*) exhibit different widths than the intervals in (*c*). Five clusters are considered for each interval *h* and the prototypes are represented by black dots. 58

Figure 5.3. Worst and best results for the total coverage obtained for synthetic data, using $p = 5$ intervals, different fuzzification coefficients, and different number of clusters, the intervals are obtained by using EP.. 60

Figure 5.4. Worst and best results for the total coverage obtained for synthetic data and 9 intervals, the intervals are obtained by using EP. 61

Figure 5.5. Coverage vs. specificity for the Boston housing dataset when using EP intervals. ... 63

Figure 5.6. Worst and best results for the total coverage obtained for synthetic data, 5 intervals, different number of fuzzification coefficients, and different number of clusters using DE. .. 65

Figure 5.7. Worst and best results for the total coverage obtained for synthetic data, 9 intervals, different number of fuzzification coefficients, and different number of clusters using DE. .. 67

Figure 5.8. Coverage vs. specificity for the Boston housing dataset and when optimizing the intervals with DE. 68

Figure 5.9. Coverage vs. specificity for the Boston housing dataset when re-arranging the number of clusters.. 69

Figure 5.10. Optimized granular model with (a) its interpretation in the form of "if-then" statements and (b) the projection of the prototypes to their individual input space, where s, m, and h are *small*, *medium*, and *high* labels of the prototypes. 72

Figure 5.11. Techniques applied to the construction of the output information granules. 74

Figure 6.1. Synthetic input data with five overlapping clusters. 82

Figure 6.2. Hyperboxes formed by using the cluster centers found in each context and by following (6.2). Plots (a) – (d) show each of the hth intervals, and (e) shows all the superimposed intervals... 83

Figure 6.3. Resulting hyperboxes after the overlap elimination by using Simpson's contraction method (step 3 in Section 3.5). (a) – (d) show each of the hth intervals, and (e) shows all the intervals superimposed... 84

Figure 6.4. Upper and lower corners of the hyperboxes constructed with the information contained in a differential vector and by using (6.4), for two clusters in interval h in a two-dimensional space. 86

Figure 6.5. Reduction of the upper and lower corners of a hyperbox in the first dimension by using their corresponding $ajhi$ values and by following (6.5). 87

Figure 6.6. Hyperboxes created by using the min-max NN algorithm, where 93.9% of the synthetic data are covered and 35 hyperboxes are created. 90

Figure 6.7. Hyperboxes created by using the proposed method, (95.6% of the data covered with the use of 20 hyperboxes). 91

Figure 7.1. The two stages envisioned for the granular classification model, where the learning is performed in two stages, unsupervised and supervised. Here, the clusters coming from FCM are assessed and passed to a refinement stage. 97

Figure 7.2. Modification of prototype $\boldsymbol{v}_i$ controlled by the level of change ε for variables 1 and 2; in this case the levels of change are equal and the directions of change are positive (+) for both variables. 97

Figure 7.3. Different strategies of re-allocation of prototypes, where (*a*) all the levels of change are equal, and (*b*) the levels are asymmetrically distributed. 101

Figure 7.4. Target dataset shown as black circles with (*a*) original prototypes obtained with the FCM algorithm and (*b*) optimized prototypes; all the prototypes are shown as larger gray crosses surrounded by the boundaries of each cluster (shown as curves). 105

Figure 7.5. (*a*) Initial shape and position of each cluster and (*b*) shape and position of optimized clusters. 106

Figure 7.6. Overall classification rates for the synthetic dataset by using all the protocols, (*a*) for different number of clusters, and (*b*) the levels of change. 107

Figure 7.7. Overall classification rates with all the protocols for (*a*) the seed dataset, and (*b*) the wine dataset. 113

List of Symbols

$\boldsymbol{x}$	Input datum
y	Target datum
$\mathbb{R}$	Set of real numbers
N	Number of instances in the data
n	Number of input dimensions (features)
R	Fuzzy relation
V	Prototype matrix
U	Partition matrix
m	Fuzzification coefficient
c	Number of clusters
Q	Performance index
D	Interval positioned in the output space
p	Number of output intervals
B	Hyperbox
$\boldsymbol{d}$	Vector representing an individual or particle
τ	Generation

1. Introduction

Commonly, when humans observe objects that are not familiar to them, for example a set of unknown symbols or characters from another language or dialect, we tend to group them together by similarity, shape, or size. This process of forming such an abstract view is done in an almost subconscious manner. We tend to make groups of objects (information granules) to achieve a better understanding of newly acquired information. It is in our nature to make groups (clusters) of objects, people, or various shapes existing within our surroundings, thereby making our ensuing cognitive process more effective and far more advanced over processes realized by *intelligent machines*. An information granule can be defined as a conceptual entity that captures the essence of the overall data in a concise manner. In data mining, for example, the pivotal goal is to make sense of the data by representing these as information granules. In [114] it is highlighted that the goal of information granularity is to try to convert clouds of data into tangible information granules, to provide a better understanding of the topology and distribution of the data under observation. In the literature we can encounter a plethora of different approaches employed to construct information granules from numerical data, either by clustering [27], [28], [52], [83], [84], [141], [147], [162], [169], [178] or by engaging in classification tasks [1], [21], [31], [38], [46], [82], [90], [95], [145], [146], [156], [157].

Fuzzy models can serve as reference or as a vehicle for the construction of information granules, and in consequence construct granular fuzzy models, given that fuzzy models offer a convenient way to describe complex and nonlinear systems. Fuzzy relational equations, viewed as a certain class of fuzzy models, play a pivotal role in fuzzy modeling. Their theory supports ways in which these type of equations could be solved and offers a characterization of the resulting families of solutions. Moreover, in the construction of fuzzy models, various clustering techniques are often used, in particular Fuzzy C-Means (FCM) [19], [20]. Nevertheless, in spite of the truly remarkable diversity of fuzzy models architectures and ensuing design approaches [11], [30], [60], [63], [64]-[67], [75], [129], [154], these models share one common feature. Fuzzy models, regardless of the use of technology of fuzzy sets, produce *numeric* results.

In contrast, linguistic models [8], [22], [25], [26], [47], [120], [125], [128], [155], [160], [171] form an interesting conceptual and design alternative in the plethora of fuzzy models in the sense that their structure is intuitively appealing and the results are inherently coming as information granules; for these kind of models clustering is used more intensively. Granular fuzzy modeling, based on the principles of fuzzy modeling, augments the well-defined design methodology of fuzzy models. In a nutshell, the parameters in the fuzzy model are represented as information granules rather than single numeric entities. The information granules themselves are formed based on the numerical values of the original fuzzy model. The granular information associated with the granular model plays an important role by quantifying the performance of the original (numeric) model. Alternatively, clustering can lead to the formation of information granules in the output space and subsequently produce induced information granules positioned in the input space.

To put the subject matter in a more general setting, granular fuzzy models are essentially about building models at the level of information granules, not numeric evidence, making these types of models more reflective of reality. Since experimental numeric data lead to information granules, these models are structured in the form of information granules. Subsequently, these information granules are linked together by developing a backbone (blueprint) of the model. We focus on the design and construction of these linkages, to bring a directionality component onto these granular fuzzy models.

In this study we offer several different alternatives to build and evaluate granular fuzzy models. In one approach, we construct a fuzzy relational model based on numeric evidence. If the systems of relational equations are solvable, several analytical methods to build the model are provided. If this essential assumption is not satisfied, we resort to approximate solutions and optimization techniques. Once the fuzzy relational model is constructed, we focus on the generalization (abstraction) of the model by arriving at its granular counterpart. Given that information granularity is viewed as an important design asset, it is subject to optimization. In this regard, we study several essential schemes (protocols) of information granularity allocation, and endow them with some optimization criterion, requesting that the granular results of granular models are made as specific as possible. These granular constructs are viewed as an alternative to the resolution or approximation of the system of equations for the fuzzy relational models. To construct these granular models, we focus on the allocation of information granularity across the parameters of fuzzy models, and an assessment of the quality of the original fuzzy models is realized in this way. At the end, approximate solutions to the system of relational equations are provided.

Alternatively, we exploit a concept of information granularity by developing a model coming as a network of intuitively structured collection of interval information granules described in the output space and a family of induced information granules (in the form of fuzzy sets) formed in the input space. In other words, the formation of information granules is realized in two phases. First, information granules are constructed in the output space. Second, for a given information granule from the output space, a collection of induced information granules is built by engaging some clustering mechanism. In this sense, information granules emerging in the input space are directly implied by the structure already established in the output space. Furthermore the associations among information granules in the input and output spaces are transparent, and in this way we can easily form the linkages of the model. This approach helps build a model that is a direction-oriented construct. Since clustering itself produces direction-free building blocks, the formation of information granules realized this way brings a directionality component onto the granular fuzzy model.

Furthermore, by following the idea of the information granules induced in the input space, we take advantage of the information granules, and use the cluster centers (prototypes) as coordinates to construct a set of *hyperboxes*. We highlight the importance of information granules coming as hyperboxes for clustering or classification tasks due to their simplicity, effectiveness, and robustness. These facilitate the search of structures and topologies in the data, and help represent the essence of such findings through a set of information granules. This makes the data more abstract by observing the distribution and size of the clusters that capture them. At the end, each hyperbox can be interpreted as a fuzzy rule, and since the hyperboxes are formed by enclosing (capturing) surrounding data, these are allowed to demonstrate their essential features. Two different novel methods are proposed to construct the hyperboxes. For the first one, a numerical constraint is implemented, where the mean distance between the prototypes is used to construct the hyperboxes. For the second method, the hyperbox expansion is optimized by allocating granularity of the data positioned in the vicinity of the prototypes, furthermore if overlaps among hyperboxes are formed, these are optimally reduced or eliminated to retain as much data covered as possible. With this in mind, we develop an appropriate performance index and determine an adequate optimization scheme.

To conclude this dissertation, we implement a novel clustering optimization technique, where optimized clusters are used on classification or reconstruction tasks. As it is well-known, the

FCM algorithm is already a useful unsupervised learning technique, since it helps us analyze and group the data into information granules. These come in the form of a partition matrix and a set of prototypes. From the partition matrix, one could derive a hard classification by simply assigning a class label with the maximum membership value to each sample. And from the granular information captured by the clusters, one can decode or reconstruct the original data. Nevertheless, in some cases using only the FCM algorithm, one might obtain low classification results, due to complex topologies, outliers, or noise in real-world datasets. On the other hand, due to inaccuracies introduced in the granular representation of the data, the reconstruction error is usually nonzero, where larger errors mean lower quality of the clusters. The idea behind the approach pursued in this study is to strategically re-allocate the original position of the prototypes, and in consequence, disturb the membership degrees of the data to the clusters to minimize the reconstruction or classification errors. To optimize and relocate the cluster centers, we implement several strategies (protocols), each with different parametric flexibilities, along with a level of change used to control the migration of the prototypes. The main goal is to enhance the clustering, reconstruction, and classification abilities of the clusters, along with the quality of the granular model.

1.1. Research objectives and originality

The key objectives of this research are:

- Develop several novel strategies to design, build, analyze, and optimize granular fuzzy models, and make the resulting granular structures as specific as possible.
- Design, construct, and optimize fuzzy relational structures to form fuzzy models, and subsequently obtain their granular counterpart by an optimal allocation of granularity.
- Design and develop direct associations between input and output data, and by using this construct, build a model at the level of information granules.
- Propose several approaches to construct and optimize hyperboxes in the input space by using direct associations between input-output information granules.
- Cluster data by using FCM, and strategically migrate the prototype coordinates to enhance the quality of information granules, and optimize their clustering, classification, and reconstruction performance.

The research discussed here exhibits several novel facets and raises and addresses new interesting issues. The question of optimization of input-output information granules is of paramount relevance and it is addressed in this dissertation. We study two algorithms for the optimization of the granular constructs, namely Particle Swarm Optimization (PSO) and Differential Evolution (DE). As for the optimization criteria, coverage, specificity, reconstruction, and classification are also studied here. Information granules serve as sound descriptors of data. Each information granule comes with its own well-defined semantics and as such, the granules can be associated with a certain linguistically sound meaning. In this way one can produce a general sound view at the data.

This research exhibits a significant level of originality:

- A novel technique based on granular relational structures with direct generalizations for the solvability of fuzzy relational equations and the quality assessment of the constructs.
- A new granular fuzzy model which exploits the concept of information granularity by developing a model coming as a network of association among information granules.
- A novel approach for the construction and optimization of hyperboxes and an optimal reduction or elimination of overlaps among them.
- An innovative fuzzy clustering optimization technique used to enhance and refine the quality of clusters to help these classify data, or use their granular information to efficiently reconstruct the numeric entries.

1.2. Construction and Optimization of the Granular Fuzzy Models: a General Structure.

Granular models are formed on the basis of information granularity and provide a superior understanding of the data under observation. To have a better grasp of the aim of this dissertation, a general structure was developed by splitting the main components into modules. Each module describes parts of the processing performed over the numerical experimental data, and the results of each module are passed to the subsequent connected module.

1.2.1. Overall Scheme

In Figure 1.1 the main components of the algorithms proposed in this study are displayed as a general structure. Each component in this structure plays an important role in the design of the granular fuzzy models.

As it is displayed in the figure, the first module represents the experimental data. This may come as input-output pairs of synthetic numerical data, or from real-world measurements. This data is usually normalized to values between the unit interval or more specifically, between the closed interval [0, 1]. However, in some cases the data is processed without any normalization. After the data is prepared, it is fed onto the ensuing modules as follows;

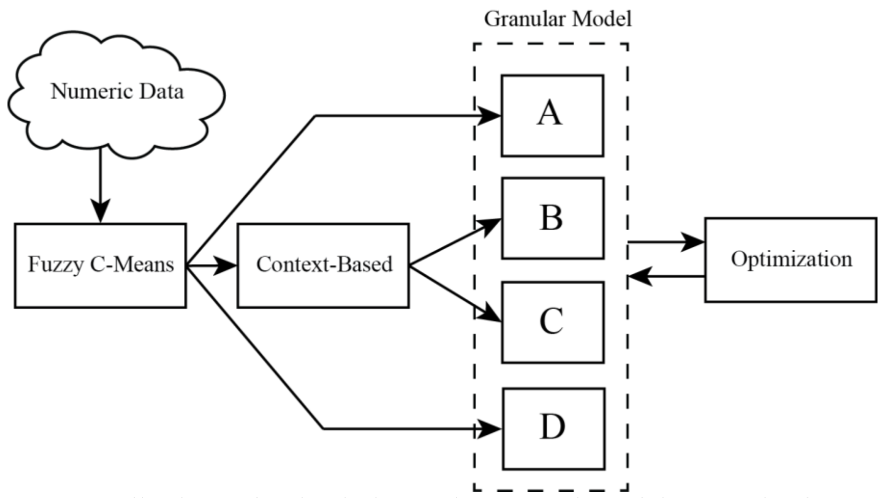


Figure 1.1. Overall scheme for the design and construction of the granular fuzzy models.

Fuzzy C-Means: This is probably the most important module of the framework, since it is used to find the partition in the data and construct information granules (clusters, fuzzy sets). These come represented as a partition matrix and a set of prototypes (the coordinates of each cluster center).

Context-based: A specialized conditional fuzzy clustering algorithm is employed in the construction of granular models. This so-called context-based FCM algorithm helps brings a directionality component into the granular constructs, to form associations (relations) between the input and output spaces. This algorithm serves as a basis to construct two granular models.

Granular Model: Once the information granules are obtained, the granular model is formed based on the information provided by the fuzzy partition found in the input space. In this research we explore several alternatives to construct the granular models around the experimental data. In the first approach (A), the partition matrix is used to train a Fuzzy Logic Processor (FLP), where its granular counterpart is constructed based on the granulation of a fuzzy relation between information granules already stablished in the input and output space; this generalizes the optimized structure, and it is used to assess it. In the subsequent approaches, the prototypes are used to construct and optimize the granular models (B), where a network of associated information granules is constructed by using the context-based FCM algorithm, while in another approach (C), a set of hyperboxes is constructed from the coordinates of the prototypes obtained at the conditional fuzzy clustering stage. In the last approach (D), the partition matrix is used to associate each cluster to a class, and a classification performance of the granular model is obtained. Alternatively, the reconstruction performance is used to assess the quality of the information granules.

Optimization: After a granular model has been designed, it is fine-tuned by employing group-based optimization algorithms. The algorithms used in this dissertation are the Particle Swarm Optimization (PSO) and Differential Evolution (DE). These algorithms were chosen due to their robustness and effectiveness on optimization tasks. Each granular structure, regardless of its construction, is optimized by means of these algorithms. Each model has an objective function, tailored to fit the granular model around numerical experimental data. The results of all granular models come represented as information granules.

1.3. Dissertation Organization

The subsequent chapters are structured as follows:

Chapter 2. State of the Art

In this chapter we offer a focused literature review of several approaches for the construction of fuzzy models, granular fuzzy models, hyperboxes, and classification techniques with clusters.

Chapter 3. Background Knowledge

Several methodologies and algorithms are discussed in this chapter, which have been applied to the construction of fuzzy models and granular fuzzy models in the past. Consequently, we highlight the techniques employed to build and optimize the various granular models proposed in this research.

Chapter 4. Granular Fuzzy Relational Structures

The underlying methodology applied to construct a fuzzy relational model, known as a fuzzy logic processor, is presented in this chapter. Moreover, we incorporate granular information to the model to construct a granular fuzzy relational model which will help us assess the performance of the original non-granular model, and at the same time offer an alternative solution to the system of fuzzy relational equations.

Chapter 5. Granular Fuzzy Models

The overall architecture of a granular model, coming as a network of intuitively structured collection of information granules, is described in this chapter. Since this model heavily relies on information granules, it is essential to focus on its design. Moreover, the details of the construction of the information granules and their optimization are presented, where we focus on a specialized fuzzy clustering algorithm; a so-called context-based FCM algorithm.

Chapter 6. Granular Fuzzy Modeling with Evolving Hyperboxes

In this chapter we describe the essential functional modules that are used to compose a novel granular model. We exploit the concept of context-based FCM where interval-based information granules are used to induce clusters in the input space. The coordinates of the cluster centers (prototypes) serve as a basis to construct and optimize a set of hyperboxes, which are formed by allocating an optimal level of granularity in the data around each prototype.

Chapter 7. Refinements of Prototypes to Enhance the Classification and Reconstruction Performance of FCM

Fuzzy C-Means (FCM) is a powerful unsupervised learning technique, and it is used as a first-plane processor to build a classifier. The idea is to use FCM to find a partition in the input data and observe its classification and reconstruction performance. Then, the prototypes found are strategically repositioned, in a supervised manner, to enhance the quality of the clusters. The objectives are to enhance the clustering, classification, and reconstruction performance of the granular model, and at the same time, preserve the results obtained in the initial clustering stage.

Chapter 8. Conclusions and Future Studies

A number of conclusions are drawn from our research in this chapter, and we suggest a number of directions for future works.

2. State of the Art

A central topic of engineering and science is the development of mathematical models from data available in real-world systems. The various usages of these models include prediction, decision support, system analysis, and control design, among other applications. Moreover these models can be viewed as useful sources of knowledge. When modeling a system, we must have a thorough understanding of the system's behavior, and the suitable modeling techniques that will lead to a usable model. However, in real-world situations it can be difficult to efficiently build a model, due to inaccurate information from different sources, from the complexity of the noise, or non-stationary behavior of the system. Some success has been achieved by using two-valued logic and set theory in system modeling. Nevertheless, for many real-world applications only the input and output data acquired from running the process is accessible for estimation. Here is where the granular fuzzy models provide formal tools to reason and handle such uncertainty found in the data. In this chapter we review several different techniques used to construct fuzzy models and granular fuzzy models over the years.

2.1. Fuzzy Relational Models

Fuzzy set theory was proposed five decades ago as a generalization of set theory [175]. In that research, several approaches were suggested to process data and information affected by non-probabilistic imprecision. The concept of fuzzy sets was designed to represent the vagueness and uncertainty of linguistic problems in a mathematical way by generalizing the concept of classical set theory. Moreover, fuzzy set theory has been proven to be a key component for modeling static and dynamic systems. In general, fuzzy set theory can be used to handle incomplete or vague knowledge in systems, adequately process imprecise information, and on transparent modeling and identification. In processes where a mathematical structure is unknown, a modeling approach based on a system of relations is appropriate since it can provide multiple input-output relations. However, in many cases, a two-valued logic approach is undesirable.

Almost four decades ago, and by generalizing the idea of Boolean equations in a similar fashion as fuzzy logic generalized the fundamental idea of two-valued logic, the concept of fuzzy relations and fuzzy relational equations was introduced [138]. A fuzzy relation is used to represent and quantify associations between objects; in the case of a Boolean relation, it represents the presence or absence of an association, this concept is generalized in fuzzy relations by allowing various degrees of association between numerical entities [71], [119].

Generally speaking, the relationships among objects are captured in terms of fuzzy relations, and their transformation is done through some composition operators. These compositions are realized by using triangular norms [92], [140]. A triangular norm is a binary operation used in the framework of probabilistic metric spaces and in multi-valued logic, specifically in fuzzy logic. There are different types of compositions, and these depend on the choice of a specific *t-norm* or *s-norm*. The most important compositions are the *sup-t* and *inf-s* (*max-t* and *min-s*) compositions.

A fuzzy relational equation is a dependency among input and output variables, expressed in the form of a fuzzy relation *R*. A generic fuzzy relational model comes in the form of the expression $\boldsymbol{x} \operatorname{op} R = \boldsymbol{y}$, where 'op' denotes a fuzzy relational operator. By looking at this equation, we arrive at the previously mentioned fundamental problems;

i) Given $\boldsymbol{x}$ and $\boldsymbol{y}$, determine *R* (*estimation* problem), and
ii) Given $\boldsymbol{y}$ and *R*, determine $\boldsymbol{x}$ (*inverse* problem)

In the past, researchers have proposed analytical methods for the resolution of these two fundamental problems. The resolution of the *estimation* problem [138] helped find a potential maximal solution to the system of relational equations, and was used in medical diagnosis applications. In [105] an analytical method was proposed to find the potential maximal solution for the *inverse* problem. In some cases, only a unique potential maximal solution exists [37], but in other cases the solution comes with a finite set of minimal solutions, as demonstrated in [53]. Later, in [32], [36] the structure of the solution set for the fuzzy relational equations was extensively analyzed, especially the complete set of lower solutions, when a max-min composition was used. Also in [33], [34] the resolutions of relational equations were analyzed by considering the energy and entropy measure of fuzziness, to form a set of solutions for a given fuzzy relational equation. The resolution of the system of equations was extended to other *t-norm* compositions, like the *max-product* composition [35].

The process of finding these sets of solutions still remains a challenge, since in real-world situations the complete sets of conditions are not easy to find [119], causing the subject matter to remain considerably open. This problem has encouraged researchers to find approximations to the systems of equations [48], [108], by employing several different methods including numerical methods [106], [107], [127], [153], [172] and optimization methods such as gradient-based (GB) methods [109], [119], [134]. In Recent approaches, fuzzy logic has been used along with population-based algorithms for unsupervised fuzzy clustering [68] and pattern recognition tasks [76], [79], [91].

2.2. Granular Fuzzy Relational Equations

If precise information about membership values is not known, and a system of relational equations is unsolvable, a bound (interval) of possible membership grades can be admitted. Then the knowledge of a numeric entity can be defined by the size of this interval. The concept of fuzzy tolerance was suggested as an extension of the solvability of the system of equations, to solve the *estimation* [165] and *inverse* problems [166]. A solution to two more problems were also suggested; given a set of upper tolerances, how to determine the lower tolerances, and vice versa, in order to obtain at least one solution. In [167] the concept of tolerances was generalized and a solution for composite interval-valued fuzzy relational equations with the min-max operator was proposed. This work was continued in [79], and these relations were transformed into a fuzzy relation inequality system, simplifying the solution methodology. More research followed, and focused on the solvability of interval-valued fuzzy relational equations based on these previous works [80], [151], [168]. The semantics of the membership degrees can be captured by intervals of possible membership grades, instead of only focusing on single values. This was proposed in [112], where a fuzzy model was generalized and abstracted (granulated) by optimal allocation of granularity [114], [119], [124]. When using interval-based fuzzy sets, an interval-valued membership degree is assigned to each entity of the universe, e.g. *x*, and this is considered as only the 'evidence for *x*'. This definition was extended in [47] where the concept of *vague sets* was proposed. Here, two independent vague values called the *evidence for x* and *evidence against x* were considered, incrementing with this the judgment about the membership grade and making it more expressive when capturing the vagueness of data.

By taking as a basis these interval-valued fuzzy sets, a model known as granular fuzzy relational equations can be constructed. These granular equations are formed by approximating a system of fuzzy relational equations between input and the output data. Then, the fuzzy relation is converted into its granular counterpart by allocating an optimal level of granularity. This technique has been successfully used in group decision-making tasks [125], in the design Type-2 fuzzy models [22], on characterization of electrocardiogram signals [46], and to extract knowledge from existing fuzzy models [123].

To find an approximation to the system of fuzzy relational equations, in [109], a learning method was proposed to construct a fuzzy neural network, and the learning was completed separately from each output node. In [134], the network was modified for n −inputs/m −outputs, where the learning process was performed on all the output nodes at the same time. By using as a reference this logic neuron, the fuzzy logic processor (FLP) was developed [119]. This model had a well-structured semantic mapping of the input and output data, therefore it transformed the "black-box" nature of a traditional ANN into a "white-box". This allowed the model to be translated easily, by extracting the semantic meaning of the knowledge acquired during the FLP training. The FLP follows a gradient-based learning to tune its weights. But the derivations are applied to the logic operations that form the fuzzy relations in the model, namely the *t-norms* and *s-norms* [119]. However, when training these models, many researchers have found that gradient-based learning often experiences some shortcomings. Some arise with large complex problems, which may cause the network to get trapped in local minima. This made researchers turn to group-based optimization algorithms. These type of algorithms, particularly Particle Swarm Optimization (PSO) [70] and Differential Evolution (DE) [130], have been widely used to overcome this shortcoming, and to optimize these structures.

Evolutionary and group-based optimization techniques do not depend on the gradient information. This makes them more robust or suitable for tasks where there is the need of handling complex problems. In [173], it was mentioned that the best optimization algorithm is always problem-dependent. It was also suggested that for complex optimization tasks, it is advisable to use hybrid algorithms, since these tend to perform better than others for a larger set of problems [9], [139]. The main idea behind hybridization is to use a population-based algorithm to find a good set of initial weights in a learning model (e.g. an ANN or FLP), and then use a local search algorithm, like the gradient-based algorithm. This helps the algorithm to find the global optimal weights.

2.3. Granular Fuzzy Models

As it has been discussed so far, in contrast to most fuzzy models encountered in the literature, granular fuzzy models produce results in the form of information granules rather than plain numeric entities. Nevertheless, a fuzzy model can serve as a basis for the construction of a granular fuzzy model. To design a fuzzy models, various clustering techniques are often used, in particular FCM.

In [162], a modified FCM algorithm, combined with a back-propagation algorithm, was implemented to construct detailed fuzzy models by using a weighted FCM algorithm. Later in [164], FCM was used to generate discrete interval-valued type-2 fuzzy models. A fuzzy C-Regression model was proposed in [81] to automatically determine a suitable number of rules from a given fuzzy model. In [78], a fuzzy clustering technique was developed to generate hyperplane-like clusters, which helped improve the estimation of Takagi-Sugeno (T-S) fuzzy models. Another clustering technique, based on a combination of FCM and switching regression algorithms [23], was used to construct enhanced fuzzy models with an improved fuzzy clustering and better structure identification. A different clustering approach helped identify T-S fuzzy models from clusters obtained with a modified Gath-Geva algorithm [6]. A face recognition approach used partition matrices obtained with the FCM algorithm as input patterns to train artificial neural networks (ANNs) [85]. Other architectures that are similar to ANNs, are the neural fuzzy controller (NEFCON) [96]-[98] and neuro-fuzzy classification (NEFCLASS) model [99]. These structures used fuzzy sets as weights, and can be interpreted as linguistic "if-then" rules. The obtained results are defuzzified (decoded) and produced numeric values.

In contrast for linguistic (granular) models clustering is used more intensively, leading to the formation of information granules in the output space, and subsequently producing induced information granules positioned in the input space. One of the earliest models falling under this category was proposed in [152], where the FCM algorithm was used to identify the structure of a fuzzy model in many-input single-output systems. In this approach, the ordinary fuzzy partition of the input space is avoided. Instead, the output data is clustered with FCM. As a result, every output datum associated with the membership degree belonging to a cluster, is projected into the individual coordinates of the multidimensional input space. This produces fuzzy rules that can be defuzzified into numeric entities. Interval-based clusters were proposed in [86] to granulate time-series and improve the forecasting accuracy of a model. These intervals are optimized by gradually adjusting their widths until they become more reasonable and informative. In other studies [2], [5], [158], [161], the information granules are induced by projecting hyperbox clusters from the multidimensional input space onto each input coordinate spaces. In other approaches, a set of intervals formed in the output space are used as classes to turn an approximation problem into a classification task, and to induce the formation of hyperboxes [4] and hyper-ellipsoids [159] in the input space, nonetheless the values at the end are defuzzified into numeric values. In [131], information granulation is used to induce consistent granular structures, which help build a knowledge distance that is used to form a lattice model that reveals the essence of information granularity. Lattice computing is used in several studies to enhance fuzzy models, where intervals are used to represent fuzzy sets, either described as α-cuts [119], or as fuzzy membership functions. Since these intervals are partially-ordered, they are considered as a lattice, hence mathematical lattice theory can be applied to build tunable fuzzy models [64]-[66], [104]. In [142] and [143], the objective function of the clustering algorithm was modified to introduce a directionality component to the formation of the prototypes, the resulting information granules are used to form "if-then" rules. Also in [118] particle swarm optimization (PSO) was used to construct interval-valued prototypes, where the widths of the intervals are adjusted through a granulation-degranulation scheme. In this way, a granular version of the original data was formed. By using a coverage criterion as an objective function, it was possible to allocate the granularity of the data.

2.4. Granular Modeling with Hyperboxes

As discussed in the previous Section, an alternative process for the construction of granular models is by clustering the data with hyperboxes, instead of hyperspherical or hyperellipsoidal clusters. The Hyperboxes have been used due to their inherent flexibility to model smooth boundaries between classes [113]. In the literature, one can find a plethora of novel methods applied to the formation of these special clusters. The idea of a min-max neural network (NN) was proposed by Simpson to classify [145], [146] and cluster [147] data. In these studies the author dealt with numeric data to construct the hyperboxes (clusters). Moreover, the use of a membership function was proposed to help discover data and expand the hyperboxes, to produce information granules with largely varying sizes in various dimensions. Also, a method was used to minimally contract the hyperboxes when an overlap between them was discovered with the help of four rules. However, in some cases the overlap elimination could cause loss of covered data. Finally, the maximum size of the hyperboxes was controlled by using a certain user-defined variable, which regulates their size and at the same time, the number of hyperboxes obtained at the end of the training process. In a later approach [44], an alternate training technique for the min-max neural networks was proposed. These studies addressed the common overlap problem among hyperboxes, which exhibits a detrimental effect that causes the coverage of the data to become irrelevant. To address this issue in [3], [4], [158], the authors proposed to label the overlaps as *inhibition* hyperboxes, and then repeat Simpson's algorithm recursively, until no more overlap were observed. This technique exhibits several advantages. First, the hyperboxes are not contracted, so the data included in the original hyperboxes remain covered, and in consequence, included in the information granule. Second, the hyperboxes can be formed around complicated topologies in the data. A drawback of this method is that it is not clear how many times it is needed to repeat these iterations, and how complex it can be to represent the final model. This technique was applied for function approximation [4], and to classify data such as license plates [3] and the iris dataset [158]. It is worth mentioning that when the method was used for function approximation, the output data were divided into intervals, and each interval was used as a class, which has certain similarities to the algorithms proposed in Chapters 5 and 6. In another approach, a rotation of the hyperboxes was proposed in [87], [137] to improve the coverage of data in complex topologies, here the rotation resulted in a better coverage of data with fewer and smaller hyperboxes.

Up to this point there was not much attention paid to improve the basic algorithm proposed by Simpson, until in [45] an improved membership function was proposed, and the parameter used to control the size of the hyperboxes dynamically adjusted to each clustering problem, and therefore it was no longer user-defined. Similarly in [13], the authors proposed partially labeled data for partial supervision learning, which greatly improved the performance. Moreover, in [116] several new measures were proposed to have a more detailed and controlled construction of the hyperboxes, these include cluster compatibility, inclusion, and sparsity measures, and the discovery of the data or small hyperboxes was done by using a certain distance measure between numeric vectors (which represent hyperbox coordinates) instead of the original membership function. Later in [14], [15], [16], the overlaps among hyperboxes were defined as *exclusion* hyperboxes, which are similar to the inhibition hyperboxes, and these could avoid the contraction of the hyperboxes by simply excluding the covered data from the overlaps. The exclusion hyperboxes helped cover complex topologies, but again resulted in loss of covered data.

With regard to more recent approaches, a genetically optimized fuzzy NN was introduced in [113], this NN was developed around fuzzy tolerance neurons, optimized by using a standard Genetic Algorithm (GA). The NN was mainly used to reveal the topology of numeric data as fuzzy hyperboxes. On another study [133], a clustering algorithm based on hyperboxes was developed to classify unlabeled data. This algorithm used an Ant Colony Optimization (ACO) algorithm to recognize topological information in the data. Similarly, in [176] the original min-max NN was modified and used for classification tasks. In this approach, a new *overlap neuron* was defined, which was added to the NN to represent the overlapping area of the hyperboxes. In another approach [29], also by using as a reference the original algorithm, the authors proposed a multi-level tree structure to classify data. This approach used various hyperbox sizes at different levels to increase the accuracy and performance of the NN. The main idea was to retain the overlaps and construct smaller hyperboxes in each overlap detected, creating a new level in the NN. Another enhancement of the original algorithm can be found in [94], where the authors added three new heuristic rules to each hyperbox construction phase, viz. expansion, overlap test, and contraction. Concisely, they introduced a new constraint for the overlap expansion, added five new rules to the original four rules for the overlap detection, and did the proper adjustments to the hyperboxes with all nine rules in the hyperbox contraction phase. In [148], an algorithm was designed to form hyperboxes around class-patterns in data, forming with this a model called dendrite morphological NN. And finally in [41], the combination of supervised learning and unsupervised clustering was applied to different regions of the data. Here, a set of hyperboxes is constructed to identify and classify groups of data with similar patterns.

Most of these previous studies focus on the construction of hyperboxes by using as a reference Simpson's method; therefore some similarities are found. Namely, how the hyperboxes are expanded and how these eliminate overlaps. However, there is still room for some improvement, and different approaches could be considered when constructing these information granules.

2.5. Classification Techniques with the use of Clusters

Up to this point it is clear that clustering constitutes a prerequisite for many granular models, and many classification methods. However the assessment of the quality of the clusters has not been addressed. To evaluate the clusters we might turn to validity indices, classification performance, or a reconstruction criterion. But the quality of the clusters is also dependent on their size, number, and shape. As it is well-known clusters form geometric objects (information granules) in the feature space. Information granules determine regions in the feature space, which are characterized by high homogeneity in terms of the class membership of the clustered patterns. Quite commonly in the generic FCM algorithm, the Euclidean distance or its variants are considered. This implies a geometry of the produced groups in the form of hyperspherical shapes of clusters. Furthermore, it has been observed that classification methods which rely solely on FCM algorithms are vulnerable to outliers [73], [100], as demonstrated in [72]. This study shows that, despite the stability of the FCM algorithm, it is not robust when it comes to outliers in the data. This vulnerability could compromise the performance of the ensuing clusters. In [56] this issue has been addressed by using an iteratively reweighted least-squares technique (IRLS) [54], combined with the Mahalanobis distance to make the clustering algorithm more robust to outliers. This work was continued in [57] where the same technique was used to construct a C-means clustering classifier. An early work proposing the use of the Mahalanobis distance can be found in [174], where the idea is to change the shape of the fuzzy sets to hyperellipsoidal one. The authors noted that by using different distances in the FCM algorithm, different partitions of data can be obtained. In this work, the values positioned on the diagonal of a definite positive matrix, used to determine the Mahalanobis distance, are optimized by using a genetic algorithm, with the *partition coefficient* [17] being used as an objective function to validate the clusters. Based on this study, in [179] the classifier was extended by exploiting the concept of the Cholesky factorization, or the fact that any positive-definitive matrix can be decomposed into the product of a lower triangular matrix (with positive entries on the main diagonal) and its conjugate transpose. The entries of the lower triangular matrix were optimized with a genetic algorithm. Here the objective function was a distance measure between the fuzzy partition and a previously known partition (classes). A second version of the algorithm was designed by using the *Xie-Beni* validity index [170] as an objective function to assess the quality of clusters. In [180] the authors proposed the use of a different definitive positive matrix for each cluster. In these two previous studies the resulting clusters were used to classify transient components in nuclear systems for diagnostic purposes. This work was continued in [93] to design a possibilistic clustering algorithm. Here the FCM algorithm was modified to avoid the initial condition in which (typically) a membership value of a numeric entity to a cluster depends directly on its membership value to other clusters. This modification allowed the clusters to classify and detect atypical patterns whenever those exhibit low membership degrees to all clusters. This method was used to classify possible events in dynamic event tree analyses. Finally in [11] the FCM classifier, which had the independent Mahalanobis distance metric for each cluster, was used in an ensemble of FCM classifiers to try to improve the classification of nuclear transient components in nuclear power plants.

A different and simpler approach was studied in [144], where the traditional FCM algorithm was used to build a fuzzy relational classifier. First the number of clusters and fuzzification coefficient were selected by using the Xie-Beni validity index, once a plausible fuzzy partition was found, a fuzzy relation was constructed to find a logic mapping between the clusters and the class labels. This algorithm was used to classify livestock from sound sequences. The fuzzification factor in FCM plays an important role, given that its value significantly affects the shape of the obtained clusters. Typically, the common value found in the literature is assumed to be equal to 2.0, where the membership functions tend to resemble a Gaussian-like shape. Nevertheless, some studies show that this is not always the best value [115], [122], and in several cases lower values of this factor offer better granular results.

In [177], an automatic edge detection method was proposed by using a multiscale wavelet transform to extract wavelet features from greyscale digital images. From these features an input dataset was formed which contained the gradient information of the image in various directions. The data are then classified (partitioned) by the FCM algorithm, and from the resulting partition matrix, a binary map regarded as an edge map is constructed. Moreover, in [88] the Xie-Beni validity index is used as an objective function for a genetic algorithm (GA) to find the best partition in the data. Here, each individual proposes a possible number of clusters c, and a set of prototypes to avoid a randomized initialization of the partition matrix. However, the optimization of the fuzzification factor is not considered, and it needs to be set by the end-user beforehand. This technique was applied to the classification of pixels from remote sensing imagery. A similar classification approach can be found in [103], but in this study the GA algorithm is combined with a so-called PBMF validity index, which was proposed earlier by the same authors [102]. In a second approach [89], Differential Evolution (DE) was used to find the best initial prototypes, and the *Minkowski* score [62] was used as a validity index. A downfall that this approach has is that the number of clusters and fuzzification factor need to be fixed before the DE algorithm starts. These algorithms were used to classify synthetic data, real-world data, and for pixel classification of remote sensing imagery.

All of these studies exhibit some similarities, given that they focus mainly on the shape and number of the clusters to enhance the classification results. Some studies depend on a validity index to optimize the quality of the clusters, while others use the classification error to assess the partition of the data. In [58] – [60], [117], [122], it was demonstrated that a proper selection of number of clusters and fuzzification factor can increase the quality of the clusters, and the information captured by the granular representation of the data can be used to reconstruct an approximation of the original numeric entities, obviously with a nonzero error. Evidently, if this reconstruction error is small, then the quality of the clusters is high. Regardless of the method used to optimize the information granules, and in consequence the number of clusters and fuzzification coefficients, we need to decide what objective function to use to optimize them. It may come as a validity index, reconstruction criterion, or as another mechanism to help assess the clustering performance of the FCM algorithm. Further in this dissertation we deal with labeled data, then the classification rate will be used as the performance index to refine and define a new position for the prototypes, where the main goal is to maximize the classification of the fuzzy clusters. Once this performance is maximized, it is assumed that the quality of the clusters has increased, so we look at the reconstruction capabilities of the information granules. Another approach is to use the reconstruction error as an objective function, and then look at the classification rate. Once the model has been optimized, its generalization and recalling capabilities are tested.

2.6. Summary

The ideas discussed in this Chapter are focused on the design and optimization of fuzzy models and granular fuzzy models. Nevertheless, there is still room for some improvement (and for alternative ideas), to help maximize the performance of these models in order to obtain sound results. In this dissertation we develop several novel strategies to design, analyze, and assess different types of granular models. In their design, we have to focus on several factors. These include data representation, number of information granules needed for each dataset, the proper fuzzification factor, among other parameters as required for each individual model. Furthermore, we need to select a proper optimization mechanism, and find a suitable representation of the problem, along with a proper objective function to be optimized. The design issues will be described in each chapter, along with the proposed solutions, and alternative strategies designed to build usable granular fuzzy models.

3. Background Knowledge

In this chapter we present the methods and algorithms used to construct the granular fuzzy models. We start by presenting the main components of fuzzy relational equations, their analytical solutions, and techniques used to find approximations to these equations. Also, the methods used to transform real-world data into fuzzy sets. Moreover, we describe the main components used to allocate the granularity in the data by offering an overview of the principle of justifiable granularity used to construct the granular models. Also, we describe alternatives to build granular models without requiring a fuzzy model as a base for their construction. Furthermore, a description of the construction of hyperboxes used to build information granules is presented. To finalize this chapter, the mechanisms used for the optimization of the granular models are presented.

3.1. Fuzzy Relational Models

When representing knowledge, it has to be presented in a way it matches intuition and captures the semantics of the intended representation. It is also required that this representation comes with sufficient flexibility to adapt to the application requirements. Fuzzy logic and fuzzy set theory are good candidates for such purpose due to their capabilities of knowledge representation including a spectrum of logical processing associated with them [119].

3.1.1. Fuzzy Relational Equations

The relations between objects can be captured in terms of fuzzy relations, and their transformation is done by using the composition of triangular norms [92], [140]. These triangular norms (*t-norms*) are used to represent a logical conjunction in fuzzy logic, or an AND operation, in fuzzy set theory. The *t-conorms* (*s-norms*) are dual to the *t-norms*. These are used to represent logical disjunction, or an OR operation. Different types of compositions can be realized, depending on the selection of *t-norms* or *s-norms*. The most commonly used compositions are the *sup-t* and the *inf-s* compositions. The choice of the *t-norms* and *s-norms* and their combination, have a significant impact in the final approximation capabilities of the system of equations. In this dissertation, the *t-norms* and *s-norms* described in the following table are used to construct fuzzy relational models.

Table 3.1. Selected fuzzy relational operations.

	t-norm	s-norm
Min – Max	$\min(x, r)$	$\max(x, r)$
Prod – Prob. Sum	$x \cdot r$	$x + r - x \cdot r$

A generic fuzzy relational model comes in the form of the following expression;

$$\boldsymbol{x} \text{ op } R = \boldsymbol{y} \tag{3.1}$$

where the fuzzy relation $R \in [0,1]^{n\times\eta}$ realizes associations between two vectors; input vector $\boldsymbol{x} \in [0, 1]^n$, and an output vector $\boldsymbol{y} \in [0, 1]^\eta$ [119]; while 'op' stands for a certain composition operator. The fuzzy relation R describes the dependencies (relationships) between the system's inputs and outputs. The membership function of $\boldsymbol{y}$ obtained for the *sup-t* composition is determined as follows [71]:

$$\boldsymbol{y} = \boldsymbol{x} \circ R,\ \ y_\imath = \max_{j=1,2,\ldots,n}[x_j \text{ t } r_{j\imath}],\ \imath = 1,2,\ldots,\eta \tag{3.2}$$

This *sup-t* composition represents the possibility measure of two fuzzy sets, or the degree of overlap of $\boldsymbol{x}$ and R, and is the composition that has been most extensively studied. On the other hand, for the *inf-s* composition, the aggregation of the inputs represents the necessity measure of two fuzzy sets, or the degree of inclusion of the complement of $\boldsymbol{x}$ (e.g. $\bar{x} = 1 - x$) in R. The membership function of $\boldsymbol{y}$ is calculated as,

$$\boldsymbol{y} = \boldsymbol{x} \bullet R,\ \ y_\imath = \min_{j=1,2,\ldots,n}[x_j \text{ s } r_{j\imath}],\ \imath = 1,2,\ldots,\eta \tag{3.3}$$

In [138] a fundamental method was suggested to determine the unique maximal solution for the *sup-t* composition, and the unique minimal solution for the *inf-s* composition, to deal with the estimation problem. To obtain the solution to this problem, let us assume that $\mathcal{R}$ is a nonempty family of fuzzy relations R, say $\mathcal{R} = \{R \in F(\mathbb{X} \times \mathbb{Y}) \mid \boldsymbol{x} \text{ op } R = \boldsymbol{y}\}$. If $\mathcal{R} \neq \emptyset$, then there exists a unique maximal solution $\widehat{R}$. It was established that this unique maximal solution is computed with the pseudocomplement operator ϕ as follows; if (3.1) is satisfied then for a given $\boldsymbol{x}$ and $\boldsymbol{y}$ a maximal fuzzy relation $\widehat{R}$ is obtained,

$$\widehat{R} = \boldsymbol{x} \,\phi\, \boldsymbol{y},$$

or $\hat{r}_{j\iota} = x_j \,\phi\, y_\iota$. For a given *t-norm*, an operator ϕ is defined as follows, see [138],

$$x_j \,\phi\, y_\iota = \sup \{r_{j\iota} \in [0, 1] \mid x_j \text{ t } r_{j\iota} \le y_\iota\} \tag{3.4}$$

This operator is an implication induced by some *t-norm*, $x_j \,\phi\, y_\iota = x_j \Rightarrow \mathrm{y}_\iota$. If the *t-norm* is specified as the minimum operator, then the above operator reads as

$$r_{j\iota} = x_j \,\phi\, y_\iota = \begin{cases} 1 & \text{if } x_j \le y_\iota \\ y_\iota & \text{if } x_j > y_\iota \end{cases}$$

For the *inf-s* composition, the unique minimal solution is determined in the form:

$$\breve{R} = \boldsymbol{x} \,\delta\, \boldsymbol{y}$$

More specifically, we have $\breve{r}_{j\iota} = x_j \,\delta\, y_\iota$. The δ operator is implied by some *s-norm*, and comes in the form of the following expression;

$$x_j \,\delta\, y_\iota = \inf \{r_{j\iota} \in [0, 1] \mid x_j \text{ s } r_{j\iota} \ge y_\iota\} \tag{3.5}$$

If the *s-norm* is specified as the maximum operator then we have,

$$r_{j\iota} = x_j \,\delta\, y_\iota = \begin{cases} y_\iota & \text{if } x_j \le y_\iota \\ 0 & \text{if } x_j > y_\iota \end{cases}$$

The specificity of a fuzzy set (interval) A is denoted as spec(A), and the following conditions must be satisfied [119]:

1. spec(∅) = 0,
2. spec(A) = 1 if and only if $A(x_0) = 1$ and $A(x) = 0\ \forall\, x \in \boldsymbol{x}$, and
3. if A_1 and A_2 are *normal* fuzzy sets and $A_1 \supset A_2$, then $\text{spec}(A_1) \le \text{spec}(A_2)$.

The first condition states that the empty set ∅ has the minimal specificity value. The second condition implies that only sets with just one element (singletons) can have the maximum specificity value, in this case one. The third condition states that the specificity measure of a normal fuzzy set decreases as the cardinality of a fuzzy set increases. A fuzzy set A is considered *normal* if its maximum membership function is equal to one, e.g. max(A) = 1.

3.2. Fuzzy Clustering and Performance Assessment

Since real-world data does not come represented as information granules, there is the need to form a conceptual interface between the logic processing realized by the fuzzy models and the numeric data. This interface can be formed through a series of information granules built either in the input or the output space. Typically such information granules are constructed through fuzzy clustering [20]. The Fuzzy C-Means (FCM) algorithm was proposed by Dunn [40] and improved by Bezdek [17], [19], and is one of the most commonly used objective function-based clustering techniques that has been successfully applied to different areas. The obtained clusters form a granular signature of the data, and as such, form a prerequisite for pursuing other processing, usually carried out in a supervised mode. In this section, we briefly recall a construction of information granules through fuzzy clustering, and look at the ways of assessing the quality of results with regard to their classification, and reconstruction abilities.

3.2.1. Fuzzy C-Means

Fuzzy C-Means (FCM) partitions a collection of N data $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_\mathrm{N}$ into c clusters, where $1 < c < N$. As a result, a collection of c prototypes, $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c$, and a partition matrix, $U = [\upsilon_{ik}]$, $i = 1, 2, \dots, c$; $k = 1, 2, \dots, N$ are formed. The partition matrix satisfies the intuitively appealing conditions; $\upsilon_{ik} \in [0, 1]$, $\sum_{i=1}^{c} \upsilon_{ik} = 1 \; \forall \, k$, and $0 < \sum_{k=1}^{N} \upsilon_{ik} < N \; \forall \, i$. In the FCM method, the following objective function is minimized,

$$J = \sum_{i=1}^{c} \sum_{k=1}^{N} \upsilon_{ik}^{m} \|\boldsymbol{v}_i - \boldsymbol{x}_k\|^2$$

Where $\|\cdot\|$ stands for some distance function, and m ($m > 1$) is a fuzzification coefficient, which affects the shape (form) and overlapping of the resulting membership functions. Typically the value of this coefficient is equal to $m = 2$, where the membership functions resemble a Guassian-like shape. Values closer to one, yield membership functions that resemble characteristic functions of sets, while values higher than 2 produce “spiky” membership functions, along with a rippling effect [119]. This behavior has a direct impact in the final result of the fuzzy relational and granular models. Since it is certain that real-world records are expressed in spaces of very different dimensionalities, we need to adhere to the weighted (normalized) Euclidean distance to avoid any bias towards any particular component. This distance is expressed as follows,

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

Where σ_j is the standard deviation of the jth feature. Assuming that the Euclidean distance is used, the prototypes are calculated as follows [20];

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{N} \upsilon_{ik}^{m} \boldsymbol{x}_k}{\sum_{k=1}^{N} \upsilon_{ik}^{m}} \tag{3.6}$$

And the entries of the partition matrix are updated by using the following expression,

$$\upsilon_{ik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{v}_i - \boldsymbol{x}_k\|}{\|\boldsymbol{v}_j - \boldsymbol{x}_k\|} \right)^{\frac{2}{m-1}}} \tag{3.7}$$

The entire process is repeated until there are no significant changes to the entries of the partition matrix U reported in the two successive iterations of the algorithm. The time complexity of the FCM algorithm is $O(Nnc^2\tau)$, where N is the total number of data clustered, n is the number of variables, and τ is the total number of iterations. For comparison purposes, the K-Means algorithm has a complexity of $O(Nnc\tau)$. It is clear that FCM requires more computation time, due to the fuzzy measures and the involvement of calculations in the algorithm. Nevertheless, K-means produces non-overlapping information granules with hard boundaries, while FCM generates overlapping clusters, which is in an advantage that makes FCM a superior algorithm. Here, the information granules built by FCM offer a better result for overlapped data with complex topologies.

Another advantage of FCM is that the algorithm performs unsupervised learning of unlabeled data, and it is always convergent. Moreover, the algorithm is simple and very efficient when representing the essence of the numerical data. The limitations of this algorithm is that it is sensible to the initialization of the partition matrix, and to noise and outliers. Other algorithms like Possibilistic C-Means (PCM) are able to handle noisy data samples. Nevertheless, the algorithm is more sensitive to good initializations, and at the end of clustering, coincident information granules may result. In contrast, the Fuzzy Possibilistic C-Means (FPCM) algorithm is able to ignore the noise sensitivity deficiency of FCM, and it manages to overcome the problem of coincident clusters that PCM has. Nevertheless, the algorithm has more numerical constraints than FCM. In this study we selected the FCM algorithm mainly due to its clustering efficiency, robustness, and because it has been extensively (and successfully) used in a wide range of engineering and scientific disciplines, for instance, medicine imaging, pattern recognition, data mining, and bioinformatics, just to name a few. However, the performance of the clustering algorithm is always dependent on the data under study. In this dissertation we exploit the FCM algorithm and explore different approaches (and modifications) to efficiently construct associations between the input and output spaces, and at the same time increase the quality of the information granules produced, as well as their clustering abilities.

When using FCM to construct the information granules, there is the need of using a certain methodology to help us assess and quantify the performance of the clusters. Two categories of problems can be formulated here:

(i) Representation problems. Here we are interested in quantifying how good the prototypes are in representing (especially reconstructing) the original data.
(ii) Classification problems. This class of problems is concerned with the use of obtained structural information (prototypes and the partition matrix) to classify individual data.

Since the best number of clusters c or fuzzification coefficient m for a given dataset are not known *a priori*, one can test different combinations of these parameters, and quantify the performance of the information granules by using a so-called *reconstruction criterion* or the *classification rate* of the clusters formed (if labeled data exists).

3.2.2. Reconstruction Criterion

The view at the quality of the clusters produced by the FCM algorithm is established by looking at a way in which the structural information obtained through the clustering process leads us to the reconstruction of the original data. This reconstruction process is part of an overall process of granulation-degranulation [122]. In essence, the quality of reconstruction of the original data is expressed as the following sum,

$$Q = \frac{1}{N}\sum_{k=1}^{N} ||\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k||^2 \tag{3.8}$$

where $\hat{\boldsymbol{x}}_k$ is the reconstructed version of $\boldsymbol{x}_k$ determined as,

$$\hat{\boldsymbol{x}}_k = \frac{\sum_{i=1}^{c} \upsilon_{ik}^{m}\boldsymbol{v}_i}{\sum_{i=1}^{c} \upsilon_{ik}^{m}}. \tag{3.9}$$

Is important to stress that the weighted Euclidean distance is used in (3.9) to determine the discrepancy (distance) between the original n-dimensional entities (patterns), and the reconstructed ones.

3.2.3. Classification Rate

Assuming that the class labels of the clustered data are available, we can assess the quality of the clusters by quantifying their homogeneity as far as class labels are concerned [49], [77], [115]. Ideally, given c clusters and ϖ classes where $\varpi \leq c$, we may wish to see that each cluster groups the patterns belonging to a single class only. The determination of the classification rate (or the classification error, respectively) is completed in the following way. Denote by $\boldsymbol{X}_i$ a collection of patterns that belong to the ith cluster at the highest degree (in comparison to their membership values in other clusters), namely

$$\boldsymbol{X}_i = \{\boldsymbol{x}_k \mid \upsilon_{ik} > \max_{j=1,2,\ldots,i-1,i+1,\ldots,c}\upsilon_{jk}\}$$

Let the number of patterns in $\boldsymbol{X}_i$ be denoted by N_i. Furthermore, the most frequent class of patterns in $\boldsymbol{X}_i$ has been determined. Then we count the number of patterns that belong to $\boldsymbol{X}_i$ but their class label is different from the one being dominant in this cluster; denote this number by M_i. It is very likely that $M_i = N_i$ and we rather encounter a situation when $M_i < N_i$. Repeating the same assessment of the remaining clusters, we form a classification error through the following expression

$$Q = \frac{M_1 + M_2 + \ldots + M_c}{N_1 + N_2 + \ldots + N_c}$$

equivalently, a classification rate is proposed in the form,

$$Q = 1 - \frac{M_1 + M_2 + \ldots + M_c}{N_1 + N_2 + \ldots + N_c} \tag{3.10}$$

If the classification rate is required as a percentage, one simply multiplies the result from (3.10) by 100.

3.3. Construction of Granular Fuzzy Models

Once a fuzzy model is formed, we can generalize it by constructing and optimizing its granular counterpart. To construct a granular fuzzy model, the information granules have to be carefully formed and their size should be well-justified. The *principle of justifiable granularity* deals with the formation of a representative information granule Ω, based on the experimental evidence regarded as a single-dimensional numeric data set, e.g. $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$. An information granule has to obey two requirements:

i) The numerical evidence gathered within the bounds of Ω has to be as high as possible. This guarantees that the presence of the information granule is well-justified, and is reflective of the existing experimental data.

ii) At the same time the information granule Ω should be as detailed (specific) as possible meaning that it comes with a well-defined semantics.

Since the data are single-dimensional, Ω can be treated as an interval to be constructed around the data and represented by $A = [a^-, a^+]$, where a^- and a^+ are lower and upper bounds respectively. If the interval is narrow (specific), the knowledge about Ω is relatively precise, however if the interval is large, the specificity decreases along with the knowledge about Ω. If the lower and upper bound are equal, the knowledge about Ω is exact and it is considered a "standard" fuzzy set [46]. The first requirement is evaluated (quantified) by counting the number of data that fall within the bounds of Ω, or that are covered by this interval. We can consider a function of the cardinality of Ω, $f_1(\text{card}\{x_k \in \Omega\})$, where f_1 is an increasing function. The second requirement can be quantified by looking at the size of the interval Ω, in this case the length of the interval serves as a specificity measure, namely $f_2(\text{length}(\Omega))$, where $\text{length}(\Omega) = |a^+ - a^-|$ and f_2 is a decreasing function. These two measures seem to be conflict since as the number of values that fall within the interval increases, the length of the interval Ω increases and becomes less detailed (specific) [112].

An interesting idea would be to construct the interval of Ω by finding a numeric representative of the data $\mathcal{D}$. This representative can be determined by the median $\text{med}(\mathcal{D})$, which is usually a value that comes as one of the elements of $\mathcal{D}$. The determination of the lower and upper bound are realized independently. Here we describe how to calculate the upper bound a^+; the calculation of the lower bound a^- is performed in an analogous fashion. The cardinality measure functional is the measure of all the elements that are positioned to the right of the median, this is:

$$f_1(\text{card}\{x_k \in \Omega | x_k > \text{med}(\mathcal{D})\})$$

and the functional of the length now is given by,

$$f_2(|\text{med}(\mathcal{D}) - a^+|)$$

Since these two requirements are in conflict, we could turn to multiobjective optimization, or maximize the product $Q = f_1 * f_2$. Furthermore, we can consider the inverse of the length and a parameter α that can calibrate the impact of the specificity criterion on the constructed information granule [112]; this will change the second functional to,

$$f_2 = e^{-\alpha|\text{med}(D) - a^+|} \tag{3.11}$$

3.3.1. Granular Fuzzy Relational Equations

Taking into account the formation of justified information granules, we can use these conditions to build granular fuzzy models by using a fuzzy model as a guide. Information granularity is viewed as an important design asset and one should carefully control how the information granules start forming around a model. It is essential that the granulation (abstraction) of the model is performed in an efficient manner, therefore the allocation of granularity is subject to optimization. In case that the system of fuzzy relational equations is unsolvable, we find an approximation to construct a relation R by using optimization mechanisms. Then the granulation of the information acquired by the construct will generalize the relation R in a way that it has more affinity to the results within the model constructed. This approach can be viewed as a granular solution to solve a system of fuzzy relational equations, and to tackle the *estimation* problem.

To construct the information granules, let us assume that we have experimental input-output data pairs $(\boldsymbol{x}_k, \boldsymbol{y}_k)$, $k = 1, 2, \ldots, N$, $\boldsymbol{x}_k \in [0, 1]^n$, $\boldsymbol{y}_k \in [0, 1]^\eta$, and a *level of granularity*, viz. $\varepsilon \in [1, 0]$. The available granularity level ε is allocated to the parameters of the mapping R, where $\dim(R) = n\eta$, so that an optimization criterion is maximized, as long as the allocation of the level of granularity satisfies the following constraint,

$$\sum_{\iota=1}^{\eta}\sum_{j=1}^{n} \varepsilon_{j\iota} = \varepsilon n\eta \tag{3.12}$$

where $\varepsilon_{j\iota}$ is associated with the (j, ι)th parameter of the original relation R. By making the entries of R granular, the Equation in (3.1) becomes a granular fuzzy relational equation in the following form,

$$\boldsymbol{x}_k \text{ op } G(R) = Y_k \tag{3.13}$$

Similar to the previous section, both coverage and specificity can be considered for the optimization of the information granule. For the coverage criterion, and for each $\boldsymbol{x}_k$, the information granule $Y_k = f(\boldsymbol{x}_k, G(R))$ is computed. And since $Y_k = [y_k^-, y_k^+]$, the degree of inclusion of y_k in the information granule formed is determined by $\text{incl}(y_k, Y_k) = y_k \in Y_k$, or the inclusion of the kth output datum in the kth information granule. Then the average sum of all the degrees of inclusion taken over all the output features is calculated as,

$$Q(\varepsilon) = \frac{1}{N\eta}\sum_{k=1}^{N}\sum_{\iota=1}^{\eta} \text{incl}\left(y_{k\iota}, \left(\boldsymbol{x}_k \circ G(R_\iota)\right)\right) \tag{3.14}$$

Where R_t is a fuzzy relation approximated for each output feature. Since the information granule comes in the form of an interval, each inclusion will return a Boolean value.

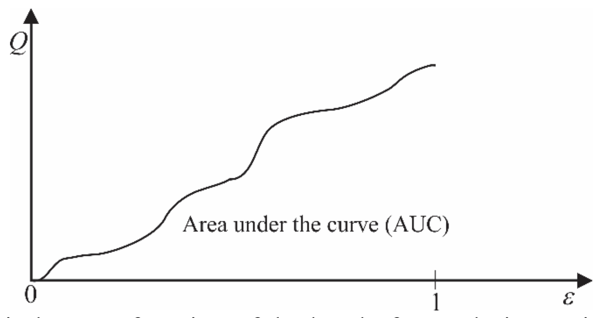
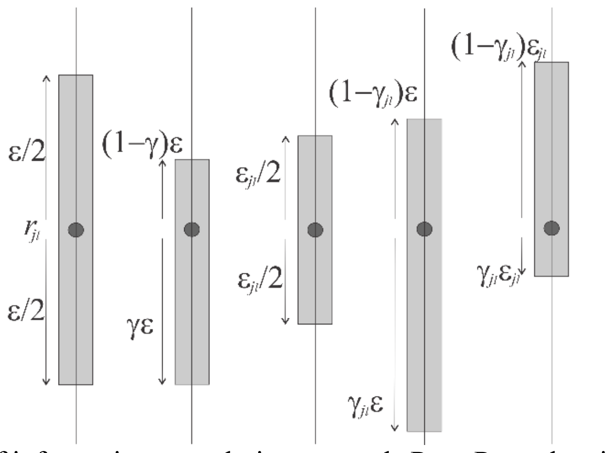


Figure 3.1. Performance index as a function of the level of granularity ε with an area under the curve (AUC).

The second criterion is the specificity criterion, which is viewed as a decreasing function of the length in the interval. Then the specificity criterion can be expressed as the average sum of the length of the intervals as,

$$\ell = \frac{1}{N}\sum_{k=1}^{N} |y_k^{+} - y_k^{-}| \tag{3.15}$$

The inverse of the length of the interval Y_k could be computed as $e^{(-\text{length}(Y_k))}$, if a single-objective optimization problem is desired.


ε/2
r_{ji}
ε/2
(1−γ)ε
γε
$\varepsilon_{ji}/2$
$\varepsilon_{ji}/2$
$(1-\gamma_{ji})\varepsilon$
$\gamma_{ji}\varepsilon$
$(1-\gamma_{ji})\varepsilon_{ji}$
$\gamma_{ji}\varepsilon_{ji}$
Figure 3.2. Allocation of information granularity protocols P_1 to P_5, and an illustration of the resulting granulation of the fuzzy relations.

To assess the performance of the granularity level, the area under the curve (AUC) is measured. This measure is deemed as a global quality descriptor of the granular model. Here, an integration or summation of the values $Q(\varepsilon)$ is calculated. In other words the AUC is represented by $\int_0^1 Q(\varepsilon)\, d\varepsilon$, where $Q(\varepsilon)$ is calculated as described in (3.14) [112], [123].

To allocate the granularity in the data, and justify the formation of the information granule, a certain information allocation protocol can be followed. Several main classes of such protocols are proposed, and a general example is displayed in Figure 3.2.

Protocol 1: Uniform allocation of granularity. The level of granularity ε is distributed symmetrically around the original parameters of the mapping. This is the simplest process since it does not require an optimization method. All the numeric values are used in the same way and become replaced by intervals of the same length. The numeric parameter of the relation r_{ji} is replaced by the information granule $G(r_{ji})$. This interval is obtained as follows.

$$\left[r_{ji} - \frac{\varepsilon}{2},\ r_{ji} + \frac{\varepsilon}{2}\right] \tag{3.16}$$

Protocol 2: Uniform allocation of granularity with asymmetric position of the intervals. The intervals are the same length but asymmetrically distributed around the numeric parameter. A value $\gamma \in [0, 1]$ controls this level of asymmetry in the interval. Notice that Protocol 2 subsumes Protocol 1 when $\gamma = 1/2$. This protocol also does not recall any optimization process to allocate the asymmetric level of granularity.

$$\left[r_{ji} - \gamma\varepsilon,\ r_{ji} + (1-\gamma)\varepsilon\right] \tag{3.17}$$

Protocol 3: Non-uniform allocation of information granularity with symmetrically distributed intervals of information granules. The level of granularity ε has various levels of asymmetry (ε_{ji}), also associated with their corresponding entries of r_{ji}. These levels of asymmetry also have to be adjusted with a certain optimization mechanism, and the expression is as follows,

$$\left[r_{ji} - \frac{\varepsilon_{ji}}{2}, r_{ji} + \frac{\varepsilon_{ji}}{2}\right] \tag{3.18}$$

Protocol 4: Non-uniform allocation of information granularity with asymmetrically distributed intervals of information granules. Various levels of asymmetry are admitted (γ_{ji}). In this case the matrix of the levels of asymmetry γ_{jl} must be adjusted by engaging on some optimization mechanism. This expression is described by,

$$\left[r_{jl} - \gamma_{jl}\varepsilon, r_{jl} + \left(1 - \gamma_{jl}\right)\varepsilon\right] \tag{3.19}$$

Protocol 5: Non-uniform allocation of information granularity with asymmetrically distributed intervals of information granules. Various levels of asymmetry are admitted for the level of asymmetry γ and the level of granularity ε; γ_{jl} and ε_{jl} respectively, and both parameters have to be adjusted.

$$\left[r_{jl} - \gamma_{jl}\varepsilon_{jl}, r_{jl} + \left(1 - \gamma_{jl}\right)\varepsilon_{jl}\right] \tag{3.20}$$

Protocol 6: As a point of reference, and to help asses a relative performance of previous protocols, the random allocation of granularity is considered. This will allow to demonstrate how the optimized process of granularity allocation performs over a purely random allocation process.

3.4. Granular Fuzzy Modeling through Associations among Information Granules

An alternative method used to construct the granular fuzzy models is by forming the model without any guide by an initial fuzzy model. To build these granular models, several main phases are required; *i*) define and refine a set of intervals positioned in the output space; and *ii*) complete a so-called *conditional fuzzy clustering* for the input data induced by the intervals initially constructed. We are interested in studying the impact that several design parameters have when these granular models are constructed. Namely, the number of output intervals, size of each interval, number of clusters per interval, and the fuzzification factor of the clustering method.

3.4.1. Interval-Based Information Granules

As usual in system modeling, a granular fuzzy model is constructed on a basis of certain data which may come in the form of a collection of input-output data pairs $(\boldsymbol{x}_k, y_k)$, $k = 1, 2, \ldots, N$, where $\boldsymbol{x}_k \in \mathbb{R}^n$ and $y_k \in \mathbb{R}$. To build the model, a collection of p interval information granules are formed in the output space, namely $D_1, D_2, \ldots, D_p$. These intervals form a partition of the output space, meaning that those intervals are pairwise disjoint and cover the entire output space. Several intuitively appealing techniques of building this collection of intervals are sought:

Equal size intervals: We make the intervals of the same length by splitting the range of the output space into intervals (bins) of equal size. The method is simple however the intervals formed in this manner do not reflect the nature of the data: there could be a few data points for which a separate interval is formed while a large number of data (exhibiting potentially high diversity) are accommodated within the same interval (which could have a detrimental impact on the performance of the model).

Equal probability (EP) criterion: The intervals are formed in such a way so that each interval includes the same number of data (a fraction p/N, to be specific). In other words, the intervals are relatively narrow if there is a large number of data in this region, and become broad if the density of data is low.

Irrespectively of the way the intervals are formed, there is a common limitation associated with them – they do not reflect the nature of input data (by being exclusively focused on the output space and not taking into consideration the input-output relationships).

Optimization of intervals: The intervals are optimized by forming them in a way it models the input – output dependencies (which are the crux of any model). With this regard, we need to formulate a suitable performance index and establish an optimization scheme.

3.4.2. Conditional Fuzzy Clustering

The formation of the output intervals is critical to the functioning of the overall granular model as subsequently they shape up a collection of information granules emerging in the input space. The rationale here is to map the output interval to the corresponding regions of the input. The essence is to run clustering, but in contrast to the "standard" FCM, here we invoke a so-called *conditional* or context-based FCM [110], [111]. To have a better understanding of this method, let us assume that we have a collection of interval-based information granules $D_1, D_2, \ldots, D_p$ positioned in the output space. Let $D_h = [d_h^-, d_h^+]$, $h = 1, 2, \ldots, p$ denote the lower and upper bounds of the hth interval (context). We cluster the data that are falling within a context of D_h – in other words, the clustering is conditioned by the output interval.

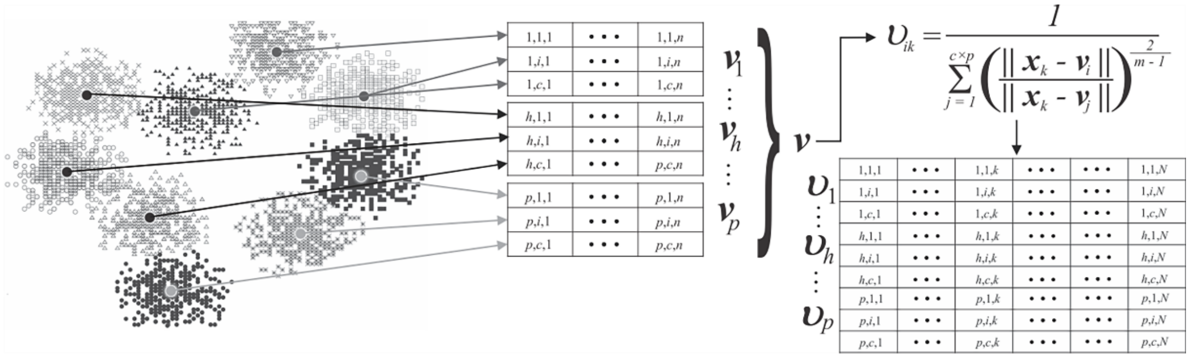


Figure 3.3.Calculation of the new partition matrix by using the prototypes obtained in the clustering stage by applying (3.21) and (3.22).

For each of data set falling within D_h, the task is to determine its structure and obtain a collection of cluster centers (prototypes) by clustering the data with FCM. For simplicity let us consider that the number of clusters c produced for each interval is equal. The FCM algorithm is realized iteratively in each context h by updating the values of the partition matrix and the prototypes. To calculate the cluster centers, (3.6) is modified as follows, [110],

$$\boldsymbol{v}_{hi} = \frac{\sum_{k=1}^{N} \upsilon_{hik}^{m} \boldsymbol{x}_{hk}}{\sum_{k=1}^{N} \upsilon_{hik}^{m}} \tag{3.21}$$

where i = 1, 2, …, c, k = 1, 2, …, N, h = 1, 2, …, p. Furthermore $\boldsymbol{v}_{hi}$ is the ith prototype in the hth context. The entries of the partition matrix are updated by modifying (3.7) as follows;

$$\upsilon_{hik} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|\boldsymbol{v}_{hi} - \boldsymbol{x}_{hk}\|}{\|\boldsymbol{v}_{hj} - \boldsymbol{x}_{hk}\|} \right)^{\frac{2}{m-1}}} \tag{3.22}$$

Where υ_{hik} represents the element of the partition matrix induced by the ith cluster and the kth data in the hth context, $\boldsymbol{x}_{kh}$ is the kth data in the hth context. As portrayed in Figure 3.3, the prototypes obtained for each context, are combined and utilized to determine a new partition matrix for the input data, now having $c \times p$ clusters. This method helps bring a directionality component onto the granular fuzzy model, given that the information granules emerging in the input space are directly implied by the structure already established in the output space.

3.5. Granular Modeling with Hyperboxes

Due to the plethora of different structures and topologies found in real-world data, it is a good idea to search for alternative methods to help represent the essence of the data through a set of information granules. A popular method found in the literature is the use of information granules coming as hyperboxes, due to their simplicity, effectiveness and robustness in clustering or classification tasks.

The approach developed by Simpson [145], [146], [147], follows a three-step method to build a so-called min-max neural network (NN), where *c* hyperboxes are constructed and used to cluster or classify data. These steps are 1) hyperbox expansion, 2) overlap test, and 3) hyperbox contraction. In the expansion phase, the algorithm identifies hyperboxes that are capable of reaching newly discovered data points and expands the suitable hyperbox to cover them. After step one is finished the overlap test starts, where the algorithm performs a dimension-by-dimension comparison between hyperboxes, if an overlap exists between two hyperboxes, the smallest overlap along any dimension (variable *j*) is retained and used during the contraction phase. This helps keep the hyperbox size as large as possible. After an overlap is detected, the hyperbox contraction phase starts, and eliminates the overlap by minimally adjusting the hyperbox.

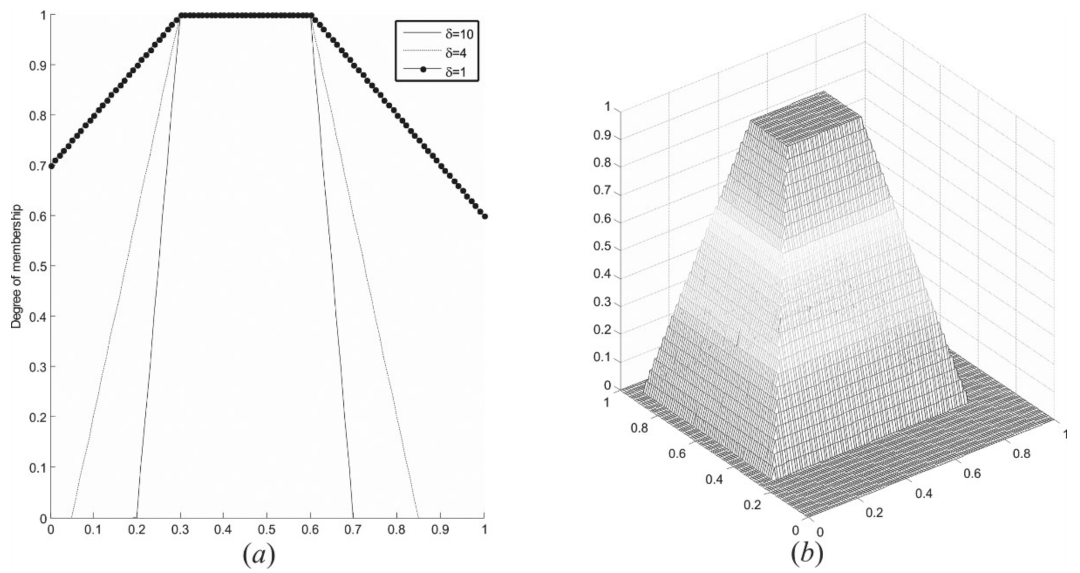


Figure 3.4. (*a*) Different degrees of membership obtained with different values of δ. (*b*) Two dimensional fuzzy set has been defined with the use of (3.23) with the lower corner at (0.5, 0.3), the upper corner at (0.7, 0.6), and $\delta = 4$.

To maintain conciseness of this section, we consider that the data are previously normalized to the unit hypercube, that is to say $(\boldsymbol{x}_k, y_k)$, $k = 1, 2, \ldots, N$, $\boldsymbol{x}_k \in [0, 1]^n$, $y_k \in [0, 1]$. Moreover, a hyperbox defined in $[0, 1]^n$ and denoted by B can be fully described by its lower ($\boldsymbol{l}$) and upper ($\boldsymbol{u}$) corners (coordinates). Hence a more specific notation comes in the form of $B = \{\boldsymbol{l}, \boldsymbol{u}\}$, where $\boldsymbol{l}$, $\boldsymbol{u} \in [0, 1]^n$. Obviously a strict inclusion relation should hold, i.e. $\boldsymbol{l} \leq \boldsymbol{u}$. If $\boldsymbol{l} = \boldsymbol{u}$, then the hyperbox reduces to a single numeric entity. For more hyperboxes ($1 < c < N$), we have $B_i = \{\boldsymbol{l}_i, \boldsymbol{u}_i\}$, $i = 1, 2, \ldots, c$, where $\boldsymbol{l}_i$, $\boldsymbol{u}_i$ are vectors of the corresponding coordinates, or $\boldsymbol{l}_i = [l_{1i}, l_{2i}, \ldots, l_{ni}]$ and $\boldsymbol{u}_i = [u_{1i}, u_{2i}, \ldots, u_{ni}]$ for the ith cluster. To discover a datum near a hyperbox, this method uses a *hypercube membership function* that helps obtain the membership degree of a datum to a given hyperbox computed as follows [16], [147];

$$\mu_{B_i}(\boldsymbol{x}_k, \boldsymbol{l}_i, \boldsymbol{u}_i) = \min_{j=1,2,\ldots,n}\left(\min\left(\left[1 - f\left(x_{jk} - u_{ji}, \delta\right)\right], \left[1 - f\left(l_{ji} - x_{jk}, \delta\right)\right]\right)\right)$$
$$f(r, \delta) = \begin{cases} 1 & \text{if } r\delta > 1 \\ r\delta & \text{if } 0 \leq r\delta \leq 1 \\ 0 & \text{if } r\delta < 0 \end{cases} \tag{3.23}$$

where $f(r, \delta)$ is a two-parameter function in which r represents the distance of a data point to the hyperbox, and δ is a sensitivity parameter which regulates how fast the membership decreases as the distance between $\boldsymbol{x}_k$ and B_i increases, in other words, it regulates the degree of change from a full membership (within the hyperbox) to a reduced membership (outside the hyperbox), see Figure 3.4(a).

A third parameter used in this method is a user-defined parameter ($0 \leq \theta \leq 1$), which is a value used to control the size of the hyperboxes. This parameter bounds a hyperbox to expand to a maximum size, therefore the following constraint must be met,

$$\sum_{j=1}^{n}\left(\max\left(u_{ji}, x_{jk}\right) - \min\left(l_{ji}, x_{jk}\right)\right) \leq \theta$$

If there exists a datum close to a hyperbox, and if the hyperbox is constrained to its maximum size, then a new hyperbox is created with the corners equal to the coordinates of the newly discovered datum. If the datum is within the reach of the hyperbox, then the hyperbox is expanded to include it. Though, if the datum is already included inside the hyperbox, the later remains unmodified. This process is repeated until a set of hyperboxes that cover the data has been produced; in the sequel the next two phases take place.

It has to be highlighted that the resulting number of hyperboxes heavily relies on the chosen value of θ. If the value is small then many small hyperboxes are obtained, where each of them could cover a relatively small number of data, increase the network complexity, and result in a highly accurate model. In contrast, larger hyperboxes will include a higher amount of data, decrease the network complexity, and could lead to a less accurate model. Therefore, careful measures must be taken when setting the value of this parameter. When clustering data, it might be counterproductive to end with many small hyperboxes, instead it is ideal to have a smaller number of large and descriptive hyperboxes. Moreover, choosing the correct value of θ for each dataset can be a time-consuming task.

To eliminate overlaps among hyperboxes, each corner of the hyperboxes is compared in a dimension-by-dimension basis. If an overlap exists among two hyperboxes the smallest overlap along any dimension is memorized ($\Delta = j$) and used during the contraction phase. To exemplify the process, let us assume that there are two hyperboxes B_1 and B_2 that were expanded after completing step 1. Next the overlap test takes place by checking each hyperbox corner as follows;

$$\begin{aligned}
&\text{Case 1: } l_{1j} < l_{2j} < u_{1j} < u_{2j} \\
&\text{Case 2: } l_{2j} < l_{1j} < u_{2j} < u_{1j} \\
&\text{Case 3: } l_{1j} \le l_{2j} \le u_{2j} \le u_{1j} \\
&\text{Case 4: } l_{2j} \le l_{1j} \le u_{1j} \le u_{2j}
\end{aligned}$$

If an overlap is detected, it is eliminated only for dimension Δ, which as mentioned before, represents the smallest overlap. The hyperboxes are then modified by considering the following four cases:

Case 1: if $l_{1\Delta} < l_{2\Delta} < u_{1\Delta} < u_{2\Delta}$

$$u_{1\Delta}^{new} = l_{2\Delta}^{new} = \frac{u_{1\Delta}^{old} + l_{2\Delta}^{old}}{2}$$

Case 2: if $l_{2\Delta} < l_{1\Delta} < u_{2\Delta} < u_{1\Delta}$

$$u_{2\Delta}^{new} = l_{1\Delta}^{new} = \frac{u_{2\Delta}^{old} + l_{1\Delta}^{old}}{2}$$

Case 3a: if $l_{1\Delta} \le l_{2\Delta} \le u_{2\Delta} \le u_{1\Delta}$ and $(u_{2\Delta} - l_{1\Delta}) < (u_{1\Delta} - l_{2\Delta})$

$$l_{1\Delta}^{new} = u_{2\Delta}^{old}$$

Case 3b: if $l_{1\Delta} \le l_{2\Delta} \le u_{2\Delta} \le u_{1\Delta}$ and $(u_{2\Delta} - l_{1\Delta}) > (u_{1\Delta} - l_{2\Delta})$

$$u_{1\Delta}^{new} = l_{2\Delta}^{old}$$

Case 4a: if $l_{2\Delta} \leq l_{1\Delta} \leq u_{1\Delta} \leq u_{2\Delta}$ and $(u_{2\Delta} - l_{1\Delta}) < (u_{1\Delta} - l_{2\Delta})$

$$u_{2\Delta}^{new} = l_{1\Delta}^{old}$$

Case 4b: if $l_{2\Delta} \leq l_{1\Delta} \leq u_{1\Delta} \leq u_{2\Delta}$ and $(u_{2\Delta} - l_{1\Delta}) > (u_{1\Delta} - l_{2\Delta})$

$$l_{2\Delta}^{new} = u_{1\Delta}^{old}$$

In this step the hyperboxes are contracted, and all the overlaps are eliminated in a single pass, while retaining at the same time most of the data initially covered.

3.6. Particle Swarm Optimization

To optimize the granular fuzzy models we use the ensuing Particle Swarm Optimization (PSO) algorithm [70]. PSO was discovered through a simplified social model, a study of bird flocks, fish schooling, and swarming theory. The key feature is to optimize nonlinear functions using particle swarm methodology.

In a nutshell, this algorithm maintains a swarm of ρ particles, and each particle represents a solution for the granular models, and the size of this particles is suited to the objective function used to optimize them. The particles are *flown* trough a multidimensional search space, where the position of each particle is adjusted according to its own experience and the experience of its neighbors in a given generation. This algorithm uses a velocity vector $\mathrm{v}_{gq}(\tau)$ which controls the movements of the particles along with the optimization process, by reflecting both the experiential knowledge of the particle and the socially exchanged information from the particle's neighborhood. Let $d_{gq}(\tau)$ denote the position of the gth particle in the qth dimension at time step τ. The position of this particle is updated by adding the velocity to its current position. The velocity is calculated by using the following expression [42], [70]:

$$\mathrm{v}_{gq}(\tau+1) = \omega \mathrm{v}_{gq}(\tau) + c_1 r_{1q}(\tau)\left(\Psi_{gq}(\tau) - d_{gq}(\tau)\right) + c_2 r_{2q}(\tau)\left(\widehat{\Psi}_q(\tau) - d_{gq}(\tau)\right) \tag{3.24}$$

where $\mathrm{v}_{gq}(\tau)$ is the velocity of the gth particle in the qth dimension at time step τ, $q = 1, ..., n_d$, and n_d is the dimension of the particle. The personal best position $\boldsymbol{\Psi}_g(\tau)$, associated with particle g is the best position the particle has visited since $\tau = 1$, and $\widehat{\boldsymbol{\Psi}}(\tau)$ is the global best position found by the swarm so far. Furthermore, $r_{1q}(\tau), r_{2q}(\tau) \in (0, 1)$ are a uniform distribution of random values in the range (0, 1) obtained for an equal number of elements in the particle for every new velocity computation. Finally, c_1 and c_2 are positive acceleration constants used to scale cognitive and social components respectively; usually $c_1 = c_2 = 2$, but some other values could be used, see [42]. The particle's new position is updated by using the following expression:

$$d_{gq}(\tau + 1) = d_{gq}(\tau) + \mathrm{v}_{gq}(\tau + 1)$$

The variable ω in (3.24) is a weight applied as a mechanism to control the exploration and exploitation abilities of the swarm. This is an inertia weighting factor that can be constant or it can be initialized as a large value, usually 0.9, and then linearly or non-linearly decreased through the course of the PSO run, usually being reduced to 0.4 [70]. This is done because large inertia weights facilitate a global search, while small inertia weights facilitate a local search.

3.7. Differential Evolution

Another strategy used to optimize the fuzzy and granular fuzzy models is the Differential Evolution algorithm (DE), which is a stochastic, population-based search strategy developed by Price and Storn [149], [150]. This algorithm exhibits some resemblance with other evolutionary algorithms (EA), but it differs considerably in the sense that distance and direction from the current population is used to guide the search process. Another difference is that mutation is applied first to generate a trial vector, and secondly a crossover operator is utilized to produce an offspring.

In this dissertation, the DE algorithm is used to optimize all the granular models proposed, henceforth the optimization process is similar for all models, and each model has a suitable individual representation. To start, a population of ρ individuals is initially generated, each representing a vector $\boldsymbol{d}$ initialized by using a normalized random distribution in $\mathbb{R}^n$, or more specifically the one located in $(0, 1)^n$. For each parent $\boldsymbol{d}_g(\tau)$, $g = 1, 2, \ldots, \rho$, a trial vector $\boldsymbol{o}_g(\tau)$ is generated as follows [42]; randomly select a target vector $\boldsymbol{d}_{g_1}(\tau)$ from the population, then randomly select two other individuals $\boldsymbol{d}_{g_2}(\tau)$ and $\boldsymbol{d}_{g_3}(\tau)$ such that $g \neq g_1 \neq g_2 \neq g_3$. With these individuals, the trial vector is computed by mutating the target vector as follows;

$$\boldsymbol{o}_g(\tau) = \boldsymbol{d}_{g_1}(\tau) + \beta\left(\boldsymbol{d}_{g_2}(\tau) - \boldsymbol{d}_{g_3}(\tau)\right) \tag{3.25}$$

where $\beta \in (0, \infty)$ is a scaling factor used to control the amplification of the differential variation. Next we proceed by applying a discrete recombination of the trial vector $\boldsymbol{o}_g(\tau)$ and the parent vector $\boldsymbol{d}_g(\tau)$ to produce offspring $\boldsymbol{d}_g'(\tau)$, as follows;

$$d'_{gq}(\tau)=\begin{cases} o_{gq}(\tau) & \text{if } \ q \in \Theta \\ d_{gq} & \text{otherwise} \end{cases}$$

where $d_{gq}(\tau)$ is the qth element of vector $\boldsymbol{d}_g(\tau)$, and Θ is a set of element indices that will suffer mutation, or the set of crossover points. These crossover points are randomly selected from a set of possible crossover points by using *binomial crossover* [42], [130], [150], where a *probability of crossover* value is used; the larger the value of the probability, the more crossover points are selected. In other words, more elements of the trial vector will be used to produce offspring; this process is repeated for a number of generations τ. At each generation the fitness of the offspring is compared to the fitness of the parent; if the performance of the offspring is higher, it replaces its parent, if the parent's performance is better it remains in the population.

With DE, with a larger population more differential vectors become available, and more directions can be explored. But it should be kept in mind that large population sizes increase the computational complexity per generation, and can cause the search to degrade to a parallel random search. Small population sizes would decrease the exploration abilities of the algorithms; however, the size of the population is always problem dependent.

To try to move the trial vector quickly towards the global optima, a different approach is introduced. To calculate new a trial vector $\boldsymbol{o}_g(\tau)$, the global best individual $\widehat{\boldsymbol{\Psi}}(\tau)$ found so far is used as a target vector, and two other individuals $\boldsymbol{d}_{g_1}(\tau)$ and $\boldsymbol{d}_{g_2}(\tau)$ are randomly selected, such that $g \neq g_1 \neq g_2$, then the trial vector is computed as follows;

$$\boldsymbol{o}_g(t) = \widehat{\boldsymbol{\Psi}}(\tau) + \beta\left(\boldsymbol{d}_{g_1}(\tau) \ - \ \boldsymbol{d}_{g_2}(\tau)\right) \tag{3.26}$$

Next, for each individual, a random value $r_g(\tau)$ is calculated. If $r_g(\tau) < \beta$ then (3.26) is used to compute a new trial vector, otherwise (3.25) is selected. Moreover, the scaling factor is calculated as $\beta = 1/\left(1 + e^{-(1/\tau)}\right)$, $\tau = 1, 2, \ldots, T$, where T is the total number of generations. Note that as the number of generation increases, the value of β decreases in the range of [1, 0.5], which results in a lower probability of using (3.26). Also, this method allows the contribution of the differential variation to decrease as the contribution of the global best individual increases.

3.8. Summary

The algorithms included in this Chapter are essential to the construction of the granular models proposed in this dissertation. Throughout this research, we show how important parameters are selected, and how some equations are altered and suited for each approach. Moreover, some strategies are used differently than originally intended, others are exploited more frequently due to their effectiveness and performance. But, at the end, the combination of these algorithms serve to achieve one purpose, build efficient granular fuzzy models to help us capture the essence of data, and give us a more abstract view of topology and distribution of the numerical entities under observation.

4. Granular Fuzzy Relational Structures

In this chapter, a fuzzy logic processor (FLP) is constructed and optimized. This model resembles an Artificial Neural Network (ANN), but the main difference is that the weights are adjusted by means of fuzzy relational equations. From the information captured by the trained FLP, a granular fuzzy relational model is formed. To build the FLP, several methods are proposed, namely the gradient based (GB) method, particle swarm optimization (PSO), and differential evolution (DE). The objective of this study is to examine and compare the performance of these learning mechanisms in the context of fuzzy relational models. Next, we compare the results with those coming from hybrid algorithms. The idea is to find a good initial set of values in the system of relational equations, by employing PSO or DE, and then use the GB algorithm to perform a local search. Furthermore, several compositions of diverse *t-norm* and *s-norms* are tested and used to construct the FLP. After the connections (weights) of the fuzzy model are optimized, these are converted into granular weights by allocating an optimal level of granularity. At the end, a granular fuzzy relational model is constructed and optimized; this will help assess the performance of the original fuzzy relational model, and at the same time offer a solution to the system of relational equations.

4.1. Problem Formulation

When seeking for good approximations to the system of relational equations, there are a number of important design factors that we have to consider. First, to build and optimize a fuzzy relational model, the data has to be transformed into information granules (fuzzy sets). If the FCM algorithm from Section 3.2 is used, we need to find a suitable number of clusters, and a proper fuzzification coefficient. Second, since the FLP is structurally similar to an artificial neural network (ANN), designing its architecture is imperative for a good performance of the model. Then, it is important to select a suitable number of logic neurons for the hidden layer. And third, to find the best model approximation, an appropriate training mechanism has to be selected. Once the fuzzy relational model is constructed, we generalize it by transforming the already optimized connections (weights) into granular weights. Granular fuzzy models, based on the principles of fuzzy modeling, are viewed as a vehicle for alternative solutions to the development of models. The structure of the granular model is constructed from the information already captured by the fuzzy models (FLPs). The methods developed in the ensuing sections are aimed to pursue an important goal; acquire knowledge, and allow the extraction of such learning. Moreover, by incorporating granular information, the performance of the original non-granular model is quantified, and since the model is being abstracted, the knowledge acquired can be transferred to other models formed with data coming from similar sources. These granular constructs are also viewed as an alternative solution to the system of fuzzy relational equations.

4.2. Fuzzy Relational Models

Fuzzy relational structures can be used for pattern recognition/classification tasks, where the pairs of input and output data $\boldsymbol{x}_k \in [0, 1]^n$, $y_k \in [0, 1]$, $k = 1, 2, \ldots, N$, are used to help build a fuzzy relational model, which is a logic mapping between the information granules specified in the input and output space. The dependency among the information granules is captured by the logic operators described in Section 3.1.1. The key problem here is to estimate the fuzzy relation R, as defined by (3.1), in this case represented by a vector of weights $\boldsymbol{w} = [w_1\ w_2 \ldots w_n]$, $\boldsymbol{w} \in [0, 1]^n$, which similarly to R, represents the association between the input and output vectors.

The logic neuron [109], [119] is a processing unit that takes advantage of the clearly defined semantics within the logic expressions, and has the sufficient plasticity that allow the model to have learning abilities. Similarly as with fuzzy relations, a logic single-output neuron realizes a logic mapping from $[0, 1]^n$ to $[0, 1]$. These logic neurons are classified into two main classes; OR and AND neurons. An OR neuron realizes a logic aggregation of the input value x_j with its corresponding weight w_j to obtain an output y; this combination of logic procedures can be addressed as an s-t aggregation, where $\boldsymbol{w}$ serves as a calibration mechanism. The modification of the values of these weights would make the aggregation of the input values generate the desired output. Since we want to estimate the weights that will produce an output y_ι, we insert this vector into the system of equations in (3.2); hence the following equation is formed:

$$y_\iota = s_{j=1,2,\ldots,n}\left(x_j \text{ t } w_j\right),\ \iota = 1,2,\ldots,\eta \tag{4.1}$$

For the AND neuron, the logic aggregations are used in an opposite direction (t-s aggregation), again by replacing the relational matrix R in (3.3) with the vector $\boldsymbol{w}$, we obtain:

$$y_\iota = t_{j=1,2,\ldots,n}\left(x_j \text{ s } w_j\right),\ \iota = 1,2,\ldots,\eta \tag{4.2}$$

The characteristics of the OR and AND neurons make them come with potential plasticity, and if these are used to help construct learning models, these characteristics become very beneficial [109].

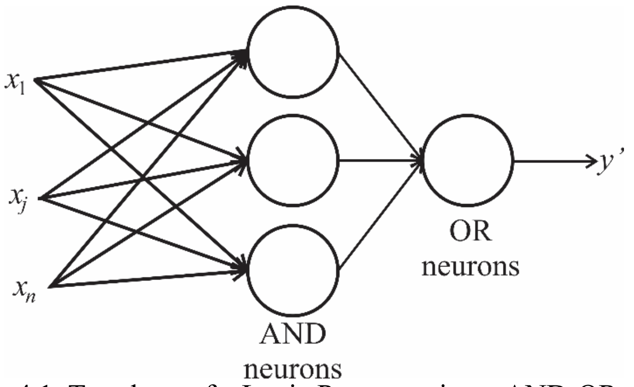


Figure 4.1. Topology of a Logic Processor in an AND-OR mode.

A Fuzzy Logic Processor (FLP) combines several logic neurons into its structure. The architecture of this processor comes from the Shannon theorem of decomposition of Boolean functions [123]. The essence of which is that, a representation scheme can come in two different forms, always involving a two-layer network. Here, the first layer consists of AND neurons with outputs being aggregated by using a single OR neuron. The other form is just the converse topology, where the outputs of a layer of OR gates is aggregated by a single AND neuron. Having this theorem in mind the topology of the fuzzy logic processor is formed, as illustrated in Figure 4.1.

Based on this discussion, a model resembling a traditional artificial neural network (ANN) structure is obtained. This results in a network with an input layer composed of AND gates, and an output layer composed of OR gates, when an AND-OR mode is applied. It has to be taken into account that logic neurons are highly nonlinear processing units, and the detailed nonlinear mappings are dependent of the specific values of the logic connectives (weights), and the selection of the *t-norms* and *s-norms*.

The FLP can be optimized by using GB learning [109], where the input data $\boldsymbol{x}_k$ is passed through the FLP to obtain an output y_ι'. This output is compared with the target output y_ι and an error is measured. The error is back-propagated into the network and the weights (relations) connecting the logic neurons are updated. This procedure is repeated until the error is reduced to a desired value, or until a number of iterations have not exceeded a certain predetermined value.

The gradient-based method works in a similar fashion that the one employed by ANNs, but since the multiplication and summation operations are replaced by fuzzy logic operators, the first derivative of these operators is used to calculate the error and perform the required adjustments to the network. The detailed derivations can be found in [119].

An alternative option to construct fuzzy relational models is to represent the vector of weights of the FLP as a particle or an individual for the PSO or DE algorithms from Sections 3.6 and 3.7 respectively, and use the distance between $\boldsymbol{y}'$ and $\boldsymbol{y}$ as a fitness function to be minimized. The size of the individuals for these optimization mechanism depends on the number neurons in the hidden layer (AND neurons). If the end-user selects h neurons, then the size of the individual is $(n * h) + h$, to represent each weight inside the structure of the FLP (refer to Figure 4.1). Moreover, one FLP is trained independently for each output y_ι, $\iota = 1, 2, \ldots, \eta$.

4.2.1. Experimental Studies

To construct and train the FLP, two datasets found in publicly-available repositories [10] are used. These are the miles per gallon (MPG) and Auto-price datasets. The MPG dataset consists of 398 instances with eight attributes, and the mpg attribute treated as an output variable. The Auto-price dataset has several attributes that could be used as an output value. This dataset consists of 159 instances, with 16 continuous values and the price attribute, treated here as an output variable. Both datasets are separated into a training set (80%) and testing set (20%). We have to highlight that all experiments were performed with the same data separation, meaning that the training data never changes; in order to maintain a consistency in all experiments.

When clustering the input data, the FCM algorithm is used, and based on the results produced by several initial experiments, it was decided to use $c = 5$ to transform the input data into fuzzy sets, the fuzzification factor m varies for each dataset. The output variable was processed by using two overlapping fuzzy sets (1/2 overlap) distributed over the entire data space, where $y_{\min}$ and $y_{\max}$ are used as modal values describing the lower and higher values of the signal. Notice that processing the output data this way, there is zero degranulation (reconstruction) error.

To build the FLP, we decided to use nine AND neurons in the hidden layer, as this number of neurons seemed to perform better in most of the initial explorative experiments. With these parameters, and two vectors forming the output signal, two FLPs are constructed with five input gates, nine hidden AND neurons, and one OR neuron. For all the learning algorithms, the number of iterations or generations were set to 500, and in the case of PSO and DE, 50 particles or individuals were used. It was observed that the performance index did not improve after around 500 iterations on all three algorithms. The number of particles or individuals was decided because a larger population can cause the search to degrade, as mentioned in Section 3.7. However the size of the population is always problem-dependent [42]. The performance index is computed using a root mean squared error (RMSE), expressed as follows;

$$Q = \sqrt{\frac{1}{N\eta}\sum_{k=1}^{N}\sum_{\iota=1}^{\eta}\left(y_{k\iota} - y'_{k\iota}\right)^2} \tag{4.3}$$

where the number of fuzzy sets in the output space is set to 2 ($\eta = 2$) and $y_{k\iota}$, $k = 1, 2, \ldots, N$, $\iota = 1, 2, \ldots, \eta$ is ιth element of the kth output datum. The output of the model is denoted by $y'_{k\iota}$. To find the best value of the fuzzification coefficient for the input data, a value close to one and larger than two is used; $m = 1.1, 1.2, \ldots, 3.0$, when using (3.6) and (3.7) to cluster the date. As for the *t-norms* and *s-norms*, four different combinations of the logic operators shown in Table 3.1 are used, viz. *max-min*, *max-prod*, *sum-min*, and *sum-prod*, to observe which combination performs better. The training of the FLPs is done by using GB method, PSO, DE, and the combinations PSO-GB, and DE-GB. In this way we were able to offer a detailed comparative analysis.

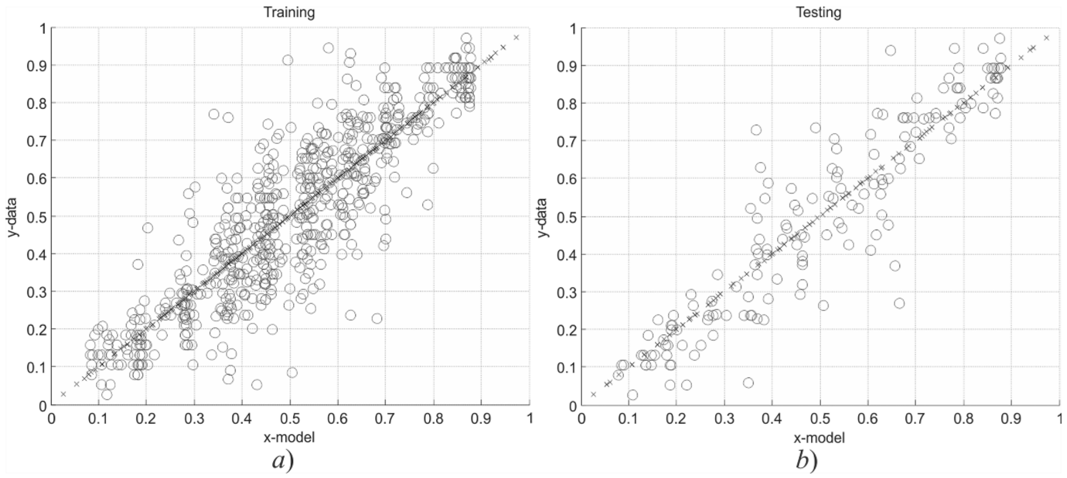


Figure 4.2. Best approximations obtained for the MPG for the (*a*) training and (*b*) testing datasets.

After the data was separated into training and testing sets, the FLPs were initialized for each value of the fuzzification coefficient. First the GB method was tested, with the different combinations of fuzzy logic operators. For the MPG dataset, the best result, computed by using (4.3), was achieved with the *sum-prod* logic operators, and m = 2.5, resulting in Q = 0.113 for the training data, and Q = 0.116 for the testing data.

Table 4.1. RMSE values obtained for learning scenarios and various combination operators for the training and testing datasets.

Dataset	MPG				
Method	GB	DE	PSO	DE-GB	PSO-GB
Logic Operators	*sum-prod*	*max-prod*	*max-prod*	*sum-prod*	*max-prod*
m	2.5	2.5	2.8	2.5	2.8
Train (Q)	0.113	0.111	0.114	0.111	0.107
Test (Q)	0.116	0.105	0.106	0.102	0.109

Next the PSO and DE algorithms were tested, again with all four combinations of the logic processors. The best result was achieved by DE with the *max-prod* combination and m = 2.5, for PSO the best result was found with m = 2.8; these results are displayed in Table 4.1.

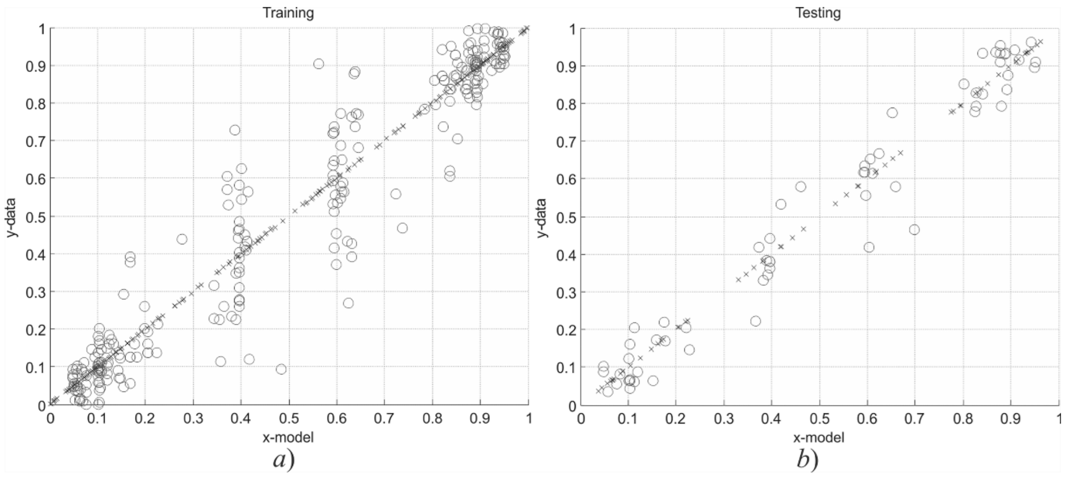


Figure 4.3. Best approximations for obtained the Auto-price for the (*a*) training and (*b*) testing sets.

When testing the hybrid algorithms, the result obtained with DE-GB was not improved. Nevertheless, the PSO-GB combination achieved the best approximation, since this hybridization minimized the error even further. Hence, the best approximation was achieved with a *max-prod* combination, m = 2.8, and Q = 0.107 for the training set, these results are also included in Table 4.1. Overall, the worst result was obtained by the PSO algorithm, with the *sum-min* combination of logic operators, and m = 1.6, obtaining Q = 0.313 and Q = 0.321 for the training and testing sets respectively. The best approximation, achieved by PSO-GB, is displayed in Figure 4.2. However, the “best” fuzzification coefficients and learning algorithms can be different for other datasets, as demonstrated in the following experiment.

For the Auto-price dataset, the GB method reported the best results when the fuzzification coefficient was m = 1.8 with Q = 0.099, and a *max-prod* combination. The DE algorithm achieved a result of Q = 0.094, with m = 1.6, and a *sum-prod* combination. And lastly PSO resulted in Q = 0.096, with the *sum-prod* combination, and m = 1.8. Overall, the best approximation was found with DE-GB and an *s-t* combination of *sum-prod*, with m = 1.6, obtaining Q = 0.085, for the training set, and Q = 0.066 for the testing set. For both experiments, GB helped improve the results, after the group-based algorithms were finished. The worst result was obtained with the PSO algorithm, which achieved a result of Q = 0.209 for the training set and Q = 0.202 for the testing set, with a *max-prod* combination, and m = 2.6. The best results for all the experiments are included in Table 4.2, and Figure 4.3 shows the best overall result. This proves that the selection of the initial parameters is always problem-dependent.

Table 4.2. RMSE values obtained for learning scenarios and various combination operators.

Dataset	Price				
Method	GB	DE	PSO	DE-GB	PSO-GB
Logic Operators	*max-prod*	*sum-prod*	*sum-prod*	*sum-prod*	*max-prod*
m	1.8	1.6	1.8	1.6	1.8
Train (Q)	0.099	0.094	0.096	0.085	0.094
Test (Q)	0.104	0.078	0.088	0.066	0.076

Now that the best FLP model is found, we proceed to the construction of its granular counterpart. To form it, the optimized weights are transform to granular weights by using the protocols described in Section 3.3.1.

4.3. Granular Fuzzy Relational Models

Granular computing becomes of great importance when designing and building granular models, based on diverse information provided by complex structures and topologies in the data. The interest is to construct a meaningful information granule, based on experimental evidence, to efficiently encompass this diverse information. It is clear that each source of knowledge is diverse, and this variety has to be considered and carefully quantified. Obviously, the resulting universal model is going to be more general and abstract, therefore the allocation of information granularity is viewed as an important design asset, and as such, it is subject to optimization.

4.3.1. Experimental Studies

To construct the granular fuzzy relational models, the FLPs that achieved the best approximations from the previous section are used. Since there are two layers in each FLP, there are two relations constructed, each having a size of $n \times h$, and $h \times 1$, which correspond to n inputs, h hidden neurons, and one output neuron. For each relation, the weights (relations) are transformed into granular weights by using each protocol from Section 3.3.1, and the performance is determined by measuring the overall coverage with Equation (3.14). Moreover, the 80% of the data used to train the FLP is used here to allocate the granularity in the data. The optimization algorithms are only applied to protocols 3 to 5, or Equations (3.18) to (3.20), recalling that the first two protocols, (3.16) and (3.17), do not require any optimization mechanisms, and protocol 6 is only used as a point of reference.

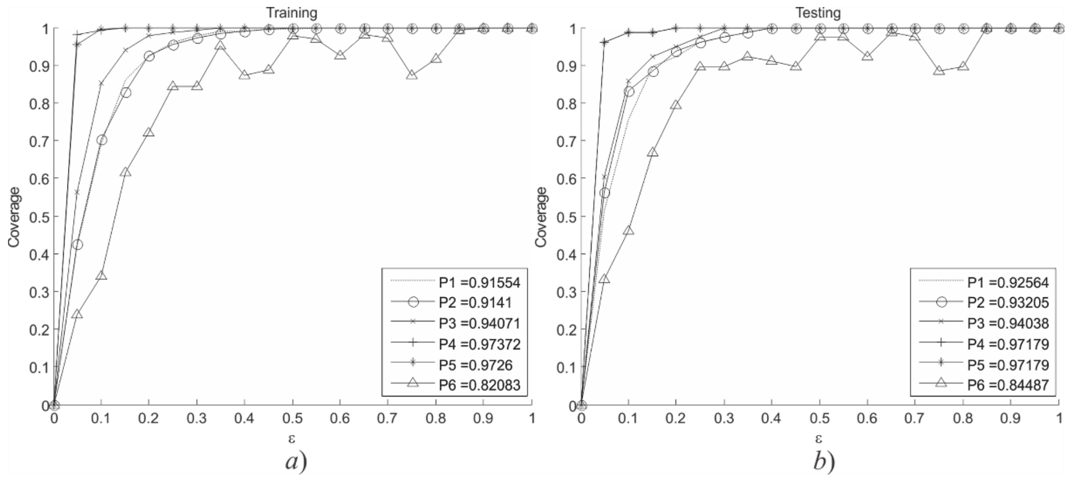


Figure 4.4. Increasing values of ε vs coverage for protocols 1 to 6 by using the MPG (*a*) training and (*b*) testing sets.

To allocate the granularity the DE algorithm is employed, given that this algorithm yielded the best overall results. For the DE algorithm, 60 individuals were used and ran for 800 generations, as for the level of granularity ε the value is increased from 0.05 to 1 with a step size of 0.05. At the end of the optimization, the remaining 20% of the data is used to review the performance (coverage) of the final granular fuzzy relation. The overall performance of the final granular model is determined by measuring the AUC.

The highest coverage achieved by all protocols is obviously when $\varepsilon = 1$, since the intervals produced by ε almost cover the entire output space. Nevertheless, a full data coverage was observed for small values of ε. For instance, when the MPG dataset was employed, full coverage was observed when $\varepsilon = 0.15$ for protocols 4 and 5 with the training set. For the testing set full coverage was achieved with $\varepsilon = 0.2$. On the other hand, for the auto-price dataset, the full coverage was observed when $\varepsilon = 0.25$ and $\varepsilon = 0.2$, for protocols 4 and 5 respectively, but for the testing set the full coverage was achieved when $\varepsilon = 0.05$. This last behavior is obvious when looking at the best approximation for the FLP model displayed in Table 4.2 (GB-DE), where the performance was higher for the testing set. When protocol 6 was tested, the AUC is lower, this demonstrates how the optimization of the allocation of granularity performs better when the information granules are carefully constructed and designed. The detailed results are summarized in Table 4.3.

Table 4.3.Area under the curve for each protocol and for the training and testing data.

	MPG					
Datasets	P1	P2	P3	P4	P5	P6
Training	0.916	0.914	0.941	0.974	0.974	0.821
Testing	0.926	0.932	0.940	0.972	0.972	0.845
	Auto-price					
Training	0.918	0.919	0.945	0.969	0.971	0.843
Testing	0.946	0.946	0.962	0.975	0.975	0.886

The AUC helps asses the performance of the original fuzzy model. Large AUC values mean that the data was almost entirely covered with small values of ε, while smaller AUC values means the opposite. If we observe Figures 4.4 and 4.5, for protocols 1 to 5, more than 90% of the data was covered when $\varepsilon = 0.2$, but for Protocols 4 and 5 this behavior was observed with $\varepsilon = 0.05$. This means that the initial fuzzy model has a good approximation, and that the model can be generalized by using very specific (narrow) information granules. Also, it is noticeable how each protocol outperforms the previous one, except for protocol 6, which demonstrates that the systematic strategy of granularity allocation exhibits superiority over the random allocation of granularity.

4.4. Summary

In this chapter a granular fuzzy relational model has been designed and optimized, where the structure is formed based on a *numerical* fuzzy relational model. It was demonstrated that building the fuzzy relational model is not an easy task, and the best algorithm used to construct them is problem-dependent. In this study, when dealing with hybrid algorithms, two scenarios were observed; *i*) The DE and PSO algorithms found a set of optimal weights (relations), and the GB method performed a local search and improved these weights; *ii*) The DE or PSO algorithms found the best global weights, and the GB method did not improve the learning. However, it was observed that the hybrid algorithms exhibited the best performance, for both training and testing sets. It was also demonstrated that different datasets would require a new set of experiments to find the best learning algorithm, logic operators, and fuzzification coefficients. It is also important to consider the suitable number of hidden neurons and the amount of cluster centers to transform the input data. Moreover, we transformed the optimized fuzzy model into a granular fuzzy model. The development of these granular fuzzy relational structures required an optimization of the allocation of information granularity by means of DE, which helped convert the original model into a more abstract and general model. For all the experiments the protocol 5 from Section 3.3.1 achieved the best overall results, due to its parametric flexibility. These granular constructs are viewed as a sound alternative for system modeling, and are considered an important design asset, which can help us asses the performance of the original numerical model, and also provide a sound solution to the system of relational equations.

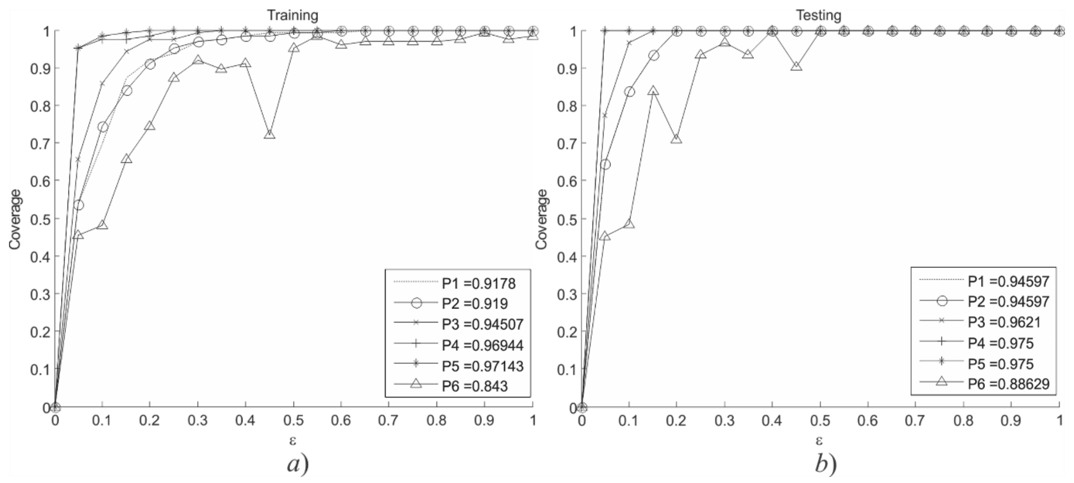


Figure 4.5. Increasing values of ε vs coverage for protocols 1 to 6 by using the auto-price (*a*) training and (*b*) testing sets.

5. Granular Fuzzy Models

In this Chapter we study the design of different granular fuzzy models. The design of these models differs to the previous approach, where a fuzzy model is used as a guide to construct a more abstract (global) model. For these models we exploit a concept of information granularity by developing a model coming as a network of intuitively structured collection of interval information granules (described in the output space), and a family of induced information granules (in the form of fuzzy sets) formed in the input space. In contrast to most fuzzy models encountered in the literature, the results produced by granular models are information granules rather than plain numeric entities. The design of the model concentrates on a construction of information granules that form a backbone of the overall construct. Interval information granules positioned in the output space are built by considering intervals of equal length, equal probability, and by developing an optimized version of the intervals, as described in Section 3.4.1. The induced fuzzy information granules localized in the input space are realized by running a so-called context-based FCM algorithm (Section 3.4.2). The performance of the model is assessed by considering criteria of coverage and information specificity (information granularity). Further optimization of the model is proposed along the line of an optimal re-distribution of input information granules induced by the individual interval information granules located in the output space. Experimental results involve some synthetic low-dimensional data and publicly available benchmark data sets.

5.1. Problem Formulation

When dealing with real-world dynamic systems, we have to be aware that these exhibit a certain amount of diversity, meaning that similar input signals can spawn several different outcomes. This diversity has to be taken into consideration and carefully quantified. In this sense, granular models fully acknowledge a notion of variable granularity whose range could cover detailed numeric entities and very abstract (general) information granules. Until now, there are no ideal *numeric* models which can capture the data without any modeling error, meaning that the numerical output of a certain model is usually not equal to the output data for all inputs forming the training data [114]. Therefore we concentrate on building a model where the outputs are not plain numeric entities, but comes as interval-based information granules.

Throughout this study we demonstrate that in granular modeling, information granules play an important and multifaceted role; *i*) the model is built as a network of associations among information granules. This supports interpretability of the model, which becomes easily translated into a collection of rules with condition and conclusion parts being formed by the constructed information granules; *ii*) Information granules form conceptual sound building blocks (supported by data) and in light of their functionality, can be used in the formation of a variety of relationships among input and output variables; and *iii*) Information granules serve as sound descriptors of data. Each information granule comes with its own well-defined semantics and as such the granules can be associated with a certain linguistically sound meaning. In this way one can produce a general sound view at the data. To construct the granular fuzzy model, the ensuing Differential Evolution (DE) [130], [149], [150] is invoked to serve as a key optimization vehicle. A number of vital design issues concerning parameters of the information granules are also raised and discussed.

5.2. General Architecture of the Model and its Underlying Processing

In comparison with the studies on linguistic models reported in [51], [74], [75], [120], [121], the model proposed in this study is inherently structured around information granules and form an architectural perspective as it arises as a network of connected information granules. The general topology of the model is shown in Figure 5.1; it might be helpful to also refer to [126]. It becomes beneficial to analyze this figure in more detail to gain a better view at the topology and the ensuing design of the model.

As visualized there, we encounter a collection of connected information granules. Moreover, there is a collection of information granules in the form of intervals $D_1, D_2, \ldots, D_p$ positioned in the output space. Let $D_h = \left[d_h^-, d_h^+\right]$, $h = 1, 2, \ldots, p$ denote the lower and upper bounds of the hth interval, while p is the number of intervals. These intervals form a partition of the output space (meaning that those intervals are pairwise disjoint and cover the entire output space). Each D_h is projected onto the input space and reflected there through a collection of information granules (fuzzy sets) in $\mathbb{R}^n$ located at the first layer of the structure of the model. In general, for each D_h, there could be a certain number of clusters formed, say $c_1, c_2, \ldots,$ and c_p, respectively.

There are two essential key features of the model. First, we build a collection of information granules (both in the input and output space) and those in the input space are directly induced by the output information granules (see Figure 5.2). Second, the results produced by the model are *granular* rather than *numeric* ones. This becomes apparent by looking at the computing realized by the model.

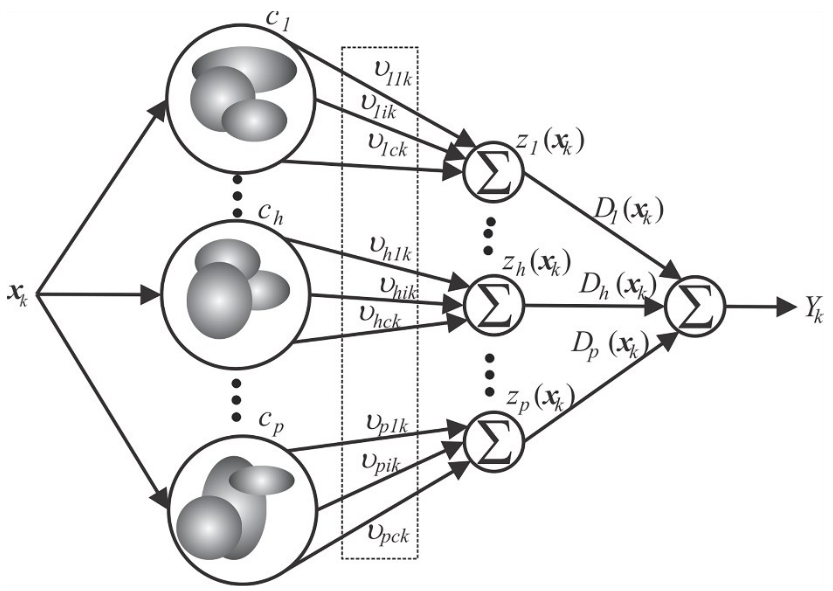


Figure 5.1. General architecture of the granular fuzzy model, where a collection of induced information granules c_h is formed by engaging a context-based clustering mechanism, giving rise to several membership degrees υ_{hik}, which are summed at each context h to trigger the corresponding activation level $z_h(\boldsymbol{x}_k)$.

The induced input space is clustered by means of the context-based FCM algorithm described in Section 3.4.2. Any numeric input datum $\boldsymbol{x}_k$, $k = 1, 2, \ldots, N$ "activates" the input information granules and produces the corresponding degrees of membership. Those being produced by the fuzzy sets implied by the same output interval are then aggregated (summed), thus giving rise to the overall activation level $z_h(\boldsymbol{x}_k)$ for the hth interval. To be more specific, the elements of the partition matrix that correspond to each interval are summed as $z_h(\boldsymbol{x}_k) = \sum_{i=1}^{c_h} \upsilon_{hik}$. Since in the traditional FCM algorithm from Section 3.2.1 the partition matrix has to satisfy the condition $\sum_{i=1}^{c} \upsilon_{ik} = 1$, then in this case the next condition has to be satisfied,

$$\sum_{h=1}^{p} z_h(\boldsymbol{x}_k) = 1 \tag{5.1}$$

Each interval impacts the performance of the clustering mechanism; if $z_h > z_{h+1}$ then the data placed under D_h produces a partition matrix whose result will have a higher membership degree in its corresponding node, henceforth it produces an interval that is formed around its corresponding numeric output value. In the sequel, the summation of the partial results is performed as follows;

$$Y_k = \sum_{h=1}^{p} \left(z_h(\boldsymbol{x}_k) \otimes D_h(\boldsymbol{x}_k) \right) \tag{5.2}$$

where $z_h(\boldsymbol{x}_k)$ and $D_h(\boldsymbol{x}_k)$ indicate that these variables are dependent of the input datum $\boldsymbol{x}_k$. We denote the algebraic operation by $\otimes$ to emphasize that the underlying computing operates on a collection of intervals, e.g. $a \otimes D = [a * d^-, a * d^+]$, note that the symbol $*$ is used as the "conventional" multiplication. Following calculations known in interval mathematics [69], the final result is an interval with bounds determined on a basis of the bounds of the individual intervals and their associated activation levels, namely

$$Y_k = [y_k^-, y_k^+] = \sum_{h=1}^{p} \left(z_h(\boldsymbol{x}_k) * [d_h^-(\boldsymbol{x}_k), d_h^+(\boldsymbol{x}_k)] \right)$$
$$y_k^- = \left(z_1(\boldsymbol{x}_k) * d_1^-(\boldsymbol{x}_k)\right) + \left(z_2(\boldsymbol{x}_k) * d_2^-(\boldsymbol{x}_k)\right) + \ldots + \left(z_p(\boldsymbol{x}_k) * d_p^-(\boldsymbol{x}_k)\right)$$
$$y_k^+ = \left(z_1(\boldsymbol{x}_k) * d_1^+(\boldsymbol{x}_k)\right) + \left(z_2(\boldsymbol{x}_k) * d_2^+(\boldsymbol{x}_k)\right) + \ldots + \left(z_p(\boldsymbol{x}_k) * d_p^+(\boldsymbol{x}_k)\right)$$

As usual in system modeling, a granular fuzzy model is constructed on a basis of certain data which come in the form of a collection of input-output data pairs $(\boldsymbol{x}_k, y_k)$, where $\boldsymbol{x}_k \in \mathbb{R}^n$ and $y_k \in \mathbb{R}$. The formation of information granules in the output space is guided by some optimization criterion. Once these have been constructed, these induce a family of information granules in the input space.

Considering the architecture of the granular model portrayed in Figure 5.1, it can be translated into a series of rules in a straightforward fashion as the condition and conclusion components are shown explicitly in the structure. The clusters (described by their prototypes) form the condition part of the rules while the intervals in the output space are located in the conclusion part of the rule. For the *h*th interval D_h, we have c clusters positioned in the input space and consequently a union of the input fuzzy sets, thus giving rise to the rule of the following format;

$$\text{if } A_{h1} \text{ or } A_{h2} \text{ or } \ldots \text{ or } A_{hc} \text{ then } D_h$$

where $A_{h1}, A_{h2}, \ldots, A_{hc}$ are the fuzzy sets in $\mathbb{R}^n$ with the membership functions computed on a basis of a set of prototypes $\{\boldsymbol{v}_{h1}, \boldsymbol{v}_{h2}, \ldots, \boldsymbol{v}_{hc}\}$ respectively, where $\boldsymbol{v}_{hi}$, $i = 1, 2, \ldots, c$ describes the coordinates of the ith cluster center of the hth context computed in the clustering stage, refer to Equations (3.21) and (3.22). In the sequel, if more detailed interpretability is required, the prototypes can be projected onto the individual input variables and then the condition part can be read as a Cartesian product of fuzzy sets defined in the individual input spaces, say $A_{hi} = A_{hi}(1) \times A_{hi}(2) \times \ldots \times A_{hi}(n)$, that reads as

$$A_{hi}(1) \text{ and } A_{hi}(2) \text{ and } \ldots \text{ and } A_{hi}(n)$$

As portrayed in Figure 5.2(a), a two-dimensional synthetic dataset consisting of 1,000 points around five points is randomly generated by using a normal distribution in the open interval (-1, 1). This leads to five clouds of overlapping data. A symmetric function is then used to calculate a target output value described as $y = x_1 * x_2 * \exp\left(-x_1^2 - x_2^2\right)$. Furthermore, the output space is divided into four intervals D_1, D_2, D_3, and, D_4, with a different size and position of the cutoff points, refer to Figures 5.2(b) and 5.2(c), in order to construct five clusters in each induced input space.

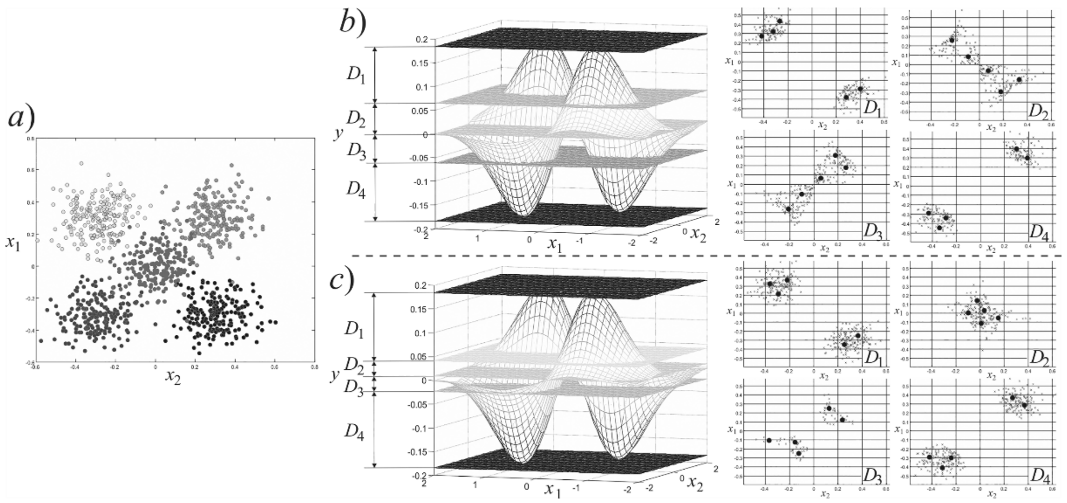


Figure 5.2. Output information granules and their influence on the clusters induced in the input space. The intervals placed in the output space (b) exhibit different widths than the intervals in (c). Five clusters are considered for each interval h and the prototypes are represented by black dots.

As depicted there, these clusters have a direct influence on the data and on the distribution of the prototypes (black dots) induced in the input space. Henceforth, the formation of the output information granules is crucial and several strategies of their development are investigated. The intervals can be designed as equal size intervals, equal probability (EP) intervals, or optimized intervals; these are fully described in Section 3.4.1. The input information granules are constructed for the individual output information granules and those are built by using an augmented version of Fuzzy C-Means (FCM), known as a so-called conditional or context-based FCM (refer to Section 3.4.2).

5.2.1. Performance Evaluation: Coverage and Specificity Criteria

To evaluate the granular fuzzy model, two criteria are taken into account: coverage and specificity. Here the dominant index is concerned with the coverage of the target data, and this coverage can be affected by a number of different design parameters. These include the number of intervals, the number of clusters in the input space, and the fuzzification coefficient. The selection of these initial values is problem dependent, and finding them is a matter of further optimization.

The key objective studied here is the coverage criterion expressed as follows;

$$f_1 = \frac{1}{N}\sum_{k=1}^{N}\left(y_k \in Y_k\right) \tag{5.3}$$

where the belongingness predicate returns 1 if y_k belongs to Y_k. For the specificity criterion, the length of the output intervals is taken into consideration. There is a compelling reason considering it: if the length is broad then the specificity of the information granule is low, and if the length of the interval decreases, the specificity starts to increase. The measure of specificity is computed as the following sum

$$f_2 = \frac{1}{N}\sum_{k=1}^{N}\left(y_k^{+} - y_k^{-}\right) \tag{5.4}$$

These coverage and specificity measures are in conflict as the number of elements covered by the information granule increases, the length of the interval might increase and cause a decrease in the specificity, thus becoming less detailed, refer to Section 3.3. In order to resolve this conflict and to calibrate the impact of the specificity criterion we consider the following form of f_2;

$$f_2 = e^{-\lambda(l/L)} \tag{5.5}$$

where λ is a non-negative parameter, l is the total average length of the output intervals, and L is the total range of the output variable (i.e. $L = y_{\max} - y_{\min}$). With the modification of the second objective, we can maximize the performance index as $Q = f_1 \times f_2$, but in this case we have to take into consideration the value of λ, given that a value close to zero can have little to no impact in the specificity criterion, whereas large values of λ could have an significant impact in the final coverage. Finding a suitable value of λ may call for another optimization problem.

5.2.2. Experimental Studies

To clearly illustrate the design process of the proposed granular model, we discuss the following examples. First the model is tested with synthetic data, then real-world numeric data coming from publicly-available repositories are used.

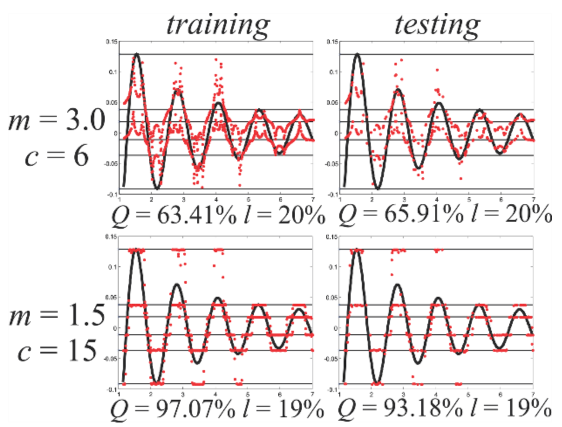


Figure 5.3. Worst and best results for the total coverage obtained for synthetic data, using $p = 5$ intervals, different fuzzification coefficients, and different number of clusters, the intervals are obtained by using EP.

a) Synthetic single-input single-output data

A one-dimensional synthetic dataset is generated by using the function $y = \sin(5x)/5x$. The input variable ranges from 1.15 to 7.00; in total 586 data points are generated lying within an interval ranging from -0.091 to 0.128. The data are randomly separated into 70% and 30% to train and test the granular model, and this separation never changes, to maintain consistency in the experiments. The output intervals are formed by using the EP criterion. In the first experiment $p = 5$ intervals were used. For each experiment, the induced data are clustered by using different combinations of the fuzzification coefficients $m \in \{1.3, 1.5, 2, 3\}$ and the number of clusters $c \in \{6, 9, 12, 15\}$. We only report the best and the worst results for all the combinations being tested.

In the reported experiments, the best result is obtained when m = 1.5 and c = 15, with 95.39% of the data covered and an average length of l = 19%, refer to (Figure 5.3). If the number of intervals is increased to p = 9 (Figure 5.4), the best coverage is obtained with the same parameters, and 93.59% of data gets covered, but in this case the average length is l = 11%, making the network of information granules more specific. There is a significant impact on the coverage with different values. Nevertheless, these results are bound to change when a different dataset is used. The final intervals or *envelopes* displayed in Figures 5.3 and 5.4 are represented as red dots. This because the synthetic data are previously separated, henceforth the results show the intervals of the training and testing data in two different plots, with the results of the interval covering only the training or test sets. If both images are to be superimposed, the total interval can be observed.

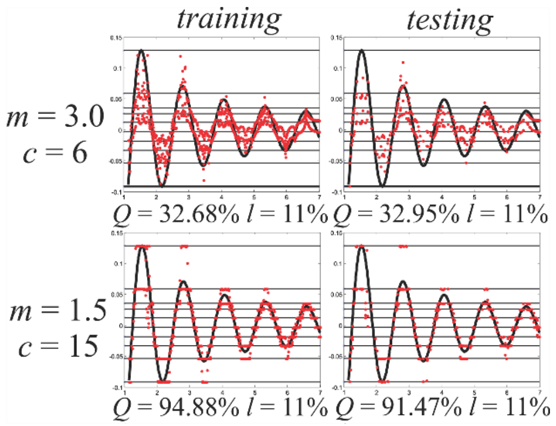


Figure 5.4. Worst and best results for the total coverage obtained for synthetic data and 9 intervals, the intervals are obtained by using EP.

b) Real-world data

For these experiments we used three publicly available datasets [8], [10]: Boston housing, delta elevators, and delta ailerons datasets. The Boston housing dataset contains prices of houses from suburbs in Boston. This dataset has 13 continuous values, including the target attribute which is the median value of owned-occupied homes in $1,000's with a range of 5 to 50, one binary-valued attribute, and a total of 506 instances [10]. The delta elevators and delta ailerons datasets are obtained from the task of controlling the elevators and the ailerons from a F16 aircraft but both datasets have different domains. The delta elevators dataset contains 9,517 examples, with six continuous values, including the target attribute with a range of -0.014 to 0.013, and one integer value. The delta ailerons dataset contains 7,129 instances, with six continuous values, which include the target value with a range of -0.0021 to 0.0022 [2].

Table 5.1. Worst and best overall results obtained for the Boston housing dataset and by using EP with selected values of the parameters.

			4 intervals		*5 intervals*	
			$c = 2$		$c = 2$	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	*Training*	44.07	0.28	35.31	0.21
		Testing	40.13	0.26	34.87	0.20
			2 intervals		*3 intervals*	
			$c = 10$		$c = 12$	
	2.0	*Training*	**90.96**	0.50	83.05	0.34
		Testing	**89.47**	0.50	82.24	0.34

For all datasets, the same procedure applied to the synthetic data is used here; the instances are randomly separated into the training and testing sets. Once the intervals have been formed with the EP criterion, the coverage of the granular model is evaluated. A number of experiments were completed to explore different outcomes of the model by testing with different values of the parameters, these included $p \in \{2, 3, 4, 5\}$, $m \in \{1.1, 1.3, 1.5, 2, 3\}$, and 2 to 12 clusters for each data set. We concentrate on the best and worst results, and also include results where both coverage and specificity show an acceptable performance.

Table 5.2. Worst and best overall results obtained for the delta elevators dataset and by using EP with selected values of the parameters.

			4 intervals		*5 intervals*	
			$c = 10$		$c = 9$	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	*Training*	41.93	0.25	35.90	0.22
		Testing	41.73	0.25	35.22	0.22
			2 intervals		*3 intervals*	
			$c = 5$		$c = 8$	
	3.0	*Training*	**98.23**	0.50	94.92	0.33
		Testing	**98.23**	0.50	94.90	0.33

By using the EP strategy and the Boston housing dataset (Table 5.1), the best result was found with a fuzzification factor of $m = 2.0$ and $c = 10$ clusters, and when the output space comes with two intervals, 90.51% of the data is covered, this percentage reflects all the data covered (e.g. training and testing sets combined). With three intervals, 82.81% of the data is covered, with the same fuzzification coefficient, but in this case c = 12. These results show how increasing the number of intervals decreased the coverage of the data, nevertheless the information granules are more specific. It is noticeable that the fuzzification coefficient plays an important role in the final output of the model. With more intervals, higher fuzzification coefficient, and more clusters the coverage does not increase or starts to decrease. This is a reason why we have only selected the results obtained with up to five intervals.

Table 5.3. Worst and best overall results obtained for the delta ailerons dataset and by using EP with selected values of the parameters.

			4 intervals		*5 intervals*	
			c = 4		c = 3	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	*Training*	60.62	0.26	53.33	0.25
		Testing	60.69	0.26	53.86	0.25
			2 intervals		*4 intervals*	
			c = 11		c = 10	
	2.0	*Training*	**99.78**	0.50	95.87	0.20
		Testing	**99.77**	0.50	94.81	0.20

For the delta elevators dataset, the best results are found when the output space is split into two intervals; with m = 3.0, c = 5 with 98.23% of the data set covered. On the other hand, with the three intervals and c = 8, we observe good coverage and narrower intervals, where 94.91% of the data is covered, see Table 5.2.

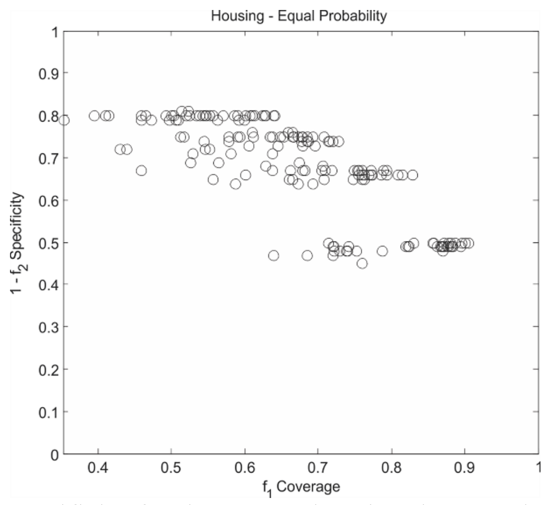


Figure 5.5. Coverage vs. specificity for the Boston housing dataset when using EP intervals.

When testing the model with the delta elevators dataset and by partitioning the output space into two intervals, 99.78% of the data is covered when $c = 11$, and by using four output intervals and $c = 10$ then 95.55% data is covered and the specificity increases; for both results $m = 2.0$ and these results are shown in Table 5.3. Notice that the average length of the intervals is reflective of the number of partitions tested in all datasets, which is owing to the data being separated into equal quantities when using the EP criterion, but only for the delta elevators dataset, the specificity increases with four intervals and becomes reflected in the produced narrower intervals.

Until now, the results show how the specificity decreases as the coverage increases. This behavior is illustrated in Figure 5.5. For this plot, the coverage and specificity are mapped by using f_1 and $1 - f_2$ from (5.3) and (5.4) respectively, so *specificity* values closer to 1 mean narrower information granules. Overall, by using the EP criterion, the coverage achieves a higher value but the specificity significantly decreases.

5.3. Further Refinements of the Model

So far the model has been built by practically *hardwiring* the existing information granules with the relationships among them being established in a very intuitive fashion, and pivot entirely upon some explicit point of view articulated by the designer. No refinements were considered for the previous design, thus there is still some room for improvement. In this section, we discuss two approaches that bring some algorithmic enhancements to the topology of the granular model. In the first approach we optimize the sizes of the output information granules, given a number of cutoff points, to maximize the coverage of the data. The second approach is concerned with a distribution of the overall number of information granules. Both of these approaches are optimized by using the DE algorithm from Section 3.7, whereas (5.3) is treated as the fitness function.

5.3.1. Optimization of Output Information Granules

While the EP criterion used in the original construction is sound to some extent, an improved strategy could be developed. Considering the same performance index in Equation (5.3), we optimize the coverage by adjusting the position of cutoff points. In other words, the optimization task is expressed as follows;

$$Q = \sum_{k=1}^{N} (y \in Y_k) \rightarrow \max \tag{5.6}$$

In case y_k falls within the Y_k interval, a value of 1 is returned (*True*). We use the count of the elements that fall within the information granule. The aim is to maximize the number of elements covered by the interval Y_k. For this task DE from Section 3.7 is applied to optimize the cutoff points of the output intervals.

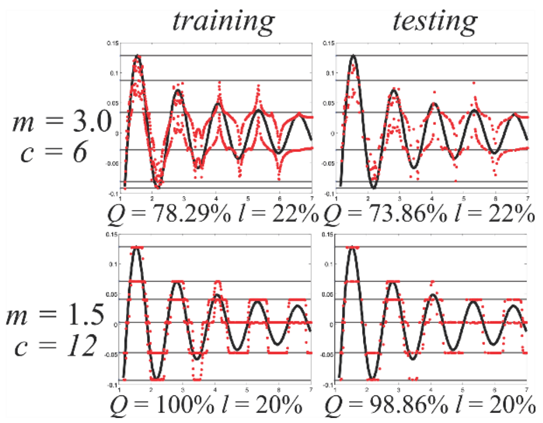


Figure 5.6. Worst and best results for the total coverage obtained for synthetic data, 5 intervals, different number of fuzzification coefficients, and different number of clusters using DE.

To initialize the algorithm, an initial population is prepared by randomly generating a set of individuals as vectors of random values in the range [0, 1], all with a size equal to the number of intervals (p) to represent the width of each interval. These individuals are later normalized into phenotypes that represent the length of the intervals in the output range, and consequently, the cutoff points that granulate (divide) the output space;

$$\boldsymbol{D} = [D_1, D_2, \ldots, D_p] = [d_1^-, d_1^+; d_2^-, d_2^+; \ldots; d_p^-, d_p^+] \tag{5.7}$$

Obviously $d_1^- = y_{\min}$ and $d_p^+ = y_{\max}$. In this study, a population of 30 individuals and 50 generations were used. We observed that the performance index did not improve after more generations and/or when increasing the population sizes over 30 individuals.

5.3.2. Experimental Studies

a) Synthetic data

To test the optimization algorithm, the synthetic data from the previous experiments is used along with the same combinations of number of clusters and fuzzification coefficients. Furthermore, for consistency and comparison purposes the data previously separated to train and test the granular model are the same as in the previous experiments.

Table 5.4 Worst and best results obtained for the Boston housing dataset when using DE with selected values of the parameters.

			4 intervals		*5 intervals*	
			c = 2		*c* = 2	
			Q	*l*	*Q*	*l*
Fuzzification coefficient (*m*)	1.1	*Training*	57.34	0.39	51.41	0.32
		Testing	53.29	0.39	46.71	0.31
			2 intervals		*4 intervals*	
			c = 5		*c* = 8	
	2.0	*Training*	**96.05**	0.73	84.18	0.31
		Testing	**94.74**	0.71	79.61	0.31

The initial coverage obtained by using the EP criterion is used as a reference point for the DE algorithm. By partitioning the output space into five intervals, the best coverage is observed when $m = 1.5$ and $c = 12$, where a total of 99.66% of the data set gets covered (Figure 5.6). It can be observed how both, the worst and best results, improve when compared to the results displayed in Figure 5.3, therefore the network of information granules is optimized, and the specificity slightly decreases as the coverage increases.

When the output space consists of 9 intervals, refer to Figure 5.7, the best result is obtained with the same fuzzification coefficient and number of cluster observed in the previous results, here 98.46% of the data is covered, and a small increase is observed in the specificity of the information granules, hence we have better coverage with narrower intervals.

b) Real-world data

When working with the real-world datasets, the overall results are increased, but the toll we have to pay is a decrease in the specificity of the information granules. For the Boston housing dataset (Table 5.4), if the output space is partitioned into two intervals, the best result is obtained with $m = 2.0$ and $c = 5$, where 95.65% of the data are covered, but the specificity substantially decreases. If the output space comes with four intervals, a sound result is obtained when $m = 2.0$ and $c = 8$, where 82.81% of the data is covered, which is the same coverage as with the previous experiment (Table 5.1) but with the improved specificity.

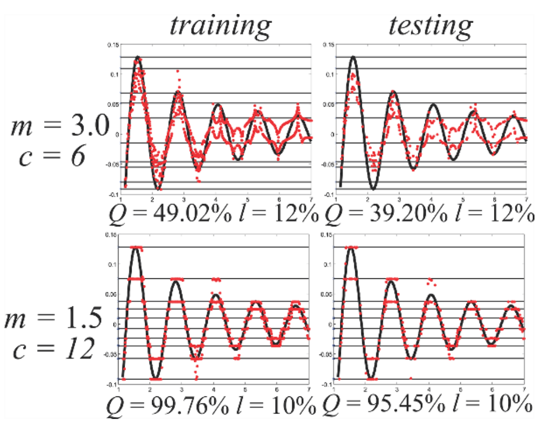


Figure 5.7. Worst and best results for the total coverage obtained for synthetic data, 9 intervals, different number of fuzzification coefficients, and different number of clusters using DE.

Table 5.5. Worst and best results obtained for the delta elevators dataset with the use of DE with selected values of the parameters.

			4 intervals		*5 intervals*	
			$c = 2$		$c = 2$	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	*Training*	56.54	0.35	46.10	0.28
		Testing	54.54	0.35	46.16	0.29
			2 intervals		*3 intervals*	
			$c = 8$		$c = 3$	
	3.0	*Training*	**99.61**	0.50	96.17	0.33
		Testing	**99.05**	0.50	96.57	0.33

With the delta elevators dataset (Table 5.5), and by using two partitions, the best coverage is obtained when $m = 3.0$ and $c = 8$, here a total of 99.51% of data gets covered. When the output space is divided into three intervals, an acceptable result is obtained with the same fuzzification coefficient and $c = 3$, where 96.29% of the data is covered, showing better coverage with the specificity not being affected, compared to the results displayed in Table 5.2.

Table 5.6. Worst and best results obtained for the delta ailerons dataset when using DE with selected values of the parameters.

			4 intervals		*5 intervals*	
			$c = 5$		$c = 5$	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	Training	77.33	0.24	69.54	0.18
		Testing	77.33	0.24	71.01	0.18
			2 intervals		*5 intervals*	
			$c = 11$		$c = 5$	
	2.0	Training	**99.86**	0.50	95.95	0.19
		Testing	**99.72**	0.50	94.90	0.19

By using the delta ailerons dataset and two partitions in the output space, the best result is obtained when m = 2.0 and c = 11, where 99.82% of the data is covered, refer to Table 5.6. If the output space is split to five intervals, 95.64% of the data gets covered when c = 5, and specificity of the information granules improves, compared with the results previously obtained with EP (Table 5.3). Notice how the worst results improve on coverage and specificity for this dataset.

With these results in place, it is observed how DE improves the coverage of the output data, but in some cases the specificity is sacrificed in the process. This behavior is depicted in Figure 5.8, where the coverage achieves its highest result with less specific intervals. Nonetheless, acceptable results can be found, where good data coverage is obtained with narrower information granules. A second approach is studied where the number of clusters is distributed across the input data, this might help reduce the impact that the coverage has on the specificity of the network of information granules.

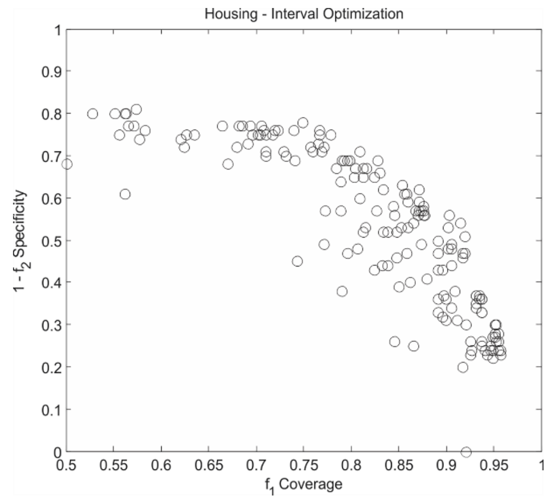


Figure 5.8. Coverage vs. specificity for the Boston housing dataset and when optimizing the intervals with DE.

5.4. Distribution of the Number of Clusters (Information Granules)

Different number of clusters behind each output interval could be beneficial to have a smaller impact on the specificity of the information granules. Here the allocation of information granularity is being carried by the input information granules, by distributing the number of clusters across the input data. We require a total number of information granules to be retained and the optimization process is again carried by DE.

The intervals are initially constructed by using the EP criterion, and the information granules are further developed in the input space to increase the coverage of the output data. To achieve this, we take into account the number of clusters c at each node and the number of partitions p. It is obvious up to this point that the size of the final collection of prototypes has $c \times p$ clusters, so by taking this amount of prototypes as a reference, we use DE as an optimization mechanism to distribute the number of clusters across input nodes and take into account the following conditions; $\sum_{h=1}^{p} c_h = c \times p$ and $c_h \geq 2$, $h = 1, 2, \ldots, p$.

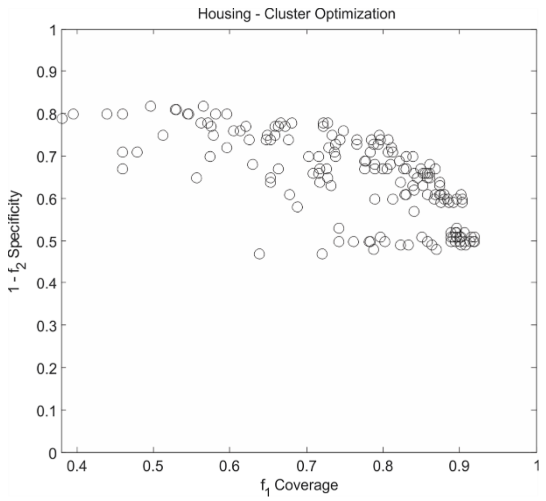


Figure 5.9. Coverage vs. specificity for the Boston housing dataset when re-arranging the number of clusters.

With the help of the DE algorithm, many different allocations of clusters are tested to enhance the performance of the information granules, and the final output of the model. For this version of the DE Algorithm, the size of the individuals is again equal to the number of contexts (p), and in the optimization process the genotypes, which initially assumes values in the range [0, 1], are normalized into a phenotype in the form $\mathrm{c} = [c_1, c_2, \ldots, c_h]$, which correspond to the number of clusters in each context. The objective of the DE algorithm is to find an individual that satisfies both conditions and increases the coverage of the target data by using (5.6). Hence these two circumstances are used as the fitness function in the optimization process. In these experiments, a population of 20 individuals and 50 generations are used; again a larger population size and/or number of generations do not improve the performance of the model.

5.4.1. Experimental Studies

a) Synthetic data

By using the synthetic data and five partitions, the best coverage is obtained when $m = 1.5$ and $c = 15$, where the total number of training and testing sets covered is 99.75% and 97.15% respectively, or 98.98% of the total data; the final distribution of cluster centers that are associated with the interval formation in the output space is 11, 12, 19, 14, and 19. The total average length of the output intervals is l = 19%; by comparing the results for the EP criterion in Section 5.3, we observe that the coverage increases while the specificity does not change. If the number of partitions is increased to nine, the best coverage is obtained with the same parameters as in the previous experiments. Here 99.76% of the training data and 96.59% of the testing data are covered or 98.17% of the total data. The distribution of the cluster centers in this case is 12, 14, 17, 17, 13, 13, 14, 18, and 17; the coverage is also increased in these experiments and the specificity does not change (l = 10%), compared to the specificity from the experiments in Section 5.4, where the output intervals are optimized with DE.

b) Real-world data

With the Boston housing dataset (Table 5.7), if the output space is partitioned into two intervals, the best coverage of the training and testing sets is obtained when $m = 2.0$ and $c = 10$, with 91.90% of the total data covered, the distribution of the cluster centers in this case is 11 and 9. It is noticeable that by using this method there is no direct impact on the specificity, since there is no substantial decrease of specificity as observed when the output intervals were optimized. If the output space is divided into five partitions, $m = 3.0$, and $c = 12$, then 83.79% of the data gets covered, and the distribution of cluster centers is 35, 5, 2, 2 and 19, here we observe a good coverage and better specificity than similar results shown in Tables 5.1 and 5.4.

Table 5.7. Worst and best results obtained for the Boston housing dataset using DE to distribute the number of clusters, with selected values of the parameters.

			4 intervals			*5 intervals*	
			$c = 2$			$c = 2$	
			Q	l		Q	l
Fuzzification coefficient (m)	1.1	Training	48.31	0.29		38.42	0.22
		Testing	40.13	0.27		36.84	0.20
			2 intervals			*5 intervals*	
			$c = 10$			$c = 12$	
	2.0	Training	**92.09**	0.50	3.0	85.59	0.30
		Testing	**91.45**	0.50		79.61	0.29

When the delta elevators dataset is considered (Table 5.8), with two partitions, the best result is obtained when m = 3.0 and c = 12, where 99.39% of the total data is covered and the specificity remains the same, the cluster centers in each partition are 11 and 13. If the output space is split into five intervals, a good coverage is obtained with the same fuzzification factor and c = 6, where 95.44% of the data is covered, the final distribution of cluster centers is 11, 2, 3, 2, and 12. In this case the specificity also remains the same but the coverage improves when compared to the results shown in Table 5.2.

Table 5.8. Worst and best results obtained for the delta elevators dataset with the use of DE to distribute the number of clusters, with selected values of the parameters.

			4 intervals		*5 intervals*	
			c = 2		c = 2	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	Training	45.92	0.30	40.45	0.24
		Testing	46.23	0.30	40.91	0.24
			2 intervals		*5 intervals*	
			c = 12		c = 6	
	3.0	Training	**99.49**	0.50	95.51	0.33
		Testing	**99.16**	0.50	95.27	0.33

Table 5.9. Worst and best results obtained with the delta ailerons dataset with the use of DE to distribute the number of clusters, with selected values of the parameters.

			4 intervals		*5 intervals*	
			c = 9		c = 4	
			Q	l	Q	l
Fuzzification coefficient (m)	1.1	Training	68.08	0.30	61.04	0.23
		Testing	66.81	0.29	60.54	0.23
			2 intervals		*4 intervals*	
			c = 10		c = 5	
	3.0	Training	**99.80**	0.50	96.63	0.20
		Testing	**99.77**	0.50	95.65	0.20

The last dataset tested is the delta ailerons dataset (Table 5.9), where the best coverage obtained when the output space has two partitions, m = 3.0 and c = 10 is 99.79%, the distribution of the clusters is 8 and 12. And if four intervals are used, the total coverage is 96.34% when c = 5, the distribution of the clusters in each interval is 10, 2, 2, and 6; notice how the specificity is not affected while the coverage increases, when comparing with the similar results shown in Table 5.3.

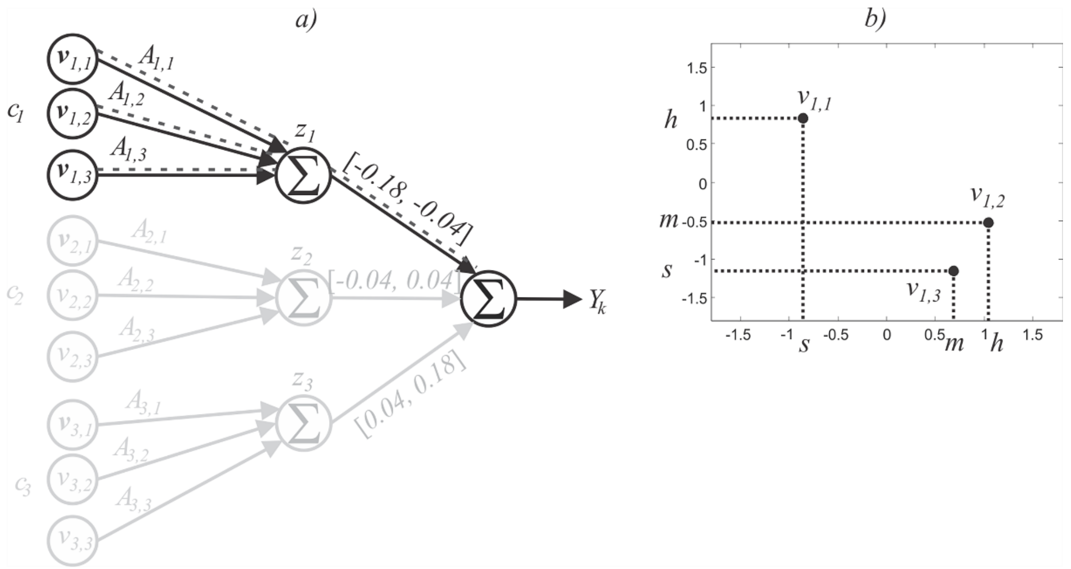


Figure 5.10. Optimized granular model with (*a*) its interpretation in the form of "if-then" statements and (*b*) the projection of the prototypes to their individual input space, where *s*, *m*, and *h* are *small*, *medium*, and *high* labels of the prototypes.

Overall the coverage increases, when compared to all the results obtained with the EP criterion. In these experiments an improvement is observed on the specificity in some cases, while in others there is no change, this is illustrated in Figure 5.9, where the mapping of the results show a similar behavior as to those observed in Figure 5.5, but upon closer inspection, it can be seen how the coverage increases and the specificity is not affected. This is different to the results depicted in Figure 5.8, where the specificity significantly decreases, which signifies that the re-arrangement of the clusters also helps in the optimization of the granular model, and does not have a negative impact on the size of the information granules.

Table 5.10. Impact of different incremental values of λ on the specificity and coverage of the output data, where *lgt, tr*, and *te* are, length, training set and testing set, respectively.

λ	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90	1.00	1.10	1.20	1.30
lgt	0.72	0.52	0.44	0.37	0.40	0.36	0.35	0.35	0.36	0.36	0.35	0.35	0.33	0.33
tr	96.05	95.81	90.11	88.98	89.55	88.14	87.57	88.42	88.42	88.70	87.85	87.57	60.73	58.76
te	94.74	92.84	88.26	76.97	78.29	76.97	75.00	76.97	75.66	76.32	75.00	73.68	52.63	51.32

Once the model has been optimized, it can be translated into a set of rules. As an example, let us take a two-dimensional dataset, with three intervals, $c = 3$, and $m = 1.5$. After optimization, the intervals resulted in $D_1 = [-0.18, -0.04]$, $D_2 = [-0.04, -0.04]$, and $D_3 = [0.04, 0.18]$. The extraction of the rules is straightforward, as shown in Figure 5.10(*a*). By using the clusters described by their prototypes to form the condition part, and the optimized intervals to form the conclusion, the rules will read as;

- if $A_{1,1}$ or $A_{1,2}$ or $A_{1,3}$ then $[$-0.18, -0.04$]$,

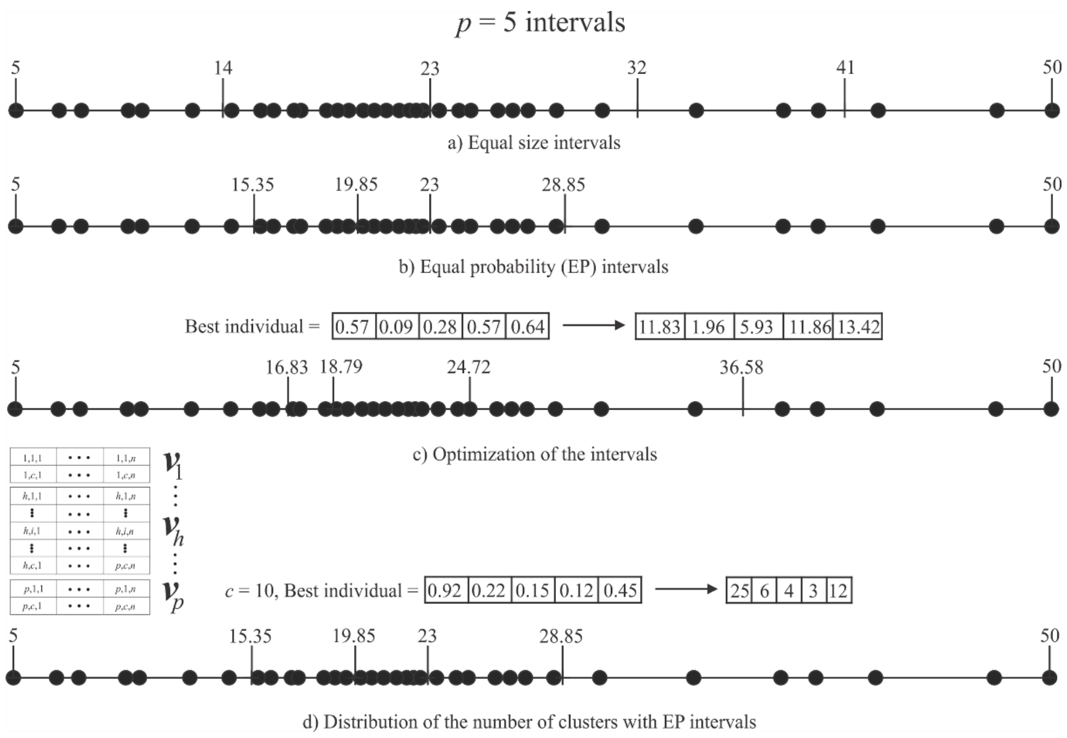
The projection of the prototypes can be seen in Figure 5.10(b), where each prototype is projected onto the individual input variables. In this case the condition part reads as

- $A_{1,1} = A_{1,1}(1) and\ A_{1,1}(2)$

To show the impact of different values of λ, and by considering the coverage and specificity criterions, we used the performance index $(Q = f_1 \times f_2)$ from (5.3) and (5.5) as the fitness function for the DE algorithm. In these experiments the Boston housing dataset is used, the output space is partitioned into three intervals and c = 5, m = 2.0, since these are the parameters where the best results were found; these are shown in Table 5.10. As it is observed, after λ = 0.2 the specificity increases but the coverage decreases. Here we can say that an optimal value of λ is around 0.2, since the coverage is still acceptable, and there is an increment in the specificity of the information granules.

5.5. Summary

In this chapter we studied several different techniques applied to the construction of the output information granules, we envisioned equal length intervals, but as shown in Figure 5.11(a), their size and position heavily depend on the distribution of the data. So it was decided to split the output data into equal amounts, or equal probability, refer to Figure 5.11(b), but this method required an almost manual construction of the information granules. Instead, we let DE fine-tune the intervals, as depicted in Figure 5.11(c). This method proved to achieve a better coverage, but in some cases a decrease in the specificity was observed. Alternatively, it was decided to optimize the number of clusters positioned in the induced data, and use the intervals constructed with the EP criterion, as illustrated in Figure 5.11(d). This method yielded good coverage, with little to no impact on the specificity. At the end, it was also shown that different factors affect the final output of the model, namely the amount of intervals and their size, the amount of induced clusters formed in the input space along with their distribution, and the value of the fuzzification factor. Henceforth, it is ideal to have an optimization algorithm which intuitively constructs the network of information granules, and improves the overall performance of the granular fuzzy model, this means finding a model that produces specific (narrow) information granules, and at the same time, covers most of the data.



p = 5 intervals
5
14
23
32
41
50
a) Equal size intervals
5
15.35
19.85
23
28.85
50
b) Equal probability (EP) intervals
Best individual = 0.57 0.09 0.28 0.57 0.64
11.83 1.96 5.93 11.86 13.42
5
16.83
18.79
24.72
36.58
50
c) Optimization of the intervals
v1
vh
vp
c = 10, Best individual = 0.92 0.22 0.15 0.12 0.45
25 6 4 3 12
5
15.35
19.85
23
28.85
50
d) Distribution of the number of clusters with EP intervals

Figure 5.11. Techniques applied to the construction of the output information granules.

6. Granular Fuzzy Modeling with Hyperboxes

In previous Chapters we constructed granular models by either forming granular fuzzy relations, or by associating information granules formed in the input and output space. In this chapter we look at a different type of information granules to cluster the input data. Clustering has been applied to numerous areas, including signal and image processing. Many approaches have been developed over the years to efficiently construct granular models on a basis of numerical experimental data. In this chapter, we propose a novel approach to construct a granular model that is fundamentally designed around information granules regarded as hyperboxes. Several studies have been focused on building a set of hyperboxes around data; one of them being a Min-Max Neural Network (NN) algorithm. Here we develop two different methods to construct these information granules, nevertheless some essential similarities to previous studies can be found. In particular, hyperboxes are constructed by using some reference data, and they are endowed with some parametric flexibility to facilitate controlling their size, whereas the construction of the hyperboxes involve elimination or reduction of possible overlaps between them. Experimental studies involve synthetic data and publicly available real-world data. The results are compared with the outcomes produced by the algorithm proposed by Simpson. The performance of the method is quantified and it is demonstrated that the obtained results are substantially better when dealing with multi-dimensional data.

6.1. Problem Formulation

Most studies, previously found in the literature, focus on the construction of hyperboxes by using as a reference the min-max NN algorithm described in Section 3.5; therefore many methods construct these information granules by following some essential steps. First, the hyperboxes are constrained to expand to a maximum size, this could result in a very complex model; in some cases, a large number of small hyperboxes could result when clustering multivariate scattered data, and some hyperboxes could end up covering only one data point. This could have a negative impact on the final number of hyperboxes, and the complexity of the model. Second, the hyperboxes are contracted to eliminate overlaps, which might result in loss of covered data. Some previous studies try to solve this problem by admitting some overlap among hyperboxes. For the granular model proposed in this Chapter, given a set of input and output data pairs, we construct interval-based information granules to partition the output space (viz. the space of the output variable). On a basis of these intervals, and similar to the previous Chapter, we carry out a so-called context-based Fuzzy C-Means algorithm to construct cluster centers (prototypes) in the multivariable input space. These prototypes now serve as hyperbox cores. To construct the information granules, two methods are studied: one develops a family of hyperboxes by realizing some constrictions, while the other one engages DE to realize further optimization. To reduce overlap, two methods are tested: one being previously proposed for the min-max NN and a new one, which engages DE to optimize the overlap reduction. The advantage of the proposed method is that the user can specify how many hyperboxes are to be constructed, and then analyze their contents by using several strategies. Moreover, the overlaps are not completely eliminated, but rather reduced to try to keep most of the data covered.

6.2. Hyperbox Construction and their Underlying Processing

To construct a set of hyperboxes around numerical data, usually there needs to be some kind of mechanism that restricts their maximum size. This may come as a control variable to help constraint the growth of the hyperboxes. This mechanism would help us not only to control their size, but also control how detailed or complex the model can be. In other words, many hyperboxes may describe the data better than a few large hyperboxes, but numerous small hyperboxes may erode the original interpretability of the data. In this sense, we need to decide on how large or how specific the hyperboxes should be to achieve the best abstraction of the dataset under examination.

The model construction starts with a collection of input and output data pairs (points in $\mathbb{R}^n$ and $\mathbb{R}$, respectively). The output data are first split into interval-based information granules. This partition is realized by using the equal probability (EP) criterion, described in Section 3.4.1. Once the output intervals are formed, the input data, associated with the output data in each interval, are then clustered by completing conditional FCM, the prototypes obtained at this stage are used to construct the hyperboxes around the data. In previous studies, the discovery of data was completed and quantified by using membership functions [145], [146], or a *compatibility measure* [13], but the construction of information granules in some cases rely on a statistical analysis of the data (e.g., the mean or median value) as a starting point for the hyperbox construction.

In this study, we use the information from a set of prototypes discovered in the conditional FCM stage, to find the hyperboxes cores and start to construct them from these points, this helps us build intervals around each prototype, and allows us to construct all the hyperboxes almost simultaneously. At the end the information provided by the hyperboxes can be easily translated into a set of rules to facilitate the interpretability of the data. By building the hyperboxes this way, we are forming direct associations among information granules, weather these come as interval-based or as prototypes. Once these building blocks are located, the hyperbox construction is initiated.

The optimal expansion and reduction of the hyperboxes is achieved by using a group-based algorithm, namely DE [130], [150]. Furthermore, we implement a three-step verification process to evaluate the data captured by the optimized hyperboxes, for this we consider; *i*) full inclusion of the data to a hyperbox, *ii*) smallest Euclidean distance between the data point and the prototypes, and *iii*) the membership degree to which the data falls within the hyperboxes, by using a modified version of equation (3.23). The numerical results are then compared with the outcomes produced by the min-max neural network (NN) algorithm from Section 3.5.

6.2.1. Detailed Description of the Granular Model

Recalling the granular fuzzy model from the previous chapter, the model proposed in the ensuing sections is inherently structured around information granules. It is formed by a collection of clusters coming in the form of intervals $D_1, D_2, \ldots, D_p$ which are positioned in the output space (output variable). Each D_h, $h = 1, 2, \ldots, p$ is then used to generate an induced structure into the input space. In general, for each D_h there exists a certain number of fuzzy sets constructed in the input space, say $c_1, c_2, \ldots,$ and c_p, respectively.

The induced input data is clustered by using the context-based FCM algorithm, described in Section 3.4.2. The centers of the fuzzy sets (prototypes) are then used as initial reference coordinates for the hyperbox construction. The information granules are constructed on the basis of certain experimental data, which come as a collection of input-output data pairs $(\boldsymbol{x}_k, y_k)$, $k = 1, 2, \ldots, N$, where $\boldsymbol{x}_\mathrm{k} \in \mathbb{R}^n$ and $y_k \in \mathbb{R}$. More specifically $\boldsymbol{x}_k \in [0, 1]^n$ and $y_k \in [0, 1]$, since the data are previously normalized to the unit hypercube.

Recalling the algorithm from Section 3.5, and to maintain conciseness of the presentation, the hyperboxes are defined in $[0, 1]^n$ and denoted by B. These can be fully described by its lower ($\boldsymbol{l}$)

or upper ($\boldsymbol{u}$) corners. However, since the hyperboxes are constructed in the induced input space, we have to consider that we have "$c*p$" hyperboxes, located in the n-dimensional input space. In this sense, a hyperbox is fully described by vectors of its lower and upper corners (coordinates) as $B_{hi} = \{\boldsymbol{l}_{hi}, \boldsymbol{u}_{hi}\}$, $h = 1, 2, \ldots, p$, $i = 1, 2, \ldots, c$, where $\boldsymbol{l}_{hi}$ and $\boldsymbol{u}_{hi}$ are vectors of the corresponding coordinates, or $\boldsymbol{l}_{hi} = [l_{1hi}, l_{2hi}, \ldots, l_{nhi}]$ and $\boldsymbol{u}_{hi} = [u_{1hi}, u_{2hi}, \ldots, u_{nhi}]$, in the hth partition and for the ith cluster.

After the hyperboxes are constructed, the input data included in each hyperbox are counted. If a datum is included in an overlap from other hyperboxes located in a different interval (context), then the datum is excluded from the count, but if the overlapping occurs in the same interval, then the datum is considered and counted. Once the data included in the hyperboxes have been counted, the algorithm proceeds with the contraction of the hyperboxes, to reduce or eliminate possible overlaps while retaining as much data being covered as possible. The formation of the information granules (hyperboxes) and the resolution of possible overlaps are guided by the DE algorithm.

Once the information granules have been formed and optimized, the granular model can be easily translated into a series of rules in a straightforward fashion. The input clusters, which are described by the hyperbox coordinates, form the condition part of the rules, while the intervals in the output space are located in the conclusion part of the rule. For the hth interval D_h in the output space, we have c_h hyperboxes B_{hi}, $i = 1, 2, \ldots, c$ located in the input space, which gives rise to the following rule;

$$\text{if } \boldsymbol{x}_k \text{ is included in } B_{hi} \text{ then } D_h$$

where $B_{hi} = [\boldsymbol{l}_{h1}, \boldsymbol{u}_{h1}], [\boldsymbol{l}_{h2}, \boldsymbol{u}_{h2}], \ldots, [\boldsymbol{l}_{hc}, \boldsymbol{u}_{hc}]$. This will tell us in which context the datum is located, and not only that, when examining the hyperboxes, we can obtain a detailed description of the data by analyzing their contents. In other words, by obtaining the standard deviation, mean, or any other statistical information from the data included in the each hyperbox, we can conduct a detailed assessment of the entities covered by the information granule.

6.2.2. Development of the Model

Information granules, in the form of intervals placed in the output space, are critical to the model's performance, since they subsequently imply a collection of information granules being formed in the input space. We decided to form the intervals by using the equal probability criterion (EP), as described in in Section 3.4.1. Recalling that each interval includes the same or similar amount of data, or *N/p* to be specific.

The information granules constructed in the input space and coming in the form of *n*-dimensional hyperboxes are determined on the basis of the intervals already formed in the output space. Here the main idea is to map the output interval to the corresponding regions of the input space. Once these regions have been set, the data are clustered but instead of the standard (viz. condition-free) clustering, we perform a so-called *conditional* FCM algorithm by following equations (3.21) and (3.22), from Section 3.4.2. As a result, the clustering reveals groups of data $\boldsymbol{x}_k$ in context D_h. Then the data that fall within each context are clustered, with this the clustering is conditioned by the output intervals. This procedure is beneficial to the model, since the relationships among the clusters in the input and output spaces are transparent and in this way we can easily form the associations of the granular model and in consequence effortlessly translate the emerging hyperboxes into a set of rules.

Fuzzy clustering is normally used to help group numeric data into a set of information granules, and these information granules form a granular *backbone* or *blueprint* of the granular model. It is obvious that the more clusters we decide to construct, the more detailed the resulting blueprint is. Deciding upon the number of clusters in each interval plays an important role in the final outcome of the model.

The prototypes obtained in each context *h*, e.g. $\boldsymbol{v}_{h1}$, $\boldsymbol{v}_{h2}$, …, $\boldsymbol{v}_{hc}$, are used as coordinates for the hyperbox cores, and two conditions have to be considered once these cluster centers are obtained. First, the hyperboxes are constructed around the prototypes, this means that the cluster center has to be included in the hyperbox. Second, when a hyperbox is contracted, it can be reduced to the coordinates of the prototype itself, hence the prototype remains included all the time, as stated by the first condition

6.2.3. Performance Evaluation: Dataset Coverage

To evaluate each hyperbox, we concentrate on the coverage of the data. To compute the coverage the number of input data included in one or more hyperboxes are counted, while verifying at the same time whether such hyperboxes belong to only one context, since we are finding prototypes and building hyperboxes in each partition of the input space. Furthermore, the coverage is directly affected by a number of different design parameters. These are the number of output intervals p and the number of input clusters c. Some other parameters might include the fuzzification coefficient, but since we are concentrated on obtaining only the prototypes (cluster centers) to be employed as cores for the hyperboxes, then we set this value to $m = 2.0$. As for the size of the output intervals, this can be sought out in future experiments. It seems obvious that the selection of the values of these parameters is always problem-dependent, and finding these values requires more attention and further optimization. The key performance index studied here is specified as the coverage criterion, expressed as follows,

$$f_{hi} = \frac{1}{N}\sum_{k=1}^{N}(\boldsymbol{x}_{hk} \in B_{hi})$$

where f_{hi} is the result of the number of data falling in the ith hyperbox for the hth interval. It is worth mentioning that an input value is considered *included* in a hyperbox, if and only if all of the numeric points that conform the input vector $\boldsymbol{x}_{hk}$ are included in the hyperbox, then the predicate can return 1 (true). This is shown as follows,

$$x_{1hk} \in B_{1hi} \text{ and } x_{2hk} \in B_{2hi} \text{ and } \ldots \text{ and } x_{nhk} \in B_{nhi} \tag{6.1}$$

Additionally, as mentioned before, it is important to verify if the input data are included in the hyperbox that belongs to only a single context. In cases where the data exhibits full inclusion to two or more hyperboxes that belong to the same context h, or more specifically,

$$\left(\boldsymbol{x}_{hk} \in B_{hi_1}\right) \text{ or } \left(\boldsymbol{x}_{hk} \in B_{hi_2}\right)$$

where $i_1 \neq i_2$, it can be considered and counted as covered, since in this case it belongs only to context D_h, but if the datum is included in two different hyperboxes from different contexts, or for example,

$$\left(\boldsymbol{x}_{h_1k} \in B_{h_1i_1}\right) \text{ and } \left(\boldsymbol{x}_{h_1k} \in B_{h_2i_2}\right)$$

where $h_1 \neq h_2$, then the datum becomes ambiguous and cannot be considered; it is not clear to which context it belongs to. In these cases, the hyperboxes might seem to cover most of the data, but if these are included in two or more hyperboxes from different intervals, the coverage of the data becomes irrelevant, and the number of data that are included in only one information granule (or more in the same interval) can be low. So it would be necessary to eliminate possible overlaps between hyperboxes that lie in different intervals.

In the situation where an input datum is located in two or more hyperboxes from only one interval and counted as *covered*, the next step will be to assign this input datum to only one hyperbox. To achieve this, we would have to measure the distance between the datum and the prototypes that generated the hyperboxes, or measure the membership degree to which the datum falls within each of the hyperboxes. By using this information we can assign the data point to only one hyperbox. To measure the membership degree, we used a *hypercube membership function* implemented by Simpson, which is thoroughly explained in Section 3.5, but instead of using the membership function to expand the hyperboxes [145], we employ it only to designate the data point to a single information granule in order to evaluate its contents.

6.3. Hyperbox Construction with a Numerical Constraint

For the first approach, the construction of the hyperboxes is realized as follows; the prototypes (cluster centers) are first ordered coordinate-wise, then for the prototype with the smallest coordinate value, we search for the closest prototype in a dimension-by-dimension basis, and compute the mean distances between them at each dimension. These distances are used to obtain an upper (or lower) corner of the hyperbox, and a lower (or upper) corner of both prototypes. This process is repeated for all prototypes and it is governed by the following expression,

$$u_{jhi} = l_{jhi+1} = \frac{v_{jhi+1} - v_{jhi}}{2} \tag{6.2}$$

Where $j = 1, 2, \ldots, n$ and $v_{jhi+1} \geq v_{jhi}$. By looking at each variable j of ordered prototypes, for the first or the last prototype, which obviously are closer to the *minimum* or the *maximum* coordinates of the input space, the lower or upper corners are expanded to those coordinates, or $l_{jh1} = 0$ and $u_{jhc} = 1$, respectively, considering that the data is located in the unit hypercube $[0, 1]^n$. This will help the hyperboxes cover the data that initially formed the core (prototype) of the hyperbox, and will hopefully result in a well-defined cluster.

6.3.1. Experimental Studies

a) Synthetic two-dimensional data

A two-dimensional synthetic dataset consisting of 1,000 points was randomly generated around 5 points by using a uniform distribution over the open interval $(-1,\ 1)$ with the following coordinates; $(0.3,\ 0.3)$, $(-0.3,\ 0.3)$, $(-0.3,\ -0.3)$ $(0.3,\ -0.3)$, and $(0,\ 0)$. As a result we obtain five clouds of overlapping data, as shown in Figure 6.1.

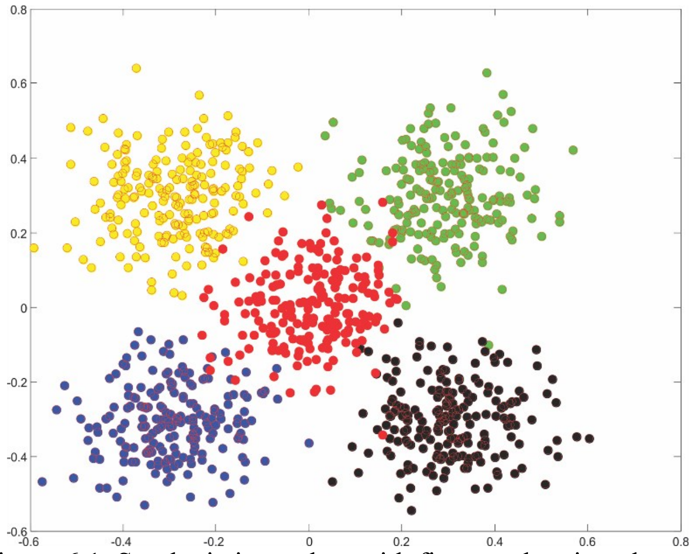


Figure 6.1. Synthetic input data with five overlapping clusters.

After generating the data, the following symmetric function was used to calculate a target value

$$y = x_1 x_2 \exp\left(-x_1^2\ -\ x_2^2\right)$$

First, we divided the output space into four intervals by using the equal probability criterion, then we chose $c = 5$ for each context. After obtaining the prototypes with the conditional FCM algorithm by using (3.21) and (3.22), we constructed the hyperboxes by following (6.2). Their geometry is visualized in Figure 6.2, where the hyperbox cores (prototypes) are represented by black dots.

As shown in the figure, 20 hyperboxes are constructed and only 43.4% of the data is initially covered. The reason is that the method used to construct the hyperboxes has some restrictions. First, the hyperboxes cannot cover more data because their corners can only be expanded to a mean distance between two cluster centers, which limits the coverage of the hyperboxes. Second, as shown in Figure 6.2(*e*) if we superimpose all the intervals, we observe a high amount of overlap among the clusters from different contexts. This means that with this method we initially have a low coverage of data, and if we eliminate the overlaps as shown in Figure 6.3, we end with only 36.5% of the data being covered, which it is not a desired result. We then are required to design a better method to build the hyperboxes and achieve good initial coverage.

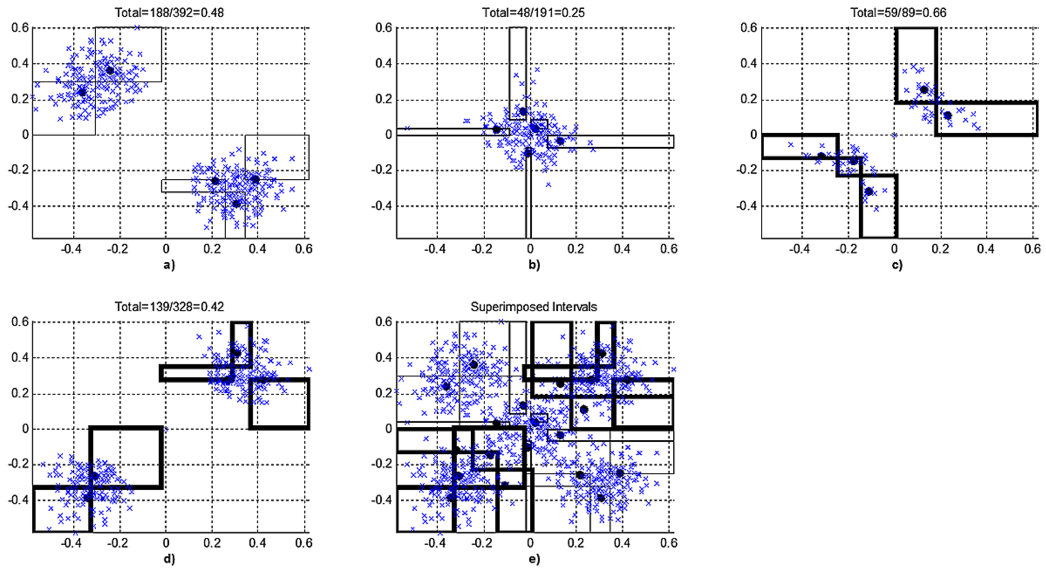


Figure 6.2. Hyperboxes formed by using the cluster centers found in each context and by following (6.2). Plots (*a*) – (*d*) show each of the *h*th intervals, and (*e*) shows all the superimposed intervals.

The overlap reduction was achieved by using the contraction step in Simpson's method (step 3 in Section 3.5), and as it can be observed in Figure 6.3(*e*), it is very efficient since there are no overlaps among the hyperboxes from different contexts, but some of the cluster centers are excluded from the hyperboxes formed, nullifying the initial condition, and completely degrading the information granules. An improved reduction of the overlaps has to be implemented. We are interested in retaining most of the data being covered, and at the same time, covering the prototypes that initially generated the hyperboxes.

6.4. Further Refinements of the Model

In the previous algorithm the hyperboxes were created almost manually, and there is no optimization algorithm that guides the hyperbox construction, so there is still room for further improvements. In the following section, we describe an optimized approach for the construction of the hyperboxes, one that employs two instances of DE to *i*) construct the hyperboxes and *ii*) reduce or eliminate the overlaps. Since we want to cover as much data as possible, the first instance of DE is used to perform an optimal *allocation of granularity* (the protocols in Section 3.3.1), this method helps expand the hyperboxes in an asymmetrical manner, in order to cover data that is asymmetrically distributed around the vicinity of the core. The second instance of DE is used to optimally contract the hyperboxes to reduce or eliminate the overlaps among them, and retain at the same time as much data covered as possible.

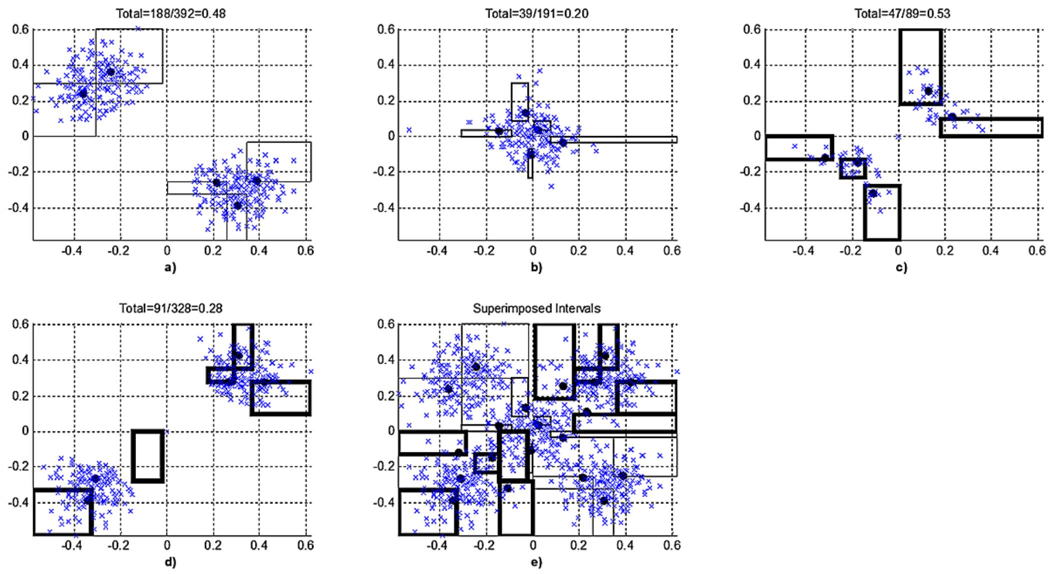


Figure 6.3. Resulting hyperboxes after the overlap elimination by using Simpson's contraction method (step 3 in Section 3.5). (*a*) – (*d*) show each of the *h*th intervals, and (*e*) shows all the intervals superimposed.

6.4.1. Optimal Construction of the Hyperboxes

After the prototypes have been determined for each context by using equations (3.21) and (3.22), their coordinates are used as a reference to start building the hyperboxes. However, in contrast to the previous method, this method expands the corners of each hyperbox in an asymmetric manner to find where most of the data are located. Due to the high performance observed when using protocol 5 in (3.20) to construct the granular fuzzy relational models from Chapter 4, we decided to use a modified version here to construct the hyperboxes. First, we use DE which proposes and optimizes values of the two parameters; ε_{jhi} and γ_{jhi}. Here, we use as reference the ε_{jhi} value, which is incremented gradually from 0.1 to 2 (with a step size of 0.1). Here, for each *j*th dimension, ε_{jhi} represents the distance between the corners l_{jhi} and u_{jhi}. Similar to Section 4.3, the second value $\gamma_{jhi} \in [0, 1]$ is used to asymmetrically distribute this distance around the prototype. Since real-world data are usually not symmetrically distributed, we are interested on expanding the hyperboxes in an asymmetric manner, so the value of ε is asymmetrically distributed across all the center coordinates. This value is similar to the θ parameter used by Simpson, and it is also used to control the maximum size of the hyperboxes. To achieve the asymmetric distribution, condition (3.12) is modified as follows, for each hth interval;

$$\sum_{j=1}^{n}\sum_{i=1}^{c}\varepsilon_{jhi} = \varepsilon nc, \tag{6.3}$$

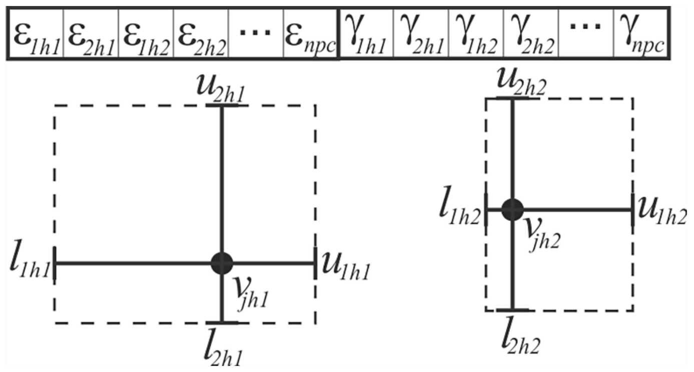
where $h = 1, 2, \ldots, p$. Considering this expression, a population of randomly generated individuals with a dimensionality of $2ncp$ are initialized, which represent the ε and γ values for each jth dimension, in the hth context, and the ith prototype. Both corners l_{jhi} and u_{jhi} are expanded at each dimension to allocate the granularity of the data. More specifically, the corners of the cluster center for each variable (dimension) are expanded by modifying (3.20) as follows,

$$\begin{gathered} l_{jhi} = v_{jhi} - \left(\gamma_{jhi}\varepsilon_{jhi}\right) \\ u_{jhi} = v_{jhi} + \left(\left(1 - \gamma_{jhi}\right)\varepsilon_{jhi}\right) \end{gathered} \tag{6.4}$$

Both values ε_{jhi} and γ_{jhi} are suggested by each differential vector $\boldsymbol{d}$ in the population, and only the first half of the vector, representing each ε_{jhi} is normalized to fulfill condition (6.3), the second half, which represents each γ_{jhi}, is used to asymmetrically distribute each ε_{jhi} around the cluster center (prototype), as shown in Figure 6.4. After this process, all the hyperboxes generated by each individual are tested for coverage, and the result is used as the fitness function. This process creates and expands the hyperboxes to hopefully achieve better coverage of the data located in the vicinity of the core.

As a numerical example, let us assume that $\varepsilon = 0.7$, $n = 2$, and $c = 2$. Due to condition (6.3) we must normalize the values coming from the differential vector corresponding to the interval h which are originally values between [0, 1], since $\varepsilon nc = 4.2$ then $\varepsilon_{1h1} + \varepsilon_{2h1} + \varepsilon_{1h2} + \varepsilon_{2h2} = 4.2$. After normalization let us assume that we obtain the first value $\varepsilon_{1h1} = 1.24$. Furthermore, from the second portion of the differential vector we obtain the value $\gamma_{1h1} = 0.63$, which is not normalized. By following (6.4), the corners of the hyperbox B_{1h1} will expand as $l_{1h1} = v_{1h1} -$ 0.78 and $u_{1h1} = v_{1h1} + 0.45$. Furthermore by using (6.4) we have two extreme scenarios; if $\gamma_{jhi} = 0$

then u_{jhi} will expand to the full value of ε_{jhi}, and l_{jhi} will be the equal to v_{jhi}, in case $\gamma_{jhi} = 1$ the opposite will occur. Whichever is the case, (6.4) guarantees that the distance provided by ε_{jhi} is always asymmetrically distributed around the prototype under observation. Only if $\varepsilon_{jhi} = 0.5$, the corners will have an equal distance around the prototype, but this scenario is highly unlikely due to the randomness and the operations involved to obtain a new offspring.



ε_{1h1} ε_{2h1} ε_{1h2} ε_{2h2} ⋯ ε_{npc} γ_{1h1} γ_{2h1} γ_{1h2} γ_{2h2} ⋯ γ_{npc}
u_{2h1}
l_{1h1}
u_{1h1}
v_{jh1}
l_{2h1}
u_{2h2}
l_{1h2}
u_{1h2}
v_{jh2}
l_{2h2}

Figure 6.4. Upper and lower corners of the hyperboxes constructed with the information contained in a differential vector and by using (6.4), for two clusters in interval h in a two-dimensional space.

This method differs from the one proposed by Simpson: the main difference is that the hyperbox construction is guided by a cluster center, which points at the center of a cloud of data. Not just a randomly selected datum. With the method developed by Simpson, the construction of the hyperboxes highly depends on the first datum used to construct the first hyperbox, which has to be close to other data in order to expand. This might cause a generation of smaller isolated hyperboxes. This selection at the same time creates a larger number of hyperboxes. For the approach pursued here, the hyperbox expansion is directly guided by its corresponding cluster center, giving us the possibility of building a desired number of hyperboxes, and efficiently control their maximum size. Nevertheless, since the main objective of the construction of the hyperboxes is to cover as much data as possible, there is a possibility that overlaps will emerge among hyperboxes.

6.4.2. Optimal Overlap Elimination

To reduce possible overlaps, or completely eliminate them, another instance of DE is initiated to help optimally reduce the clusters. The main goal is to reduce the overlap as much as possible, while at the same time, retain most of the data covered. To reduce the hyperboxes, we look at all the corners of the hyperboxes, and assign an α value to each corner, where $\alpha \in [0, 1]$ is a reduction percentage, applied to the total distance between the hyperbox core (prototype) and each corner, and since this value is guided by the DE algorithm, it will optimally reduce the overlaps among the hyperboxes, refer to Figure 6.5.

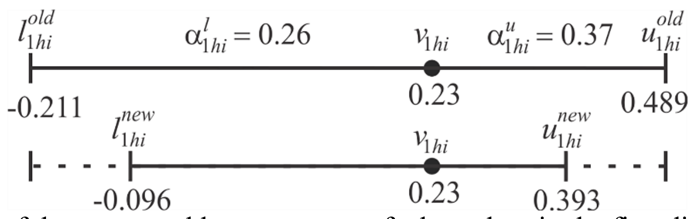


Figure 6.5. Reduction of the upper and lower corners of a hyperbox in the first dimension by using their corresponding α_{jhi} values and by following (6.5).

To denote each α value, we generate an individual that represents a vector $\boldsymbol{\alpha}$. An important consideration for the overlap reduction algorithm is that the hyperbox cannot be reduced further than the cluster center that generated it, and since we want to reduce both *i*th corners, at each *h*th partition, on every *j*th dimension, each individual must have a dimensionality of 2*ncp* to represent the corresponding values α_{jhi}^{l} and α_{jhi}^{u}.

The values of α, represented by each individual, are initially generated with a uniform distribution over [0, 1]. Each one of these values defines an amount of reduction that the hyperbox will suffer in the each dimension. As a numerical example and by using as a reference Figure 6.5, let us assume that two values are proposed by an individual in dimension $j = 1$, these are $\alpha_{1hi}^{l} = 0.26$ and $\alpha_{1hi}^{u} = 0.37$; to reduce the dimensionality of the hyperbox the distance between the cluster center and both corners is calculated, and the results are multiplied by its corresponding α values, this means that the distance between the lower and upper corners are reduced by 26% and 37% respectively, this process is described by the following general expressions;

$$\begin{aligned} l_{jhi}^{new} &= l_{jhi}^{old} + [\left(v_{jhi} - l_{jhi}^{old}\right)\alpha_{jhi}^{l}] \\ u_{jhi}^{new} &= u_{jhi}^{old} - [\left(u_{jhi}^{old} - v_{jhi}\right)\alpha_{jhi}^{u}] \end{aligned} \tag{6.5}$$

By using this method, we have two extreme scenarios; if both values of α_{jhi}^{l} and α_{jhi}^{u} are equal to zero, no reduction will be observed, and if the values are equal to one, the corners will be equal to the prototype in the observed dimension. This method, along with the fitness function, will help the DE algorithm reduce or eliminate the overlaps, and at the same time, keep most of the data covered at all times. The fitness function is specified as follows;

$$Q = \frac{d}{cov} + \frac{cov}{Tcov}$$

Where d represents the number of data considered *counted*, in other words not positioned within an overlap between hyperboxes from other intervals, *cov* is the total number of data covered by all the hyperboxes, included or not in an overlap. The main idea behind this first part of the fitness function is that if the number of data covered (*cov*) is equal to d, then we have complete overlap elimination. The second part of the fitness function had to be included because it was observed that the DE algorithm significantly contracted the hyperboxes to satisfy the condition, and never considered the data initially covered, also reducing the final coverage of the data. So in the second part *Tcov* is equal to the number of data covered by the initial unmodified hyperboxes, included or not in an overlap. So, with these criteria in mind, the best case scenario would be to have a fitness value equal to two, meaning that it achieved a total coverage of the data, and completely eliminated the overlaps.

The reduction of the overlaps is different from the one being proposed by Simpson, where each corner is compared against the corners of other hyperboxes; if this method is used, the hyperboxes would contract and uncover data, the main interest in this study is to preserve the covered data as much as possible, and even retain a few small overlaps instead of totally eliminating them.

6.4.3. Detailed Evaluation of the Hyperboxes

Once the hyperboxes are constructed and the overlaps have been eliminated (or reduced), we verify the covered data that fall within in each hyperbox. To verify these contents we implemented a three-step verification process. For each datum $\boldsymbol{x}_k$;

1. Verify whether the data point is fully included in any of the hyperboxes with (6.1),
2. Compute the Euclidean distance between the datum and all the prototypes obtained in the fuzzy clustering stage, and
3. Measure the membership degree to which the datum falls within each of the hyperboxes by modifying (3.23) as follows,

$$\mu_{B_{hi}}(\boldsymbol{x}_{hk}, \boldsymbol{l}_{hi}, \boldsymbol{u}_{hi}) = \min_{j=1,2,\ldots,n}\left(\min\left(\left[1 - f\left(x_{jhk} - u_{jhi}, \delta\right)\right], \left[1 - f\left(l_{jhi} - x_{jhk}, \delta\right)\right]\right)\right) \tag{6.6}$$

where i = 1, 2, …, c, k = 1, 2, …, N, and h = 1, 2, …, p. This modification allows us to obtain the membership degree of each datum to each ith hyperbox in each hth context.

With these three steps, if the datum is included in only one hyperbox then it belongs to that cluster, and we assign the corresponding y_k value to that hyperbox. If the datum has full inclusion to two or more hyperboxes from a single hth context, then we search for the smallest Euclidean distance between the datum and the hyperbox core, and assign its corresponding target value to the corresponding hyperbox. If none of the hyperboxes cover the datum, then we look at the maximum membership degree to a hyperbox, and assign its corresponding target value to a corresponding hyperbox. For all the scenarios we label the corresponding target datum with the information of the corresponding hyperbox. In other words, if the ith hyperbox from the hth interval includes the datum under observation, then this information (i and h) is included along with the target datum. At the end we look at these labels, and if y_k is part of partition h then is counted as correctly clustered. The last step would be to analyze the information captured by each hyperbox i.

With this method we can look at the information contained in each hyperbox with great detail, assess how many data points are being clustered, and obtain more information about the data contained, such as the mean and standard deviation of the clustered data.

6.4.4. Experimental Studies

For the following experiments we used the synthetic data and five publicly available datasets: Boston housing, California housing, delta elevators, delta ailerons, and MV datasets. The Boston housing dataset contains housing values from suburbs in Boston. This dataset has 13 continuous values, including the target attribute which is the median value of owned-occupied homes in \$1000's with a range of 5 to 50, one binary-valued attribute, and a total of 506 instances [10]. The California housing dataset contains information about all block groups in California from the 1990 census, the task is to approximate the median house value for each block, and contains 20,640 instances and 8 features; 3 real 5 integer values [101]. The delta elevators and delta ailerons datasets are obtained from the task of controlling the elevators and the ailerons from a F16 aircraft but both datasets have different domains. The delta elevators dataset contains 9517 examples, with 6 continuous values, including the target attribute with a range of -0.014 to 0.013, and 1 integer value. The delta ailerons dataset contains 7129 instances, with 6 continuous values, which include the target value with a range of -0.0021 to 0.0022 [8]. The MV dataset is an artificial dataset with dependencies between the attribute values, it contains 40,768 instances and 10 features; 7 real and 3 integer values [50].

a) Synthetic data

The data were first normalized into the unit interval $[0, 1]^n$, then the intervals that divide y_k were defined, at the same time separating the corresponding input data $\boldsymbol{x}_k$; for the synthetic data, four intervals were considered. For comparison, we implemented the complete min-max NN algorithm, and ran it for each interval, with $\theta = 0.2$ and $\delta = 4$, refer to (3.23). We observed that these values would generate an acceptable number of clusters, where each hyperbox contains a large amount of data. The results can be observed in Figure 6.6, where 93.9% of the data were covered and a total of 35 hyperboxes are formed, with 11, 7, 6 and 11 clusters constructed in each interval.

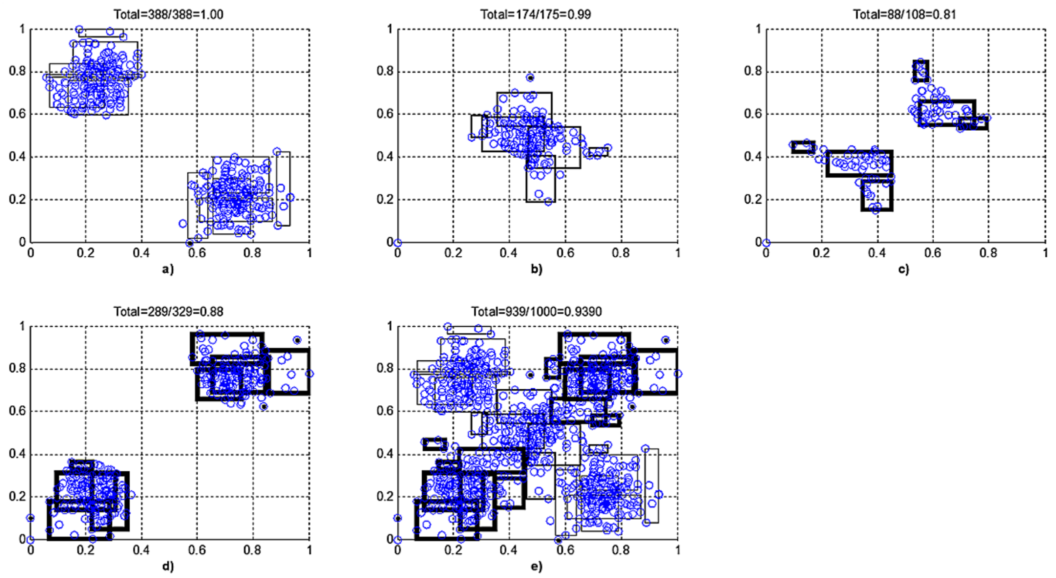


Figure 6.6. Hyperboxes created by using the min-max NN algorithm, where 93.9% of the synthetic data are covered and 35 hyperboxes are created.

The same synthetic data were used to form the clusters in the same number of intervals with $\varepsilon = 0.8$, and the same $\delta = 4$ for the membership function in (6.6), here we selected $c = 5$ for the conditional FCM algorithm. The DE algorithm used 50 individuals and ran for 150 iterations.

We choose this number of individuals because there is a tradeoff between using large or small population sizes, as explained in Section 3.7, and this specific number of iterations was used because there was no significant improvement observed after more than 150 iterations. When the hyperbox construction is finished, the overlap reduction is started, here the DE algorithm used 60 individuals and ran for 350 iterations. At the end, we obtained 95.6% coverage of the data, and obviously obtained 20 hyperboxes; these results are displayed in Figure 6.7.

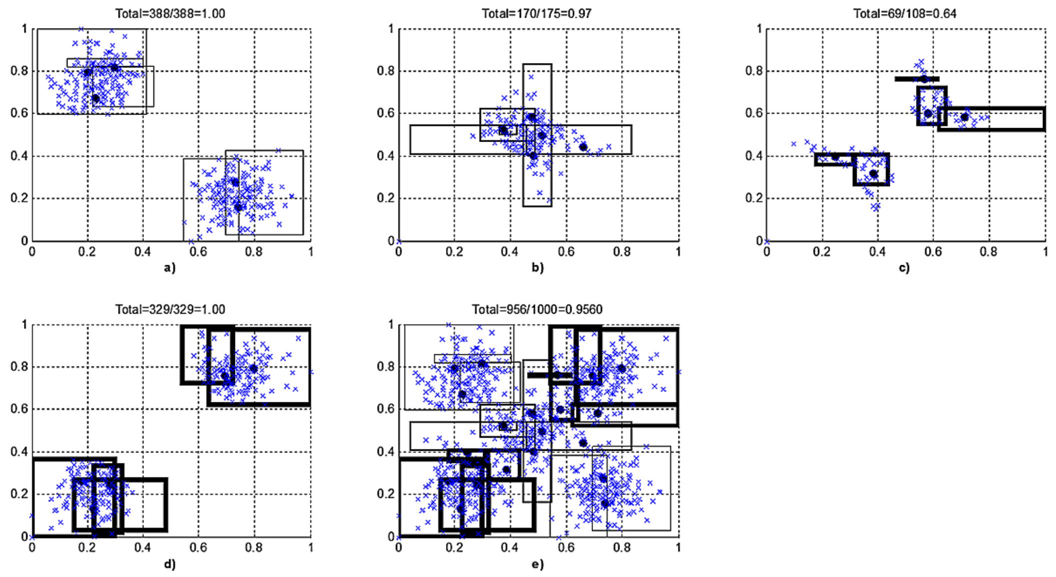


Figure 6.7. Hyperboxes created by using the proposed method, (95.6% of the data covered with the use of 20 hyperboxes).

As it can be observed, both algorithms are very efficient, when clustering datasets with a small number of samples. Upon close inspection, we can see in Figure 6.6(*d*) how three hyperboxes are created around three single data points (circles with a dot inside them), whereas by using the proposed method, in Figure 6.7(*d*) the hyperboxes were expanded to cover those points. This is due to the maximum size restriction used in the Simpson's method for the hyperbox construction. In contrast, we allow the hyperboxes to expand where required. In Figure 6.7 we included the hyperbox cores or prototypes that were found by running conditional FCM in each interval, and these are represented by black dots. At the end, the proposed method provides a slightly larger coverage than Simpson's method, if a more detailed model is desired, the value of θ can be decreased, but this would increase the number of hyperboxes, and for large amounts of data we would have to create another algorithm to group clusters of hyperboxes.

b) Real-world data

These experiments are designed to explore several different scenarios and combinations of number of intervals and cluster centers. For all datasets, we divided y_k into 2 to 5 *equal probability* intervals. Also, we used context-based FCM to obtain 2 to 6 cluster centers per interval. And finally we used different values of ε from 0.1 to 2.0 (with a step size of 0.1) where the main goal was to find a value that helps obtain hyperboxes large enough to cover the data, but not too large because this can cause a problem when reducing the overlaps. For both instances of the DE algorithms, used for the hyperbox construction and overlap reduction/elimination, we used the same settings as with synthetic data. With Simpson's algorithm, we set θ to 0.2, this value was the lowest value we could use, which generated an acceptable number of hyperboxes; lower values would significantly increase the number of hyperboxes, making a highly complex network, very small hyperboxes, and considerably increase the computational time.

Table 6.1. Results obtained with Simpson's algorithm and the Differential Evolution Hyperbox (DEH) construction method, the *Hyper.* column shows how many hyperboxes are obtained at the end.

Dataset/Method	Simpson	Hyper.	DEH	Hyper.
D. Ailerons	64.67%	116	**92.64%**	10
D. Elevators	62.34%	132	**82.96%**	11
MV	37.24%	756	**73.04%**	8
Boston	**70.16%**	259	69.96%	8
California	54.24%	808	**62.94%**	7

As shown in Table 6.1, the method used to construct hyperboxes by using DE improves the coverage in almost all the datasets tested, only for the Boston Housing dataset we see how Simpson's method has slightly better coverage, nevertheless it can observed that Simpson's algorithm generates a significantly larger number of hyperboxes to achieve this result, while in our approach the number of hyperboxes is controlled by the number of prototypes calculated in each interval at the beginning; if we take into account the number of hyperboxes generated for the Boston housing dataset, we can assume that it generated an average of one hyperbox for every two data points, making the model very complex. With the proposed approach, the algorithm only generated eight hyperboxes and covered almost the same amount of data.

For all datasets, the output was divided into two intervals ($h = 2$), this value yielded the best clustering results. Nevertheless, the values c and ε were not the same in all cases. For the Delta Ailerons dataset the best result was found with six clusters for each interval (12 in total) and with

$\varepsilon = 0.6$, but two clusters were empty, or are hyperboxes included inside a bigger hyperbox, so only 10 significant clusters are defined at the end. For the Delta Elevators dataset, we found the best results with six clusters per interval and $\varepsilon = 0.6$, with one empty hyperbox. The best result achieved with the MV dataset was found with six clusters per interval and $\varepsilon = 0.6$, in this case four clusters were found empty. The California housing dataset achieved the best coverage results with four clusters per interval and $\varepsilon = 0.3$. And finally, the Boston housing dataset achieved the best result with four clusters per interval and $\varepsilon = 0.7$, with no empty clusters. In Table 6.2 we include the number of data captured by each hyperbox, as well as the mean and standard deviation of the data captured in each cluster.

Table 6.2. Detailed contents of each Hyperbox, including number of data being covered, mean, and standard deviation (Std).

Delta Ailerons

			Cluster					
			1	2	3	4	5	6
Intervals	1	Data	77	3051	380	0	1	63
		Mean	0.438	0.433	0.442	0	0.465	0.416
		Std	0.029	0.042	0.031	0	0.000	0.041
	2	Data	14	9	37	127	2845	0
		Mean	0.523	0.540	0.547	0.549	0.548	0
		Std	0.015	0.025	0.041	0.036	0.049	0

Delta Elevators

		Cluster					
		1	2	3	4	5	6
1	Data	22	5	4	3	3860	168
	Mean	0.428	0.459	0.343	0.444	0.436	0.441
	Std	0.057	0.033	0.161	0.037	0.053	0.048
2	Data	1503	0	1	7	2198	124
	Mean	0.592	0	0.556	0.587	0.592	0.586
	Std	0.047	0	0.000	0.058	0.047	0.041

MV

			Cluster					
			1	2	3	4	5	6
Intervals	1	Data	0	807	16652	2065	0	0
		Mean	0	0.114	0.086	0.075	0	0
		Std	0	0.019	0.051	0.044	0	0
	2	Data	0	1739	45	2108	1460	4902
		Mean	0	0.425	0.427	0.363	0.333	0.426
		Std	0	0.317	0.300	0.229	0.164	0.282

California Housing

		Cluster			
		1	2	3	4
1	Data	32	3063	4431	4
	Mean	0.035	0.034	0.035	0.034
	Std	0.015	0.013	0.012	0.009
2	Data	3300	51	2109	0
	Mean	0.121	0.092	0.112	0
	Std	0.083	0.035	0.070	0

Boston Housing

			Cluster			
			1	2	3	4
Intervals	1	Data	27	73	38	56
		Mean	0.156	0.274	0.319	0.204
		Std	0.076	0.063	0.033	0.097
	2	Data	12	64	48	36
		Mean	0.606	0.494	0.606	0.544
		Std	0.293	0.129	0.199	0.150

As it can be seen in Table 6.2, some hyperboxes capture large amounts of data, and others only capture less than 10 data points. There are also some clusters that are dominant, or capture large amounts of data, this can happen when there is a large amount of similar data, confirming at the same time that the hyperboxes are well-defined. Also, after verifying the contents of the hyperboxes, we observe that some hyperboxes are empty, in such cases we can assume that these hyperboxes could be included inside such dominant hyperboxes. If we observe the data captured when using the MV dataset, we can see how the data captured by each hyperbox in the first interval have small variations, as suggested by the standard deviations. But for the second interval the data in each hyperbox have larger variations. If we consider both scenarios, where some hyperboxes are empty and others have variable numeric data, we can think on an improvement for the algorithm, where a different number of clusters is defined in each interval, for example the first interval of the MV dataset only needed three hyperboxes to capture the data, while the second interval might need a larger number of hyperboxes (or a division of this single interval into smaller intervals) to capture less variable data. We can also think on another way to divide the initial output intervals, given that this step plays an important role in the construction of the information granules, one can contemplate a method where similar data are separated, and not only focus on similar amounts of data for each interval.

6.5. Summary

In this chapter we constructed a set of information granules coming in the form of hyperboxes. To build these hyperboxes first the output space was divided into intervals by using the EP criterion. Each interval is then used to induce data in the input space, which is clustered by using the context-based FCM algorithm. In this sense, the granulation vehicle is the context-based fuzzy clustering algorithm, or a so-called conditional FCM. We also reported on a series of numerical experiments, based on the construction of the hyperboxes by using a numerical constraint. The granular model is further refined by employing DE on two main design phases: *a*) a construction of the hyperboxes, and *b*) their overlap reduction. We also included a series of numerical experiments with synthetic data and selected real-world data coming from publicly available machine-learning repositories. The results showed how the method is comparable to previous methods found in the literature. Nevertheless, the results produced by the proposed method are better when dealing with large multi-dimensional data. If the number of data is low with a few dimensions, then the method developed by Simpson performs similarly. However, the results show that the Min-Max neural network algorithm produces a very complex granular model, due to the high amount of hyperboxes formed in the input space, and if the dimensionality in the data increases, the number of hyperboxes becomes quite large.

7. Refinements of Prototypes to Enhance the Classification and Reconstruction Performance of FCM

Owning to their abilities to reveal structural relationships in data, fuzzy clustering plays a pivotal role in fuzzy modeling, pattern recognition, and data analysis, as it has been discussed and demonstrated throughout this study. As supporting an unsupervised mode of learning, fuzzy clustering, brings about unique opportunities to build a structural backbone of numerous constructs in the areas identified above. A follow-up phase is required when the structural findings developed in the form of fuzzy clusters need some refinements, usually when the clustering results are used in the supervised model. Following this general line of thought, in this Chapter we propose a novel approach to optimize the clustering and classification performance of the Fuzzy C-Means (FCM) algorithm. Proceeding with a collection of clusters (information granules) produced by the FCM, we carry out the refinements of the results (in order to improve the representation or classification capabilities of fuzzy clusters) by adjusting a location of the prototypes so that a certain performance index becomes optimized. At this phase, the optimization is carried out in a supervised mode with the aid of Differential Evolution (DE). We propose five different strategies to adjust a location of the prototypes. Experimental studies completed on synthetic data and publicly-available real-world data quantify the improvement of the representation and classification abilities of the clustering method.

7.1. Problem Formulation

Fuzzy clustering is a useful unsupervised learning technique aimed at analyzing, organizing, and revealing structures in data sets such as patterns. This strategy helps partition a given input space into a certain number of regions (groups, categories), depending upon some predefined similarity/dissimilarity measure. In contrast to two-valued clustering, fuzzy clustering offers higher flexibility by admitting the degrees of membership to individual clusters to be positioned in the unit interval. Meaning that a data point could belong to several clusters with some membership values. While useful in general, FCM (and all other clustering methods) suffer from some limitations, including difficulties to cope with outliers, selection of a suitable number of clusters, and a choice of a specific distance expressing similarity between two patterns. Quite often, clustering is regarded as a prerequisite for further tasks, being either focused on more detailed data analysis, or system modeling. The genuine advantage of clustering is its *unsupervised* mode of operation. This, however, acts as a double edge sword: we may anticipate that further improvements/adjustments of structural findings (clusters) could be beneficial prior to use the clustering results in further constructs of system modeling or pattern classifiers. The improvements could be based on some mechanisms of supervised learning.

In this Chapter we focus on the enhancement of the classification and reconstruction capabilities of the FCM algorithm by exploiting the concepts of *unsupervised* learning (clustering) followed by *supervised* learning [39]. More specifically, the development strategy consists of two essential phases.

1) Discovery of a structure in the data. This is achieved in an unsupervised mode realized by the the FCM algorithm.
2) Modification (refinement) of the prototypes. In this phase, realized in the supervised mode of learning, we optimize a certain reconstruction criterion (when representing the data) or a classification rate (when dealing with the classification problem).

We propose and study five different strategies (protocols) supporting an allocation of prototypes. Each protocol comes with a different parametric flexibility. The approach proposed in this Chapter is experimented with the use of synthetic data and real-world data coming from publicly-available repositories.

7.2. Architecture and Main Phases of Processing

The model proposed in this study consists of two stages (refer to Figure 7.1); (*i*) an unsupervised clustering stage, and (*ii*) a supervised refinement stage. The number of fuzzy sets constructed in the input space is equal to c and the fuzzy sets are described as $A_1, A_2, \ldots, A_c$. Each fuzzy set (cluster) is then associated to a class $\ell = 1, 2, \ldots, \varpi$, by looking at individual similarities between the data captured by each cluster and the output class. The formation of these associations plays an important role in the construction of the model. First, the data captured by each information granule can be associated to a class, this helps us form fuzzy association rules. Second, the model can be used as a classifier, and as such, it could be adjusted by an optimization mechanism to reduce possible performance issues.

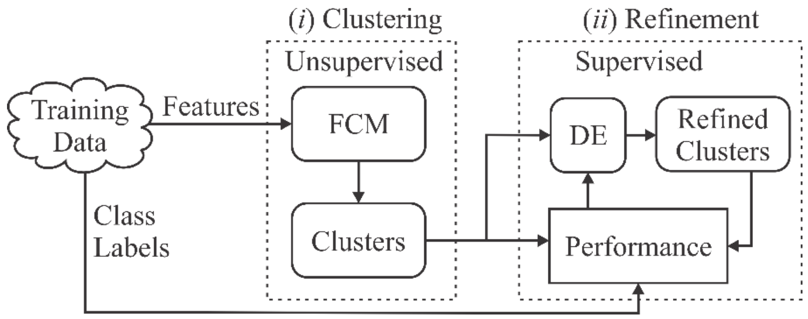


Figure 7.1. The two stages envisioned for the granular classification model, where the learning is performed in two stages, unsupervised and supervised. Here, the clusters coming from FCM are assessed and passed to a refinement stage.

At the first stage the FCM algorithm, described in Section 3.2.1, determines the structure of the *unlabeled* numerical data and a set of clusters is obtained, which are pairwise disjoint, non-empty, and come represented as the partition matrix and a set of coordinates characterizing the cluster centers (prototypes). Once the structure in the data has been stablished, the quality of the clusters is assessed by considering two criteria; the reconstruction error in Section 3.2.2, expressed in (3.8), and the classification rate from Section 3.2.3, provided by (3.10). To classify the data, each cluster is associated to a certain class (label). This approach allows the number of clusters to be independent of the number of actual classes. Next, the input data that belongs to a cluster and also to the cluster's class are counted and a classification rate is obtained. This rate serves as a reference (starting point) for the optimization of the model. Alternatively, the prototypes are assessed by looking at the reconstruction error, or the distance between the original numeric entities in $\boldsymbol{x}_k$ and the reconstructed numeric entities $\hat{\boldsymbol{x}}_k$. It is obvious that if this distance is zero then we have achieved a perfect reconstruction of the data. The reconstruction error also serves as an objective function to be used in the optimization phase.

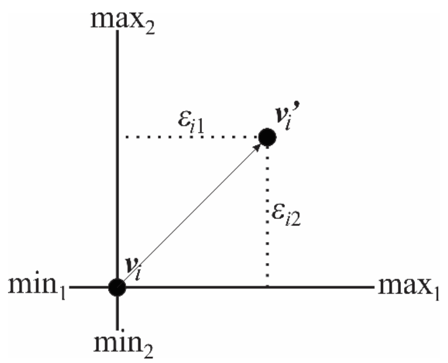


Figure 7.2. Modification of prototype $\boldsymbol{v}_i$ controlled by the level of change ε for variables 1 and 2; in this case the levels of change are equal and the directions of change are positive (+) for both variables.

At the second stage, the ensuing DE algorithm (Section 3.7) serves as the optimization vehicle to adjust (refine) the structure already developed in the input space by strategically relocating the coordinates of the prototypes in a dimension-by-dimension basis to alter the degree of belongingness of the data points to the clusters. Here, each individual proposes a new set of coordinates for the prototypes represented by $\boldsymbol{v}'_i$, which are used to generate a new partition matrix U', these are assessed to obtain the performance of the proposed clusters. With these movements the goal is to discover new classes or outliers, enhance and refine the boundaries between classes, and improve the quality of the clusters. If the classification rate is used, then the objective is to maximize this performance, and then look at the reconstruction error of the optimized clusters. On the other hand, if the reconstruction error is used then the goal is to minimize this error, and at the end look at the classification rate. This process is repeated until a number of generations, selected by the end user, have been reached. At the end, the generalization capabilities of the enhanced clusters are tested by presenting the model with previously unseen data, which is separated beforehand for this purpose.

To control the modifications of the prototypes, we use a control variable, viz. ε, regarded as the total *level of change* (modification). Obviously if $\varepsilon = 0$ the prototypes are not modified, and as this value increases, the prototypes will move further from the origin as depicted in Figure 7.2. We also use a certain condition to constrict the modifications of the prototypes. This condition allows the DE algorithm to strategically move the cluster centers around the original coordinates in a controlled manner, which will alter the shapes of the fuzzy sets, and at the same time, the degree of belongingness of the data to the information granules. Nevertheless, this condition will only control the distance that a prototypes moves from its origin, and we also need to consider the *direction of change*. This direction, can either move the coordinate closer to the minimum range ($-$) or the maximum range (+) in each variable. For example, in Figure 7.2 the direction of change of prototype $\boldsymbol{v}_i$ is positive (and of equal length) for both variables. Having this in mind, when designing the DE algorithm, we have to consider two alternatives;

i) the levels of change are equal for all the prototypes and for all variables, therefore we have to implement a mechanism to get the directions of change; and *ii)* the levels of change are different, then we need to decide how to distribute these levels for all the prototypes for all variables, in addition to getting the directions of change.

In the ensuing sections we describe several alternatives to solve these contemplations. Once we have decided how to change the position of the prototype, and the clusters are optimized and assessed, the granular model can be translated into a series of rules in a very straightforward fashion. The input clusters, which are now labeled to belong to a class, form the condition part of the rules, while the classes are located in the conclusion part of the rule. Every input datum $\boldsymbol{x}_k$ belonging to fuzzy set A_i, $i = 1, 2, \ldots, c$, gives rise to the following rule;

$$\text{if } \boldsymbol{x}_k \text{ is included in } A_i \text{ then class } \ell$$

This will allow us to associate every input datum to a cluster, and in consequence, to a class. But before modifying the prototypes and attempting to optimize the clusters, a number of important conditions have to be considered. First, the data originally clustered and associated to a class has to remain clustered and associated to that class, this is why we implemented some constrictions to move the prototypes otherwise these would move freely across the n-dimensional input space, loose the initial clustering results, and potentially decrease the initial clustering performance. Second, the prototypes cannot move outside each jth dimension of the input space, this means that for every variable, the prototype can move as far as the minimum or maximum values on each variable's range. In case a coordinate of the n-dimensional prototype is placed outside its range, e.g. j, it is adjusted to $\min_j$ or $\max_j$, depending on whichever is closer. Third, the optimized model must provide a good *generalization*, meaning that the classifier must not be optimized to a particular training set, but instead it has to be capable of recognizing novel patterns, viz. data that has not been presented in the training stage. And fourth, the minimum number of clusters has to be equal to the number of classes in the data under observation. This is intuitively obvious as each cluster has to be assigned to a class, and having fewer clusters than classes will result in a low classification rate.

7.3. Optimization of Information Granules

The FCM algorithm optimizes a certain objective function, however this performance is not directly linked with the reconstruction error or the classification rate. Hence the idea is to enhance the performance of clustering by directly linking their structure by adjusting the position of the prototypes. Given that the partition matrix contains the membership degree of every input data to an information granule, adjusting the coordinates of the prototypes will consequently produce a new partition matrix. This will have a direct impact on the membership degrees and the direct association of the data to a cluster. With these changes we are trying to adjust the membership degrees to discover new data, and also try to separate the classes that are not linearly separable. To relocate the prototypes several strategies are studied.

7.3.1. Migration of Prototypes

To change the position of the prototypes we introduce a level of change (modification) for the coordinates of the prototypes, viz. $\varepsilon \in [0, 1]$. This level affects the position of a given prototype in each of its variable j, hence, this value has to be distributed throughout all the variables while satisfying at the same time the following condition,

$$\sum_{j=1}^{n} \varepsilon_j = \varepsilon n \tag{7.1}$$

where $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \geq 0$. Moreover, since we have at least two clusters for each variable, then we must obtain a level of change for the jth variable and the ith prototype. The value of ε_j is determined as follows,

$$\varepsilon_j = \frac{\sum_{i=1}^{c} \left|\varepsilon_{ij}\right|}{c} \tag{7.2}$$

To change the position of the prototypes and decide the direction of change, we considered two alternatives. The first alternative is to have an equal level of change for all the prototypes, that is $\varepsilon_{ij} = \varepsilon$, $i = 1, 2, \ldots, c$; $j = 1, 2, \ldots, n$, as depicted in Figure 7.3(*a*). This alternative might be an attractive venue, and we would only have to decide upon the direction of change. The second alternative is to have an asymmetrical level of change distributed across all the prototypes, as illustrated in Figure 7.3(*b*). For this alternative we would have to asymmetrically distribute the levels of change across all the prototypes and also determine the directions of change.

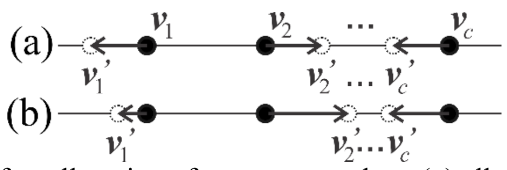


Figure 7.3. Different strategies of re-allocation of prototypes, where (*a*) all the levels of change are equal, and (*b*) the levels are asymmetrically distributed.

To observe the impact that each scenario has in the final performance, we implemented several protocols to modify the prototypes. Each protocol has a different level of parametric flexibility, and these produce different levels or directions of change depending on the value of ε assigned by the end user.

1. The levels of change ε_{ij} and the directions are random. This protocol is used as a point of reference to help assess a relative performance of all the other protocols, and will allow us to demonstrate how an optimized process performs over a purely random one. For this option an optimization is not required. To obtain the random changes, two vectors of *cn* values are initialized by using a uniform distribution of random values between the open interval (0, 1) to obtain the levels and directions of change.
2. The levels of change ε_{ij} are constant and the direction is optimized. For this protocol all the levels of change are equal to ε.
3. The levels of change ε_{ij} are random and the direction is optimized. The idea behind this protocol is to initialize a vector of *cn* random values use these to asymmetrically distribute these to the levels of change ε_{ij}, while the direction is optimized.
4. The level of change ε_{ij} is optimized and the direction is random. Here, a vector of *cn* random values is used to get the directions of change, meaning that all the directions of change are arbitrary and the levels of change are optimized.
5. The level of change ε_{ij} is optimized along with the direction of change.

By using these protocols we can study the impact that each one has over the classification performance. Depending on each protocol, and assuming that we have a vector $\boldsymbol{d}$ of *cn* values between the open interval (0, 1), which represent every *n*-dimensional coordinate of the *c* prototypes, the values in this vector are treated differently. For protocols 1-4, we transform $\boldsymbol{d}$ into a matrix $E \in \mathbb{R}^{c \times n}$. Each column in E is then normalized by using equation (7.2) to satisfy condition (7.1), with this the levels of change ε_{ij} are obtained. In case that the levels of change are constant we simply set $\varepsilon_{ij} = \varepsilon$. If the numerical values from $\boldsymbol{d}$ are used to get the direction of change, these are converted as follows; if $d_q < 0.5$ then $d'_q = -1$, else $d'_q = 1$, for $q = 1, 2, \ldots, cn$. In the case of a random vector of cn values, these will be treated equally.

If protocol 5 is used, then the vector $\boldsymbol{d}$ is processed as follows; if $d_q < 0.5$, then the entry is normalized to a value between the open interval $(-1, 0)$, and if $d_q > 0.5$, then it is normalized to a value between $(0, 1)$. By introducing the absolute value into equation (7.2), we make sure that the condition in (7.1) is fulfilled. The levels of change are then obtained in the same fashion as for the other protocols.

Once the levels of change ε_{ij} are obtained, along with the direction of change, for protocols 1-4 we transform the vector of directions into a matrix $H = [h_{ij}]$, and use these entries to obtain $\varepsilon'_{ij} = \varepsilon_{ij} h_{ij}$ and get the level and positive or negative direction of change. For protocol 5, the direction is already included in the level of change. The final step would be to add the level (and direction) of change to the coordinates of each prototype as follows;

$$v'_{ij} = v_{ij} + \varepsilon_{ij}' * \text{range}_j;\ i = 1, 2, \ldots, c; j = 1, 2, \ldots, n$$

Where $*$ represents the conventional multiplication, and $\text{range}_j = \max_j - \min_j$. The range of each variable j is included in this expression because initially the data is not normalized to have values located within the unit interval. Once the new coordinates of each partition are calculated, a new reconstruction error is obtained by using (3.8), or the classification rate by using (3.10).

7.3.2. The use of Differential Evolution

To optimize the position of the prototypes, the DE algorithm is used (Section 3.7). This optimization algorithm will propose a population of ρ individuals or differential vectors $\boldsymbol{d}$ which enclose enough information to change the position of each ith prototype a certain level and direction along each jth variable. Hence the size of these vectors is equal to cn and contain values between the open interval (0, 1).

The processing of each individual depends on the protocol that is being used and on the value of ε. For protocols 2 and 3 the differential vector will be transformed into (-1) or $(+1)$, depending on the numerical values enclosed in this vector. If protocol 4 is used, the differential vector will be transformed into the levels of change ε_{ij} by normalizing the numeric entries to satisfy condition (7.1). For the special case of protocol 5, the values will be transformed into the levels and directions of change without having to use a vector of size $2cn$, which could lead to higher processing times. Depending on the objective function that is being used, the goal of the DE algorithm is to search for an adequate position of the prototypes that could increase the performance of the clusters. If the reconstruction error from (3.8) is used as the objective function, then the goal is to minimize this error, or more specifically,

$$Q = \frac{1}{N}\sum_{k=1}^{N} \|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|^2 \rightarrow \min \tag{7.3}$$

In case that the classification rate from (3.10) is used, then the goal would be to maximize this performance. This classification rate is used as the objective function to be maximized,

$$Q = 1 - \frac{M_1 + M_2 + \ldots + M_c}{N_1 + N_2 + \ldots + N_c} \rightarrow \max \tag{7.4}$$

It has to be highlighted that the level of change ε has an obvious impact on the new set of clusters and consequently on the membership degrees, so careful measures have to be taken when setting this value. It is recommended to start with small values, e.g. $\varepsilon = 0.05$, and increase them gradually; it was observed that larger values could cause an overfitting effect on the classifier, and small values could drive the performance into local minima. Nevertheless, the value of ε has a strong relation to the number of clusters chosen; for small values of c a large level of change ε is needed, and vice versa. However this is not a rule of thumb, and the precise values needed to accurately classify a dataset is always problem dependent. There is also the issue of choosing an appropriate fuzzification factor. In the following section we offer a set of experiments and demonstrate how these parameters are selected.

7.3.3. Experimental Studies

These following experiments were designed to test the proposed protocols and explore several different scenarios; we look at different values for the number of clusters c, fuzzification coefficient m, and level of change ε. Furthermore, we tested the granular model with a synthetic dataset and eight publicly-available datasets. To assess the performance of the clusters we use two criteria.

1. Maximize the classification rate and then record the reconstruction error.
2. Minimize the reconstruction error and then record the classification rate.

Obviously the second criterion does not have as an objective the classification of the data but rather is dedicated to find a better reconstruction of the numerical entries. For both criteria the objective is to observe if the quality of clusters is optimized, hence it could be interesting to see if maximizing the classification rate would decrease the reconstruction rate (and vice versa) and at the same time increase the quality of the clusters.

a) Synthetic Data

As a point of departure, a synthetic dataset is considered. This dataset is two-dimensional, with $N = 770$ and $\varpi = 6$. The peculiarity of this dataset is that it resembles a *target* and it has four outliers that represent four classes. These outliers have values that are very different from most data. Class one has 395 instances with values within the open interval (-0.5, 0.5), while class two has 363 instances forming an outer ring within the open interval (-2, 2). The remaining four classes have three instances each, and lie outside the outer ring around the values (-3, -3), (-3, 3), (3, -3), and (3, 3). The geometry of the dataset is visualized in Figure 4.

The problem with this dataset is that when running only the FCM algorithm and after obtaining the classification rate, only two classes are correctly clustered and classified; one corresponding to the inner circle, and the second enclosing the outer ring along with the outliers, as shown in Figure 7.4(a). This scenario is observed for $c = 6$, which is the minimum number of clusters we can use, and $c = 200$; a large number of clusters were used because usually the clustering performance improves along with the number of clusters. This demonstrates that it does not matter how many clusters are chosen, the FCM algorithm in this case only clusters the two dominant classes, this results in a classification rate of around $Q = 0.98$, and the outliers are never classified in their own class.

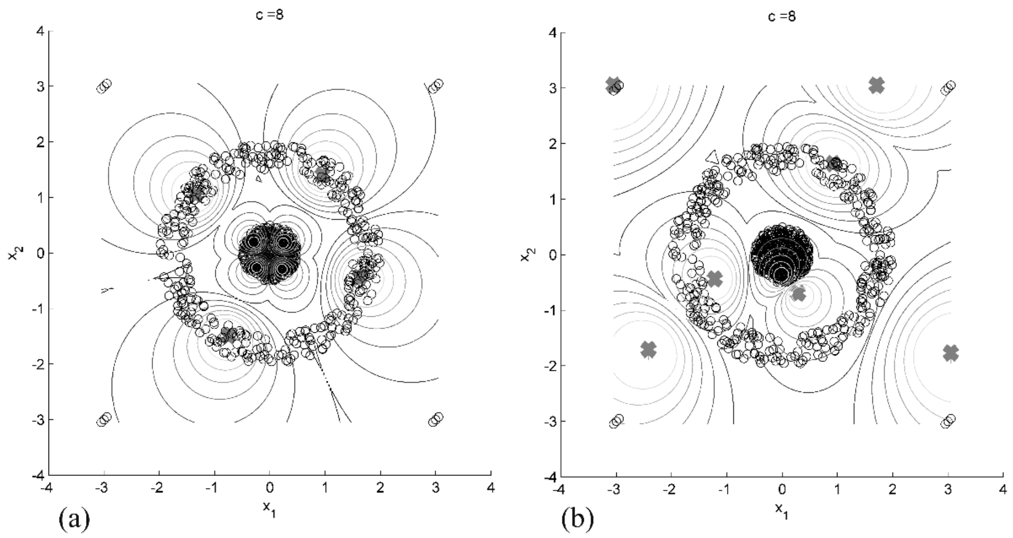


Figure 7.4. Target dataset shown as black circles with (*a*) original prototypes obtained with the FCM algorithm and (*b*) optimized prototypes; all the prototypes are shown as larger gray crosses surrounded by the boundaries of each cluster (shown as curves).

To optimize the cluster centers first the patterns were randomly permutated and then divided into 80% to train the algorithm, and the remaining data is used for testing purposes. In all the experiments, the data separated to train the algorithm never changes to maintain a consistency across all the experiments. The DE algorithm uses 60 individuals and runs for 250 generations. We choose these values because we observed that increasing the number of individuals or generations did not improve the classification rate. For the FCM algorithm, several number of clusters and fuzzification coefficients were used, namely c = 6, 7, …, 15 and m = 1.1, 1.2, …, 3. As for the level of change the values ε = 0.05, 0.1, …, 0.3 were used. In Table 7.1, we include the best classification rates for all the protocols and included the reconstruction error.

Table 7.1. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for the training, testing, and the reconstruction error for the synthetic dataset.

Method	FCM	P1	P2	P3	P4	P5
c	14	14	10	10	8	8
m	1.4	1.4	2.2	2.8	2.6	1.8
ε	--	0.15	0.2	0.25	0.4	0.35
Train (%)	98.86	99.51	100	100	100	100
Test (%)	98.70	99.35	100	100	100	100
Total (%)	98.83	99.48	100	100	100	100
Rec. Error	0.088	0.023	0.022	0.019	0.031	0.029

To obtain the results from Table 7.1, we selected the rates from each protocol that attained the best classification with the lowest number of clusters and level of change. Overall, almost all the protocols reached a classification rate of 100%, and the lowest result was obtained with protocol

1, due to the random levels and directions of change. Also, the data used for testing was also completely classified, meaning that the refined prototypes retained their generalization. Furthermore, protocols 4 and 5 achieved the best classification with only 8 clusters, and the level of change for protocol 4 is larger, when compared to protocol 5. Moreover, the reconstruction error for FCM is larger than the errors for the optimized clusters, meaning that these clusters not only increased their classification capabilities but also their quality of reconstruction. Here protocol 3 achieved the lowest reconstruction error, but needed 10 clusters to classify the data.

To have a better visualization of the classification result obtained with protocol 5, refer to Figure 7.4(*b*) and 7.5(*b*) which show the optimized cluster centers, after modifying the prototypes. As pictured in Figures 7.4(*a*) and 7.5(*a*) initially, with the FCM algorithm and $c = 8$, four prototypes are positioned in the inner circle to cluster only class one, while the remaining four are positioned close the outliers to cluster parts of the outer ring along with each outlier. Moreover, the boundaries of each cluster show the inclusion of each outlier in a cluster, refer to Figure 7.4(*a*). When these prototypes are moved and optimized, we observe how only one prototype is needed to cluster class one (Figure 7.5(*b*) - 3), three to cluster class two (Figures 7.5(*b*) - 1, 7, 8), and the remaining four to cluster each outlier (Figures 7.5(*b*) - 2, 4, 5, 6). We can also visualize how some of the prototypes originally positioned in the inner circle moved to cluster the outer ring, while other prototypes moved further, as allowed by the constraint set by the level of change ε. It is clear that each prototype movement affected the membership degree of the outliers, the outer ring, and the inner circle in all the clusters.

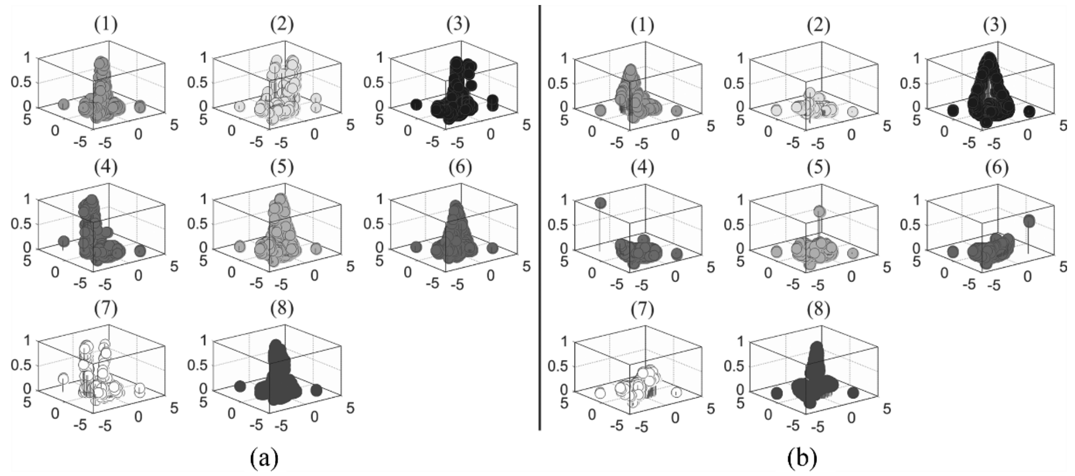


Figure 7.5. (*a*) Initial shape and position of each cluster and (*b*) shape and position of optimized clusters.

For an overall insight of the results from all the protocols, and for comparison purposes, we plotted the results in Figures 7.6(*a*) and 7.6(*b*). In the results from Figure 7.6(*a*), which is the average classification rate for all the values *m* and *ε* for each cluster. Here the FCM algorithm had an overall performance of around 98.4% for $c = 3$ to $c = 13$, and slightly increased with $c = 14$ and $c = 15$, while protocol 1 had the worst overall results, since it lost the classification rate along with the clustered data; only with $c = 15$ it almost reached the classification rate of FCM. Protocol 2 had worst results than FCM from $c = 6$ to $c = 8$, after that it started to reach the results obtained with Protocol 3. Lastly, Protocols 4 and 5 had a similar performance, nevertheless, the best overall performance was obtained for protocol 5. In general, the classification rates increase as the clusters increase, as expected. A similar behavior can be seen in Figure 7.6(*b*), which is the average classification rate for all the values of *m* and *c* and each level of change. Here, if the levels of change increase, protocols 3, 4 and, 5 start obtaining better results. Only protocol 2 lost its overall performance with $\varepsilon = 0.3$, and for protocol 1, the increasing levels had an overall negative effect, due to the randomness of this protocol.

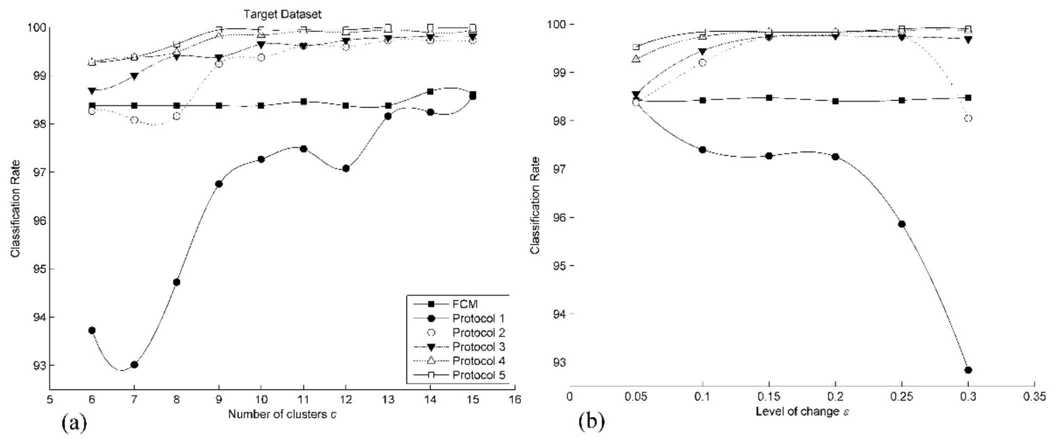


Figure 7.6. Overall classification rates for the synthetic dataset by using all the protocols, (*a*) for different number of clusters, and (*b*) the levels of change.

It is important to highlight that for this synthetic dataset, larger values of *ε* along with smaller values of *c* caused and overfitting effect, meaning that the refinement of the prototypes resulted in a perfect classification of the training samples (100%), but it resulted in a low classification for the novel patters, in this case the testing set.

If the objective function of DE is changed to minimize the reconstruction error, the results shown in Table 7.2 are obtained. The results were selected by looking at the lowest reconstruction error, then we found the results where the classification rate was higher.

By looking at these results it might be obvious that if the objective of the DE is changed, then the algorithm is not dedicated to obtain the best classification rate, but rather a better reconstruction of the data, hence the classification performance would be reduced. Nevertheless, if we look at the highest classification rate, protocols 2 and 5 manage to classify the data and, at the same time, maintain a low reconstruction error. But if we focus on the results where the reconstruction error is lower, the classification rates sometimes decrease, especially for protocols 1 to 3. Here protocol 1 achieved a better classification result than protocols 2 and 3, nevertheless it has the highest reconstruction error (among the protocols), and this result is not guaranteed to be repeated due to the randomness of the levels and directions of change. Only Protocols 4 and 5 achieved better results than FCM, and their reconstruction errors are slightly lower, when compared to the ones with the maximum classification rate. Overall, the reconstruction errors are smaller on the protocols that are optimized by DE, and by losing a small amount of reconstruction error, a higher classification rate is obtained, this points to the fact that not only the reconstruction performance is enhanced, but also the classification capabilities of the information granules.

Table 7.2. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for the training and testing data, for the minimum reconstruction error, and maximum classification rate for the synthetic dataset.

Minimum reconstruction error						
Method	FCM	P1	P2	P3	P4	P5
c	14	10	14	15	15	15
m	1.4	2.8	2.2	2.2	2.4	1.8
ε	--	0.4	0.2	0.2	0.15	0.1
Train (%)	98.86	83.44	71.10	77.92	99.67	99.68
Test (%)	98.70	79.87	74.02	79.87	99.35	100
Total (%)	98.83	82.72	71.68	78.31	99.61	99.74
Rec. Error	0.088	0.018	0.015	0.014	0.011	0.011
Maximum classification performance						
Method	FCM	P1	P2	P3	P4	P5
c	15	14	14	14	13	14
m	1.2	2.4	1.4	1.6	1.6	1.8
ε	--	0.3	0.2	0.2	0.15	0.1
Train (%)	99.18	99.83	100	99.67	99.67	100
Test (%)	98.70	100	100	99.35	99.35	100
Total (%)	99.09	99.87	100	99.61	99.61	100
Rec. Error	0.09	0.018	0.017	0.018	0.014	0.012

b) Real-World Data

For the following experiments we used eight publicly available datasets: iris, thyroid, seeds, vertebral column, image segmentation, wine, and chillanto datasets. The iris dataset is perhaps the best known dataset to be found in the literature. This dataset contains three classes with four features and 50 instances each, and each class describes a type of iris plant, viz. 1 – setosa, 2 – versicolor, and 3 – virginica [10], [43]. Moreover, this dataset has one linearly separable class, while the other two are not linearly separable from each other. The thyroid dataset has five features and 215 instances, the task is to detect if a given patient is 1 – normal, or suffers from 2 – hyperthyroidism or 3 – hypothyroidism [8], [132]. The seeds dataset has seven attributes and 210 instances, and contains the measurements of geometrical properties of kernels belonging to three different varieties of wheat, 1 – Kama, 2 – Rosa, and 3 – Canadian [24]. The vertebral column dataset has six features and 310 instances, it contains six biomechanical features used to classify orthopedic patients into three classes, 1 – normal, 2 – disk hernia, or 3 – spondylolisthesis. Another variant of this dataset is to recognize only two classes, 1 – normal or 3 – abnormal [10]. The image segmentation dataset contains 19 features and 2310 instances. The instances are randomly drawn from a database of seven outdoor images, used as classes, where each instance encodes a 3 × 3 pixel region [10]. The wine dataset has 13 features and 178 instances, and the data are the results of a chemical analysis of wines grown in Italy, all coming from the same region but derived from three different cultivars, which are used as classes [7]. And finally, the infant cry dataset, known as the *Chillanto* dataset [135], [136], which contains cry recordings of 98 Mexican babies categorized into three classes 1 – normal babies, 2 – deaf babies and 3 – babies with asphyxia. These recordings where divided into one-second samples and processed to extract their acoustic features, resulting in 1049, 879, and 340 samples from normal, deaf, and babies with asphyxia respectively, for a total of 2268 instances and 144 features.

The experiments were designed to explore several different scenarios and combinations of number of clusters, fuzzification coefficients, and level of change. For all datasets, the minimum value for c is equal to the number of classes, while the maximum number of clusters was variable and dependent on each individual dataset. We used different values for the fuzzification coefficient m from 1.1 to 3, with a step size of 0.2, and finally the level of change ε from 0.05 to 0.5, with a step size of 0.05. Also, all datasets are randomly permutated by using a uniform distribution, and divided into 80% training and 20% testing sets, with no further modification of the data for all experiments. The selection of the classification rates for each protocol, and each dataset is similar to the previous experiment; we selected the best overall classification rate attained with the lowest number of clusters, lowest level of change, and with the best generalization capabilities. The reconstruction error is also reported for these results. The results from all the experiments are included in Table 7.3.

Table 7.3. Classification rates obtained with the initial prototypes (FCM) and with all the protocols for all the datasets. The maximum classification rate and reconstruction error for all sets are shown.

	Iris							Thyroid					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	8	5	6	4	3	8		7	5	7	7	5	4
m	2.6	2.2	2	3	1.8	2.2		1.8	1.4	2.8	2.2	1.2	1.2
ε	--	0.15	0.35	0.15	0.15	0.15		--	0.1	0.2	0.25	0.25	0.15
Train (%)	97.5	96.7	99.17	99.17	99.17	100		87.21	88.95	96.51	96.51	100	100
Test (%)	100	100	100	100	100	100		95.34	95.35	95.34	95.34	97.67	97.67
Total (%)	98	97.33	99.33	99.33	99.33	100		88.83	90.23	96.28	96.27	99.53	99.53
Rec. Error	0.18	0.131	0.162	0.129	0.14	0.086		0.325	0.148	0.143	0.158	0.246	0.149
	Seed							Column 2					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	6	8	7	6	6	7		12	11	11	12	12	12
m	2.6	2.6	2.8	1.4	2.4	1.4		1.8	2.8	2.6	2.6	2	1.6
ε	--	0.15	0.45	0.3	0.4	0.15		--	0.05	0.1	0.1	0.15	0.25
Train (%)	92.26	92.85	97.02	95.83	97.02	97.03		82.25	83.87	91.13	91.53	92.34	93.95
Test (%)	90.47	90.47	95.23	92.85	95.24	97.62		82.25	93.54	90.32	90.32	91.94	91.94
Total (%)	91.90	92.38	96.67	95.23	96.67	97.14		82.25	85.80	90.97	91.29	92.26	93.55
Rec. Error	0.262	0.12	0.238	0.191	0.24	0.155		0.189	0.115	0.101	0.101	0.111	0.131
	Column 3							Image segmentation					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	10	9	10	9	10	10		10	9	9	10	10	10
m	1.8	3.6	1.8	4.6	1.8	2.2		1.2	2.4	4.8	1.4	4.4	4.8
ε	--	0.15	0.1	0.1	0.3	0.1		--	0.05	0.1	0.5	0.1	0.15
Train (%)	81.85	83.06	89.52	89.92	91.13	91.94		68.72	63.69	71.43	82.52	90.42	94.85
Test (%)	85.48	83.87	90.32	90.32	95.16	93.55		67.97	62.77	68.61	82.9	90.91	93.50
Total (%)	82.58	83.22	89.67	90	91.94	92.26		68.57	63.50	70.87	82.59	90.52	94.58
Rec. Error	0.202	0.113	0.112	0.123	0.122	0.118		0.105	0.079	0.091	0.160	0.087	0.091
	Wine							Infant cry					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	8	7	7	6	7	5		7	7	7	7	8	7
m	5	2.2	1.8	2.6	3	3.6		1.2	1.4	1.2	1.4	1.2	2.8
ε	--	0.05	0.45	0.45	0.15	0.4		--	0.05	0.25	0.4	0.25	0.15
Train (%)	77.46	77.46	95.07	91.55	91.55	97.89		91.81	91.81	96.84	96.49	96.16	97.01
Test (%)	72.22	75	94.44	88.89	88.89	94.44		93.66	93.22	95.02	96.15	94.57	96.38
Total (%)	76.40	76.97	94.94	91.01	91.01	97.19		92.17	92.09	96.47	96.42	95.84	96.88
Rec. Error	0.416	0.27	0.434	0.498	0.341	0.414		0.331	0.27	0.406	0.446	0.433	0.315
	Infant cry 5 PCs							Infant cry 35 Pcs					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	7	7	7	8	7	6		7	7	6	7	6	5
m	1.4	1.4	1.8	1.6	2.6	2.8		1.2	1.2	2.8	2.4	2.8	2.8
ε	0.2	0.05	0.1	0.1	0.25	0.35		--	0.05	0.05	0.1	0.05	0.1
Train (%)	90	90.16	94.69	94.86	96.77	97.06		90.56	89.1	95.48	95.02	95.48	96.67
Test (%)	92.53	90.95	95.48	95.02	95.24	95.02		92.76	89.82	95.70	94.34	95.7	94.79
Total (%)	90.51	90.35	94.85	94.89	96.47	96.65		91	89.24	95.52	94.89	95.52	96.29
Rec. Error	0.062	0.034	0.043	0.043	0.053	0.058		0.149	0.142	0.142	0.146	0.142	0.144

The best overall results from each protocol show a similar behavior to the previous result obtained with the synthetic dataset; the classification rate achieved by FCM is enhanced by the optimized prototypes. The only protocol that obtained a lower result in some cases was the first one, however it managed to cluster and classify the training and testing data without losing too much performance. It is clear from these results that protocol 5, which has the highest parametric flexibility, reached the best general performance, and only protocol 4 had similar results. Perhaps the most notable results the ones obtained with the image segmentation and wine datasets, where in the former the classification rate increased from 68.57% to 94.58%, while the latter increased the rate from 76.96% to 97.19%. The wine dataset in particular, has separable classes, but only one classifier has achieved a 100% accuracy, namely the Regularized Discrimination Analysis (RDA), moreover the 1-Nearest-Neighbor (1NN) achieved 96.1% accuracy by using z-transformed data; these results were obtained by using the leave-one-out technique [7].

Another comparable result is obtained when using the infant cry dataset, and in this case the experiments were performed in a different fashion for comparative purposes. The results reported in [135] and [136] were obtained by using a feed-forward time delay neural network (FFTD) to classify the infant cry samples. In these experiments, since the dataset has 340 samples from asphyxiating babies, only 340 samples were randomly selected from the remaining two classes. Furthermore, the experiments compared the results obtained by using the 144 input features against results obtained by reducing these features with principal component analysis (PCA) to 50 and 35 PCs, and by optimally selecting 50 features with *evolutionary strategies* [42], [55]. The results reported a classification accuracy of 93.33% for data reduced to 50 PCs, 96.40% for data with no reduction, 96.79% for the optimal selection of 50 features, and 97.39% for the data reduced to 35 PCs. In our experiments, we decided to use all the 2268 samples, instead of randomly selecting samples from each class. We experimented with 144 features, and also reduced the input vectors to 35 PCs, and 5 PCs to compare the results, and test the robustness of the refined clusters.

Overall, the reconstruction error is smaller in almost all the results, in some experiments this error increased or remained similar to the one obtained with FCM. Nevertheless the classification rate is higher for the protocols that optimized either the direction or level of change (or both), and in all experiments protocol 5 has a lower reconstruction error, meaning that the parametric flexibility included in this protocol optimizes the quality of the clusters and their reconstruction and classification abilities.

Table 7.4. Comparison of the original prototypes found by the FCM algorithm, their new position after the optimization process, and the minimum and maximum values for the thyroid dataset, for $c = 3$, and $\varepsilon = 0.1$.

	Initial prototypes				
c/n	**1**	**2**	**3**	**4**	**5**
1	88.197	**16.840**	**4.403**	1.601	1.0451
2	109.148	9.457	1.728	1.630	2.669
3	**125.463**	7.875	1.742	5.460	7.067
	Optimized prototypes				
1	120.074	**16.885**	**4.673**	0.1	-0.7
2	119.633	1.961	3.621	12.549	6.104
3	**120.426**	15.214	0.2	0.1	1.18918
	Minimum and Maximum values				
Min	65	0.5	0.2	0.1	-0.7
Max	144	25.3	10	56.4	56.3

In general, the optimized clusters were able to find data that the FCM algorithm could not cluster into its corresponding class. This behavior was observed in every experiment executed that used DE; the results always showed an increase in the clustering performance after the optimization. In some cases, especially for protocols 4, and 5, we observed that some prototypes for a given variable j, almost remained in their original position, this allowed other the prototypes to have a larger level of change, due to the asymmetrical distribution implemented in these protocols. This behavior can be seen in the prototype coordinates found by protocol 5 included in Table 7.4 (highlighted in bold fonts). This indicates that the original prototypes already have a good position in a certain dimension, and partially remain in their place to avoid loss of correctly clustered data. At the end the results shown in this section seem to indicate that the approach presented is of practical importance for clustering applications.

By plotting the overall results for selected datasets, we see a similar behavior than with synthetic data, where the classification rate increases along with the number of clusters. In Figure 7.7(*a*) this behavior is more notable for the seed dataset, moreover, it the difference between the overall results from the original prototypes is significant when compared with the refined prototypes from protocol 5. This significant variation is also seen for the wine dataset, as depicted in Figure 7.7(*b*), nonetheless, in this particular case, the protocol 4 had a lower overall performance. If we see how the level of change affected the results, for the wine dataset, these did not affect much the results when ε reached the value of 0.1. On the other hand, a higher level of change had a significant impact to the final result, since it required values of around 0.35 to 0.5 to obtain higher classification rates. These behaviors are reflected in the results from Table 7.4.

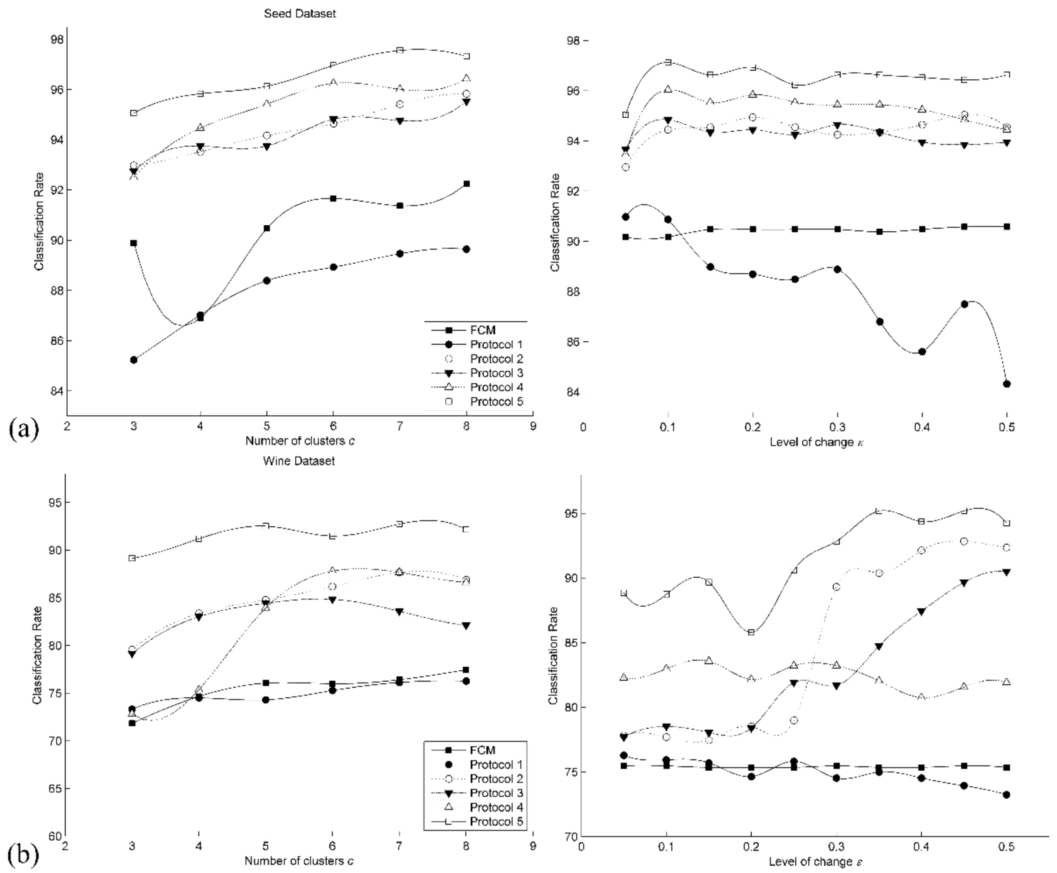


Figure 7.7. Overall classification rates with all the protocols for (*a*) the seed dataset, and (*b*) the wine dataset.

Lastly, to assess the quality of the clusters, the reconstruction criterion was used. Recalling that the quality of a cluster is measured by looking at the smallest distance between the original and the reconstructed data; it will point to the best combination of number of clusters and fuzzification coefficient. For the following experiments, the reconstruction error is used as the objective function, and at the end we report the classification rate. The results included in Table 5 only report the minimum reconstruction error found by DE along with the classification rate.

From the results in Table 7.5 it is clear that the DE algorithm had a different objective than the previous version, which is to find the best number of clusters and fuzzification factor to reconstruct the data. Similar to the results from Table 7.3, protocol 5 shows the lowest reconstruction error in most results, and in some the error remained the same as by using FCM.

Nevertheless, it is clear that the classification rate now it is lower, meaning that there is a tradeoff between having a good reconstruction of the original data, and correctly clustering each class. Moreover, in these experiments, if we find the highest classification rate, the reconstruction error also increases, but these results do not resemble the results from Table 7.3. At the end this situation becomes a multi-objective problem, where we need to find a balance between having the smallest reconstruction and classification errors to obtain clusters with the highest performance.

Table 7.5. Reconstruction errors obtained with the initial prototypes (FCM) and with all the protocols for all the datasets. The minimum reconstruction error and classification rate for all sets are shown.

	Iris							Thyroid					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	8	7	8	8	8	8		8	8	8	8	8	8
m	1.6	2.6	3	2.8	3	2.8		1.4	1.2	1.6	1.6	2.8	3
ε	--	0.05	0.3	0.25	0.15	0.4		--	0.05	0.05	0.1	0.25	0.3
Train (%)	96.67	89.17	86.67	93.33	90	90.83		84.88	83.13	83.72	84.88	90.12	86.05
Test (%)	100	90	93.33	93.33	100	100		93.02	88.37	93.02	95.34	90.69	90.7
Total (%)	97.33	89.33	88	93.33	92	92.67		86.51	84.19	85.58	86.98	90.23	86.98
Rec. Error	0.062	0.067	0.059	0.059	0.05	0.047		0.081	0.095	0.081	0.081	0.077	0.071
Dataset	Seed							Column 2					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	8	8	8	8	8	8		11	11	11	11	11	9
m	1.6	1.4	2.2	2.4	1.8	2.4		1.2	1.4	4.2	3.2	2.4	3.4
ε	--	0.05	0.05	0.05	0.05	0.05		--	0.05	0.4	0.4	0.3	0.4
Train (%)	89.88	90.48	87.5	90.48	91.07	91.67		78.22	79.03	80.24	76.03	80.24	80.24
Test (%)	90.48	88.1	88.1	88.1	88.1	90.48		82.26	80.64	88.71	85.48	88.70	85.48
Total (%)	90	90	87.61	90	90.47	91.43		79.03	79.35	81.94	78.06	81.94	81.29
Rec. Error	0.083	0.09	0.082	0.083	0.082	0.08		0.08	0.089	0.077	0.075	0.065	0.06
Dataset	Column 3							Image segmentation					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	11	11	10	11	11	11		10	10	9	10	10	10
m	1.2	1.4	5	4	2.4	2.8		1.6	1.4	1.2	2.2	1.6	2
ε	--	0.05	0.4	0.5	0.2	0.25		--	0.05	0.05	0.05	0.05	0.05
Train (%)	78.62	79.83	61.29	72.17	74.19	74.19		67.80	60.5	65.09	66.23	67.31	71
Test (%)	85.48	88.71	66.13	77.41	90.32	79.03		67.53	63.64	62.98	66.88	65.58	69.7
Total (%)	80	81.61	62.26	73.23	77.41	75.16		67.74	61.13	64.67	66.36	66.97	70.74
Rec. Error	0.08	0.092	0.08	0.073	0.065	0.057		0.065	0.073	0.065	0.065	0.065	0.064
	Wine							Infant cry					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	7	8	7	7	6	6		8	8	8	8	8	8
m	1.4	1.6	2.4	2.4	3	1.6		1.2	1.2	1.4	1.4	1.4	1.4
ε	--	0.05	0.05	0.05	0.05	0.05		--	0.05	0.05	0.05	0.05	0.05
Train (%)	73.94	75.35	75.35	76.06	71.12	75.35		91.75	91.58	90.96	89.83	90	90.90
Test (%)	72.22	77.78	72.22	72.22	77.77	69.44		93.66	91.86	91.62	92.31	91.62	92.08
Total (%)	78.09	75.84	74.72	75.28	72.47	74.15		92.13	91.64	91.09	90.33	90.35	91.13
Rec. Error	0.251	0.264	0.251	0.251	0.251	0.251		0.247	0.268	0.247	0.247	0.247	0.247
	Infant cry 5 PCs							Infant cry 35 Pcs					
Method	FCM	P1	P2	P3	P4	P5		FCM	P1	P2	P3	P4	P5
c	8	8	8	7	8	8		7	8	8	8	8	8
m	1.2	1.8	2.6	2.6	2.8	3		1.2	1.4	1.4	1.6	1.6	1.4
ε	--	0.05	0.5	0.5	0.25	0.4		--	0.05	0.05	0.05	0.05	0.05
Train (%)	91.36	88.64	67.96	82.48	81.07	82.99		90.56	79.83	83.44	83.72	79.15	85.03
Test (%)	92.08	88.01	69	81.9	82.58	82.35		92.76	81.44	85.74	86.87	84.38	86.42
Total (%)	91.5	88.52	68.17	82.37	81.37	82.87		91	80.15	83.91	84.35	80.19	85.30
Rec. Error	0.03	0.034	0.028	0.028	0.026	0.024		0.133	0.139	0.133	0.133	0.133	0.133

7.4. Summary

In this Chapter, a clustering optimization algorithm was presented, where the idea is to relocate the position of the cluster centers (prototypes) initially determined by running the FCM algorithm on a dataset. The strategy pursued here is to determine how many labeled data points were correctly classified (clustered) to obtain a classification rate. Alternatively, the reconstruction criterion is implemented to decode the original data, and obtain a reconstruction error. Then the DE algorithm was used to modify and refine the prototypes by using the classification rate or reconstruction criterion as its fitness function. The prototypes are optimized by following five different strategies (protocols), which help strategically re-allocate their *n*-dimensional coordinates, to modify the clusters. Once an optimal performance is found, the information granules are tested with novel data to assess their generalization capabilities. The strategy was tested with synthetic and real-world data coming from publicly available repositories. An enhancement of the performance occurred in almost all experiments, especially when using protocol 5. The improvement was observed irrespectively of the values of the fuzzification factor, number of clusters, and the intensity of modifications of the prototypes. Moreover, the results exhibit good generalization capabilities without any overfitting effects. The experiments show that this strategy is beneficial to improve the clustering performance, both with regard to the classification and reconstruction abilities.

8. Conclusions and Future Studies

Granular fuzzy models offer a new and attractive venue for system modeling by building models at the level of information granules. These models are aimed on building clusters that are able to augment the homogeneity of the clustered patterns associated with the input and output space. To optimize these associations, we tested and implemented several strategies to introduce a directionality component into the granular constructs. Most of these optimizations were carried out by the Differential Evolution (DE) algorithm, due to its robustness, performance, and implementation flexibility. Nevertheless, finding the best optimization algorithm is always problem dependent. For the fuzzy relational models, DE combined with the gradient-based method (GB) exhibited the best performance, for both training and testing sets. As stand-alone optimization algorithms, the best performance was observed when DE was utilized, followed by GB and PSO. The abstraction (granulation) of the fuzzy model indicated that it achieved good approximations, since it required small levels of granularity to cover the entire input space. Moreover, the experimental results indicated that the protocol 5 (Section 3.3.1) displayed the best performance, since it required a low level of granularity (specificity) to cover the entire output space. These experiments made us adhere to this protocol to help in the construction of other granular models.

We also demonstrated that granular fuzzy models offer a new and attractive venue for system modeling. To build them, a numerically inclined optimization was carried out to construct a collection of intuitively appealing associations among information granules. The experimental studies demonstrated that the optimization of the interval-based information granules (positioned in the output space) is crucial as all input clusters are induced by them. Moreover, granular fuzzy models represented as hyperboxes helped describe the essence of the data, and the information captured by the information granules reflected their topology, which is similar to fuzzy clusters. The hyperboxes formed by the numerical constraint showed a poor coverage performance, which was further degraded after the overlap elimination took place. In contrast, the optimized formation of the hyperboxes, along with the optimal overlap reduction, offered better comparable results. Furthermore, the three-step coverage criteria helped us assess the clustering abilities of the algorithm, and gave us an insight of the data being captured by the information granules. On the other hand, it was demonstrated that by carefully migrating the position of the prototypes, the clustering performance of Fuzzy C-Means (FCM) was improved. This improvement was also observed with regard to the classification and reconstruction abilities of the information granules. Moreover, the quality of the clusters also increased, given that the reconstruction error decreased even when it was not included as part of the fitness function. In contrast, if the reconstruction error is used as a fitness function it is considerably minimized, but the toll we had to pay is a reduction on the classification rate. Overall, the methods proposed in this dissertation show some interesting highlights,

- The granular models produce granular results instead of numeric entities, this reflected the variability in the data.
- The proposed methods captured the diversity of the data and handled the lack of numerical precision, which makes the granular results more reflective of reality.
- The granular fuzzy relational models helped us assess the performance of fuzzy relational models, and resulted on an alternative solution to the system of fuzzy relational equations.
- The parametric flexibility of the protocols combined with DE optimized the allocation of the granularity in the data, and also provided the best coverage with small (specific) levels of granularity.
- The number of hyperboxes can be selected by the end-user, instead of bounding their size to create multiple clusters that could cover only one datum and make the system more complex.
- The optimization of the overlap reduction helped us preserve the covered data as much as possible, while retaining a few small overlaps instead of totally eliminating them
- The clustering performance, with regard of the classification and reconstruction abilities, can be augmented by using supervised methods.
- The strategies proposed to refine the clusters (coming from FCM) offer an interesting alternative for cluster construction and assessment.

8.1. Possible Limitations of the Research

While the experimental studies displayed a good performance, it has to be taken into account that for each different dataset we would require a new set of experiments to find the best number of information granules and fuzzification factor. Moreover, for the granular fuzzy relational model, we would require to find the best learning algorithms (or combination of these), combination of logic operators, and the most suitable number of hidden layers for the initial FLP. As for the induced information granules, we need to find a suitable number of interval-based information granules. Also, for the construction of the hyperboxes or the migration of the prototypes, we need to search for the best level of granularity or level of change respectively. On the other hand, the information granules positioned in the output space are numbers (have hard boundaries), and the inclusion of the data has only a two-valued outcome. Finally, the coverage criterion was intensively exploited in this dissertation, and the specificity criterion was used independently in some cases, this could cause a detrimental effect on the information granules being developed and optimized. To solve this issue, we can turn to multi-objective optimization or a bi-criteria scenario, where coverage and specificity (or other design factors) can be considered in the objective function.

8.2. Future Research Directions

First and foremost, finding the best values of the parameters for the clustering mechanisms is crucial when building the granular fuzzy models. A method to improve this search could be an interesting research topic; an algorithm that could optimize all the initial parameters and at the same time construct and optimized granular model. However, we have to consider that when using individuals (or particles) with high-dimensionality in population-based algorithms, these suffer from premature convergence, low optimization precision, large computational overhead, and even failure. In the case of the fuzzy relational structures, there is the need of finding better representation of the output signal to improve the performance of the model. An idea that could be further investigated is to cluster the output data, and use the prototypes as modal values to construct triangular membership functions, and the adjacent fuzzy sets would overlap at the 0.5 level. This would also result on a zero degranulation error.

For the granular models, we would need to thoroughly investigate a bi-criteria optimization scenario involving both coverage and specificity. Moreover, another bi-criteria scenario involving the re-distribution of induced information granules along with the optimization of the output intervals could benefit the optimization of the granular fuzzy model and the formation of the hyperboxes. There are two other interesting venues of further study directly linked with the undertaken research reported here. The topology of the proposed granular models is general – further realizations including other formalisms of information granules (rough sets, probabilistic sets) are worth exploring. In case of the interval-valued information granules, the assessment of the quality of the model can be pursued by admitting further generalizations of the output intervals and making them granular intervals (viz. intervals whose bounds are not numbers but information granules themselves). As for the hyperboxes, an interest is to further investigate their shape and size. A motivating idea would be to conclude the algorithm by shaping the limits of the hyperboxes to the minimum and maximum values of the data captured inside the hyperbox, this in theory would eliminate empty regions that the information granule is covering. Moreover, it would be of interest use the clustering optimization strategies as a first-phase processor for the construction of granular models shown in this study, and observe if their performance is enhanced.

Bibliography

[1] J. H. Abawajy, A. V. Kelarev, and M. Chowdhury, “Multistage approach for clustering and classification of ECG data,” *Computer Methods and Programs in Biomedicine*, Vol. 112, No. 3, Pp. 720-730, Dec. 2013. DOI: 10.1016/j.cmpb.2013.08.002.

[2] S. Abe, M.-S. Lan, and R. Thawonmas, “Tuning of a fuzzy classifier derived from data,” Fuzzy Systems, 1994. *IEEE World Congress on Computational Intelligence.*, *Proceedings of the Third IEEE Conference on*, Vol. 2, Pp. 786-791, Jun 1994. DOI: 10.1016/ 0888-613X(95)00076-S.

[3] S. Abe and M.-S. Lan, “A classifier using fuzzy rules extracted directly from numerical data,” *Fuzzy Systems, 1993, Second IEEE International Conference on*, Pp.1191-1198 Vol.2, 1993. DOI: 10.1109/FUZZY.1993.327561.

[4] S. Abe and M.-S. Lan, “A function approximator using fuzzy rules extracted directly from numerical data,” *Neural Networks. IJCNN ‘93-Nagoya. Proceedings of 1993 International Joint Conference on*, Vol. 2, Pp.1887-1892, Oct 1993, DOI: 10.1109/IJCNN.1993.717024.

[5] S. Abe and M.-S. Lan, “A method for fuzzy rules extraction directly from numerical data and its application to pattern classification,” *Fuzzy Systems, IEEE Transactions on*, Vol. 3, No. 1, Pp. 18-28, Feb 1995. DOI: 10.1109/91.366565.

[6] J. Abonyi, R. Babuska, and F. Szeifert, “Modified Gath-Geva fuzzy clustering for identification of Takagi-Sugeno fuzzy models,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol. 32, No. 5, Pp. 612-621, Oct. 2002. DOI: 10.1109/TSMCB.2002.1033180.

[7] S. Aeberhard, D. Coomans, and O. de Vel, ”Comparison of Classifiers in High Dimensional Settings,” *Technical Report No. 92-02*, Dept. of computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland, 1992.

[8] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, “KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework.” *Journal of Multiple-Valued Logic and Soft Computing*, Vol. 17, No. 2-3, Pp. 255-287, 2010.

[9] T. F. Araújo and W. Uturbey, “Performance assessment of PSO, DE and hybrid PSO–DE algorithms when applied to the dispatch of generation and demand,” *International Journal of Electrical Power & Energy Systems*, Vol. 47, May 2013, Pp. 205-217.

[10] K. Bache and M. Lichman, *UCI Machine Learning Repository* [Online]. Available at: http://archive.ics.uci.edu/ml, accessed on October, 2015.

[11] P. Baraldi, R. Razavi-Far, and E. Zio, "Bagged ensemble of Fuzzy C-Means classifiers for nuclear transient identification," Annals of Nuclear Energy, Vol. 38, Issue 5, pp. 1161-1171, May 2011, DOI: 10.1016/j.anucene.2010.12.009.

[12] P. Baranyi, "The Generalized TP model transformation for TS fuzzy model manipulation and generalized stability verification," *IEEE Transactions on Fuzzy Systems*, Vol. PP, No. 99, Pp. 1-1, 2013. DOI: 10.1109/TFUZZ.2013.2278982.

[13] A. Bargiela, and W. Pedrycz, "Granular clustering with partial supervision - the development of abstract models for simulation," *Proceedings of the European Simulation Multiconference* (ESM 2001), Prague June 2001.

[14] A. Bargiela and W. Pedrycz, and Tanaka, M., "Exclusion/Inclusion Fuzzy Classification Network," *Knowledge-Based Intelligent Information and Engineering Systems*, Lecture Notes in Computer Science (LNCS). Vol.2773, Pp.1236-1241, Springer Berlin Heidelberg 2003. DOI: 10.1007/978-3-540-45224-9_167.

[15] A. Bargiela, W. Pedrycz, and M. Tanaka, "An inclusion/exclusion fuzzy hyperbox classifier," *International Journal on Knowledge.-Based Intelligent Engineering Systems*, Vol. 8, No. 2, Pp.91-98, April 2004.

[16] A. Bargiela, M. Tanaka, G. Castellano, and A.-M. Fanelli, "Adaptation of exclusion/inclusion hyperboxes for classification of complex data," *Proceedings of the 5th Asia Pacific Industrial Engineering and Management Systems Conference*, Brisbane, Pp. 17.7.1-17.7.12, December, 2004.

[17] J .C. Bezdek, "Cluster validity with fuzzy sets," *Journal of Cybernetics*, Vol. 3, Iss. 3, Pp. 58–72, 1973. DOI: 10.1080/01969727308546047.

[18] J. C. Bezdek, "Mathematical models for systematics and taxonomy," *Procceedings of the 8th International Conference on Numerical Taxonomy*, Freeman, San Francisco, CA, Pp. 143–166, 1975.

[19] J. C. Bezdek "Pattern recognition with fuzzy objective function algorithms." *Springer-Verlag US*, 1981.

[20] J. C. Bezdek, R. Erhlich, W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geoscience* 10, Pp. 191–203, 1984.

[21] W. Cai, S. Chen, and D. Zhang, “A Multiobjective Simultaneous Learning Framework for Clustering and Classification,” *Neural Networks, IEEE Transactions on*, Vol. 21, No. 2, Pp. 185-200, Feb. 2010. DOI: 10.1109/TNN.2009.2034741.

[22] O. Castillo, P. Melin, and w. Pedrycz, “Design of interval type-2 fuzzy models through optimal granularity allocation,” *Applied Soft Computing*, Vol. 11, Iss. 8, Pp. 5590-5601, 2011.

[23] J. Casillas, O. Cordon, M. J. Del Jesus, and F. Herrera, “Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction,” *IEEE Transactions on Fuzzy Systems*, Vol. 13, No. 1, Pp. 13-29, Feb. 2005. DOI: 10.1109/TFUZZ.2004.839670.

[24] M. Charytanowicz, J. Niewczas, P. Kulczycki, P.A. Kowalski, S. Lukasik, and S. Zak, “A Complete Gradient Clustering Algorithm for Features Analysis of X-ray Images,” *Information Technologies in Biomedicine*, Eds. E. Pietka, J. Kawa, Springer-Verlag, Berlin-Heidelberg, Pp. 15-24, 2010. DOI: 10.1007/978-3-642-13105-9_2.

[25] O. Cordon and F. Herrera, “A proposal for improving the accuracy of linguistic modeling,” *IEEE Transactions on Fuzzy Systems*, Vol. 8, No. 3, Pp. 335-344, Jun. 2000. DOI: 10.1109/91.855921.

[26] O. Cordon, F. Herrera, and I. Zwir, “Linguistic modeling by hierarchical systems of linguistic rules,” *IEEE Transactions on Fuzzy Systems*, Vol. 10, No. 1, Pp. 2-20, Feb. 2002. DOI: 10.1109/91.983275.

[27] O. P. Dahal, S. M. Brahma, and H. Cao, “Comprehensive Clustering of Disturbance Events Recorded by Phasor Measurement Units,” *Power Delivery, IEEE Transactions on*, Vol. 29, No. 3, Pp. 1390-1397, Jun. 2014. DOI: 10.1109/TPWRD.2013.2285097.

[28] B.-R. Dai, J.-W. Huang, M.-Y. Yeh, and M.-S. Chen, “Adaptive Clustering for Multiple Evolving Streams,” *Knowledge and Data Engineering, IEEE Transactions on*, Vol. 18, No. 9, Pp. 1166-1180, Sep. 2006. DOI: 10.1109/TKDE.2006.137.

[29] R. Davtalab, M. H. Dezfoulian, and M. Mansoorizadeh, “Multi-Level Fuzzy Min-Max Neural Network Classifier,” *Neural Networks and Learning Systems, IEEE Transactions on*, Vol. 25, No. 3, Pp. 470-482, Mar. 2014. DOI: 10.1109/TNNLS.2013.2275937.

[30] Z. Deng, Y. Jiang, K.-S. Choi, F.-L. Chung, and S. Wang, "Knowledge-Leverage-Based TSK Fuzzy System Modeling," *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 24, No. 8, Pp. 1200-1212, Aug. 2013. DOI: 10.1109/TNNLS.2013.2253617.

[31] D. Dey, B. Chatterjee, S. Chakravorti, and S. Munshi, "Rough-granular approach for impulse fault classification of transformers using cross-wavelet transform," *Dielectrics and Electrical Insulation, IEEE Transactions on*, Vol. 15, No. 5, Pp. 1297-1304, Oct. 2008. DOI: 10.1109/TDEI.2008.4656237.

[32] A. Di Nola, "Relational equations in totally ordered lattices and their complete resolution," *Journal of Mathematical Analysis and Applications*, Vol. 107, Issue 1, 1985, Pp. 148-155.

[33] A. Di Nola and W. Pedrycz, "Entropy and energy measure characterization of resolution of some fuzzy relational equations," *Busefal 10*, 1982, Pp. 44-53.

[34] A. Di Nola, W. Pedrycz, and S. Sessa, "On measures of fuzziness of solutions of fuzzy relation equations with generalized connectives," *Journal of Mathematical Analysis and Applications,* Vol. 106, Issue 2, 1985, Pp. 443-453.

[35] A. Di Nola, W. Pedrycz, and S. Sessa, and W. Pei-Zhuan, "Fuzzy Relation Equation under a Class of Triangular Norms: A survey and new results," *Stochastica*, Vol. 3, Issue 2, 1984, Pp. 99-145.

[36] A. Di Nola and S. Sessa, "On the set of solutions of composite fuzzy relation equations," *Fuzzy Sets and Systems*, Vol. 9, Issues 1–3, 1983, Pp. 275-285.

[37] A. Di Nola and S. Sessa, "Finite fuzzy relation equations with a unique solution in linear lattices," *Journal of Mathematical Analysis and Applications*, Vol. 132, Issue 1, 1988, Pp. 39-49.

[38] Y. Dong and J. Ma, "Feature extraction through contourlet subband clustering for texture classification," *Neurocomputing*, Vol. 116, No. 20, Pp. 157-164, September 2013. DOI: 10.1016/j.neucom.2011.12.059.

[39] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern Classification," 2nd Edition, *John Wiley & Sons*, 2012. ISBN: 978-0-471-05669-0 .

[40] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *Journal of Cybernetics*, vol. 3, no.3, pp. 32-57, 1974.

[41] M. Eastwood and C. Jayne, “Evaluation of hyperbox neural network learning for classification,” *Neurocomputing*, Vol. 133, No. 10, Pp. 249-257, June 2014. DOI: 10.1016/j.neucom.2013.11.011.

[42] A. P. Engelbrecht, “Computational Intelligence: An Introduction,” 2nd ed. London, UK: *Wiley & Sons*, 2007.

[43] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annual Eugenics*, 7, Part II, Pp. 179-188, 1936. Also in *Contributions to Mathematical Statistics*, John Wiley, NY, 1950.

[44] D. B. Fogel and P. K. Simpson, “Evolving fuzzy clusters,” *Neural Networks, 1993., IEEE International Conference on*, Vol.3, Pp. 1829-1834, 1993. DOI: 10.1109/ICNN.1993.298835.

[45] B. Gabrys and A. Bargiela, “General fuzzy min-max neural network for clustering and classification,” *Neural Networks, IEEE Transactions on*, Vol. 11, No. 3, 769-783, May 2000. DOI: 10.1109/72.846747.

[46] A. Gacek, and W. Pedrycz, “A characterization of electrocardiogram signals through optimal allocation of information granularity,” *Artificial Intelligence in Medicine*, Vol. 54, Iss. 2, Pp. 125-134, 2012.

[47] W. Gau, and D. Buehrer, “Vague Sets,” *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, Iss. 2. Pp. 610-614, 1993.

[48] S. Gottwald and W. Pedrycz “Solvability of fuzzy relational equations and manipulation of fuzzy data,” *Fuzzy Sets and Systems*, Vol. 18, Issue 1, 1986, Pp. 45-65.

[49] D. Graves, W. Pedrycz, "Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study," *Fuzzy Sets and Systems*, vol. 161, issue 4, pp. 522-543, 16 February 2010, ISSN 0165-0114, DOI: 10.1016/j.fss.2009.10.021.

[50] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. “The WEKA data mining software: an update,” *ACM SIGKDD Explorations Newsletter*, Vol. 11, No.1, Pp. 10-18. 2009. DOI: 10.1145/1656274.1656278.

[51] Y.H. Han and K.C. Kwak, “An Optimization of Granular Network by Evolutionary Methods,” in *9th WSEAS International Conference on Artificial Intelligence*, Pp. 65-70. 2010.

[52] H. He, Y. Tan, Hong He, and Yonghong Tan, “A two-stage genetic algorithm for automatic clustering,” *Neurocomputing*, Vol. 81, No. 1, Pp. 49-59, April 2012. DOI: 10.1016/j.neucom.2011.11.001.

[53] M. Higashi and G. J. Klir, “Resolution of finite fuzzy relation equations,” *Fuzzy Sets and Systems*, Vol. 13, Issue 1, 1984, Pp. 65-82.

[54] P. W. Holland, and R. E. Welsch, “Robust regression using iteratively reweighted least-squares,” *Communications in Statistics*, Vol. A6, No. 9, Pp. 813-827, 1977.

[55] T.S. Hussain, "An Introduction to Evolutionary Computation," Department of Computing and Information Science Queens University, Kingston, Ont. K7L3N6. 1998.

[56] H. Ichihashi and K. Honda, “Fuzzy c-Means Classifier for Incomplete Data Sets with Outliers and Missing Values,” *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, Vol. 2, Pp.457,464, 2005. DOI: 10.1109/CIMCA.2005.1631511.

[57] H. Ichihashi, K. Honda, and A. Notsu, "Postsupervised Hard c-Means Classifier," Rough Sets and Current Trends in Computing, LNCS 4259, pp. 918-927, Springer Berlin Heidelberg 2006 DOI: 10.1007/11908029_95.

[58] H. Izakian and W. Pedrycz, "Anomaly detection in time series data using a fuzzy c-means clustering," *IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS), Joint Conference*, pp.1513-1518, 24-28 June 2013 DOI: 10.1109/IFSA-NAFIPS.2013.6608627.

[59] H. Izakian and W. Pedrycz, "Anomaly Detection and Characterization in Spatial Time Series Data: A Cluster-Centric Approach," *IEEE Transactions on in Fuzzy Systems*, vol. 22, no. 6, pp. 1612-1624, Dec. 2014 DOI: 10.1109/TFUZZ.2014.2302456.

[60] H. Izakian, W. Pedrycz, and I. Jamal, "Clustering Spatiotemporal Data: An Augmented Fuzzy C-Means," *IEEE Transactions on in Fuzzy Systems*, vol.21, no.5, pp.855-868, Oct. 2013 DOI: 10.1109/TFUZZ.2012.2233479.

[61] J.-S. R. Jang, “ANFIS: adaptive-network-based fuzzy inference system,” *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, No. 3, Pp. 665-685, May/Jun. 1993. DOI: 10.1109/21.256541.

[62] N. Jardine and R. Sibson, “Mathematical Taxonomy,” *John Wiley and Sons*, 1971.

[63] C.-F. Juang and C.-Y. Chen, “An Interval Type-2 Neural Fuzzy Chip with On-Chip Incremental Learning Ability for Time-Varying Data Sequence Prediction and System Control,” *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 25, No. 1, Pp. 216-228, Jan. 2014. DOI: 10.1109/TNNLS.2013.2253799.

[64] V. G. Kaburlasos, “Granular Fuzzy Inference System (FIS) Design by Lattice Computing,” *Hybrid Artificial Intelligence Systems 2010*, LNCS Vol. 6077, Pp. 410-417, DOI: 10.1007/978-3-642-13803-4_51.

[65] V. G. Kaburlasos and S. E. Papadakis, “A granular extension of the fuzzy-ARTMAP (FAM) neural classifier based on fuzzy lattice reasoning (FLR),” *Neurocomputing,* 2009, Vol. 72, No. 10–12, Pp. 2067-2078, DOI: 10.1016/j.neucom.2008.06.024.

[66] V. G. Kaburlasos, G. A. Papakostas, T. Pachidis, and A. Athinellis, “Intervals’ Numbers (INs) interpolation/extrapolation,” *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*, Pp. 1-8, DOI: 10.1109/FUZZ-IEEE.2013.6622318.

[67] N. K. Kasabov and Q. Song, “DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction,” *IEEE Transactions On Fuzzy Systems*, Vol. 10, No. 2, Pp. 144-154, Apr. 2002. DOI: 10.1109/91.995117.

[68] A. H. Kashan, B. Rezaee, and S. Karimiyan, “An efficient approach for unsupervised fuzzy clustering based on grouping evolution strategies,” *Pattern Recognition*, Vol. 46, Issue 5, May 2013, Pages 1240-1254.

[69] R. B. Kearfott, “Interval computations: Introduction, uses, and resources,” *Euromath Bulletin*, Vol. 2, No. 1, Pp. 95–112. 1996.

[70] J. Kennedy, R. Eberhart, and Y. H. Shi, “Swarm Intelligence,” *Morgan Kaufman Publishers*, 2001.

[71] G. J. Klir and B. Yuan “Fuzzy Sets and Fuzzy Logic: Theory and Applications,” *Prentice Hall PTR*, New Jersey, 1995.

[72] R. Krishnapuram and J. M. Keller, “A possibilistic approach to clustering,” *Fuzzy Systems, IEEE Transactions on*, Vol. 1, No. 2, Pp. 98-110, 1993. DOI: 10.1109/91.227387.

[73] K. Kummamuru, A. Dhawale, and R. Krishnapuram, “Fuzzy co-clustering of documents and keywords,” *Fuzzy Systems*, 2003. FUZZ '03. The 12th IEEE International Conference on, Vol. 2, Pp. 772-777, 2003. DOI: 10.1109/FUZZ.2003.1206527.

[74] K. C. Kwak, “An Optimization of Granular Networks Based on PSO and Two-Sided Gaussian Contexts,” *(IJARAI) International Journal on Advance Research in Artificial Intelligence*, Vol. 1, No. 9, 2012. DOI: 10.14569/IJARAI.2012.010901.

[75] K.C. Kwak and W. Pedrycz, “A Design of Genetically Oriented Linguistic Model with the Aid of Fuzzy Granulation,” *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, Pp. 1-6, 18-23 Jul. 2010. DOI: 10.1109/FUZZY.2010.5584357.

[76] P. Ładyżyński and P. Grzegorzewski, “Particle swarm intelligence tunning of fuzzy geometric protoforms for price patterns recognition and stock trading,” *Expert Systems with Applications*, Vol. 40, Issue 7, 1 June 2013, Pp. 2391-2397.

[77] D.T.C. Lai and J.M. Garibaldi, "Improving semi-supervised fuzzy c-means classification of Breast Cancer data using feature selection," 2013 IEEE International Conference on Fuzzy Systems (FUZZ), pp. 1-8, July 2013, DOI: 10.1109/FUZZ-IEEE.2013.6622544.

[78] W. C. Lemos and F. Gomide, “Multivariable Gaussian Evolving Fuzzy Modeling System,” *IEEE Transactions On Fuzzy Systems*, Vol. 19, No. 1, Pp. 91-104, Feb. 2011. DOI: 10.1109/TFUZZ.2010.2087381.

[79] G. Z. Li and S. C. Fang, “Solving Interval-Valued Fuzzy Relation Equations,” *IEEE Transactions on Fuzzy Systems*, Vol. 6 Iss.2, pp. 321-324, 1998.

[80] P. Li and S. C. Fang, “A note on solution sets of interval-valued fuzzy relational equations,” *Mathematics and Statistics: Fuzzy Optimization and Decision Making*, Springer Netherlands Vol. 8 Issue 1 Pp. 115-121, 2009.

[81] J. Z. Li, B. Fu, P. Kou, and J. Xiao, “T–S Fuzzy Model Identification With a Gravitational Search-Based Hyperplane Clustering Algorithm,” *IEEE Transactions On Fuzzy Systems,* Vol. 20, No. 2, Pp. 305-317, Apr. 2012. DOI: 10.1109/TFUZZ.2011.2173693.

[82] R. Liu, Y. Chen, L. Jiao, and Y. Li, “A particle swarm optimization based simultaneous learning framework for clustering and classification,” *Pattern Recognition*, Vol. 47, No. 6, Pp. 2143-2152, Jun. 2014. DOI: 10.1016/j.patcog.2013.12.010.

[83] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, “Entropy-Rate Clustering: Cluster Analysis via Maximizing a Submodular Function Subject to a Matroid Constraint,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 36, No. 1, Pp. 99-112, Jan. 2014. DOI: 10.1109/TPAMI.2013.107.

[84] C. Lu, X. Hu, and J-R Park, “Exploiting the Social Tagging Network for Web Clustering,” Systems, Man and Cybernetics, *Part A: Systems and Humans, IEEE Transactions on*, Vol. 41, No. 5, Pp. 840-852, Sep. 2011. DOI: 10.1109/TSMCA.2011.2157128.

[85] J. Lu, X. Yuan, and T. Yahagi, “A Method of Face Recognition Based on Fuzzy c-Means Clustering and Associated Sub-NNs,” *IEEE Transactions on Neural Networks*, Vol. 18, No. 1, Pp. 150-160, Jan. 2007. DOI: 10.1109/TNN.2006.884678.

[86] W. Lu, X. Chen, W. Pedrycz, and X. Liu, J. Yang, “Using interval information granules to improve forecasting in fuzzy time series,” *International Journal of Approximate Reasoning*, Vol. 57, Pp. 1-18, February 2015. DOI: 10.1016/j.ijar.2014.11.002.

[87] F. M. F. Mascioli, A. Rizzi, M. Panella, and G. Martinelli, “Clustering with unconstrained hyperboxes,” *Fuzzy Systems Conference Proceedings 1999, IEEE International*, Vol. 2, Pp. 1075-1080, 1999. DOI: 10.1109/FUZZY.1999.793103.

[88] U. Maulik and S. Bandyopadhyay, “Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification,” *Geoscience and Remote Sensing*, IEEE Transactions on, Vol. 41, No. 5, Pp. 1075-1081, May 2003. DOI: 10.1109/TGRS.2003.810924.

[89] U. Maulik, I. Saha, “Modified differential evolution based fuzzy clustering for pixel classification in remote sensing imagery,” *Pattern Recognition*, Vol. 42, Iss. 9, Pp 2135-2149, 2009. DOI: 10.1016/j.patcog.2009.01.011.

[90] P. Melin and O. Castillo, “A review on type-2 fuzzy logic applications in clustering, classification and pattern recognition,” *Applied Soft Computing*, Vol. 21, Pp. 568-577, 2014. DOI: 10.1016/j.asoc.2014.04.017.

[91] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, “Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic,” *Expert Systems with Applications*, Vol. 40, Issue 8, 15 June 2013, Pp. 3196-3206.

[92] K. Menger, “Statistical metrics,” *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 28, 1942, Pp. 535-537.

[93] D. Mercurio, L. Podofillini, E. Zio and V.N. Dang, "Identification and classification of dynamic event tree scenarios via possibilistic clustering: Application to a steam generator tube rupture event," Accident Analysis & Prevention, Vol. 41, Issue 6, pp. 1180-1191, November 2009, DOI: 10.1016/j.aap.2008.08.013.

[94] M. F. Mohammed and C. P. Lim, “An Enhanced Fuzzy Min-Max Neural Network for Pattern Classification,” *Neural Networks and Learning Systems, IEEE Transactions on*, Vol. PP, No. 99, Pp. 1-1, Apr. 2014. DOI: 10.1109/TNNLS.2014.2315214.

[95] A. V. Nandedkar and P. K. Biswas, “A Granular Reflex Fuzzy Min–Max Neural Network for Classification,” *Neural Networks, IEEE Transactions on*, Vol. 20, No. 7, Pp. 1117-1134, Jul. 2009. DOI: 10.1109/TNN.2009.2016419.

[96] D. Nauck, F. Klawonn, and R. Kruse, “Combining neural networks and fuzzy controllers,” *Lecture Notes in Computer Science: Fuzzy Logic in Artificial Intelligence*, Vol. 695, Pp. 35-46, 1993. DOI: 10.1007/3-540-56920-0_6.

[97] D. Nauck and R. Kruse, “A fuzzy neural network learning fuzzy control rules and membership functions by fuzzy error backpropagation,” *Neural Networks, IEEE International Conference on*, Vol. 2, Pp. 1022-1027, 1993. DOI: 10.1109/ICNN.1993.298698.

[98] D. Nauck, R. Kruse, and R. Stellmach, “New Learning Algorithms for the Neuro-Fuzzy Environment NEFCON-I,” *Proceedings of Neuro-Fuzzy-Systems*, Pp. 357-364, 1995.

[99] D. Nauck, U. Nauck, and R. Kruse, “Generating classification rules with the neuro-fuzzy system NEFCLASS,” *Fuzzy Information Processing Society, NAFIPS., Biennial Conference of the North American*, Pp. 466-470, Jun 1996.

[100] C.-H. Oh, K. Honda, and H. Ichihashi, “Fuzzy clustering for categorical multivariate data,” *IFSA World Congress and 20th NAFIPS International Conference*, 2001. Joint 9th, Vol. 4, Pp. 2154-2159, 2001. DOI: 10.1109/NAFIPS.2001.944403.

[101] R.K. Pace and R. Barry, “Sparse Spatial Autoregressions,” *Statistics and Probability Letters*, No. 33 Pp. 291-297, 1997. DOI: 10.1016/S0167-7152(96)00140-X.

[102] M. K. Pakhira, S. Bandyopadhyay, U. Maulik, “Validity index for crisp and fuzzy clusters,” *Pattern Recognition*, Vol. 37, Iss. 3, Pp. 487-501, 2003. DOI: 10.1016/j.patcog.2003.06.005.

[103] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik, “A study of some fuzzy cluster validity indices, genetic clustering and application to pixel classification,” *Fuzzy Sets and Systems*, Vol. 155, Iss. 2, Pp. 191-214, 2005. DOI: 10.1016/j.fss.2005.04.009.

[104] S.E. Papadakis and V.G. Kaburlasos, “Piecewise-linear approximation of non-linear models based on probabilistically/possibilistically interpreted intervals’ numbers (INs),”

Information Sciences 2010, Vol. 180, No. 24, Pp. 5060-5076, DOI: 10.1016/j.ins.2010.03.023.

[105] C. P. Pappis and M. Sugeno, “Fuzzy relational equations and the inverse problem,” *Fuzzy Sets and Systems*, Vol. 15, Issue 1, 1985 Pp. 79-90.

[106] W. Pedrycz, “Numerical and applicational aspects of fuzzy relational equations,” *Fuzzy Sets and Systems*, Vol. 11, Issues 1–3, 1983, Pp. 1-18.

[107] W. Pedrycz, “An identification algorithm in fuzzy relational systems,” *Fuzzy sets and Systems*, Vol. 13, Issue 2, 1984, Pp. 153-167.

[108] W. Pedrycz, “Approximate solutions of fuzzy relational equations,” *Fuzzy sets and Systems*, Vol. 28, Issue 2, Pp. 183-202, 1988.

[109] W. Pedrycz, “Neurocomputations in relational systems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, Issue 3, 1991, Pp. 289-297.

[110] W. Pedrycz, “Conditional Fuzzy C-Means,” *Patter Recognition Letters*, Vol. 17, No. 6, Pp. 625-631, May 1996. DOI: 10.1016/0167-8655(96)00027-X.

[111] W. Pedrycz, “Conditional Fuzzy Clustering in the Design of Radial Basis Function Neural Networks,” *IEEE Transactions on Neural Networks*, Vol. 9, No. 4, Pp. 601-612, Jul. 1998. DOI: 10.1109/72.701174.

[112] W. Pedrycz, “From Fuzzy Models to Granular Fuzzy Models,” *LNCS 6857: Fuzzy Logic and Applications*. Springer Berlin / Heidelberg, Pp. 75-82, 2001.

[113] W. Pedrycz, “Genetic tolerance fuzzy neural networks: From data to fuzzy hyperboxes,” *Neurocomputing*, Vol. 70, No. 7–9, Pp. 1403-1413, March 2007. DOI: 10.1016/j.neucom.2006.06.001.

[114] W. Pedrycz, “Allocation of information granularity in optimization and decision-making models: Towards building the foundations of Granular Computing,” *European Journal of Operational Research*, 2012.

[115] W. Pedrycz, A. Amato, V. Di Lecce, and V. Piuri, "Fuzzy Clustering With Partial Supervision in Organization and Classification of Digital Images," *IEEE Transactions on in Fuzzy Systems*, vol. 16, no. 4, pp. 1008-1026, Aug 2008 DOI: 10.1109/TFUZZ.2008.917287.

[116] W. Pedrycz and A. Bargiela, “Granular clustering: a granular signature of data,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol. 32, No. 2, Pp. 212-224, Apr 2002. DOI: 10.1109/3477.990878.

[117] W. Pedrycz and A. Bargiela, "Fuzzy clustering with semantically distinct families of variables: Descriptive and predictive aspects," *Pattern Recognition Letters*, vol. 31, issue 13, pp. 1952-1958, October 2010, DOI: 10.1016/j.patrec.2010.06.018.

[118] W. Pedrycz and A. Bargiela, “An Optimization of Allocation of Information Granularity in the Interpretation of Data Structures: Toward Granular Fuzzy Clustering,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol.42, No.3, Pp.582-590, 2012, DOI: 10.1109/TSMCB.2011.2170067.

[119] W. Pedrycz and F. Gomide, “Fuzzy Systems Engineering: Toward Human-Centric Computing,” *Wiley-IEEE Press*, 2007.

[120] W. Pedrycz and K. C. Kwak, “Linguistic Model as a Framework of User-Centric System Modeling,” *System, Man, and Cybernetic, Part A: Systems and Humans*, *IEEE Transactions on*, Vol. 36, No. 4, Pp.727-745, 2006. DOI: 10.1109/TSMCA.2005.855755.

[121] W. Pedrycz and K. C. Kwak, “Boosting of granular models,” *Fuzzy Sets and Systems*, Vol. 157, No. 22, Pp. 2934-2953, Nov. 2006. DOI: 10.1016/j.fss.2006.07.005.

[122] W. Pedrycz and J.V. de Oliveira, "A Development of Fuzzy Encoding and Decoding Through Fuzzy Clustering," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 4, pp. 829-837, April 2008. DOI: 10.1109/TIM.2007.913809.

[123] W. Pedrycz, B. Russo, and G. Succi, “Knowledge transfer in system modeling and its realization through an optimal allocation of information granularity,” *Applied Soft Computing*, Vol. 12, Issue 8, 2012, Pp. 1985-1995.

[124] W. Pedrycz and M. Song “Analytic Hierarchy Process (AHP) in Group Decision Making and its Optimization with an Allocation of Information Granularity,” *IEEE Transactions on Fuzzy Systems*, Vol. 19, Iss. 3, pp. 527-539, 2011.

[125] W. Pedrycz and M. Song, “Granular fuzzy models: a study in knowledge management in fuzzy modeling,” *International Journal of Approximate Reasoning*, Vol. 53, Iss. 7, Pp. 1061-1079, 2012.

[126] W. Pedrycz and A. V. Vasilakos, “Linguistic Models and Linguistic Modeling,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol.29, No.6, Pp.745-757, 1999. DOI: 10.1109/3477.809029.

[127] K. Peeva, “Resolution of fuzzy relational equations – method, algorithm and software with applications,” *Information Sciences*. Vol. 234, 10 June 2013, Pp. 44-63. DOI: 10.1016/j.ins.2011.04.011.

[128] C. Porcel, A. G. López-Herrera, and E. Herrera-Viedma, “A recommender system for research resources based on fuzzy linguistic modeling,” *Expert Systems with Applications*, Vol. 36, No. 3, part 1, Pp. 5173-5183, Apr. 2009. DOI: 10.1016/j.eswa.2008.06.038.

[129] M. Pratama, S.G. Anavatti, P.P. Angelov, and E. Lughofer, “PANFIS: A Novel Incremental Learning Machine,” *IEEE Transactions in Neural Networks and Learning Systems*, Vol. 25, No. 1, Pp. 55-68, Jan. 2014. DOI: 10.1109/TNNLS.2013.2271933.

[130] K. V. Price, R. M. Storn, and J. A. Lampinen, “Differential Evolution: A Practical Approach to Global Optimization,” *Springer-Verlag*, 2005.

[131] Y. Qian, H. Zhang, F. Li, Q. Hu, and J. Liang, “Set-based granular computing: A lattice model,” *International Journal of Approximate Reasoning*, Vol. 55, Issue 3, Pp. 834-852, March 2014. DOI: 10.1016/j.ijar.2013.11.001.

[132] J. R. Quinlan, P. J. Compton, K. A. Horn, and L. Lazurus, “Inductive knowledge acquisition: A case study,” *Proceedings of the Second Australian Conference on Applications of Expert Systems*, Sydney, Australia, 1986.

[133] G. N. Ramos, Y. Hatakeyama, F. Dong, and K. Hirota, “Hyperbox clustering with Ant Colony Optimization (HACO) method and its application to medical risk profile recognition,” *Applied Soft Computing*, Vol. 9, No. 2, Pp. 632-640, Mar 2009. DOI: 10.1016/j.asoc.2008.09.004.

[134] C. A. Reyes, “On the design of a fuzzy relational neural network for automatic speech recognition,” *Doctoral Dissertation*, The Florida State University, Tallahassee, Fl. 1994.

[135] O.F. Reyes-Galaviz, A. Verduzco, E. Arch-Tirado, and C.A. Reyes-García, "Analysis of an infant cry recognizer for the early identification of pathologies," *Nonlinear Speech Modeling and Applications*, LNCS 3445, Springer Berlin Heidelberg, pp. 404–409, 2005. DOI: 10.1007/11520153_25.

[136] O.F. Reyes-Galaviz and C.A. Reyes-García, “Infant Cry Classification to Identify Hypo Acoustics and Asphyxia Comparing an Evolutionary-Neural System with a Neural Network System,” *MICAI: Advances in Artificial Intelligence*, LNCS 3798, Springer Berlin Heidelberg, pp. 949-958, 2005. DOI: 10.1007/11579427_97.

[137] A. Rizzi, F. M. F. Mascioli, and G. Martinelli, “Generalized min-max classifier,” *Fuzzy Systems*, *FUZZ IEEE 2000*. The Ninth IEEE International Conference on, Vol.1, Pp.36-41, 7-10 May 2000. DOI: 10.1109/FUZZY.2000.838630.

[138] E. Sanchez, “Resolution of Composite Fuzzy Relational Equations,” *Information and Control*, Vol. 30, Issue 1, 1976, Pp. 38-48.

[139] S. Sayah and A. Hamouda, “A hybrid differential evolution algorithm based on particle swarm optimization for nonconvex economic dispatch problems,” *Applied Soft Computing*, Vol. 13, Issue 4, April 2013, Pp. 1608-1619.

[140] B. Schweizer and A. Sklar, “Associative functions and statistical triangle inequalities,” *Publ. Mathematical Debrecen*, Issue 8, 1961, Pp. 313-334.

[141] M. Senoussaoui, P. Kenny, T. Stafylakis, and P. Dumouchel, “A Study of the Cosine Distance-Based Mean Shift for Telephone Speech Diarization,” *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, Vol. 22, No. 1, Pp. 217-227, Jan. 2014. DOI: 10.1109/TASLP.2013.2285474.

[142] M. Setnes, “Supervised fuzzy clustering for rule extraction,” *Fuzzy Systems Conference Proceedings, FUZZ-IEEE ‘99. IEEE International*, Vol. 3, Pp. 1270-1274, Aug 1999. DOI: 10.1109/FUZZY.1999.790084.

[143] M. Setnes, “Supervised fuzzy clustering for rule extraction,” *Fuzzy Systems, IEEE Transactions on*, Vol. 8, No. 4, Pp. 416-424, Aug 2000. DOI: 10.1109/91.868948.

[144] M. Setnes and R. Babuska, "Fuzzy relational classifier trained by fuzzy clustering," in IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 29, no. 5, pp. 619-625, Oct 1999 DOI: 10.1109/3477.790444.

[145] P. K. Simpson, “Fuzzy Min-Max Neural Networks,” *Neural Networks, 1991. IEEE International Joint Conference on*, Vol. 2, Pp. 1658-1669, 18-21 Nov 1991. DOI: 10.1109/IJCNN.1991.170647.

[146] P. K. Simpson, “Fuzzy Min-Max Neural Networks-Part 1: Classification,” *Neural Networks, IEEE Transactions on*, Vol. 3, No. 5, Pp. 776-786, Sep 1992. DOI: 10.1109/72.159066.

[147] P. K. Simpson, “Fuzzy Min-Max Neural Networks - Part 2: Clustering,” *Fuzzy Systems, IEEE Transactions on*, Vol. 1, No. 1, Pp.32--, Feb. 1993. DOI: 10.1109/TFUZZ.1993.390282H.

[148] H. Sossa and E. Guevara, “Efficient training for dendrite morphological neural networks,” *Neurocomputing*, Vol. 131, No. 5 Pp. 132-142 May 2014. DOI: 10.1016/j.neucom.2013.10.031.

[149] R. Storn, “On the usage of differential evolution for function optimization,” *Fuzzy Information Processing Society, 1996, NAFIPS, 1996 Biennial Conference of the North American*, Pp.519-523, Jun. 1996. DOI: 10.1109/NAFIPS.1996.534789.

[150] R. Storn and K. Price, “Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces,” *Journal of Global Optimization*, Vol. 11, No. 4. Pp. 341-359, Dec. 1997. DOI: 10.1023/A:1008202821328.

[151] C. Su and F. Guo, “Solving Interval-Valued Fuzzy Relation Equations with a Linear Objective Function,” *Sixth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, Vol. 4, Pp. 380-385, 2009.

[152] M. Sugeno and T. Yasukawa, “A fuzzy-logic-based approach to qualitative modeling,” *IEEE Transactions on Fuzzy Systems*, Vol. 1, No. 1, Pp. 7, Feb. 1993. DOI: 10.1109/TFUZZ.1993.390281.

[153] F. Sun, “Conditions for the existence of the least solution and minimal solutions to fuzzy relation equations over complete Brouwerian lattices,” *Information Sciences*, Vol. 205 Issue 1, 2012, Pp. 86-92.

[154] Y. Sun, G. Feng, and J. Cao, “A New Approach to Dynamic Fuzzy Modeling of Genetic Regulatory Networks,” *IEEE Transactions on Nanobioci*ence, Vol. 9, No. 4, Pp. 263-272, Dec. 2010. DOI: 10.1109/TNB.2010.2082559.

[155] Y. Tang and J. Zheng, “Linguistic modeling based on semantic similarity relation among linguistic labels,” *Fuzzy Sets and Systems*, Vol. 157, Issue 12, Pp. 1662-1673, Jun. 2006. DOI: 10.1016/j.fss.2006.02.014.

[156] Y. Tang, Y.-Q. Zhang, Z. Huang, X. Hu, and Y. Zhao, “Recursive Fuzzy Granulation for Gene Subsets Extraction and Cancer Classification,” *Information Technology in Biomedicine, IEEE Transactions on*, Vol. 12, No. 6, Pp. 723-730, Nov. 2008. DOI: 10.1109/TITB.2008.920787.

[157] Y. Tarabalka, J.A. Benediktsson, and J. Chanussot, “Spectral–Spatial Classification of Hyperspectral Imagery Based on Partitional Clustering Techniques,” *Geoscience and Remote Sensing, IEEE Transactions on*, Vol. 47, No. 8, Pp. 2973-2987, Aug. 2009. DOI: 10.1109/TGRS.2009.2016214.

[158] R. Thawonmas and S. Abe, “A fuzzy classifier based on partitioned hyperboxes,” *Neural Networks, IEEE International Conference on*, Vol. 2, Pp. 1097-1102, 3-6 Jun. 1996. DOI: 10.1109/ICNN.1996.549051.

[159] R. Thawonmas and S. Abe, “Function approximation based on fuzzy rules extracted from partitioned numerical data,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol. 29, No. 4, Pp. 525-534, Aug 1999. DOI: 10.1109/3477.775268.

[160] K. Trawinski, O. Cordon, L. Sanchez, and A. Quirin, “A Genetic Fuzzy Linguistic Combination Method for Fuzzy Rule-Based Multiclassifiers,” *IEEE Transactions on Fuzzy Systems,* Vol. 21, No. 5, Pp. 950-965, Oct. 2013. DOI: 10.1109/TFUZZ.2012.2236844.

[161] N. Tschichold-Gürman. “Generation and improvement of fuzzy classifiers with incremental learning using fuzzy RuleNet,” *Proceedings of the 1995 ACM symposium on Applied computing (SAC ‘95),* ACM, New York, NY, USA, Pp. 466-470, 1995. DOI: 10.1145/315891.316069.

[162] G.E. Tsekouras, “On the use of the weighted fuzzy c-means in fuzzy modeling,” Adv. Eng. Softw. 36 (5) (May 2005) Pp. 287-300, http://dx.doi.org/10.1016/j.advengsoft.2004.12.001.

[163] V.S. Tseng, C-P Kao, “A Novel Similarity-Based Fuzzy Clustering Algorithm by Integrating PCM and Mountain Method,” Fuzzy Systems, IEEE Transactions on, Vol. 15, No. 6, Pp. 1188-1196, Dec. 2007. DOI: 10.1109/TFUZZ.2006.890673.

[164] O. Uncu, I.B. Turksen, “Discrete Interval Type-2 Fuzzy System Models Using Uncertainty in Learning Parameters,” *IEEE Transactions On Fuzzy Systems,* Vol. 15, No. 1, Pp. 90-106, Feb. 2007. DOI: 10.1109/TFUZZ.2006.889765.

[165] W. Wagenknecht and K. Hartmann, “Fuzzy modeling with tolerances,” *Fuzzy Sets and Systems*, Vol. 20, Iss. 3, Pp. 325-332, 1986.

[166] W. Wagenknecht and K. Hartmann, “On direct and inverse problems for fuzzy equation systems with tolerances,” *Fuzzy Sets and Systems*, Vol. 24, Iss. 1, Pp. 93-102, 1987.

[167] H. F. Wang and Y. C. Chang, “Resolution of composite interval-valued fuzzy relation equations,” *Fuzzy sets and Systems*, Vol. 44, Iss. 2, Pp. 227-240, 1991.

[168] S. Wang, S. C. Fang, and H. L. Nuttle, “Solution Sets of Interval-Valued Fuzzy Relational Equations,” *Fuzzy Optimization and Decision Making, Mathematics and Statistics*, Springer Netherlands. Vol. 2 Iss. 1, Pp. 41-60, 2002.

[169] S. Wu, X. Feng, W. Zhou, “Spectral clustering of high-dimensional data exploiting sparse representation vectors,” Neurocomputing, Vol. 135, No. 5, Pp. 229-239, July 2014. DOI: 10.1016/j.neucom.2013.12.027.

[170] X. L. Xie and G. Beni, “A validity measure for fuzzy clustering,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, Vol. 13, No. 8, Pp. 841-847, 1991. DOI: 10.1109/34.85677.

[171] Y. Xie, V.V. Raghavan, P. Dhatric, X. Zhao, “A new fuzzy clustering algorithm for optimally finding granular prototypes,” *International Journal of Approximate Reasoning*, Vol. 40, Issues 1–2, Pp. 109-124, July 2005. DOI: 10.1016/j.ijar.2004.11.002.

[172] C. W. Xu and Y. Z. Lu, “Fuzzy model identification and self-learning for dynamic systems,” *IEEE Transactions in Systems*, Man and Cybernetics Vol. 17, Issue 4, 1987, Pp. 683-689.

[173] X. Yao, “Evolving artificial neural networks,” *Proceedings of the IEEE*, Vol. 87 Issue 9, 1999, Pp. 1423-1447.

[174] B. Yuan, G.J. Klir, J.F. Swan-Stone, "Evolutionary fuzzy c-means clustering algorithm," Proceedings of 1995 IEEE Int. in Fuzzy Systems, 1995. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., vol. 4, pp. 2221-2226, Mar 1995 DOI: 10.1109/FUZZY.1995.409988.

[175] L.A. Zadeh, “Fuzzy Sets,” *Information and Control*, Vol. 8, Issue 3, Pp. 338-353, 1965.

[176] H. Zhang, J. Liu, D. Ma, and Z. Wang, “Data-Core-Based Fuzzy Min–Max Neural Network for Pattern Classification,” Neural Networks, IEEE Transactions on, Vol. 22, No. 12, Pp. 2339-2352, Dec. 2011. DOI: 10.1109/TNN.2011.2175748.

[177] Y. Zhai, X. Liu, "Multiscale edge detection based on fuzzy c-means clustering," in 1st International Symposium on Systems and Control in Aerospace and Astronautics, 2006. ISSCAA 2006, Jan. 2006, pp.1201-1204, DOI: 10.1109/ISSCAA.2006.1627581.

[178] Y. Zhong, A. Ma, and L. Zhang, “An Adaptive Memetic Fuzzy Clustering Algorithm With Spatial Information for Remote Sensing Imagery,” Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of, Vol. 7, No. 4, Pp. 1235-1248, Apr. 2014. DOI: 10.1109/JSTARS.2014.2303634.

[179] E. Zio and P. Baraldi, "Evolutionary fuzzy clustering for the Classification of transients in nuclear components," Progress in Nuclear Energy, Vol. 46, Issues 3–4, pp. 282-296, 2005 DOI: 10.1016/j.pnucene.2005.03.010.

[180] E. Zio and P. Baraldi, "Identification of nuclear transients via optimized fuzzy clustering," Annals of Nuclear Energy, Vol. 32, Issue 10, pp. 1068-1080, July 2005, DOI: 10.1016/j.anucene.2005.02.012.