Evolving Granular Systems

Daniel Furtado Leite

Supervisor: Fernando Gomide

Co-supervisor: Pyramo Costa Jr.

Department of Computer Engineering and Industrial Automation

School of Electrical and Computer Engineering

University of Campinas

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

July, 2012

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

L536s

Leite, Daniel Furtado
Sistemas granulares evolutivos / Daniel Furtado Leite. --Campinas, SP: [s.n.], 2012.

Orientador: Fernando Antonio Campos Gomide.
Tese de Doutorado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Computação granular. 2. Sistemas inteligentes. 3. Redes neurais artificiais. 4. Aprendizado do computador. 5. Sistemas fuzzy. I. Gomide, Fernando Antonio Campos, 1951-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês: Evolving granular systems
Palavras-chave em Inglês: Granular computing, Intelligent systems, Artificial neural networks, Machine learning, Fuzzy systems
Área de concentração: Automação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora: Andre Paim Lemos, Weldon Alexander Lodwick, Fernando José Von Zuben, Romis Ribeiro de Faissol Attux
Data da defesa: 19-07-2012
Programa de Pós Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Daniel Furtado Leite

Data da Defesa: 19 de julho de 2012

Título da Tese: "Sistemas Granulares Evolutivos"

Prof. Dr. Fernando Antonio Campos Gomide (Presidente): _____

Prof. Dr. Andre Paim Lemos: _____

Prof. Dr. Weldon Alexander Lodwick: _____

Prof. Dr. Fernando José Von Zuben: _____

Prof. Dr. Romis Ribeiro de Faissol Attux: _____

Acknowledgement

First and foremost, I would like to thank my supervisor Fernando Gomide. It is difficult to imagine this thesis without his inspiring mentorship, insightful suggestions, and words of encouragement. I owe him a debt of gratitude for all I have learned from him. I am also greatly fortunate to have Pyramo Costa as my co-supervisor. Apart from providing me with numerous research feedbacks, he has been a constant source of support and stimulation.

My academic life was shaped by great professors. I would like to acknowledge Fernando Von Zuben, Romis Attux, Akebo Yamakami, and Takaaki Ohishi for lecturing some of the best classes I have ever taken, and for nurturing my early interests in computational intelligence, mathematics and optimization. I am also indebted to Professor Rosangela Ballini for her help in obtaining and analyzing datasets, and for sharing her expertise in several opportunities.

I would like to thank the committee members Andre Lemos, Weldon Lodwick, and again Fernando Von Zuben, Romis Attux, and Fernando Gomide for the constructive feedback they provided me in order to improve this thesis.

I was privileged to have known talented and enthusiastic folks in our LCA group throughout these years. Special thanks go to Joelma Costa, Glaucya Boechat, Yi Liu, Fernando Bordignon, Leandro Maciel, and Vitor Marques for being a great support in hard times and joyful company in good times. My gratitude extends to Lucas Nascimento, Alan Barbosa, Luiz Bergo, Enderson Cruz, and Israel Mendes, fellows from our LASI group at PUC Minas.

I appreciated the help of Carmen Fonseca, Mariana Silva, Noemia Benatti, Edson Filho, Maria Waldman, Carolina Velho, Jerusa Soares, and Zilda Padovan, who made their support available in countless ways whenever I needed it.

I have much gratitude for CAPES, the Brazilian Ministry of Education, for

the fellowship which enabled me to pursue this research.

Last, but certainly not least, I would like to thank Daniela and Lucia for their endless support and encouragement during this long endeavor. Their love and perseverance have kept me going through the roughest of times. To them I dedicate this thesis.

Abstract

In recent years there has been increasing interest in computational modeling approaches to deal with real-world data streams. Methods and algorithms have been proposed to uncover meaningful knowledge from very large (often unbounded) data sets in principle with no apparent value. This thesis introduces a framework for evolving granular modeling of uncertain data streams. Evolving granular systems comprise an array of online modeling approaches inspired by the way in which humans deal with complexity. These systems explore the information flow in dynamic environments and derive from it models that can be linguistically understood. Particularly, information granulation is a natural technique to dispense unnecessary details and emphasize transparency, interpretability and scalability of information systems. Uncertain (granular) data arise from imprecise perception or description of the value of a variable. Broadly stated, various factors can affect one's choice of data representation such that the representing object conveys the meaning of the concept it is being used to represent. Of particular concern to this work are numerical, interval, and fuzzy types of granular data; and interval, fuzzy, and neurofuzzy modeling frameworks. Learning in evolving granular systems is based on incremental algorithms that build model structure from scratch on a per-sample basis and adapt model parameters whenever necessary. This learning paradigm is meaningful once it avoids redesigning and retraining models all along if the system changes. Application examples in classification, function approximation, time-series prediction and control using real and synthetic data illustrate the usefulness of the granular approaches and framework proposed. The behavior of nonstationary data streams with gradual and abrupt regime shifts is also analyzed in the realm of evolving granular computing. We shed light upon the role of interval, fuzzy, and neurofuzzy computing in process-

ing uncertain data and providing high-quality approximate solutions and rule summary of input-output data sets. The approaches and framework introduced constitute a natural extension of evolving intelligent systems over numeric data streams to evolving granular systems over granular data streams.

Keywords: Granular Computing, Intelligent Systems, Computational Intelligence, Machine Learning, Artificial Neural Networks.

Resumo

Recentemente tem-se observado um crescente interesse em abordagens de modelagem computacional para lidar com fluxos de dados do mundo real. Métodos e algoritmos têm sido propostos para obtenção de conhecimento a partir de conjuntos de dados muito grandes e, a princípio, sem valor aparente. Este trabalho apresenta uma plataforma computacional para modelagem granular evolutiva de fluxos de dados incertos. Sistemas granulares evolutivos abrangem uma variedade de abordagens para modelagem on-line inspiradas na forma com que os humanos lidam com a complexidade. Esses sistemas exploram o fluxo de informação em ambiente dinâmico e extrai disso modelos que podem ser linguisticamente entendidos. Particularmente, a granulação da informação é uma técnica natural para dispensar atenção a detalhes desnecessários e enfatizar transparência, interpretabilidade e escalabilidade de sistemas de informação. Dados incertos (granulares) surgem a partir de percepções ou descrições imprecisas do valor de uma variável. De maneira geral, vários fatores podem afetar a escolha da representação dos dados tal que o objeto representativo reflita o significado do conceito que ele está sendo usado para representar. Neste trabalho são considerados dados numéricos, intervalares e fuzzy; e modelos intervalares, fuzzy e neuro-fuzzy. A aprendizagem de sistemas granulares é baseada em algoritmos incrementais que constroem a estrutura do modelo sem conhecimento anterior sobre o processo e adapta os parâmetros do modelo sempre que necessário. Este paradigma de aprendizagem é particularmente importante uma vez que ele evita a reconstrução e o retreinamento do modelo quando o ambiente muda. Exemplos de aplicação em classificação, aproximação de função, predição de séries temporais e controle usando dados sintéticos e reais ilustram a utilidade das abordagens de modelagem granular propostas. O comportamento de fluxos de dados não-estacionários com

mudanças graduais e abruptas de regime é também analisado dentro do paradigma de computação granular evolutiva. Realçamos o papel da computação intervalar, fuzzy e neuro-fuzzy em processar dados incertos e prover soluções aproximadas de alta qualidade e sumário de regras de conjuntos de dados de entrada e saída. As abordagens e o paradigma introduzidos constituem uma extensão natural de sistemas inteligentes evolutivos para processamento de dados numéricos a sistemas granulares evolutivos para processamento de dados granulares.

Palavras-chave: Computação Granular, Sistemas Inteligentes, Inteligência Computacional, Aprendizagem de Máquina, Redes Neurais Artificiais.

Publications

During the course of this research, a number of publications were produced which are based on or somehow related to the content of this thesis. They are listed below for reference.

Book chapters

- Leite, D.; Costa, P.; Gomide, F. "Interval approach for evolving granular system modeling." In: Mouchaweh, M.; Lughofer, E. (Eds.) Learning in Non-stationary Environments: Methods and Applications, Springer - New York, pp: 271-301, 2012.
- Leite, D.; Gomide, F. "Evolving linguistic fuzzy models from data streams." In: Trillas, E.; Bonissone, P.; Magdalena, L.; Kacprycz, J. (Eds.) Combining Experimentation and Theory: A Hommage to Abe Mamdani (Studies in Fuzziness and Soft Computing), Springer - Verlag, pp: 209-223, 2011.
- Leite, D.; Costa, P.; Gomide, F. "Granular approach for evolving systems modeling." In: Hullermeier, E.; Kruse, R.; Hoffmann, F. (Eds.) Lecture Notes in Artificial Intelligence (LNAI/IPMU), Vol. 6178, pp: 340-349, Springer - Verlag Berlin Heidelberg, 2010.

Journals

- Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural networks from fuzzy data streams." Neural Networks. (Accepted).
- Leite, D.; Ballini, R.; Costa, P.; Gomide, F. "Evolving fuzzy granular modeling from nonstationary fuzzy data streams." Evolving Systems, Springer. Vol. 3, Issue 2, pp: 65-79, 2012.
- Leite, D.; Hell, M.; Costa, P.; Gomide, F. "Real-time fault diagnosis of nonlinear systems." Nonlinear Analysis: Theory, Methods & Applications. Vol. 71, Issue 12, pp: 2665-2673, 2009.

International conferences

- Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural network for fuzzy time series forecasting." World Congress on Computational Intelligence (WCCI: IJCNN), Brisbane - AU, 8p. 2012.
- Lemos, A.; Leite, D.; Maciel, L.; Ballini, R.; Caminhas, W.; Gomide, F. "Evolving Fuzzy Linear Regression Tree Approach for Forecasting Sales Volume of Petroleum Products." World Congress on Computational Intelligence (WCCI: FUZZ-IEEE), Brisbane - AU, 8p. 2012.
- Leite, D.; Gomide, F.; Ballini, R.; Costa, P. "Fuzzy granular evolving modeling for time series prediction." IEEE International Conference on Fuzzy Systems, Taipei - TW, pp: 2794-2801, 2011.
- Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural network for semi-supervised data stream classification." World Congress on Computational Intelligence (WCCI: IJCNN), Barcelona - ES, pp: 1877-1884, 2010.
- Leite, D.; Costa, P.; Gomide, F. "Evolving granular classification neural networks." IEEE International Joint Conference on Neural Networks, Atlanta - US, pp: 1736-1743, 2009.
- Leite, D.; Attux, R.; Von Zuben, V.; Costa P.; Gomide, F. "Evolutionary neural network applied to induction motors stator fault detection." IEEE International Electric Machines and Drives Conference, Miami - US, pp: 1721-1728, 2009.
- Leite, D.; Costa P.; Gomide, F. "Interval-based evolving modeling." IEEE Symposium Series on Computational Intelligence: Workshop on Evolving Systems, Nashville - US, pp: 1-8, 2009.

Brazilian conferences

- Leite, D.; Ballini, R.; Costa, P.; Gomide, F. "Fuzzy granular evolving modeling." 10th Brazilian Symposium on Intelligent Automation, Sao Joao Del Rey - MG, 6p. 2011. (In Portuguese).
- Leite, D.; Costa, P.; Gomide, F. "Granular neural networks for semi-supervised learning." 18th Brazilian Congress on Automatics, Bonito - MS, 8p. 2010. (In Portuguese).
- Leite, D.; Costa, P.; Gomide, F. "Evolving granular systems: real-time processing of data streams." (Abstract) 1st Brazilian Congress on Fuzzy Systems, Sorocaba - SP, 2p. 2010. (In Portuguese).

- Leite, D.; Nascimento, L.; Barbosa, A.; Costa, P.; Ferreira, D.; Gomide, F. "Evolving approach for power transformer fault detection." 6th International Workshop on Power Transformers, Foz do Iguacu - PR, 8p. 2010. (In Portuguese).

- Leite, D.; Gomide, F. "Granular neural network for evolving classification." (Extended abstract) Annual Meeting of the Department of Computer Engineering and Industrial Automation, UNICAMP, Campinas - SP, 4p. 2010. (In Portuguese).

- Leite, D.; Bergo, L.; Costa, P.; Gomide, F. "Evolving granular neural networks in systems modeling." 9th Brazilian Congress on Neural Networks, Ouro Preto - MG, 5p. 2009. (In Portuguese).

- Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural networks." 9th Brazilian Symposium on Intelligent Automation, Brasilia - DF, 6p. 2009. (In Portuguese).

- Leite, D.; Costa, P.; Gomide, F. "Evolving connectionist systems." 9th Brazilian Symposium on Intelligent Automation, Brasilia - DF, 6p. 2009. (In Portuguese).

- Leite, D.; Gomide, F. "Interval-based evolving modeling for streamflow forecasting." (Extended abstract) Annual Meeting of the Department of Computer Engineering and Industrial Automation, UNICAMP, Campinas - SP, 4p. 2009.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The computing world has experienced a rapidly increasing growth of information. A proliferation of automated systems, small scale computing devices, sensor networks, and data capture technologies has contributed to the production of large volumes of data. Data set growth sometimes outpaces available storage capacity and other times data are stored with no prospective use. Broadly stated, the focus of data processing and analysis has changed from offline batch processing of data to the incremental handling of online data streams.

Online data streams originate from a variety of sources such as media entertainment, surveillance systems, mobile devices, multimedia, industrial monitoring and control, oceanographic and atmospheric systems, health care, stock market, satellites, financial and meteorological systems, web traffic and clickstreams, to name a few. Their prominence in real-world systems, along with the necessity of modeling, analyzing, and understanding these systems, has brought new challenges, greater demands, and new research directions.

Research and development of conceptual frameworks, methods, and algorithms capable of extracting knowledge from data streams have taken place motivated by a manifold of relevant applications. Data stream modeling is fundamentally based on computational learning approaches that both, process data continuously as an attempt to find similarities in their spatio-temporal features, and thereafter provide insights about the phenomenon which governs the data. The ultimate goal is to obtain more abstract (often human-centric) representations of large amounts of detailed data with no apparent value.

Modeling, processing, and disposing information become more complex as real-world systems become more complex. Data streams are characterized by nonstationarity, nonlinearity, and heterogeneity; they are potentially endless and may be subject to changes of various kinds. Direct application of machine learning and data mining algorithms to data streams is very often infeasible because it is difficult to maintain all the data in memory. A particular challenge faced in stream modeling concerns how to handle uncertainty.

The primary research question of this thesis is how to obtain accurate and interpretable human-centered models from uncertain data streams. We introduce evolving granular systems, a granular modeling framework able to capture the essence of uncertain data streams in a more abstract and compact representation. While the term 'evolving' derives from structurally adaptive models from data streams, the term 'granular' comes from granular computing theory and emphasizes comprehensible models of uncertainty. This thesis combines evolving intelligence and granular computing concepts and ideas, and explicitly realizes them into a practical evolving granular framework.

1.1 Background Research

Uncertainty is an attribute of information since our ability to perceive reality is often limited (27) (154). The more complex a system is, potentially, the more uncertain we are of the available information, and the more imprecise is our understanding of that system. The imprecision of real-world perception is evident in natural languages and also in empirical measurements where it is known that the process which generated the data is uncertain. As Kreinovich stated, measurements and expert estimates are never exact (72). Modeling complex systems raises doubts about the necessity of precise models (118). Granular computing theory (15) (89) (116) (141) (144) (151) hypothesizes that accepting some level of uncertainty may be beneficial and therefore suggests a balance between precision and uncertainty.

Information granulation for uncertainty representation is a fundamental manifestation of the human knowledge (15). Information granulation means that, instead of dealing with detailed real-world data, the data are considered in a

more abstract and conceptual perspective. The result of information granulation is called information granule - a granule being a cluster of points put together by indistinguishability, similarity, proximity, or functionality (152). Examples of granules include hyperboxes, fuzzy sets, bell-shaped probability distributions and rough sets (153). A set of granules constitutes a vocabulary of generic descriptors (115), and underlies the basic concepts of linguistic variable and rule-based systems. Put differently, granules are semantically meaningful building blocks of granular rule-based systems (18). Granular rules connect the elements of the vocabulary and therefore play an important role in computation with information described in natural language.

The notion of granulation emerged as a natural need to abstract and summarize information and data to support various processes of comprehension and decision making (15). For example, when we observe an environment we seldom take into account all of the details of that environment. Because of our physical and cognitive limitations, a reduced number of samples, variables, and attributes of interest are brought into focus. To avoid distracting details we are provided with effective abstraction mechanisms. Detailed numeric data are integrated (aggregated) into kinds of information granules where the granules themselves are regarded as sets of elements that are perceived as being functionally equivalent (116). There are close relations between granulation, data mining (135), data fusion (86), and knowledge discovery (99).

From a more practical point of view, granular computing is a framework for problem solving, complexity reduction and structured thinking (142). It deals with data granulation and granular data processing. Information granulation splits a complex problem into simpler sub-problems and treats them on an individual basis. Granular systems able to self-adapt their structures from data streams have only been formally investigated since the early 2000s.

Granular models developed from data streams can be expressed in several computationally tractable frameworks such as interval mathematics, statistics, fuzzy sets, rough sets, shadow sets, cluster analysis, decision trees, neighborhood systems, or hybrids. On top of these are generalized constraints, in the sense of Zadeh's general theory of uncertainty (154), which are used to delimit and represent granules within the different frameworks. Computing with granules

grants ample freedom to choose representative granular objects and handling tools. Regardless of the framework chosen, online granulation aims to retain the essence of stream data as granular objects. Online granular computing models consider online granular data streams under simpler (less detailed) resolutions. The fundamental objective is to extract features of interest from the data to attain efficient solutions and a better rapport with reality.

Evolving intelligent systems are a mainstream of research in online data modeling (5) (11) (65) (66) (97). These systems encompass one pass recursive algorithms (algorithms independent of previous data) and manage to build the structure of models from scratch as new information arises. We use the term 'evolving' in the sense of gradual development of the system structure (rule base or the architecture of a neural network) and their parameters. This learning paradigm mimics the evolution of individuals during their life-cycle, especially humans: learning from experience, inheritance, and gradual change. Knowledge can be generated from repetitive tasks and from data streams produced through perceptions and sent to the brain. The development of the rule-base or neural network structure is gradual, where the rules/neurons are not fixed or pre-defined. Evolving systems generate new rules (neurons) each time new data does not fit into the existing model/understanding (fuzzy rule-base or neural network), but at the same time only when this new data is informative enough (9) (11). Classification, clustering, frequent pattern mining, time series prediction, regression and control are examples of problems addressed in the evolving systems literature.

Note that stream data modeling should not be confused with time series data modeling (51). Although related, time series carry static objects that can, in principle, be analyzed offline whereas stream data require the evolution of the model structure in online mode, which is not a requirement in time series analysis (21). Particularly, conventional statistical (54), computational intelligence (44), and machine learning (101) systems do not meet the requirements of data stream modeling because they assume forms of linearity and stationarity, or demand multiple passes over entire data sets and offline processing. Very often there are real-time constraints that must be met by stream algorithms. Evolving intelligent systems arose as a framework to model online data streams and overcome the drawbacks of existing conventional systems.

Currently, a number of evolving intelligent systems have succeeded in dealing with time-varying numeric data by means of recursive clustering algorithms and adaptive local models. Notwithstanding, these systems are quite often unable to process granular data and realize granular-data-stream-oriented computing in unknown nonstationary environments. Informally, if u is a value which is known precisely (exactly), we refer to u as a singular (point) value. Conversely, if u is not known precisely, but there is some information which constrains possible values of u, then the constraint on u defines a granular value (155).

This thesis introduces evolving granular systems, which extend evolving intelligent systems in two ways. First, evolving granular systems deal with granular input and output data such as intervals, fuzzy numbers, and fuzzy intervals. Granular data may arise from expert judgment, readings from unreliable sensors, and summaries of numeric data over time periods. Interval and fuzzy data stream modeling generalizes numeric data stream modeling by allowing interval and fuzzy data granulation. Numeric (singular) data stream is a special case of granular data stream. Second, evolving granular systems provide granular approximation of functions. Granular approximation refers to an enclosure where the output data are within the granular approximation. The granular approximation may come with a linguistic description, in addition to a numeric pointwise approximation common of evolving intelligent systems. Granular output is useful for interpretability purposes and helps to enhance model acceptability.

Numeric, interval and fuzzy granular data streams and interval, fuzzy and neurofuzzy granular frameworks are of special concern to this study. Independent of the data type and framework, evolving granular systems aim to provide transparent rule-based models that are built from a data sequence. Interval mathematics (60) (69) (104) and fuzzy set theory (41) (159), as practical frameworks of granular computing, capture our innate conception of transitional set belonging and uncertainty. While 'below 100', 'around 10 and 20', and 'above 100' are examples of interval data, 'about 20', and 'around 90' are examples of fuzzy data. The fundamental distinction between the interval and fuzzy granular frameworks concerns the notion of partial membership supported by fuzzy sets. Whenever interval yes-or-no quantification of concepts becomes too restrictive, fuzzy sets offer an important feature of describing information granules whose

constituting elements may belong only partially, i.e., more-or-less quantification of concepts. Fuzzy sets avoid specifying solid borders between full belongingness and full exclusion by means of smooth transition boundaries (116). Artificial neural networks (55) are nonlinear, highly plastic systems equipped with significant learning capability. Fuzzy sets and fuzzy neurons provide neural networks with mechanisms of approximate reasoning and transparency of the resulting construction. Fuzzy sets and neurocomputing are complementary in terms of their strengths thus motivating neurofuzzy granular computing. Granules formalized in any of these frameworks, interval, fuzzy or neurofuzzy, can facilitate a vast array of human-centric pursuits (116).

1.2 Objective

The main objective of this thesis is to introduce and characterize a theoretical evolving granular modeling framework and a suite of practical approaches to learn from and process uncertain data streams with a focus on accuracy, transparency and interpretability of models.

Many issues arose during the course of this research which had to be overcome in order to achieve the main objective. The research issues included:

- how to process interval and fuzzy types of data;
- how to fit uncertain data into rule-based granular models;
- how to create, delete and refine granules and rules without the need to redesign and retrain the system structure from scratch;
- how to analyze large volumes of stream data efficiently;
- how to adjust the granularity of models based on stream data;
- how to obtain more flexible granular constructs;
- how the interval, fuzzy and neurofuzzy approaches are comparable to each other and alternative approaches.

Other, more subtle, issues are discussed in context throughout the chapters.

1.3 Contributions

The contributions of this thesis can be broadly divided into three groups: conceptual, methodological and computational.

The conceptual contribution is the introduction of a new modeling framework to represent and process granular data streams. The framework allows input and output data to be real numbers, intervals, and fuzzy sets. We discuss the notion of granular data streams as well as learning and model building driven by such data streams. Central to the proposed framework is not only computational efficiency, but also interpretability and transparency. The framework intends to develop human-oriented models whose results are readily understood. Formulations and conceptualizations are provided which are intended to establish foundations for online granular data processing and uncertainty management.

The methodological contributions of this thesis are three practical approaches to handle granular data streams. The approaches are oriented to different types of input and output data and each is supported by concepts and tools derived from different theories. In common, both approaches are designed to capture the very essence of the underlying data stream.

First, we introduce interval-based evolving modeling. Interval-based evolving modeling (IBeM) is a granular approach to enclose imprecise data and produce rule-based summary. IBeM emphasizes imprecise data manifesting as tolerance intervals and learning procedures grounded in fundamentals of interval mathematics. Antecedent and consequent parts of interval rules are interval hyperboxes, which are linked by an interval granular mapping - or inclusion function in the interval analysis terminology. Interval granular approach for systems modeling makes no specific assumption about the data including probability distributions, membership functions and belief or possibility values.

Second, we address fuzzy set based evolving modeling. Fuzzy set based evolving modeling (FBeM) employs fuzzy granular models to deal with more detailed fuzzy granular data and therefore provide a more comprehensible (human-intelligible) representation of the data. For each fuzzy model there exists an associated fuzzy rule base. The structure of the fuzzy rule base is gradually developed from incremental learning algorithm suitable to process potentially unbounded

fuzzy data streams. FBeM renders linguistic models of information systems and single-valued and granular approximation of nonstationary functions.

Third, we consider evolving neurofuzzy networks. Evolving granular neural networks (eGNN) use fuzzy granules and fuzzy aggregation neurons for information fusion. The fuzzy aspect allows a neural network to be translated into a knowledge base and a rule-based inference system that can be promptly read and understood. The eGNN learning algorithm is committed to building and adapting the neural network structure using fuzzy data streams. It may add or remove granules, neurons and respective connections whenever necessary. This means that the neural network captures new information from data streams, adapts itself to the new scenario, and avoids redesigning and retraining.

The third primary contribution of this thesis concerns an extensive set of computational results detailing the performance and demonstrating the usefulness of the proposed approaches. The interval IBeM, fuzzy FBeM, and neurofuzzy eGNN approaches are evaluated in a variety of applications such as semi-supervised classification, time series prediction, function approximation, and control. The application examples emphasize the difficulty of currently existing methods to deal with nonstationary data streams. The results demonstrate the competitiveness of the proposed evolving granular approaches and framework.

1.4 Organization

This thesis is organized into eight chapters as summarized below.

- This chapter contains a general statement of the problem dealt with in this thesis and places the research into a broader perspective by connecting it to well-established information systems theories.

- Chapter 2 provides a review on concepts of granular computing and uncertainty processing. We provide essentials from interval analysis, fuzzy sets, and aggregation operators to form a background of concepts and support our developments.

- Chapter 3 covers the state-of-the-art research in evolving intelligent systems

and introduces a theoretical framework for the analysis and representation of granular data.

- Chapter 4 introduces an interval learning method. Interval-based evolving modeling is an approach to deal with stream interval data. Interval granules are characterized by sharp lower and upper bounds and empty content. We present details along with some intuition behind learning heuristics of interval algorithms.

- Chapter 5 presents a fuzzy extension of the aforementioned interval method. Fuzzy set based evolving modeling uses fuzzy data streams to develop rule-based fuzzy granular models. Fuzzy sets avoid specifying solid borders between full belongingness and full exclusion by means of sets with partial membership.

- Chapter 6 proposes evolving granular neural networks. These networks use fuzzy granules and fuzzy neurons for information fusion and uncertainty representation. The underlying granular construction is incrementally evolved from a learning algorithm. It pictures a set of fuzzy rules and a fuzzy inference system, which are obtained from fuzzy data streams.

- Chapter 7 addresses application examples of evolving granular systems in semi-supervised classification, function approximation, time series prediction, and control problems. They are accompanied with discussions and comparisons. We contrast the methods introduced in chapters 4, 5 and 6 with the state-of-the-art online and traditional offline methods.

- Chapter 8 concludes this thesis and proposes future research directions.

Chapter 2

Foundations of Granular Computing

This chapter provides definitions and principles of granular computing. Essential notions of interval analysis and fuzzy sets are addressed from the granular computing point of view. Some notation to be used throughout this thesis is introduced. The chapter also covers different types of aggregation operators which map several real inputs in the unit hypercube onto a single output in the unit interval. Aggregation operators perform information fusion by gathering large volumes of dissimilar information into a more compact form. Intervals and fuzzy sets are instances of practical frameworks of granular computing.

2.1 Introduction

Theories and methodologies that make use of granules to solve problems featured by supplying huge amounts of data, information, and knowledge label a new area of multi-disciplinary study called Granular Computing (15) (89) (117) (141) (144) (151). Granular computing as a paradigm of information processing spotlights multiple levels of data detailing to often provide useful abstractions and approximate solutions to hard real-world problems (18) (19) (118) (146).

Granular information systems have appeared under different names in related fields such as interval analysis, fuzzy and rough sets, divide and conquer, quotient

space theory, information fusion, and others (see (141)). Elementary processing units in granular systems are referred to as information granules. An information granule is defined as a clump of entities that may originate at the numeric (singular) or granular level and are arranged together due to their similarity, proximity, indistinguishability, or coherency.

The goal of a granule is to catch the very essence of the overall data in a concise and explainable manner (15) (118); it defines a subset of a universal set and conveys an internal representation. Granules may be interpreted from two points of view: from the perspective of uncertainty theory, they are units lacking precise knowledge; from that of knowledge engineering, they are units of elementary knowledge.

Granular computing is intended to identify manifestations of granules from moving back and forth among granularities to yield more or less differentiation. Too much detail is wasteful whereas too little renders a system useless. In general, there is no universal level of granularity of information: the size of granules is problem-oriented and user-dependent. Granularity is defined as the extent to which a larger and more complex system is broken down into smaller and simpler parts. We can quantify the granularity of a granule, for example, by counting its number of elements. The more elements are located in a granule, the lower is its granularity, and the higher is its generality (116). High granularities can produce substantial computational overhead for data storage. In excess, granularities and granules bring undesirable scalability issues such as incapacity to satisfy the required throughput. The granularity of information that is explicitly inbuilt into granules provides useful features in information systems modeling such as transparency and flexibility.

Let the result of data granulation be designated as a granular structure. A granular structure is a family of granules which, when considered together, reassemble the more complex original system. Handling a complex phenomenon by means of granular structures allows us to arrive at meaningful solutions. Based on some carefully chosen granularity, granular computing systems attempt to solve a problem by isolating its loosely connected sub-problems and handling them on an individual basis.

Granules of multiple sizes are related to the depth of penetration that charac-

terizes a system. A coarse granular structure contains fewer number of granules compared to a fine granular structure. This can be stated more precisely as follows. A coarse granular system regards a small amount of large granules usually characterized by low precision and high interpretability. A fine granular system regards a large amount of small granules, high precision, and limited interpretation. Low-level refined granules provide details about the system functionality. More abstract, high-level granules are easier to manage and interpret, but may lose important minutiae.

Input and output data sets generate input and output granular structures, respectively, which should be somehow connected. We name the correspondence between input and output granular structures as granular mapping. A granular mapping is defined over information granules lying in an input space and maps them into a collection of granules expressed in some output space. Granular mappings can be encountered quite frequently in rule-based systems, where the mapping is given as If-Then statements (18).

In granular computing, everything, including data, variables and parameters, is allowed to be granular. In general, inaccurate measurements and perception-based information are granular, for example: 'x is small', 'approximately 90', 'temperature is high', 'probability is high', [20, 25]. In this sense, a granular system provides NL-capability (126), that is, capability to operate on information described in Natural Language. NL-capability is important because much of human knowledge is described in natural language. Imprecision of human sensory organs and brain is passed on to natural language (154). More specifically, when a proposition expressed in a natural language is represented as a system of generalized constraints (153), it is, in effect, a granular system. Computation with information described in natural language ultimately reduces to computation with granular values.

Computing with granules brings together existing formalisms of interval analysis, fuzzy sets, rough sets, etc. under one roof. In spite of several visible distinct underpinnings of these theories, they exhibit fundamental synergies, which are exploited in the granular computing framework (117).

2.2 Interval Analysis

Interval analysis is a branch of mathematics that provides reliable numerical tools for problem solving; it treats an interval both as a set and as a number (53) (60) (69) (103) (104) (109). While arithmetic performs operations on numbers, interval arithmetic performs operations on intervals. Generally speaking, intervals are instances of granules. Granular computing materializes in the framework of interval analysis and provides features for interpretability.

Interval analysis is a theory oriented toward computational implementation because it supports the development of interval-based granular algorithms. These algorithms are mainly designed to automatically provide rigorous bounds on approximation errors, rounding errors, and propagated uncertainties in initial data. This is of utmost importance because modeling of complex systems must compromise complexity and precision. Operations involving imprecise objects must consider the nature of the imprecision.

The main concern of the interval analysis is to provide a guaranteed approximation of the set of solutions of the underlying problem. 'Guaranteed' in this context means that outer approximations (enclosure) of intervals can always be obtained and, moreover, be made as precise as desired when further information yields intervals of narrower width. Intervals acknowledge limited precision by associating with a variable of the model under investigation a set of reals as possible values. For ease of storage and fast computation, these sets are restricted to intervals (56). Essentials of interval theory, which form a background of fundamentals for our investigations, are summarized next.

2.2.1 Interval Vectors

An interval I is a closed bounded set of real numbers

$$[l, L] \;=\; \{x : l \leq x \leq L\}, \tag{2.1}$$

where l and L denote its endpoints. An n-dimensional interval vector is an ordered n-tuple of intervals (I_1, ..., I_j, ..., I_n). If I is, e.g., a two-dimensional

interval vector, then $I = (I_1, I_2)$ for some $I_1 = [l_1, L_1]$ and $I_2 = [l_2, L_2]$.

Set-theoretic operations of intersection, $\cap$, and union, $\cup$, are applicable to intervals. The intersection of two intervals, I^1 and I^2, is empty, $I^1 \cap I^2 = \emptyset$, if either $l^1 > L^2$ or $L^1 < l^2$. This indicates that I^1 and I^2 have no common points. Otherwise, the intersection of I^1 and I^2 is again an interval:

$$I^1 \cap I^2 \quad = \quad [max(l^1, l^2),\ min(L^1, L^2)]. \tag{2.2}$$

The intersection of interval vectors is empty if the intersection of any of their items is empty. Otherwise, for $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$ we have:

$$I^1 \cap I^2 \quad = \quad (I_1^1 \cap I_1^2, ..., I_j^1 \cap I_j^2, ..., I_n^1 \cap I_n^2). \tag{2.3}$$

If two intervals have nonempty intersection, then their union,

$$I^1 \cup I^2 \quad = \quad [min(l^1, l^2),\ max(L^1, L^2)], \tag{2.4}$$

is an interval. Disconnected sets must not be expressed as a single interval.

The convex hull of two interval vectors, I^1 and I^2, namely $ch(I^1, I^2)$, is the smallest interval vector containing all their elements. Then,

$$ch(I_j^1, I_j^2) \quad = \quad [min(l_j^1, l_j^2),\ max(L_j^1, L_j^2)],\ j = 1, ..., n. \tag{2.5}$$

Hull computation is an efficient procedure to combine sets independently of their connection. It follows that $I^1 \cup I^2 \subseteq ch(I^1, I^2)$ for any I^1 and I^2.

If $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$ are interval vectors, then

$$I^1 \subseteq I^2 \quad \text{if and only if} \quad I_j^1 \subseteq I_j^2,\ j = 1, ..., n. \tag{2.6}$$

We denote the width of an interval vector, namely $wdt(I)$, as the length of its largest side:

$$wdt(I) \quad = \quad max(wdt(I_1), ..., wdt(I_j), ..., wdt(I_n)), \tag{2.7}$$

where,

$$wdt(I_j) \quad = \quad L_j - l_j, \;\; j = 1, ..., n. \tag{2.8}$$

Finally, it is worth defining the midpoint of an interval I:

$$mp(I) \quad = \quad \frac{l + L}{2}. \tag{2.9}$$

Analogously, if $I = (I_1, ..., I_j, ..., I_n)$ is an interval vector, then:

$$mp(I) \quad = \quad (mp(I_1), ..., mp(I_j), ..., mp(I_n)). \tag{2.10}$$

2.2.2 Interval Arithmetic

Operations on real numbers can be extended to intervals. Interval arithmetic treats intervals as numbers: adding, subtracting, multiplying, and dividing them.

The rules for interval addition and subtraction are:

$$I^1 + I^2 \quad = \quad [l^1, L^1] + [l^2, L^2] = [l^1 + l^2, L^1 + L^2], \tag{2.11}$$

$$I^1 - I^2 \quad = \quad [l^1, L^1] - [l^2, L^2] = [l^1 - L^2, L^1 - l^2]. \tag{2.12}$$

Operations of addition and subtraction for interval vectors are understood to be component-wise. For two interval vectors, $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$, we have

$$I^1 + I^2 = (I_1^1 + I_1^2,\ ...,\ I_j^1 + I_j^2,\ ...,\ I_n^1 + I_n^2), \tag{2.13}$$
$$I^1 - I^2 = (I_1^1 - I_1^2,\ ...,\ I_j^1 - I_j^2,\ ...,\ I_n^1 - I_n^2). \tag{2.14}$$

For the product of two independent intervals, I^1 and I^2, we get

$$I^1 I^2 = \{x^1 x^2 : x^1 \in I^1, x^2 \in I^2\}. \tag{2.15}$$

Clearly, the result is again an interval, say I^3, whose endpoints are

$$[l^3, L^3] = [min(l^1 l^2,\ l^1 L^2,\ L^1 l^2,\ L^1 L^2),\ max(l^1 l^2,\ l^1 L^2,\ L^1 l^2,\ L^1 L^2)]. \tag{2.16}$$

The reciprocal of an interval I yields:

$$1/I = \{1/x : x \in I\}. \tag{2.17}$$

If I is an interval not containing the number 0, then $1/I = [1/L, 1/l]$ if $l > 0$; or $1/I = [1/l, 1/L]$ if $L < 0$. In case I contains 0 so that $l \leq 0 \leq L$, then the set is unbounded and cannot be represented as an interval whose endpoints are real numbers. For the quotient of two intervals, we have:

$$I^1/I^2 = I^1(1/I^2) = \{x^1/x^2 : x^1 \in I^1, x^2 \in I^2\}. \tag{2.18}$$

I^1/I^2 is again an interval if 0 is not contained in I^2. I^1 and I^2 are independent.

The product and quotient operations for interval numbers hold for interval vectors. For two interval vectors, $I^1 = (I_1^1, ..., I_j^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_j^2, ..., I_n^2)$, it follows that:

$$I^1 I^2 = (I_1^1 I_1^2,\ ...,\ I_j^1 I_j^2,\ ...,\ I_n^1 I_n^2), \tag{2.19}$$
$$I^1/I^2 = (I_1^1/I_1^2,\ ...,\ I_j^1/I_j^2,\ ...,\ I_n^1/I_n^2). \tag{2.20}$$

2.2.3 Distance Between Intervals

A suitable metric to measure the distance between two intervals, I^1 and I^2, is:

$$d(I^1, I^2) = max(|l^1 - l^2|, |L^1 - L^2|). \tag{2.21}$$

With this metric, the correspondence between the interval number system and the real number system, $[x, x] \leftrightarrow x$, holds (106). The metric $d(.)$ preserves the distance between the corresponding items. We have that

$$d([x^1, x^1], [x^2, x^2]) = max(|x^1 - x^2|, |x^1 - x^2|) = |x^1 - x^2| \tag{2.22}$$

for any x^1 and x^2. The real line is isometrically embedded into the metric space of intervals (106).

The distance between two interval vectors, $I^1 = (I_1^1, ..., I_n^1)$ and $I^2 = (I_1^2, ..., I_n^2)$,

$$d(I^1, I^2) = (max(|l_1^1 - l_1^2|, |L_1^1 - L_1^2|),\ ...,\ max(|l_n^1 - l_n^2|, |L_n^1 - L_n^2|)), \tag{2.23}$$

is an interval vector. Sometimes, we are more interested in a number to represent the overall distance between interval vectors. A measure for the overall distance between two interval vectors, I^1 and I^2, is

$$D(I^1, I^2) = max(d(I^1, I^2)). \tag{2.24}$$

2.2.4 Interval Functions

Consider a real-valued function $f(x)$ and a corresponding interval-valued function $f(I)$. $f(I)$ is a united extension of $f(x)$ if $f(I) = f(x)$ for any value of $x \in I$. If the parameters of $f(I)$ are degenerated, then $f(I)$ is a degenerated interval equal to $f(x)$. Formally, the image of an interval I under a real mapping f is

$$f(I) \quad = \quad \{f(x) : x \in I\}. \tag{2.25}$$

More generally, the image of a specified n-dimensional vector I admitting a multivariable real function f is:

$$f(I_1, ..., I_j, ..., I_n) \quad = \quad \{f(x_1, ..., x_j, ..., x_n) : x_j \in I_j \ \forall j\}. \tag{2.26}$$

Generally, the image of an interval through f is not a box (see Fig. 2.1) and it may be difficult to obtain in closed form. In practice, $f(I)$ can be approximated by an inclusion function $F(I)$, which is a box in the range of f if f is continuous.

An interval function F from $\mathbb{IR}^n$ to $\mathbb{IR}^m$ is called an interval inclusion function of f if

$$f(I) \subseteq F(I) \ \forall I \in \mathbb{IR}^n. \tag{2.27}$$

Inclusion functions are not unique and they depend on how we choose F. An inclusion function is optimal if $F(I)$ is the interval hull of $f(I)$. In other words, the optimal interval inclusion function for $f(I)$ is the smallest box $F^*(I)$ that contains $f(I)$. Figure 2.1 illustrates the idea. $F^*(I)$ is unique.

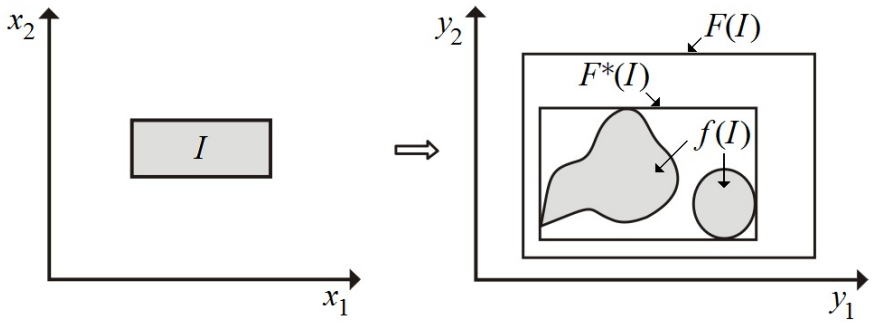


Figure 2.1: Image f of box I and inclusion functions F and F^*

In particular, for degenerated intervals I, it follows that:

$$F(I) = f(I) = F^*(I). \tag{2.28}$$

Consider f monotonically increasing in $I = [l, L]$. Then, assuming continuity or upper semicontinuity of f, we can obtain $f(I)$ using:

$$f(I) \quad = \quad [f(l), f(L)]. \tag{2.29}$$

Consequently,

$$f(x) \subseteq [f(l), f(L)] \; \forall x \in I. \tag{2.30}$$

With monotonic decreasing functions, we order the resulting endpoints properly. In these cases $f(I) = [f(L), f(l)]$, i.e. strict inclusion relationship holds.

Nonmonotonic functions could be monotonic under endpoint constraint. For example, $f(I) = sin(I)$ is not monotonic in general but defining $I = [-\Pi/2, \Pi/2]$, then $f(I)$ is monotonic and $f(I) = sin(I) = [sin(l), sin(L)]$.

An interval function $f(I)$ is inclusion isotonic when for any interval vectors, I^1 and I^2,

$$\text{if } I^1 \subset I^2, \text{ then } f(I^1) \subset f(I^2). \tag{2.31}$$

Finite interval arithmetic (104) is inclusion isotonic. Let $\bullet$ denote the operations of addition, subtraction, multiplication and division, thus

$$I^1 \bullet I^2 \subset I^3 \bullet I^4 \tag{2.32}$$

holds whenever $I^1 \subset I^3$ and $I^2 \subset I^4$. In this thesis all interval enclosures are inclusion isotonic interval extensions of real-valued continuous functions.

An interval function $f(I) \in \mathbb{IR}$ is called 'thin' when it involves only degenerate interval parameters or, equivalently, singular parameters. For instance,

$$f(I) = a_0 + \sum_{j=1}^{n} a_j I_j \tag{2.33}$$

is thin for $(a_0, ..., a_n)$ degenerated intervals. When an interval function involves at least one interval parameter of nonzero width, it is called 'thick'. This thesis considers thin interval functions only.

Interval analysis goes far beyond what has been covered in this section. For instance, we do not address interval statistics (49), intervals in fuzzy set theory (105), interval integration (106), complex interval arithmetic (120), but the essential to the completeness of this work.

2.3 From Interval Analysis to Fuzzy Set Theory

While interval analysis arose out of a need to analyze error and uncertainty on digital computers (103), fuzzy set theory arose from a need of more complete and inclusive mathematical models of uncertainty (149). Relationships between fuzzy set theory and interval mathematics have been reported by Lodwick (93).

Fuzzy arithmetic (67) is defined by means of the extension principle for fuzzy sets (70) (149). The extension principle for fuzzy sets is the united extension in the interval analysis terminology when the fuzzy set is restricted to be an interval (93). When intervals and fuzzy sets are non-interactive, arithmetic on alpha level sets is a united extension arithmetic. Both concepts are related fundamentally through what is known as set functions (131).

From the point of view of intervals as sets, interval analysis can be considered as a subset of the fuzzy set theory. For instance, an interval $[l, L]$ is a trapezoidal fuzzy set $[l, \lambda, \Lambda, L]$ where $l = \lambda$ and $\Lambda = L$ (138).

Fuzzy interval analysis (40) and interval type-2 fuzzy logic systems (100) (150) are explicit examples of joint efforts between fuzzy set theory and interval analysis to overcome the difficulties of uncertainty modeling.

Interval analysis and fuzzy set theory are instances of practical frameworks used to represent granular information and construct granular mappings. Conceptually, intervals and fuzzy sets are different ways to model imprecise quantities and capture our inherent notion of approximate numbers. 'Above 100' and 'around 1.5 and 1.7' are instances of intervals whereas 'approximately 100' and 'around 1.6' are instances of fuzzy sets.

A striking difference between intervals and fuzzy sets comes from the idea of partial membership intrinsic to fuzzy sets. Whenever interval quantification becomes too restrictive, fuzzy sets provide an important feature of describing information granules whose constituting elements may belong only partially. Fuzzy sets prevent defining hard borders between full belongingness and full exclusion by means of smooth transition boundaries. Granules formalized in the language of fuzzy sets support a vast array of human-centric pursuits (116).

2.4 Fuzzy Sets

Fuzzy sets (70) (149) constitute one of the most influential notions in science and engineering. A fuzzy set captures in a granular way the essential in which much of physical phenomena is observed and described. Fuzzy information granulation underlies the basic concepts of linguistic variables, fuzzy rules, and fuzzy rule base (116). In fuzzy set theory, objects, variables and concepts are a matter of degree. In particular, fuzzy information granulation allows both, incorporation of domain knowledge and knowledge discovery from data.

Fuzzy sets extend the notion of set by assigning to each element of a reference set a value representing its degree of membership in the fuzzy set. Membership values correspond to the degree the element is similar with typical elements representing the concept associated with the fuzzy set. This characteristic of fuzzy sets facilitates the management of the uncertainty carried by such elements.

Concepts and definitions related to fuzzy sets which are useful for our investigations are summarized in next.

2.4.1 Fuzzy Set Definitions

Fuzzy sets are fully characterized by their membership functions. Any function $A : X \rightarrow [0, 1]$ may serve as a membership function of fuzzy set A. In this thesis we assume trapezoidal membership functions, which are piecewise linear functions described by four parameters (l, λ, Λ, L). The membership degree of an element x in the trapezoidal fuzzy set A is

$$A(x) = \begin{cases} 0, & x < l \\ \frac{x-l}{\lambda-l}, & x \in [l, \lambda[\\ 1, & x \in [\lambda, \Lambda] \\ \frac{L-x}{L-\Lambda}, & x \in]\Lambda, L] \\ 0, & x > L \end{cases} \tag{2.34}$$

A fuzzy set A is normal if it produces a membership degree equal to 1 for at least one element x of the universe X. Denote *sup* as the supremum value of A for some element x; then A is normal if

$$sup_{x \in X} A(x) = 1. \tag{2.35}$$

We denote support and core of a trapezoidal membership function A respectively as the set of elements of X with nonzero membership degrees in A, and the set of elements of X with membership degrees equal to 1, that is, for a trapezoidal membership function A,

$$supp(A) = \{x \in X | A(x) > 0\} = [l, L], \quad \text{and} \tag{2.36}$$

$$core(A) = \{x \in X | A(x) = 1\} = [\lambda, \Lambda]. \tag{2.37}$$

The α-cut of a fuzzy set A, A_α, is a set containing all elements of X whose membership degrees are greater than the value α. We have

$$A_\alpha = \{x \in X | A(x) > \alpha\}. \tag{2.38}$$

Support ($\alpha = 0$) and core ($\alpha = 1$) are boundary cases of α-level sets.

A fuzzy set is convex if for all $x^1, x^2 \in X$ and all $\kappa \in [0, 1]$ it follows that

$$A(\kappa x^1 + (1 - \kappa)x^2) \geq min(A(x^1), A(x^2)). \tag{2.39}$$

A fuzzy set A^1 is a subset of A^2 if and only if every element of A^1 is also an element of A^2:

$$A^1(x) \leq A^2(x), \text{ for all } x \in X. \tag{2.40}$$

The midpoint and width of a membership function A are, respectively:

$$mp(A) = \frac{\lambda + \Lambda}{2}, \tag{2.41}$$

$$wdt(A) = L - l. \tag{2.42}$$

Intersection and union of two fuzzy sets, say A^1 and A^2, are defined as

$$(A^1 \cap A^2)(x) = min(A^1(x), A^2(x)) \ \forall x \in X, \tag{2.43}$$

$$(A^1 \cup A^2)(x) = max(A^1(x), A^2(x)) \ \forall x \in X. \tag{2.44}$$

The convex hull of two trapezoidal fuzzy sets A^1 and A^2 is a trapezoidal fuzzy set determined as follows:

$$ch(A^1, A^2) = (min(l^1, l^2), min(\lambda^1, \lambda^2), max(\Lambda^1, \Lambda^2), max(L^1, L^2)). \tag{2.45}$$

2.4.2 Fuzzy Interval

Granular data may take various forms depending on how they are modeled. They can be intervals, probability distributions, rough sets, fuzzy numbers, and fuzzy intervals (42). Fuzzy intervals and fuzzy numbers are instances of fuzzy granular data. Fuzzy data arise in the realm of expert knowledge, whenever measurements are inaccurate, variables are hard to be precisely quantified, or pre-processing steps introduce uncertainty in singular data.

A membership function $A : X \rightarrow [0, 1]$ is upper semi-continuous if the set $\{x \in X | A(x) > \alpha\}$ is closed, that is, if the α-cuts of A are closed intervals. If the universe X is the set of real numbers and A is normal, $A(x) = 1 \; \forall x \in [\lambda, \Lambda]$, then A is a model of a fuzzy interval, with monotone increasing function ϕ_A: $[l, \lambda[\rightarrow [0, 1]$, monotone decreasing function ι_A: $]\Lambda, L] \rightarrow [0, 1]$, and zero otherwise. A fuzzy interval A has the following canonical form:

$$A(x) = \begin{cases} \phi_A, & x \in [l, \lambda[\\ 1, & x \in [\lambda, \Lambda] \\ \iota_A, & x \in]\Lambda, L] \\ 0, & \text{otherwise} \end{cases}, \tag{2.46}$$

where x is a real number in X. The fuzzy interval A satisfies the conditions of normality ($A(x) = 1$ for at least one $x \in X$) and convexity ($A(\kappa x^1 + (1 - \kappa)x^2) \geq min\{A(x^1), A(x^2)\}$, $x^1, x^2 \in X$, $\kappa \in [0, 1]$). If

$$\phi_A = \frac{x - l}{\lambda - l} \text{ and} \tag{2.47}$$

$$\iota_A = \frac{L - x}{L - \Lambda}, \tag{2.48}$$

then the fuzzy membership function (2.46) reduces to the model of a trapezoidal membership function (2.34). Moreover, when $\lambda = \Lambda$, then $A(x) = 1$ for one element x. In this case the corresponding fuzzy entity is called a fuzzy number (116). Fuzzy data generalize numeric data by allowing fuzziness.

2.4.3 Similarity Between Fuzzy Sets

Granular data and models are fuzzy objects of trapezoidal nature. In this case, a useful similarity measure for trapezoids, say A^1 and A^2, is:

$$S(A^1, A^2) = 1 - \frac{|l^1 - l^2| + |\lambda^1 - \lambda^2| + |\Lambda^1 - \Lambda^2| + |L^1 - L^2|}{4}. \quad (2.49)$$

This measure translates the relation between the trapezoids in a number. It returns 1 for identical trapezoids (indicating the maximum degree of matching between them) and decreases linearly when A^1 and A^2 withdraw from each other. Particularly, equation (2.49) is a Hamming-like metric (52) where the parameters of the trapezoids are compared one by one. A thorough discussion of similarity and compatibility measures can be found in (33).

The distance between two vectors of trapezoids, say $A^1 = (A^1_1, ..., A^1_n)$ and $A^2 = (A^2_1, ..., A^2_n)$,

$$S(A^1, A^2) = 1 - \frac{1}{4n}\sum_{j=1}^{n}(|l^1_j - l^2_j| + |\lambda^1_j - \lambda^2_j| + |\Lambda^1_j - \Lambda^2_j| + |L^1_j - L^2_j|), \quad (2.50)$$

is also a number, which quantifies their relationship.

2.5 Aggregation Operators

Aggregation operators $\mathbb{C} : [0,1]^n \to [0,1]$, $n > 1$ combine input values in the unit hypercube $[0,1]^n$ into a single output value in $[0,1]$. They must satisfy two fundamental properties: (***i***) monotonicity in all arguments, i.e., given $x^1 = (x^1_1, ..., x^1_n)$ and $x^2 = (x^2_1, ..., x^2_n)$, if $x^1_j \leq x^2_j$ $\forall j$ then $\mathbb{C}(x^1) \leq \mathbb{C}(x^2)$; (***ii***) boundary conditions: $\mathbb{C}(0, 0, ..., 0) = 0$ and $\mathbb{C}(1, 1, ..., 1) = 1$. Important classes of aggregation operators are summarized below. See (20) (116) for details.

2.5.1 T-norm Aggregation

T-norms (T) are commutative, associative, and monotone operators on the unit hypercube whose boundary conditions are $T(\alpha, \alpha, ..., 0) = 0$ and $T(\alpha, 1, ..., 1) = \alpha$, $\alpha \in [0, 1]$. The neutral element of T-norms is $e = 1$. An example is the minimum operator:

$$T_{min}(x) = \min_{j=1,...,n} x_j, \tag{2.51}$$

which is the strongest T-norm because

$$T(x) \leq T_{min}(x) \text{ for any } x \in [0, 1]^n. \tag{2.52}$$

The minimum is also idempotent, symmetric, and Lipschitz-continuous. Further examples of T-norms include the product,

$$T_{prod}(x) = \prod_{j=1}^{n} x_j, \tag{2.53}$$

and the Lukasiewicz T-norm,

$$T_L(x) = max(0, \sum_{j=1}^{n} x_j - (n - 1)). \tag{2.54}$$

2.5.2 S-norm Aggregation

S-norms (S) are operators on the unit hypercube which are commutative, associative, and monotone. $S(\alpha, \alpha, ..., 1) = 1$ and $S(\alpha, 0, ..., 0) = \alpha$ are the boundary conditions of S-norms. It follows that $e = 0$ is the neutral element of S-norms.

S-norms are stronger than T-norms. The maximum operator:

$$S_{max}(x) = \max_{j=1,...,n} x_j, \tag{2.55}$$

is the weakest S-norm, that is,

$$S(x) \geq S_{max}(x) \geq T(x), \text{ for any } x \in [0,1]^n. \tag{2.56}$$

Other examples of S-norms include the probabilistic sum,

$$S_{prob}(x) = 1 - \prod_{j=1}^{n}(1 - x_j), \tag{2.57}$$

and the Lukasiewicz S-norm,

$$S_L(x) = min(1, \sum_{j=1}^{n} x_j). \tag{2.58}$$

The dual $\mathbb{C}_D$ of an aggregation operator $\mathbb{C}$ is

$$\mathbb{C}_D(x_1, ..., x_n) = 1 - \mathbb{C}(1 - x_1, ..., 1 - x_n). \tag{2.59}$$

Maximum and minimum, probabilistic sum and product, and Lukasiewicz S and T-norms are examples of self-dual aggregation operators.

2.5.3 Uninorm Aggregation

Uninorms (U) are bivariate, associative and symmetric operators closed under duality. Similarly as with T-norms and S-norms, associativity allows n-ary extension of uninorms. Uninorms $U : [0,1]^n \to [0,1]$ generalizes triangular norms by relaxing the assumption about the neutral element e to get values in $[0,1]$. Input values higher than e are interpreted as beneficial, a positive evidence; input values lower than e are considered detrimental, a negative evidence. Naturally, when e is equal to 0 a uninorm turns into an S-norm and when $e = 1$ the uninorm becomes a T-norm.

This work considers the following family of uninorms:

$$U(x) = \begin{cases} e \;\; T \;\; \left(\frac{x_1}{e}, ..., \frac{x_n}{e}\right) & \text{if } x \in [0, e]^n \\ (e + (1 - e)) \;\; S \;\; \left(\frac{x_1 - e}{1 - e}, ..., \frac{x_n - e}{1 - e}\right) & \text{if } x \in [e, 1]^n \\ T(x_1, ..., x_n) & \text{otherwise,} \end{cases} \tag{2.60}$$

where $e \neq 0$ and $e \neq 1$. Any pair of T and S norms may be used to construct the uninorm U independently of their properties or duality.

2.5.4 Averaging Aggregation

An aggregation operator $\mathbb{C}$ is averaging if for every $x \in [0, 1]^n$ it is bounded by

$$T_{min}(x) \leq \mathbb{C}(x) \leq S_{max}(x). \tag{2.61}$$

The basic rule is that the output value cannot be lower or higher than any input value. An example of averaging operator is the arithmetic mean:

$$M(x) = \frac{1}{n} \sum_{j=1}^{n} x_j. \tag{2.62}$$

Averaging operators are assumed to be idempotent, strictly increasing, symmetric, homogeneous, and Lipschitz continuous.

2.5.5 Compensatory T-S Aggregation

Compensatory T-S operators combine T-norms and S-norms to counterbalance their opposite effects. Contrary to uninorm aggregation, T-S aggregation is uniform in the sense that it does not depend on parts of the underlying domain.

T-S operators use both a T-norm and a S-norm and averages the two values obtained by means of a weighted quasi-arithmetic mean. The linear convex operator

$$L(x) = (1 - v)T(x_1, ..., x_n) + vS(x_1, ..., x_n), \tag{2.63}$$

where $v \in [0, 1]$, is an example of T-S operator of the family of weighted quasi-arithmetic means. T-S operators need not to be dual in terms of T and S. It follows that:

$$S(x) \geq L(x) \geq T(x), \text{ for any } x \in [0, 1]^n. \tag{2.64}$$

2.6 Summary

This chapter has addressed principles and definitions of granular computing that are useful for the comprehension of subsequent chapters. We argued that information granulation plays a primary role both in handling data of uncertain nature and in representing concepts described in natural language. We emphasized interval and fuzzy granular computing frameworks - with intervals and fuzzy sets being instances of information granules. When processing granular data we are in fact handling a significant number of similar individual elements at the same time and therefore ignoring details. This chapter also covered aggregation operators which are pertinent for information fusion within granular computing environment.

Chapter 3

Evolving Granular Systems

Evolving granular systems are a modeling framework that considers online granular data stream processing and structurally adaptive rule-based models. As uncertain data prevail in stream applications, excessive data granularity becomes unnecessary and inefficient. This chapter starts with the motivation which led to the development of evolving intelligent systems. We briefly summarize the main historical landmarks of the research area leading to the state of the art. Next, we introduce evolving granular systems, which extend evolving intelligent systems allowing data, variables and parameters to be granular (intervals and fuzzy sets). The aim of evolving granular systems is to fit the information carried by input-output data streams from online nonstationary processes into rule-based models and, at the same time, provide granular approximation of functions and linguistic description of the system behavior.

3.1 Introduction

Adaptability is of paramount importance for intelligent systems. As Darwin quoted (35), it is neither the strongest nor the most intelligent that survives, but the most adaptable to change. Building adaptive models from large volumes of real-world online data flows requires developing non-conventional learning algorithms able to continuously track system and environment changes. Rethinking traditional data mining and modeling techniques is primordial to support struc-

tural adaptation of information systems based on sequences of data, possibly of uncertain nature.

Because data acquisition systems and small scale computing devices became mere components of complex systems, large amounts of data have been produced uninterruptedly. Storage of large-scale data sets and offline processing are frequently impractical, especially in online applications. In addition, data from different sources may be temporally and spatially related. Online learning algorithms should benefit from time and space data stream correlations to capture essential information and recursively translate it into structured knowledge. The effectiveness of data stream-oriented learning algorithms is rooted in their aptitude to quickly evolve models from nonstationary data.

Learning system models from data streams in online mode is a challenging task for most statistical and computational intelligence methods. Adaptive - and naturally non-adaptive - learning methods face a number of drawbacks when dealing with evolving data streams including: (*i*) difficulty in choosing the model structure since data sets and related information are not available; (*ii*) forgetfulness when trying to acquire new information after concept changes; and (*iii*) limited transparency and interpretability of the resulting model. In particular, there is a need for developing recursive learning methods that explore the nature of data streams (11) and at the same time fulfill accuracy, transparency and interpretability requirements (4).

3.2 Evolving Intelligent Systems

Approaches to extract meaningful information from data streams have recently been developed (1) (9) (10) (24) (25) (32) (46) (59) (66) (75) (79) (83) (84) (94) (122) (124) (125) (134). Methods and algorithms directed toward this end are known as Evolving Intelligent Systems. Evolving intelligent systems focus on nonstationary processes and embody online learning methods and one-pass incremental algorithms that evolve or gradually change individual models to guarantee life-long learning and self-organization of the system structure.

Evolving systems are a step toward a higher level of adaptability compared to conventional adaptive systems from control theory (13), classical identification

systems (92), and traditional data mining systems (54) (135). While the term 'intelligent' comes from the use of fuzzy and neuro-fuzzy (computational intelligence) techniques, the evolving aspect of these systems accounts for unbounded (infinite) amounts of data, changing concepts, and structural adaptation of models.

Formally stated, a system is said to be evolving if it:

- learns continuously from data streams;
- does not store previous samples;
- does not depend upon prior structural knowledge;
- self-adapts its structure when needed;
- is independent of statistical properties of data; and
- does not use 'prototype' initialization.

Moreover, it is much desired that evolving systems assimilate knowledge fast using small memory requirements to support real-time applications. Evolving systems must account for the fact that the unknown is likely to matter.

In terms of implementation, evolving systems usually achieve their final purpose in software level, but they may be performed in physical embodiments including intelligent agents, embedded systems, and ubiquitous computing (11).

3.2.1 Historical Landmarks

In the beginning of this century, two mainstreams of research in evolving intelligent systems were introduced: evolving fuzzy systems (5) and evolving connectionist systems (65). Their origins are independent of one another.

Evolving fuzzy systems (eFS) were proposed by Angelov (5), being evolving Takagi-Sugeno (eTS) fuzzy systems (6) a milestone in the field of structurally adaptive rule-based systems. The eTS is an eFS paradigm for function approximation and control that fulfils the requirements for flexible and adaptive approaches of a variety of modern applications such as automation processes,

autonomous systems, intelligent sensors, and defense. eTS assumes that the antecedent and consequent parameters of functional fuzzy rules as well as the number of rules in a rule base can gradually change by learning from experience based on data streams. This characteristic provides eTS approaches with the fundamental ability to pursue online modeling of time-varying nonstationary functions. Evolving fuzzy classifiers (eClass) (8) (10) are another approach derived from eFS when the consequent part of fuzzy rules is a class label. In eClass the number of classes needs not be known in advance and new classes can be incorporated at any time. eClass models were seminal to the field of evolving classifiers which possess the ability to capture both concept drift and shift (95).

Evolving connectionist systems (eCOS) were proposed by Kasabov (65) (66). eCOS are artificial neural networks that operate continuously in time and adapt their structure and functionality through interaction with the environment and other systems. A paradigm of eCOS is called evolving fuzzy neural network (EFuNN) (63), which is the earliest and perhaps most influential model of eCOS. All neurons in EFuNN are created and updated during learning. They represent membership functions and rules. Information carried by a data stream is memorized on neurons and connections, and further used for predictions. The EFuNN structure evolves from hybrid (supervised and unsupervised) algorithms. Particularly, the fuzzy aspect of EFuNN permits the neural network to be interpreted as a fuzzy rule-base. Other noteworthy approaches supporting the context of eCOS are evolving self-organizing maps (eSOM) (34) and dynamic evolving neural-fuzzy inference systems (DENFIS) (64).

Common to both eFS and neurofuzzy eCOS are fuzzy sets, which are formed on a basis of numeric data through incremental clustering. Clusters give rise to fuzzy membership functions that considered together convey a global view of the available data. In evolving systems, fuzzy membership functions play a key role as the core of modeling approaches. They aim to represent similar data in a concise manner. After cluster identification, a recursive algorithm is usually used to refine local parameters and functions. In both platforms, eFS and neurofuzzy eCOS, expert knowledge can be incorporated, but it is not compulsory.

From the granular computing point of view, eFS and great part of eCOS can be considered granular modeling frameworks. Fuzzy sets, used to represent numeric

data, are instances of granules whereas computations in eFS and eCOS are based on the result of information granulation. However, in general, evolving intelligent systems cannot be regarded as evolving granular systems in the greatest sense of the term because they do not deal with input and output granular data and quite often do not produce granular estimation. In other words, evolving systems are granular systems internally, and singular systems externally.

Since the conception of evolving intelligent systems a diversity of studies suggesting extension of the original content has taken place. Approaches regarding primarily computational intelligence principles and ideas follow the essential notions of the original evolving intelligent systems. Conversely, there exist parallel research lines where structurally adaptive learning approaches from data streams are mostly based on data mining and statistics. Such approaches are often not referred to as 'evolving'; however, the central idea of capturing gradual and abrupt changes in nonstationary data streams is the same independently of the different terminologies. The next section reviews some state-of-the-art works.

3.2.2 State of the Art

This section summarizes recent research related to learning methods capable of handling numeric data streams. We do not intend to give an exhaustive review of the literature. The purpose is to overview works closely related to the approaches addressed in this thesis.

The evolving participatory learning (ePL) approach (87) combines the concept of participatory learning (137) with evolving Takagi-Sugeno fuzzy systems. The ePL approach is based on unsupervised clustering and therefore is a candidate to find rule base structures in adaptive fuzzy modeling. ePL uses participatory learning fuzzy clustering instead of scattering or information potential-based clustering used by eTS. At each time step, ePL updates the rule base structure using convex combinations of new data samples and the closest cluster center. The parameters of the consequent part of a rule are adapted using a recursive least squares algorithm.

The evolving multivariable Gaussian approach (eMG) (84) is an evolving functional fuzzy modeling approach which, differently from eTS, uses an evolving

Gaussian clustering algorithm based on the concept of participatory learning. The clustering algorithm is one-pass and updates the eMG rule base continuously. Fuzzy sets in eMG are multivariable Gaussian membership functions which are adopted to preserve information between input variable interactions. The parameters of the membership functions, that is, cluster centers and dispersion matrices, are estimated by the clustering algorithm. A weighted recursive least squares algorithm updates the parameters of the rule consequents. The eMG clustering algorithm is particularly robust to noisy data and outliers through the use of a mechanism to smooth incompatible input data.

A data-driven incremental algorithm called flexible fuzzy inference system (FLEXFIS) was proposed in (94) to evolve Takagi-Sugeno fuzzy systems. A modified version of vector quantization was suggested for rules evolution. The FLEXFIS algorithm adapts linear functions of rules consequent and premise parameters (fuzzy membership functions) in online mode. Clusters of data are automatically generated based on the nature, distribution and quality of new data. Convergence toward the optimal parameter set in the least-squares sense has been achieved by the algorithm.

Self-organizing fuzzy modified least-square neural network (SOFMLS) (124) is a neurofuzzy network capable of adapting itself in real-time to a changing environment. In SOFMLS, parametric and structural model adaptation is performed simultaneously. The neural network generates a new rule if the smallest distance between a new numeric data vector and rule parameters is higher than a prespecified radius. A density-based pruning procedure controls the network growth over time. SOFMLS does not require retraining of the whole model and has proved to be able to escape from local minima and be stable to concept changes.

The fuzzy min-max neural network (GFMM) (46) is a generalization of the fuzzy min-max clustering and classification neural networks (129) (130). It handles labeled and unlabeled data simultaneously in a single neural model. GFMM combines supervised and unsupervised learning to give hybrid clustering and classification. The learning process places and adjusts hyperboxes (expansion-contraction paradigm) in the feature space in a few or one pass over data sets. GFMM is able to classify interval data and can be viewed as an incremental granular classifier.

Learn++.NSE (43) is an ensemble of classifiers-based approach for time-varying data distribution modeling. Learn++.NSE considers consecutive batches of data and makes no assumptions about the nature and rate of concept drift. The algorithm learns incrementally, similar to other algorithms of the Learn++ family (107) (121). Learn++.NSE trains one new classifier for each batch of data it receives and combines these classifiers using a dynamically weighted majority voting procedure. This procedure allows the algorithm to recognize and react to changes in the underlying data distributions. Since data batches are discarded after use, Learn++.NSE is suitable for online modeling of large volumes of data.

Very fast decision trees (VFDT) (38) is a method to discover knowledge in databases that builds decision trees using constant memory space and constant time to process a sample. VFDT operates on high-volume data streams and gradually creates branches and leaves if necessary. The approach uses Hoeffding bounds to guarantee that its output is asymptotically nearly identical to that of a conventional batch learner. VFDT is designed for classification purpose.

The ultra fast forest of trees (UFFT) (48) is a one-pass incremental algorithm able to detect concept drift. Trees are split according to new information appearing in a numeric data stream. In multi-class classification problems UFFT builds a binary tree for each possible pair of classes, leading to a forest of trees. Decision nodes and leaves contain naive Bayes classifiers to detect changes in class distribution and classify test examples. When changes in class distributions are detected, sub-trees rooted at representative nodes are pruned.

Differently from VFDT and UFFT, evolving fuzzy linear regression trees (eFRT) (83) (85) convey a linear regression model in each leaf. Thus, eFRT can be used for function approximation and prediction. In general, the number of tree nodes and the number of inputs can be changed given a new sample. The tree starts with a single leaf and grows replacing leaves with sub-trees and adding more variables to the regression model. The eFRT topology is updated on the fly using a statistical model selection test that considers accuracy and number of parameters to provide accurate and parsimonious trees.

Massive Online Analysis (MOA) (23) is a software environment for learning from evolving data streams. MOA supports incremental classification and clustering approaches that do not scale with the volume of information. For classifi-

cation, MOA considers boosting, bagging, and Hoeffding trees with and without naive Bayes classifiers at the leaves. For clustering, it implements the algorithms StreamKM++, CluStream, ClusTree, Den-Stream, D-Stream, and CobWeb. The aim of MOA is to provide analysis tools and insight about real-world data stream mining problems. MOA can interact with the software WEKA, the Waikato Environment for Knowledge Analysis (135).

3.3 Granular Data Streams

Physical systems change over time and usually produce considerable amounts of nonstationary data. Data streams in online environment can be granular from different perspectives. A more intuitive perspective concerns data that are granular by themselves. To elaborate, consider a simple example of predicting variable y from the last available observation x. This leads us to search for an approximand p to describe the process function f based on pairs (x, y). Here, instances x and y are singular (real numbers), and function f is single-valued. Singular data do not restrain models to be singular but rather a granular system may use granular models whose size and placement reflect the information carried by singular data. A hypothesis is that granular representation helps to assess the structure of detailed singular data and organizes the data into a more interpretable format.

Consider $x = [\underline{x}, \overline{x}]$ and $y = [\underline{y}, \overline{y}]$ as instances of a granular data stream, intervals in this case. To exemplify, $\underline{x}$ and $\overline{x}$ may denote the minimum and maximum price of an economical index during a day, and $\underline{y}$ and $\overline{y}$ the range of fluctuation of the price in the next day. In this example, data are originally granular, and models $[\underline{p}, \overline{p}]$ must be granular to support granular data. Figure 3.1 illustrates the granular modeling approach for function approximation.

Figures 3.1(a) and 3.1(b) show that granular models outer approximate single-valued and granular functions, respectively. Outer approximations of functions can always be obtained, e.g., at the top level, the coarsest possible granular approximation is the problem domain. Although merely enclosing a solution may sound at first shallower than finding the solution itself, we should reflect that the degree of satisfaction involved in embracing a solution depends strongly on the width of the enclosure obtained (60). Moreover, when processing stream data,

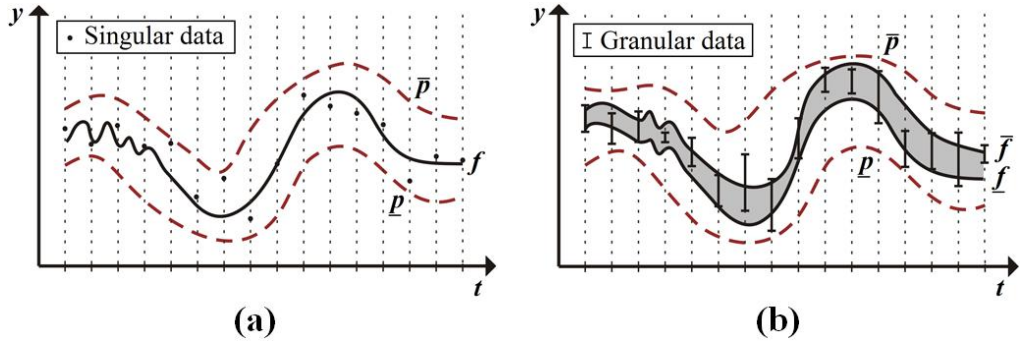


Figure 3.1: Granular models: (a) single-valued function, (b) granular function

we rarely have an idea about the error range and uncertainty associated with the data. On the contrary, if we can compute with granules containing a solution, then we can take for example the midpoint as a numeric approximation. Hence, we obtain both an approximate numeric solution and tolerance bounds on the approximation. The key task of approximating functions with granules is seeking for the tightest envelope for the approximand.

Another perspective for the materialization of granules in data streams is concerned with the uncertainty introduced during preprocessing steps. Incomplete data makes precise discrimination of examples difficult. Missing values are usually predicted through imputation methods (91) (127) where the imputed data is uncertain by the very nature of the prediction and motivates granules. In privacy-preserving data mining, uncertainty may be added to the data in order to preserve the privacy of the results (3). Additionally, noise and disturbances of bounded-error dynamic context also demand information granulation.

Granular data may arise when measurements are inaccurate or variables are hard to be quantified. For example, in sensor streams imprecision arises from inaccuracies in the underlying data acquisition equipment. Often, data are purely numerical, but the process which generated the data is uncertain. In these cases, uncertainty in data representation may be useful to improve the quality of the results. For example, an instance with greater uncertainty may not be as important as one with smaller uncertainty.

Sometimes, stream data are derived from expert knowledge. Granular com-

puting provides a general framework to represent real-world perception in natural language (42) (138). Various considerations can affect one's choice of data representation. Foremost among these is what Zadeh calls cointention (154), the ability of the representing object to convey the meaning of the concept it is being used to represent (140).

In a nutshell, stream data can be intervals, probability distributions, rough sets, and fuzzy intervals (42). We define granular data streams as a sequence of samples that conveys granular information about a process. Evolving granular models are built from granular data streams. Interval and fuzzy granular data streams generalize numeric data streams by allowing interval representation and fuzziness.

3.4 Evolving Granular Modeling

Nonstationary granular system modeling encompasses adaptive and flexible learning procedures to deal with many types of data such as numbers, intervals, and fuzzy intervals. Granular computing provides a rich framework for modeling nonstationary systems using granular data streams.

Evolving granular modeling (6) (16) (66) (75) (77) (78) (79) (80) (81) (97) (119) comes not only as an approach to capture the essence of stream data but also as a framework to extrapolate spatio-temporal correlations from lower-level raw data and provide a more abstract human-like representation of them. Research effort into granular computing toward online environment-related tasks is supported by a manifold of relevant applications such as financial, health care, video and image processing, GPS navigation, click stream analysis, etc.

Our definition of evolving granular system is as follows: evolving granular systems are systems that are able to derive interpretable rule-based models and provide granular function approximation using an incremental learning algorithm and imprecise stream data (with imprecise data being numbers, intervals, fuzzy intervals, etc.) Association rules given in the form of If-Then statements can be extracted from an evolving granular construct at any time. The evolved rule base means, in essence, a granular description of a process.

In practice, evolving granular systems extend evolving intelligent systems in

their capability to handle singular and granular input-output data, and give single-valued and granular approximations of original single-valued or granular functions. Granular approximation comes with a linguistic description in addition to a numeric, pointwise approximation typical of evolving intelligent systems.

Evolving granular systems rely fundamentally on the concepts of granular view, information granule, and granular mapping (see Section 2.1) in the process of modeling stream data. Emphasis is on the tasks of data granulation and computing with granules (143) (145) (146) (152). The granularity of information explicitly embedded into granular systems offers valuable features in dynamic modeling such as transparency and flexibility. Naturally, we are concerned with a certain way of compressing granular data into more intelligible granular models.

Granular data streams are responsible for creation, expansion and shrinkage of granular models along one or more dimensions of the input and output spaces, guide parameter adaptation, and order the most appropriate granularities. Concept change, missing and noisy values, superfluous and outlier samples are common in online environments and require automatic intervention. Whenever a sample arrives, evolving algorithms should decide whether to discard it or to use it to update the current knowledge. Evolving granular learning algorithms designed to handle online granular data face odd challenges concerning the value of the current knowledge, which reduces as the concept changes; and the impossibility to neither store nor retrieve the data once read. Learning must be one-pass. Constructive (bottom-up) and decomposition-based (top-down) mechanisms predominate.

3.5 Time and Space Granulation

Data granulation may be performed in time and space domains. Approaches to building granules regard temporal granulation earlier than spatial granulation, as illustrated in Fig. 3.2. Temporal granulation precedes spatial granulation because of several reasons. Occasionally, samples are recorded at different time intervals, e.g., as in events stream. The need for synchronized analysis of manifold data streams and search for time-correlated structures give support to the possibility of considering temporal granulation first. Temporal granulation tends to slow down

the data flow once several streaming instances can be wrapped by a granular object and further computations be based on granules. Time granules grant synchronism and smaller amount of granular data for subsequent spatial analysis. Spatial correlation uniting heterogeneous data with multiple levels of granularity and different representations (intervals, fuzzy sets, rough sets, etc.) is captured during the process of spatial granulation. Structured representation of data is preserved over time as a synopsis of the data stream; it warrants structured problem solving at the practical level.

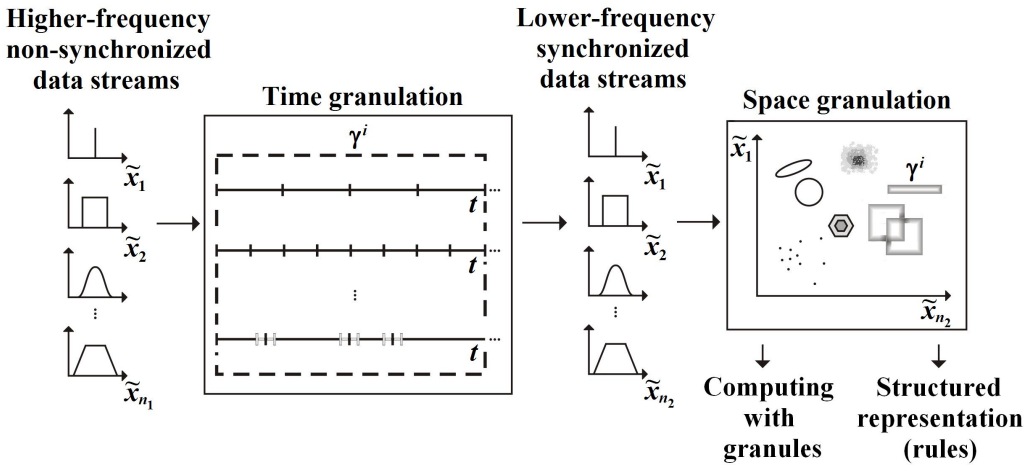


Figure 3.2: Time and space granulation

The flexibility of handling data streams using a granular computing framework enables us to describe granules in different application domains without deep knowledge about the problem. Tight time and memory constraints of online environment and interpretability requirements inspire granulated views of detailed data and computing at coarser granularities.

3.5.1 Time Domain Granulation

Time granulation aims at both reducing the sampling rate of fast data streams and synchronizing concurrent data streams that are input at random time intervals. A time granule describes the data for a certain time period.

Whenever the bounds of a time granule are aligned with significant shifts in the target function, the underlying granulation provides a good abstraction of the data. Conversely, if the alignment is poor, models may be inadequate (17). Manifold granularities require temporal reasoning and respective formalizations. Time granules and time windows are distinguished as follows.

Time window (74) (110) stands for a pre-specified or adaptive duration interval within which data samples assemble a representation. Generally, a fixed number of samplings or error values defines the size of the window. Windowing the time domain attempts to produce as few segments as possible to avoid data overfitting. Few time segments may hide information if the concept changes. Nonstationarity modifies "ideal" window lengths by its own dynamic. Approaches to testing window lengths are computationally costly and, hence, infeasible in environments with narrow time constraints. Essentially, there may exist several information granules in a time window. Data chunk analysis belongs to window-based approaches for information extraction and analysis.

A time granule groups data according to their indistinguishability in time. Since a time granule conveys similar data indexed in time, its bounds are naturally aligned with substantial changes in the function. The result of dynamic time granulation is a unique granule per segment. Time granules assume manifold levels of data abstraction and are aware of the pace of concept changes.

Event streams are examples of streams that usually come about at different time granularities. They require analysis of time-domain granules for commonalities extraction prior to space-domain analysis. Broadly stated, information evoked from time granules can be bounds of intervals, probability distributions or membership functions, and features such as frequency and correlation between events, patterns, prototypes. The internal structure of a granule and its associated variables provide full description and characterization of the granule.

Whenever manifold data streams mismatch each other at finer time granularities we resort to a granulated view of the time domain and a data mining and modeling approach. The resulting granulation should be at least as coarse as the coarsest individual stream to agree with the notion of outer approximation of functions and guaranteed solution (Section 2.1).

3.5.2 Space Domain Granulation

Data granulation over the space domain is a process of organization for comprehension (15). Granulation enables us to view different samples as being the same if low level details are neglected. Granulating the domain space is fundamental in methods of clustering (2) (17) and information integration (50). Resulting granules may compose antecedent and consequent parts of rules in rule-based systems (11) (71) (73).

Whenever variables are recorded simultaneously and the sampling frequency is not so high that we have enough time to step recursive algorithms, the time granulation stage can be ignored and efforts fully concentrated on spatial granulation. In fact, time and space granulation are somewhat related. For instance, (*i*) with the minimal and maximal values occurring in a time granule we may form an interval granular object; (*ii*) taking a representative mean or median of instances resting into a time granule and a confidence interval around it we may form a statistical granular object; (*iii*) capturing the core and the uncertainty of instances falling in a same time granule may give rise to a fuzzy granular object. Granular objects of any precedence may be taken into consideration as input to the stage of spatial granulation.

The location and size of a granule play a role in the process of granulation. Location of a granule means the position in which it resides in the space of the data. Original stream data are compressed to a few granules whose location and granularity reflect the structure of the data. There are many granulated views of the same problem. When evolving granular structures, granules are created as instances of the current knowledge. Next, granules may expand and occupy the space wherever new instances arrive. Operations on granules combine granules to form a coarser granule or decompose a granule into finer granules. Operations on granules should be consistent with the size of the granules and relations between granules; they provide the basic ingredients for the granular computing.

While concept drift and shift are terms related to the joint time-space domain (95), the descriptions of data density and information specificity (139) concern the space domain and are options to guide spatial granulation. Bargiela and Pedrycz (15) state that granules should encompass as many data as possible while

maintaining certain specificity in what they called principle of the maximization of the information density. The principle of the balanced information granularity (15) gives preference to the design of granules balanced along all dimensions rather than granules with unbalanced geometry. In particular, hyperbox-based spatial granulation provides descriptions fully compatible with the descriptions of intervals and fuzzy sets. With intervals and fuzzy sets, the pursuit of a balanced granularity and refining and coarsening of granules are reduced to operations on bounds of intervals and parameters of fuzzy membership functions.

3.6 Summary

Evolving granular systems combine granular computing and evolving intelligent systems concepts into a single framework. We argued that it is sometimes unnecessary or inefficient to discriminate numeric data precisely. Moreover, we argued that systems are better supported by a granular framework to suit uncertain, granular stream data. Numeric data is a particular case in which a granule degenerates into a singleton. The necessity of building models in finer granularities, close to the singularity, is justified only when there are clear benefits on doing so. This chapter presented the state of the art of the research in evolving granular systems and discussed adaptive rule-based modeling from granular data streams.

Chapter 4

Interval Based Evolving Modeling

This chapter introduces an interval-based evolving modeling approach to develop system models using data streams. The approach consists of a rule-based modeling scheme that gradually adapts its antecedent and consequent parts over time. Its main purpose is continuous (inductive) learning, self-organization, and adaptation to unknown, nonstationary environments. While traditional functional rule-based modeling approaches use numeric data and produce numeric results, the suggested interval approach uses interval data and presents results in numeric and granular format. Interval rule-based approaches are highly human-centric in the sense that antecedent and consequent of rules are intervals, which may convey a linguistic meaning. Granular values are more informative and comprehensible than singular, numeric values.

4.1 Introduction

Interval based evolving modeling (IBeM) is an adaptive granular framework whose idea is to enclose similar interval data into coarser albeit more interpretable interval models. The outcomes of IBeM are single-valued and interval predictions of a target function, and a rule summary, which describes the behavior of a system. IBeM emphasizes uncertain data manifesting as multi-dimensional tolerance in-

tervals and recursive learning procedures rooted in fundamentals of the interval mathematics theory. Antecedent and consequent terms of IBeM rules are determined by input and output crisp hyperboxes (granules) formed over time. Crisp hyperboxes are referred to as a modality of crisp granular precisiation in the generalized theory of uncertainty (27) (153). IBeM input and output hyperboxes are linked by a granular mapping, also called inclusion function in the interval analysis terminology (75) (78) (81).

IBeM is equipped with a one-pass-through-the-data recursive algorithm which builds its rule set gradually from scratch, captures new concepts from data streams, and copes with uncertainty. The IBeM approach makes no specific assumption about the properties of the data including probability distributions, belief intervals, possibility values, membership functions. Moreover, no human intervention is necessary during model construction. Contrarily, interval data stream guides learning and refinements. Examples of concepts translated into interval data IBeM manages to handle include: the number of red balls in the box is between 8 and 12; tonight's temperature will be from 65 to 72 degrees; the normal count of leukocytes in adult humans is 4500-11000 per cubic millimeter of blood; [0.8,0.85]. Intervals also rise after preprocessing singular data by comprising them into a smaller amount of interval data.

Interval representation of interval data streams is attractive due to several reasons: (*i*) easiness of acquiring parameters. Only two parameters related to real features (upper and lower bounds) need to be captured; (*ii*) adaptation of intervals demands basic fully-formalized operations of interval arithmetic; (*iii*) intervals make no specific assumption about the content of a granule. Higher-level interval models are everything we wish to know from large quantities of detailed, low-level, interval data; (*iv*) intervals can be translated quite easily to linguistic propositions. Interval granular precision facilitates comprehension when supported by a context. Naturally, an interval model has a great deal of appeal to represent counterpart interval data.

4.2 Related Work

Literature in interval data modeling using interval representation is scarce. Some works related to adaptive (non-evolving) interval modeling approaches that take into account interval data are summarized next. The approaches do not support online learning and require the whole data set to be available. Neural network approaches able to learn from interval data streams (for example (46) (108)) are discussed in Chapter 6.

A partitioning dynamical clustering algorithm which considers interval data and Pompeiu-Hausdorff distance was addressed in (31). The algorithm builds clusters and identifies their representative prototypes concomitantly at each processing step. Interval data are compared using adaptive Pompeiu-Hausdorff distance. This distance varies for each existing cluster according to intra-class structures. Although the clustering algorithm considers interval data, it is not suitable for evolving modeling since prototype identification is based on optimizing an adequacy criterion that requires all data samples within a cluster to be available.

An extension of the radial basis function (kernel method) approach to interval data mining is proposed in (37). Here, interval data result from the aggregation of large data sets into smaller ones to represent uncertainty. Aggregation is carried out through a Pompeiu-Hausdorff distance-based approach that clusters numeric data into crisp hyperboxes. The underlying learning approach can deal with classification, regression and novelty detection problems. The learning approach cannot handle online data streams.

Reference (12) proposes an interval analysis based adaptive approach for an extended Kalman filter. The approach is aimed at mobile robot navigation and, particularly, at obstacle avoidance and robot position estimation. Since Kalman filters are often affected by noise and drift, interval adaptive approach is useful to model and correct robot position estimates. Interval analysis methods dispense deterministic modeling of the robot system and have shown to give more accurate position estimates when compared to estimates using non-adaptive non-interval Kalman filter methods.

4.3 Structure and Processing

The mathematical formalism of the interval analysis (Section 2.2) provides a robust framework for the analysis of granular structures. Interval mathematics supports the core of the IBeM learning algorithm and gives simplicity, correctness, totality, closeness, efficiency, and optimality in the sense of Hickey et al. (56).

Let $(x, y)^{[h]}$, $h = 1, ...,$ be the h-th observation of the target function f. The output $y^{[h]}$ is known given the input $x^{[h]}$ or will be known some steps latter. In this chapter each attribute x_j of $x = (x_1, ..., x_n)$ is an interval $[\underline{x}_j, \overline{x}_j]$. The same holds true for the output y, that is, $y = [\underline{y}, \overline{y}]$. Therefore, (x, y) assembles a crisp hyperbox (granule) in the Cartesian product space $X \times Y$.

Let γ^i, $i = 1, ..., c$, be the current collection of IBeM granules built on the basis of (x, y). Granules γ^i are defined in the Cartesian product space $X \times Y$. The internal representation of γ^i in respect to input variables $x = (x_1, ..., x_n)$ is empty. This means that bounds of intervals, say $[l^i_j, L^i_j]$, $j = 1, ..., n$, are all IBeM records from the input data stream. The output variable y is granulated using bounds $[u^i, U^i]$. The content of a granule in respect to the output y conveys an additional information, an inclusion monotonic function p^i. The inclusion function uses the bounds of the input variables to produce granular approximation of f.

Rules R^i associated with granules γ^i are of the type:

$$\begin{aligned} R^i\text{: IF } & (l^i_1 \leq x_1 \leq L^i_1) \text{ AND } ... \text{ AND } (l^i_n \leq x_n \leq L^i_n) \\ \text{THEN } & (u^i \leq y \leq U^i) \text{ AND } \hat{y} = p^i(x_1, ..., x_n) \end{aligned}$$

where

$$p^i(x_1, ..., x_n) = \hat{y} = a^i_0 + \sum_{j=1}^{n} a^i_j [\underline{x}_j, \overline{x}_j]. \tag{4.1}$$

Functions p^i are thin (with single-valued parameters) and of first order in this case. In general, each p^i can be of different type, thick, and does not need to be linear. Computing p^i using x gives interval granular or single-valued approximation of f. If the width of x is equal to zero, then p^i is a single-valued approximation of f. Otherwise, if the width of x is greater than zero, then p^i is a granular approximation of f. The recursive least squares algorithm as described in Appendix B

is used to determine the coefficients a_j^i of p^i. Bounds $[u^i, U^i]$ are obtained from output data granulation. They provide an enclosure of the solution, an outer approximation of f.

IBeM exploits predominantly bottom-up incremental learning procedures to form higher level granules and interval-valued rules from finer granular data. In a sense, it performs input-output data compaction to provide more human-like models. A $\cup$-closure granular structure ensues from more specific local granules. In particular, granulation eases incremental updating and discovering of the essence of the time and space structure of the data with modest storage and processing costs. Experts usually prefer models that approximate real system outputs and provide estimates of the approximation bounds. Taking into account intervals in bounded-error context is the IBeM approach to deal with uncertainty.

The next section introduces a learning approach to construct an IBeM model from the very beginning, and adapt its structure and parameters on the fly.

4.4 Learning in IBeM

This section addresses the working principle of the IBeM learning algorithm.

The learning algorithm detailed next is used to evolve the structure and parameters of IBeM models whenever new information appears in the data stream. By IBeM structure we mean interval-type granules, If-Then rules, and a concept. From an overall point of view, when new samples do not fit current knowledge, learning creates new granules and rules managing the granules. Conversely, when new samples fit current knowledge, learning adapts parameters of existing granules and rules if necessary. Eventually, the resulting granular structure may be refined or coarsened agreeing with inter-granule relationships.

The IBeM framework grants important characteristics for online modeling. Its incremental learning algorithm spends a small and constant processing time: the processing time does not scale with the number of samples. Continuous processing on a per-sample basis enables IBeM to deal with concept drift and shift within online environment. Constructive bottom-up mechanisms of learning usually prevail over top-down, decomposing mechanisms.

4.4.1 Choosing the Granularity

Let ρ be the maximum width that interval granules may assume:

$$wdt([l_j^i, L_j^i]) \leq \rho,\ wdt([u^i, U^i]) \leq \rho,\ j = 1, ..., n;\ i = 1, ..., c. \tag{4.2}$$

Values of ρ allow different representations of the same problem in different levels of detail. ρ works as an upper bound to the level of modeling abstraction.

For normalized data, ρ takes values in $[0, 1]$. If ρ equals 0, granules cannot be expanded and each data sample is accommodated by a new granule. Conversely, if ρ equals 1, a single granule encloses the entire data set. To counterbalance these extremes means to establish a tradeoff between complexity and precision.

In the most general case, IBeM starts learning with an empty rule base and devoid of knowledge about the data stream. It is reasonable in this case to set ρ halfway to regard rule creation and rule adaptation equally. We consider $\rho^{[0]} = 0.5$ as the default initial value. A simple and fast approach to adapt the maximum width ρ allowed for granules is as follows. Let r be the number of rules created during h_r time steps. If the number of rules grows faster than a given rate η, that is, $r > \eta$, then ρ is increased,

$$\rho(\text{new}) = \left(1 + \frac{r}{h_r}\right) \rho(\text{old}). \tag{4.3}$$

The idea here is to reject large rule bases because they increase model complexity and may not help generalization. Equation (4.3) acts against outbursts of growth letting intervals and granules expand larger.

Otherwise, if the number of rules grows at a rate smaller than η, that is, $r < \eta$, then ρ is decreased as follows:

$$\rho(\text{new}) = \left(1 - \frac{(\eta - r)}{h_r}\right) \rho(\text{old}). \tag{4.4}$$

With this mechanism we maintain a data-dependent fluctuating granularity.

Alternative heuristic approaches to evolve the value of ρ over time take into account estimation errors and their derivatives as addressed in (79). Time-varying granularity avoids guesses on how fast and how often the data stream changes.

4.4.2 Time Granulation

Consider an interval data stream $(x, y)^{[h]},\ h = 1, ...$ Time granulation groups a set of successive instances $(x, y)^{[h]},\ h = h_b, h_{b+1}, ..., h_e$, where h_b and h_e denote the lower and upper bounds of a time interval $[h_b, h_e]$. The set of instances input during $[h_b, h_e]$ is considered indistinguishable and the inequalities

$$wdt(ch(x_j^{[h_b]}, ..., x_j^{[h_e]})) \leq \rho,\ j = 1, ..., n,\ \text{and}\ wdt(ch(y^{[h_b]}, ..., y^{[h_e]})) \leq \rho \quad (4.5)$$

hold true. Literally, the width of the convex hull of all samples available during $[h_b, h_e]$ is less or equal to the maximum width allowed for granules, ρ. The sample indexed by h_{e+1} conveys at least one contrasting value.

The collection $(x, y)^{[h]},\ h = h_b, h_{b+1}, ..., h_e$, produces a unique granule $\gamma^{[H]}$ whose lower and upper endpoints are:

$$[l_j^{[H]}, L_j^{[H]}] = [min(x_j^{[h_b]}, ..., x_j^{[h_e]}), max(x_j^{[h_b]}, ..., x_j^{[h_e]})],\ \text{and} \quad (4.6)$$

$$[u^{[H]}, U^{[H]}] = [min(y^{[h_b]}, ..., y^{[h_e]}), max(y^{[h_b]}, ..., y^{[h_e]})]. \quad (4.7)$$

Thus, a single granule $\gamma^{[H]}$ summarizes the content of several samples $(x, y)^{[h]}$. Both, $\gamma^{[H]}$ and $(x, y)^{[h]}$, are of the same interval nature.

In the IBeM framework, time granulation is used as a preprocessing step whenever input data arrive at different rates, for example, x_1 arrives at each 10 milliseconds and x_2 at each 30 milliseconds. Multiple time granularities allow synchronized analysis of concurrent data streams. Therefore, learning within the space domain is based on the resulting interval granule $\gamma^{[H]}$ rather than on original data $(x, y)^{[h]}$. IBeM is not exposed to all original data, which are sometimes far more abundant than time granules.

4.4.3 Creating and Adapting Granules

No IBeM rules need to be preconceived nor does the amount of granules need to be set in advance. From scratch, granules and rules are created and adapted on demand, dynamically, steered by the behavior of the target process and information mirrored in the measured data. Whenever data $(x, y)^{[h]}$ become available, a decision mechanism is trigged and new granules and rules can be inserted into the IBeM structure or existing ones can be refined.

A key decision when building IBeM models concerns when and how to create or adapt granules and rules recursively to consider never seen data samples potentially bringing new information.

Define $E^i = (E^i_1, ..., E^i_n, E^i_k)$, the expansion region of granule γ^i:

$$E^i_j = [L^i_j - \rho,\ l^i_j + \rho],\ j = 1, ..., n, \tag{4.8}$$

$$E^i_k = [U^i - \rho,\ u^i + \rho]. \tag{4.9}$$

Expansion regions help to derive criteria for deciding whether or not data should be gathered into a common granule. Figure 4.1 illustrates the expansion region E^i_j of an attribute $[l^i_j, L^i_j]$ of the granule γ^i.

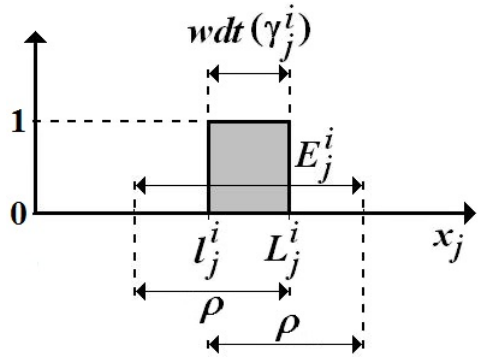


Figure 4.1: Expansion region of an IBeM granule

An IBeM granule is created when expansion regions E^i, $i = 1, ..., c$, do not fit the sample (x, y). In this case, none of the existing granules can expand its

bounds beyond the limits imposed by ρ to include the sample. Connective AND operators of IBeM rules suggest the complete enclosing of both inputs x_j and output y for the corresponding granule to be considered. The new granule γ^{c+1} matches perfectly the sample that caused its creation, that is,

$$[l_j^{c+1}, L_j^{c+1}] = [\underline{x}_j, \overline{x}_j], \text{ and} \tag{4.10}$$

$$[u^{c+1}, U^{c+1}] = [\underline{y}, \overline{y}]. \tag{4.11}$$

The parameters of the thin local function p^{c+1} are:

$$a_j^{c+1} = 0, \ j \neq 0, \text{ and } a_0^{c+1} = mp(y). \tag{4.12}$$

Adaptation of granules γ^i sets the boundaries $[l_j^i, \ L_j^i]$ and $[u^i, \ U^i]$ to enclose a sample (x, y). Meanwhile, parameters a_j^i of the local inclusion function p^i are updated using recursive least squares as described in the Appendix B. A granule γ^i is chosen to be adapted whenever (x, y) falls within its expansion region E^i. Adaptation of only one granule is enough to ensure that the information is incorporated in the model. Conflict resolution is addressed in Section 4.4.5. Figure 4.2 summarizes nine situations that may happen depending on where the data are confined and appropriate adaptations.

In Fig. 4.2, the recently arrived datum $x = [\underline{x}, \overline{x}]$ can be placed either outside, partially inside, or inside granule γ^i. Depending on the location of x, IBeM may create a new granule γ^{c+1} and/or adapt the bounds of γ^i. Expansion of granules is chiefly based on union and convex hull operations. All uncertainty is covered by some granule to guarantee outer approximation of the input and output data. Although datum and granule may have some level of overlap, two granules are forbidden to overlap as result of these adaptation procedures.

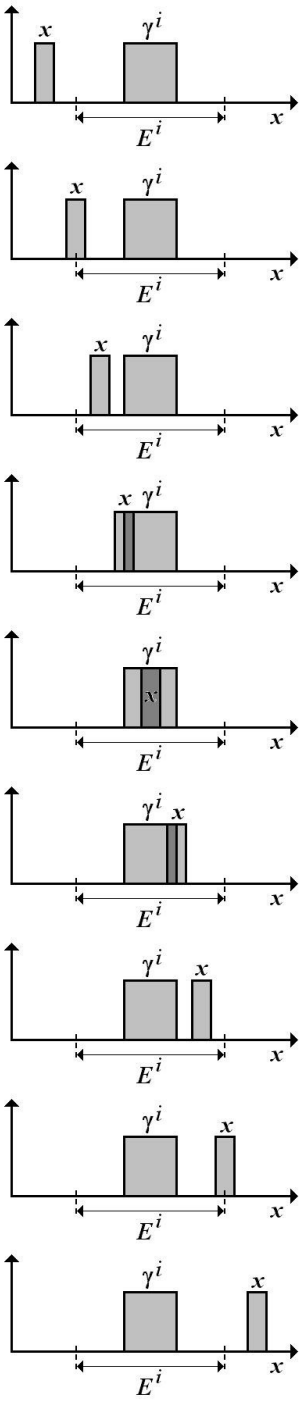


Figure 4.2: Creation and recursive adaptation of IBeM granules

4.4.4 Refining the Rule Base

Once granules are identified, IBeM analyzes the relationship among them and proceeds accordingly. Top-down and bottom-up structural operations support refining and coarsening of granules over time. Structural knowledge is generated to help visualization of relationships between different parts of the problem.

Top-down processes produce $\cap$-closure granular models by splitting a large granule into smaller granules. Situations in which the maximum width allowed for a granule reduces (see Section 4.4.1) may cause top-down refinements to fit some granules to the new value. Formally, $wdt(\gamma^i) < \rho$, $i = 1, ..., c$, may return false for some i. In this case, granule γ^i is split into γ^{i_1} and γ^{i_2} so that

$$[l_j^{i_1}, L_j^{i_1}] = [l_j^i, mp([l_j^i, L_j^i])], \text{ and} \tag{4.13}$$

$$[l_j^{i_2}, L_j^{i_2}] = [mp([l_j^i, L_j^i]), L_j^i], \ j = 1, ..., n. \tag{4.14}$$

The same splitting procedure holds for the output interval $[u^i, U^i]$. The procedure is repeated until $wdt(\gamma^i) < \rho$ for all i.

A $\cup$-closure granular model results from a bottom-up process that involves forming a larger granule from small granules. Consider the overall distance between two interval vectors as described in Section 2.2.3 and let

$$D = \begin{bmatrix} D(\gamma^1, \gamma^1) & ... & D(\gamma^1, \gamma^i) & ... & D(\gamma^1, \gamma^c) \\ \vdots & \ddots & \vdots & & \vdots \\ D(\gamma^i, \gamma^1) & ... & D(\gamma^i, \gamma^i) & ... & D(\gamma^i, \gamma^c) \\ \vdots & & \vdots & \ddots & \vdots \\ D(\gamma^c, \gamma^1) & ... & D(\gamma^c, \gamma^i) & ... & D(\gamma^c, \gamma^c) \end{bmatrix} \tag{4.15}$$

be a distance matrix relating any pair of granules. Matrix D is symmetric with zeros in the main diagonal. Neighbor granules can be located close enough to justify their combination into a coarser granule. The combination is based on the minimum entry of matrix D, say $D(\gamma^{i_1}, \gamma^{i_2})$, and depends if

$$wdt(ch([l_j^{i_1}, L_j^{i_1}], [l_j^{i_2}, L_j^{i_2}])) \leq \rho,\ j = 1, ..., n, \text{ and} \tag{4.16}$$

$$wdt(ch([u^{i_1}, U^{i_1}], [u^{i_2}, U^{i_2}])) \leq \rho. \tag{4.17}$$

Granule $\gamma^i = ch(\gamma^{i_1}, \gamma^{i_2})$ is coarsening of γ^{i_1} and γ^{i_2}.

Coarsening provides more compact rule bases and contributes to eliminate gaps between similar granules. At the top level, IBeM is closed by the most general granule formed by the convex hull of all elementary granules.

4.4.5 Conflict of Interest

A requirement when designing granular systems such as IBeM is to include all information that assembles a solution. However, at the same time it is desirable to keep the system as simple as possible. During the course of learning, conflicting situations may arise. In these cases, adaptation procedures that result in narrower granules must be considered. Conflict of interest happens when two or more

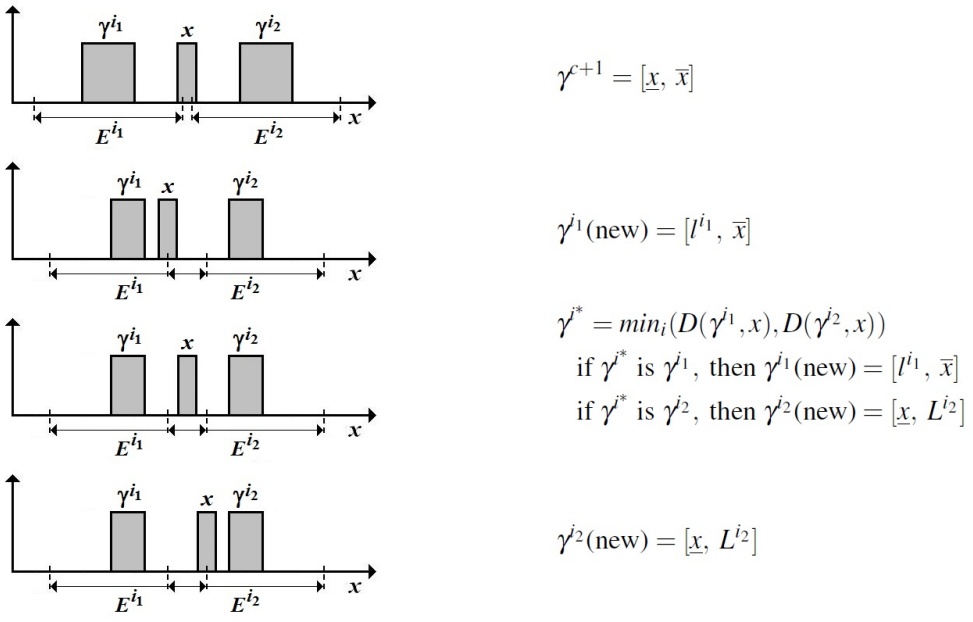


Figure 4.3: Inter-granular conflict and data accommodation

granules can be expanded to embrace a data sample. Figure 4.3 shows four typical situations considering the current input x and two granules, say γ^{i_1} and γ^{i_2}, they are: (i) $\underline{x} \in E^{i_1} = [L^{i_1} - \rho, l^{i_1} + \rho]$, but $\overline{x}$ does not. Conversely $\overline{x} \in E^{i_2}$, but $\underline{x}$ does not; (ii) $x \cap E^{i_2} \neq \emptyset$, but $x \in E^{i_1}$ and $x \notin E^{i_2}$; (iii) $x \subseteq (E^{i_1} \cap E^{i_2})$; and (iv) $x \cap E^{i_1} \neq \emptyset$, but $x \in E^{i_2}$ and $x \notin E^{i_1}$. The respective adaptation procedures are shown in the figure.

In case (i) a new granule is created to include x because γ^{i_1} and γ^{i_2} cannot expand beyond ρ. Cases (ii) and (iv) avoid redundancy and inconsistency neglecting the adaptation of the granule that cannot enclose x entirely. Case (iii) chooses the granule closer to x according to the distance D.

Inter-granular conflict resolution helps to choose which IBeM rule to adapt and prevents overlapped intervals and contradiction. The tightest envelope for the data generates a more concise description about the information it carries.

4.4.6 Removing Granules

Depending on the data sequence, small undesirable granules might be formed quite close to large granules. These small granules, often called satellites (97), contain residual information which is better neglected. The deletion procedure we have proposed is useful to retain only the necessary information.

Broadly speaking, a granule should be removed from the IBeM structure if it is inconsistent with the current concept. Common removing strategies either (i) remove granules by age, (ii) exclude the weakest granules based on error values, or (iii) delete the most inactive granules. In IBeM, the strategy is to delete inactive granules by exclusion. Old granules may still be useful in the current environment whereas weak granules are attempted to be strengthened by adapting parameters of local inclusion functions.

IBeM granules are deleted whenever they become inactive during h_r time steps. If the application requires memorization of rare events, or if cyclical drifts are anticipated, then it may be the case to let the granules remain forever. Removing inactive granules periodically helps to keep the rule set updated and concise.

4.4.7 Learning Algorithm

The IBeM modeling procedure can be summarized as follows:

BEGIN
Set parameters ρ, h_r, η, $c = 0$;
Read $(x, y)^{[h]}$, $h = 1$;
Create granule γ^{c+1} and rule R^{c+1};
For $h = 2, ...$ do
  Read $(x, y)^{[h]}$;
  Provide single-valued approximation $mp(p(x^{[h]}))$;
  Provide granular approximation $[u^i, U^i]$;
  Calculate output error $\epsilon^{[h]} = mp(y^{[h]}) - mp(p(x^{[h]}))$;
  If ($x_j^{[h]} \notin E_j^i$ || $y^{[h]} \notin E_k^i$, for $i = 1, ..., c$, and some j)
    Create γ^{c+1} and R^{c+1};
  Else
    Resolve possible conflicts;
    Update γ^i and R^i to accommodate $(x, y)^{[h]}$;
    Adapt local inclusion function parameters a_j^i using RLS;
  If $h = \alpha h_r$, $\alpha = 1, 2, ...$
    Update model granularity ρ;
    Split and combine granules when feasible;
    Remove inactive granules and rules;
END

4.5 Summary

This chapter has introduced an interval-based evolving modeling approach to assess the essence of interval data streams and simplify complex problems characterized by nonlinearity and nonstationarity. The IBeM modeling approach for interval data is based on the incremental evolution of hyperrectangle-like forms of information granules and associated interval inclusion functions. Stream data guide the construction of both granules and rule base without the need of human intervention or additional care. IBeM provides highly human-centric interval and precise numeric approximations of functions using a fast one-pass learning algorithm, modest memory requirements, and uncertain data.

Chapter 5

Fuzzy Set Based Evolving Modeling

This chapter introduces an evolving fuzzy-set based granular framework to learn from and model time-varying fuzzy input-output data streams. The framework consists of a recursive algorithm capable of developing the structure of fuzzy rule-based models on-demand. The framework is particularly suitable to handle potentially unbounded fuzzy data streams and provide single-valued and granular approximations of unknown nonstationary functions.

5.1 Introduction

A primary requirement of a broad class of evolving intelligent systems is to process a sequence of numeric data over time. The fuzzy set based evolving modeling (FBeM) framework employs fuzzy granular models to deal with more detailed fuzzy granular data and therefore provide a more intelligible exposition of the data. For each granular model there exists an associated fuzzy rule base. The antecedent part of FBeM rules consists of fuzzy hyperboxes, which are interpretable transparent descriptors of input granular data. The consequent part of FBeM rules has a linguistic and a functional component. The linguistic component arises from fuzzy hyperboxes formed by output data granulation. It facilitates model interpretation and encloses possible model outputs. The functional

component is derived from input data and real-valued local functions. This component produces more accurate approximands. The rationale behind the FBeM approach is that it looks to input-output data streams under different resolutions and decides when to adopt coarser or more detailed granularities.

The function of FBeM is to deliver simultaneous single-valued and granular function approximation and linguistic description of the behavior of a system. Local FBeM models are a set of If-Then rules developed incrementally from input-output data streams. Learning can start from scratch and, as new information is brought by the data stream, granules and rules are created and their parameters adjusted. Therefore, FBeM becomes more flexible to handle data so that redesign and retraining models all along are needless. The resulting input-output granular mapping may be eventually either refined or coarsened according to inter-granules relationships and error indices.

5.2 Related Work

This section summarizes works related to incremental learning approaches to handle data streams. The approaches described next are closely related to the approach suggested in this chapter.

Fuzzy ARTMAP (30) is an adaptive resonance neural network. Its incremental learning ability suggests its use in online data processing. Fuzzy ARTMAP is a supervised neural network characterized by weight vectors. Half of the positions of weight vectors represent one corner of a hyperbox, and the remaining half the opposite corner. When new data arrive, the smallest box that encloses the data is chosen. If no such box exists, then either the box that needs less expansion to enclose the data is selected, or a new one is created. Next, a neuron representing a hyperbox is selected and a vigilance criterion is checked. Vigilance serves to choose another box if the one selected is too large. Fuzzy ARTMAP compares the mapped class for an input with the actual label. If the labels are different, then it creates a new neuron and connects it to the actual label. Fuzzy ARTMAP is not appropriate to fit fuzzy data and provide granular function approximation.

Evolving Mamdani-Takagi-Sugeno neural fuzzy inference system (eMTSFIS) (58) is a neurofuzzy approach to model the dynamic nature of real-world prob-

lems. eMTSFIS comes with an incremental learning algorithm that evolves its structure and parameters according to time-varying numeric data streams. The learning algorithm is life-long and addresses the stability-plasticity dilemma typical of neurofuzzy constructs. Essentially, the eMTSFIS approach combines the relatively higher interpretability of Mamdani-type systems with the precision of Takagi-Sugeno fuzzy systems in numeric data stream modeling.

The uncertain micro-clustering algorithm (UMicro) (2) considers that stream data arrive together with their underlying standard error instead of assuming the entire probability distribution function of the data is known. The algorithm uses uncertainty information to improve the quality of the underlying results. UMicro incorporates a time decay method to update the statistics of micro-clusters. The decaying method is especially useful to model drifting concepts in evolving data streams. The efficiency of the UMicro approach has been demonstrated in a variety of data sets.

5.3 Structure and Processing

The formalism of fuzzy sets (Section 2.4) provides a framework for the analysis and representation of fuzzy granular structures.

Let $(x, y)^{[h]}$, $h = 1, ...,$ be the h-th observation of a data stream. The output $y^{[h]}$ is known provided the input $x^{[h]}$ or will be known some steps latter. In this chapter, each attribute x_j of $x = (x_1, ..., x_n)$ is a trapezoidal fuzzy interval $(\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j)$. The same holds for the output y, that is, $y = (\underline{\underline{y}}, \underline{y}, \overline{y}, \overline{\overline{y}})$. Therefore, (x, y) shapes a fuzzy hyperbox in the Cartesian product space $X \times Y$.

Let γ^i, $i = 1, ..., c$, be the current collection of FBeM granules built on the basis of (x, y). Granules γ^i are defined in the Cartesian product space $X \times Y$. Rules R^i governing FBeM information granules γ^i are of the type:

$$R^i\text{: IF } (x_1 \text{ is } A_1^i) \text{ AND } ... \text{ AND } (x_n \text{ is } A_n^i)$$
$$\text{THEN } \underbrace{(y \text{ is } B^i)}_{\text{linguistic}} \quad \text{AND} \quad \underbrace{\hat{y} = p^i(x_1, ..., x_n)}_{\text{functional}},$$

where A_j^i and B^i are trapezoidal membership functions built in light of input and

output data being available; p^i is a local approximation function. The collection of rules R^i, $i = 1, ..., c$, casts a rule base. Rules in FBeM are created and adapted on the fly whenever the data ask for improvement in the current model. Notice that an FBeM rule combines both linguistic and functional consequents. The linguistic part of the consequent favors interpretability since fuzzy sets may come with a label. The functional part of the consequent offers accuracy. Thus, FBeM takes advantage of both linguistic and functional consequents within a single framework.

Fuzzy sets A_j^i and B^i are generated from scattered fuzzy granulation. The scattering approach clusters the data into fuzzy sets when appropriate, and takes into account the coexistence of a manifold of granularities in the data stream. Sets A_j^i and B^i can be easily extended to fuzzy hyperboxes γ^i (granules) in the product space. Granules are positioned at locations populated by input and outputs data in the product space. Figure 5.1 illustrates the scatter granulation mechanism of fuzzy data. Note in the figure that the granularity of models is coarser than the granularity of data. This is to obtain data compression and to provide a more abstract, human-interpretable, representation of the data.

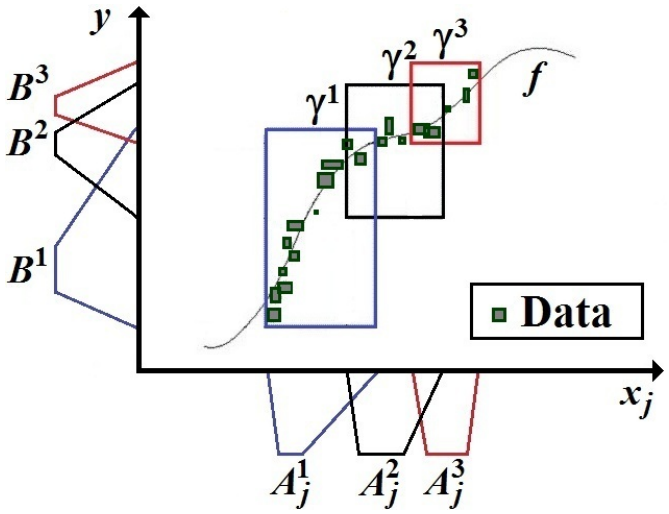


Figure 5.1: Scattering approach for fuzzy data granulation

Fitting data into conveniently placed and sized granules through scattering leaves substantial flexibility for incremental learning. The FBeM approach grants

freedom in choosing the internal structure of granules.

Yager (138) (140) has demonstrated that a trapezoidal fuzzy set $A_j^i = (l_j^i, \lambda_j^i, \Lambda_j^i, L_j^i)$ allows the modeling of a wide class of granular objects. A triangular fuzzy set is a trapezoid where $\lambda_j^i = \Lambda_j^i$; an interval is a trapezoid where $l_j^i = \lambda_j^i$ and $\Lambda_j^i = L_j^i$; a singleton (singular datum) is a trapezoid where $l_j^i = \lambda_j^i = \Lambda_j^i = L_j^i$. Additional features that make the trapezoidal representation attractive comprise: (***i***) ease of acquiring the necessary parameters: only four parameters need to be captured; (***ii***) many operations on trapezoids can be performed using the endpoints of intervals which are level sets of trapezoids; moreover, the piecewise linearity of the trapezoidal representation allows calculation of only two level sets, corresponding to the core and support, respectively, to obtain a complete implementation; (***iii***) trapezoids are easier to translate into linguistic labels.

Fuzzy sets $B^i = (u^i, v^i, \Upsilon^i, U^i)$ are used to assemble granules in the output space. The local function p^i is adapted for samples that rest inside the granule γ^i. In general, functions p^i can be of different type and are not required to be linear. Here we assume affine functions:

$$p^i(x_1, ..., x_n) = a_0^i + \sum_{j=1}^{n} a_j^i mp(x_j), \tag{5.1}$$

for simplicity. If higher-order functions p^i are used to approximate a function f, then the number of coefficients to be estimated increases, especially when the number of input variables n is large. The recursive least squares algorithm (Appendix B) is used to calculate the coefficients a_j^i of p^i.

Trapezoidal fuzzy sets and scatter granulation allow granules to overlap. Therefore, two or more granules can accommodate the same data sample. FBeM singular output is found as the weighted mean value:

$$p = \frac{\sum_{i=1}^{c} min(A_1^i(x_1), ..., A_n^i(x_n)) p^i(x_1, ..., x_n)}{\sum_{i=1}^{c} min(A_1^i(x_1), ..., A_n^i(x_n))}. \tag{5.2}$$

Granular output is given by the convex hull of output fuzzy sets B^{i^*}, where i^* are indices of granules that can accommodate the data sample. The convex hull of trapezoidal fuzzy sets $B^1, ..., B^c$ is given as follows:

$$ch(B^1, ..., B^c) = (min(u^1, ..., u^c), min(v^1, ..., v^c), \\ max(\Upsilon^1, ..., \Upsilon^c), max(U^1, ..., U^c)). \tag{5.3}$$

The granular output given by B^{i^*} enriches decision making and motivates interpretability. While being specific as determined by p we risk being incorrect, being unspecific from B^{i^*} increases our confidence to be correct.

5.4 Learning in FBeM

5.4.1 Setting the Granularity

The maximum width fuzzy sets A_j^i are allowed to expand is denoted by ρ, that is, $wdt(A_j^i) = L_j^i - l_j^i \leq \rho$, $j = 1, ..., n$; $i = 1, ..., c$. Different values of ρ yield different representations of the same data stream in different levels of granularities.

Let the expansion region of a set A_j^i be denoted by

$$E_j^i = [mp(A_j^i) - \frac{\rho}{2}, mp(A_j^i) + \frac{\rho}{2}], \tag{5.4}$$

where $mp(A_j^i) = (\lambda_j^i + \Lambda_j^i)/2$ is the midpoint of A_j^i. Expansion regions help to derive criteria for deciding whether or not data samples should be considered in the same granule.

For normalized data, ρ takes on values in $[0, 1]$. If ρ is equal to 0, then FBeM granules are not enlarged. Learning creates a new rule for each sample, which may cause overfitting and lead to excessive complexity and irreproducible optimistic results. If ρ equals 1, then a single granule may cover the entire data domain so that FBeM becomes unable to manage nonstationarity in the data. Meaningful life-long adaptability is reached choosing intermediate values for ρ.

In the most general case, FBeM starts learning with an empty rule base and without any knowledge about the data-generating process. In this case, a reasonable approach is to initialize ρ halfway to yield structural stability and plasticity equally. We consider $\rho^{[0]} = 0.5$ as the default initial value.

A fast procedure to evolve ρ over time is as follows. Let r be the difference between the current number of granules and the number of granules h_r steps earlier, $r = c^{[h]} - c^{[h-h_r]}$. If the quantity of granules grows faster than a given rate η, that is, $r > \eta$, then ρ is increased:

$$\rho(\text{new}) \quad = \quad \left(1 + \frac{r}{h_r}\right) \rho(\text{old}). \tag{5.5}$$

Equation (5.5) controls the value of ρ in such a way to reject large rule bases and therefore avoid complexity increasing. Large rule bases may not help generalization of the results.

If the number of granules grows at a rate smaller than η, that is, $r \leq \eta$, then ρ is decreased as follows:

$$\rho(\text{new}) \quad = \quad \left(1 - \frac{(\eta - r)}{h_r}\right) \rho(\text{old}). \tag{5.6}$$

This procedure keeps model granularity time-varying according to the data stream. Other granularity adaptation procedures may take into consideration estimation errors and their derivatives, as suggested in (79) (80).

Reducing the maximum width allowed for granules requires shrinking larger granules to fit them to the new value. In this case, the support of a fuzzy set A_j^i is narrowed as follows:

$$\text{If } mp(A_j^i) - \frac{\rho(\text{new})}{2} > l_j^i \text{ then } l_j^i(\text{new}) = mp(A_j^i) - \frac{\rho(\text{new})}{2}$$

$$\text{If } mp(A_j^i) + \frac{\rho(\text{new})}{2} < L_j^i \text{ then } L_j^i(\text{new}) = mp(A_j^i) + \frac{\rho(\text{new})}{2}.$$

Cores $[\lambda_j^i, \Lambda_j^i]$ are handled similarly. Time-varying granularity is useful to avoid guesses on how fast the data stream changes.

5.4.2 Time Granulation

Consider a fuzzy data stream $(x, y)^{[h]}$, $h = 1, ...$ Time granulation groups a set of successive samples $(x, y)^{[h]}$, $h = h_b, h_{b+1}, ..., h_e$, where h_b and h_e denote the lower and upper bounds of a time interval $[h_b, h_e]$. The set of instances input during $[h_b, h_e]$ produces a unique granule $\gamma^{[H]}$ whose corresponding fuzzy sets are

$$A_j^{[H]} = (min(\underline{\underline{x}}_j^{[h_b]}, ..., \underline{\underline{x}}_j^{[h_e]}), min(\underline{x}_j^{[h_b]}, ..., \underline{x}_j^{[h_e]}), \\ max(\overline{x}_j^{[h_b]}, ..., \overline{x}_j^{[h_e]}), max(\overline{\overline{x}}_j^{[h_b]}, ..., \overline{\overline{x}}_j^{[h_e]})), \tag{5.7}$$

and $B^{[H]}$, which is constructed similarly from the output stream. Instances falling within $A_j^{[H]}$, $j = 1, ..., n$, and $B^{[H]}$ are considered indiscernible and the inequalities

$$wdt(A_j^{[H]}) \leq \rho, \;\; j = 1, ..., n, \quad \text{and} \quad wdt(B^{[H]}) \leq \rho \tag{5.8}$$

hold true.

Whenever input data arrive at different rates, for example, x_1 arrives at each 2 seconds and x_2 at each 3 seconds, or the amount of data exceeds the affordable computational cost (e.g. in high-frequency applications), we resort to granulated views of the time domain. Thereafter, rule construction is based on the resulting fuzzy granules, $A_j^{[H]}$ and $B^{[H]}$, rather than on original data $(x, y)^{[h]}$. FBeM does not need to be exposed to all original data.

5.4.3 Creating Granules

No rule necessarily exists before learning starts. The incremental procedure to create rules runs whenever at least one entry of an input $(x_1, ..., x_n)$ does not belong to expansion regions $(E_1^i, ..., E_n^i)$, $i = 1, ..., c$, or the output $y \not\subset E_k^i$, $i = 1, ..., c$. Otherwise, the current rule base is not modified.

A new granule γ^{c+1} is assembled from fuzzy sets A_j^{c+1} and B^{c+1} whose parameters match the sample, that is,

$$\begin{aligned} A_j^{c+1} &= (l_j^{c+1}, \lambda_j^{c+1}, \Lambda_j^{c+1}, L_j^{c+1}) = (\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j), \\ B^{c+1} &= (u^{c+1}, v^{c+1}, \Upsilon^{c+1}, U^{c+1}) = (\underline{\underline{y}},\ \underline{y},\ \overline{y},\ \overline{\overline{y}}). \end{aligned} \tag{5.9}$$

Coefficients of the real-valued local function p^{c+1} are set to

$$a_0^{c+1} = mp(y);\ a_j^{c+1} = 0,\ j \neq 0. \tag{5.10}$$

5.4.4 Adapting Granules

Adaptation of granules either expands or contracts the support and the core of fuzzy sets A_j^i and B^i to enclose new data, and simultaneously refines the coefficients of local functions p^i to improve accuracy. A granule is chosen to be adapted whenever an instance of the data stream falls within its expansion region. In situations in which two or more granules are qualified to enclose the data, adapting only one of the granules is enough.

Data and granules are fuzzy objects of trapezoidal nature. A possible similarity measure for vectors of trapezoids is:

$$S(x, A^i) = 1 - \frac{1}{4n} \sum_{j=1}^{n} (|\underline{\underline{x}}_j - l_j^i| + |\underline{x}_j - \lambda_j^i| + |\overline{x}_j - \Lambda_j^i| + |\overline{\overline{x}}_j - L_j^i|). \tag{5.11}$$

This measure is based on Hamming or city-block distance (52); it quantifies the degree that input data match the current knowledge. Particularly, equation (5.11) returns 1 for identical trapezoids, indicating the maximum degree of matching. The output value decreases linearly as the trapezoids x and A^i move away from each other. Among all granules qualified to accommodate a particular sample, the one with highest similarity should be chosen. This procedure prevents conflict and helps to keep the FBeM construction simple. Although equation (5.11) is

simple to compute, involving only basic arithmetic operations, there are no strong principled reasons to impose this measure. In fact, there is no generally accepted consensus on a best similarity measure for a given application (33).

Adaptation of granules proceeds depending on how far an input datum x_j is from fuzzy set A_j^i. Namely,

$$\begin{array}{lll}
\text{If} & \underline{\underline{x}}_j \in [mp(A_j^i) - \frac{\rho}{2}, l_j^i] & \text{then} \quad l_j^i(\text{new}) = \underline{\underline{x}}_j \\
\text{If} & \underline{x}_j \in [mp(A_j^i) - \frac{\rho}{2}, \lambda_j^i] & \text{then} \quad \lambda_j^i(\text{new}) = \underline{x}_j \\
\text{If} & \underline{x}_j \in [\lambda_j^i, mp(A_j^i)] & \text{then} \quad \lambda_j^i(\text{new}) = \underline{x}_j \\
\text{If} & \underline{x}_j \in [mp(A_j^i), mp(A_j^i) + \frac{\rho}{2}] & \text{then} \quad \lambda_j^i(\text{new}) = mp(A_j^i) \\
\text{If} & \overline{x}_j \in [mp(A_j^i) - \frac{\rho}{2}, mp(A_j^i)] & \text{then} \quad \Lambda_j^i(\text{new}) = mp(A_j^i) \\
\text{If} & \overline{x}_j \in [mp(A_j^i), \Lambda_j^i] & \text{then} \quad \Lambda_j^i(\text{new}) = \overline{x}_j \\
\text{If} & \overline{x}_j \in [\Lambda_j^i, mp(A_j^i) + \frac{\rho}{2}] & \text{then} \quad \Lambda_j^i(\text{new}) = \overline{x}_j \\
\text{If} & \overline{\overline{x}}_j \in [L_j^i, mp(A_j^i) + \frac{\rho}{2}] & \text{then} \quad L_j^i(\text{new}) = \overline{\overline{x}}_j.
\end{array}$$

The first and the eighth rules suggest support expansion while the second and seventh recommend core expansion. The remaining cases advise core contraction. Figure 5.2 illustrates seven possible adaptation situations. In the figure, the datum $x = (\underline{\underline{x}}, \underline{x}, \overline{x}, \overline{\overline{x}})$ places either outside, partially inside, or inside granule γ^i. FBeM creates a new granule γ^{c+1} or adapts the parameters of γ^i accordingly.

Operations on core parameters, λ_j^i and Λ_j^i, require further adjustment of the midpoint of the respective granule:

$$mp(A_j^i)(\text{new}) = \frac{\lambda_j^i(\text{new}) + \Lambda_j^i(\text{new})}{2}. \tag{5.12}$$

As result, support contraction may happen in two occasions:

$$\begin{array}{lll}
\text{If } mp(A_j^i)(\text{new}) - \frac{\rho}{2} > l_j^i & \text{then} & l_j^i(\text{new}) = mp(A_j^i)(\text{new}) - \frac{\rho}{2} \\
\text{If } mp(A_j^i)(\text{new}) + \frac{\rho}{2} < L_j^i & \text{then} & L_j^i(\text{new}) = mp(A_j^i)(\text{new}) + \frac{\rho}{2}.
\end{array}$$

Adaptation of consequent fuzzy sets B^i is done similarly using output data

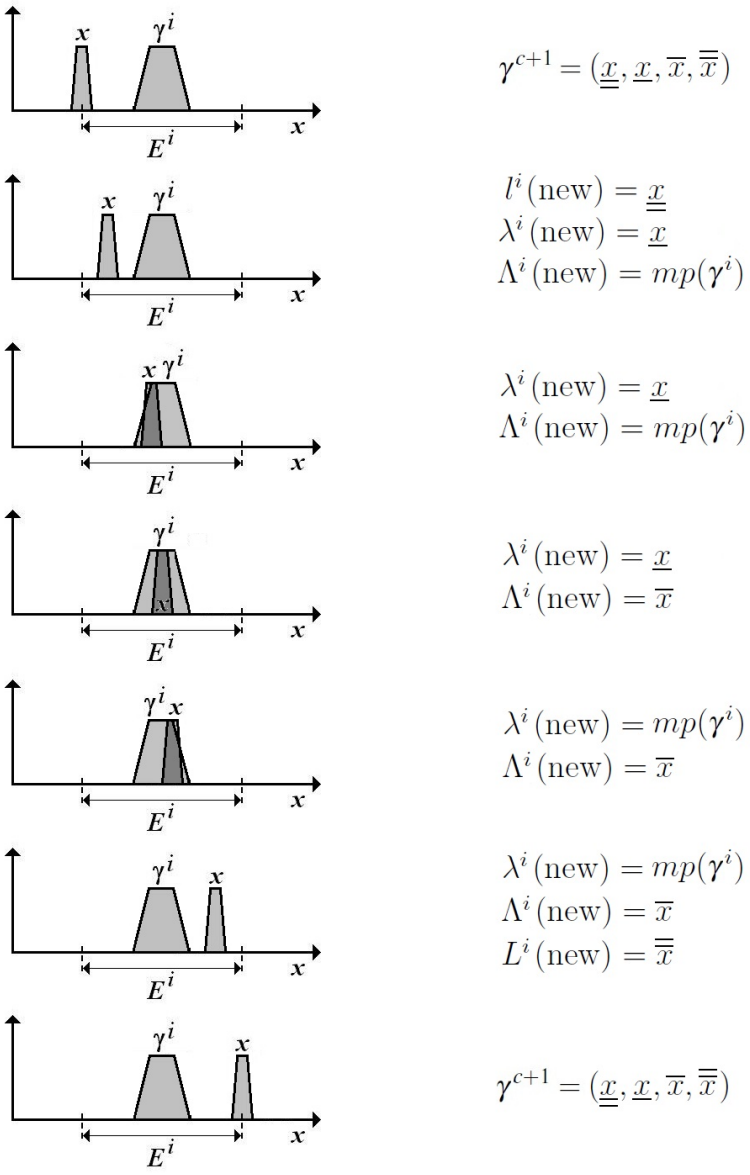


Figure 5.2: Creation and recursive adaptation of FBeM granules

y. Coefficients a_j^i of the local function p^i are updated using the recursive least squares algorithm as described in Appendix B. Storage of a number of recent instances may be useful to guide alternative coefficient identification algorithms, e.g., data chunks oriented algorithms. However, it comes with some additional cost concerning memory and processing time.

5.4.5 Coarsening the Granular Model

Relationships between granules may be strong enough to justify assembling a more abstract granule that inherits the information of lower level granules. The similarity measure (5.11) can be used to quantify granule-granule resemblance if we restate it as:

$$S(A^{i_1}, A^{i_2}) = 1 - \frac{1}{4n}\sum_{j=1}^{n}(|l_j^{i_1} - l_j^{i_2}| + |\lambda_j^{i_1} - \lambda_j^{i_2}| + |\Lambda_j^{i_1} - \Lambda_j^{i_2}| + |L_j^{i_1} - L_j^{i_2}|). \quad (5.13)$$

This measure has better discrimination capability than, for example, distance between midpoints of granules (136), and its calculation is fast.

FBeM combines granules in intervals of h_r steps considering the lowest entry of $S(A^{i_1}, A^{i_2})$, $i_1, i_2 = 1, ..., c$, $i_1 \neq i_2$, and a decision criterion. The decision may be based on whether the new granule obeys the maximum width allowed ρ.

A new granule γ^i, coarsening of γ^{i_1} and γ^{i_2}, is formed by trapezoidal membership functions A_j^i with parameters derived from $A_j^{i_1}$ and $A_j^{i_2}$ as follows:

$$\begin{aligned} l_j^i &= min(l_j^{i_1}, l_j^{i_2}) \\ \lambda_j^i &= min(\lambda_j^{i_1}, \lambda_j^{i_2}) \\ \Lambda_j^i &= max(\Lambda_j^{i_1}, \Lambda_j^{i_2}) \\ L_j^i &= max(L_j^{i_1}, L_j^{i_2}). \end{aligned} \quad (5.14)$$

The granule γ^i encloses all the content of the granules γ^{i_1} and γ^{i_2}. The same coarsening procedure is used to determine the parameters of the output membership function B^i. The coefficients of the local function of granule γ^i are:

$$a_j^i = \frac{1}{2}(a_j^{i_1} + a_j^{i_2}), \; j = 0, ..., n. \quad (5.15)$$

Combining granules reduces the size of the rule base and eliminates redundancy. The importance of reducing the number of rules in evolving rule-based systems

is discussed in (96).

5.4.6 Removing Granules

A granule should be removed from the system model if it seems to be inconsistent with the current knowledge. A common approach consists in deleting the most inactive granules (79).

Similarly as in (3), let

$$\Theta^i = 2^{(-\psi(h-h_a^i))} \tag{5.16}$$

be the activity factor associated to the granule γ^i. The constant ψ is a decay rate, h the current time step, and h_a^i the last time step that granule γ^i was processed. Factor Θ^i decreases exponentially when h increases. The half-life of a granule is the time spent to reduce the factor Θ^i by half, that is, $1/\psi$.

Half-life $1/\psi$ is a value that suggests deletion of inactive granules. As a rule, ψ is domain-dependent. Large values of ψ express lower tolerance to inactivity and higher privilege of more compact structures. Small values of ψ add robustness and prevent catastrophic forgetting. If the application requires memorization of isolated events or seasonality is expected, then it may be the case to set ψ to 0 and let granules and rules exist forever. In general, ψ should be set in $]0, 1[$ to keep model evolution active.

5.4.7 Learning Algorithm

The learning procedure to evolve FBeM models can be summarized by the algorithm described below. The algorithm underlines the essence of data stream oriented approaches where instances are read and discarded one at a time. Historical data are dispensable and evolution stands continuously on an incremental basis.

BEGIN
Set parameters ρ, h_r, η, ψ, $c = 0$;
Read $(x, y)^{[h]}$, $h = 1$;
Create granule γ^{c+1};
For $h = 2, ...$ do
  Read $(x, y)^{[h]}$;
  Provide single-valued approximation $p(x^{[h]})$;
  Provide granular approximation B^{i^*};
  Calculate output error $\epsilon^{[h]} = mp(y^{[h]}) - p(x^{[h]})$;
  If $x^{[h]}$ or $y^{[h]}$ is not into granules' expanded regions $E^i \forall i$
    Create granule γ^{c+1};
  Else
    Adapt the most active granule γ^i, $i = max_i(S(x, A^1), ..., S(x, A^c))$;
    Adapt local function parameters a_j^i using RLS;
  If $h = \alpha h_r$, $\alpha = 1, 2, ...$
    Combine granules when feasible;
    Update model granularity ρ;
    Remove inactive granules;
END

5.5 Summary

This chapter has introduced FBeM, an evolving granular fuzzy modeling framework based on fuzzy granular data streams. FBeM carries a series of properties that makes it suitable to model online nonstationary functions using fuzzy data. FBeM gives accurate and granular information simultaneously. Granular model outputs contain a range of possible values delimited by soft boundaries which turns the outputs more reliable and truthful.

Chapter 6

Evolving Granular Neural Networks

This chapter introduces evolving granular neural networks for neurofuzzy modeling of fuzzy data streams. The evolving granular neural network (eGNN) efficiently handles concept changes, distinctive events of nonstationary environments. eGNN builds interpretable multi-sized local models using fuzzy neural information fusion. An incremental learning algorithm develops the neural network topology from the information contained in data streams. We emphasize fuzzy intervals and objects with trapezoidal membership function representation. More precisely, the framework considers triangular, interval, and numeric types of data to construct granular fuzzy models as particular arrangements of trapezoids.

6.1 Introduction

Artificial neural network approaches are able to perform parallel processing, and identify and generalize patterns in data sets. Although classical neural networks can approximate any nonlinear continuous function in compact domains, they usually demand high quality training data and time-consuming offline learning. Generally speaking, neural networks give black box, non-transparent, difficult to interpret models.

Granular neural networks were introduced in (112) as a framework to process

information granules. By considering sets of objects sharing commonalities and imprecise data items instead of precise singular data items, granular neural networks avoid processing detailed and costly data. Granular neural networks do not need to consider all data, which are far more numerous than granules. Rather, data can be discarded whenever they match an existing information granule.

The eGNN approach described in this chapter, differently from (112), is committed to online modeling of potentially unbounded fuzzy data streams. We focus on fuzzy trapezoidal data, namely granular data expressed by trapezoidal fuzzy sets. Trapezoids allow some freedom in the choice of representative granules once they embody triangular fuzzy sets, intervals and real values as particular realizations (138). The eGNN structure uses fuzzy aggregation neurons as basic processing units and encodes a set of fuzzy rules and a fuzzy inference system. Its structure results from a gradual neurofuzzy construction that is transparent and interpretable. eGNN manages to discover more abstract high-level granular knowledge from finer granular data. High-level granular knowledge can be easily translated into a fuzzy knowledge base. Each rule of the knowledge base consists of two parts: rule antecedent (If part) and rule consequent (Then part). The consequent part is composed by a linguistic and a local functional (real-valued function) term. Independently of the choice of aggregation neurons, network parameters and the nature of input-output data, the linguistic term of the rule consequent provides a granular output while the functional term gives a singular (pointwise) output.

The eGNN framework addresses four issues: (***i***) non-interpretability and lack of transparency of black box neural network models; (***ii***) online processing of granular data streams; (***iii***) trading off precision and interpretability; and (***iv***) handling of large volumes of nonstationary data. The first issue is addressed using fuzzy hyperbox representations, which are interpretable and transparent descriptors of granular data, together with fuzzy neurons to aggregate data. The second issue is dealt with through incremental learning mechanisms capable of processing granular data. The third issue is handled by combining functional and linguistic fuzzy models into a single model. The last issue is managed by resorting to scalable recursive learning algorithm that works on a per-sample basis, requiring only features of a sample plus a small amount of aggregated

information such as fuzzy rule bases or neural networks. Learning should be one-pass, neglecting all previously seen data samples: each sample is processed only once and removed from memory (10).

Learning in eGNN fundamentally means to accommodate new data into existing granular models on a recursive basis. Learning may add new granules, neurons and respective connections into the network structure whenever necessary. The parameters of the real-valued functions of rule consequents are also object of learning. This means that eGNN captures new information from data streams, adapts itself to the new scenario, and avoids redesigning and retraining. The granular neural structure may be coarsened or refined depending on inter-granules relationships (transparency and interpretability) and error indices (accuracy). Practical applications of eGNN include evolving regressors, classifiers, forecasters and neurofuzzy controllers (76) (77) (82).

6.2 Related Work

There are two main approaches related with evolving granular neural networks. The first concerns granular neural networks for data granulation, granularity adaptation, and granular data processing. These are grounded in the principles of granular computing aiming at problem solving, complexity reduction, and structured thinking. In this case, granular neural networks often require multiple passes over data sets and offline learning. The second involves data stream oriented connectionist systems whose focus is on online tracking of nongranular data in nonstationary environments. Although a number of evolving fuzzy neural networks have succeeded in dealing with time-varying information, as it will be shown next, they are not able to process granular fuzzy input-output data. The evolving granular neural network framework addressed in this chapter benefits from and enhances both approaches.

6.2.1 Granular Neural Networks

On conceptualizing the world at different granularities, humans usually deal with information granules hierarchically (88) (142) (156). Human learning profits from the aggregation of local fragments to form a global picture. Granular computing offers human-like learning to be gradually embedded into modeling approaches through incremental granular data processing. Granule oriented neural networks generalize numeric data oriented neural networks because they provide mechanisms to process both singular and granular data.

Granular self-organizing maps (grSOM) (62) are neurofuzzy models for structure identification in linguistic system modeling. grSOM induces fuzzy intervals (granules) from the data using a metric tuned nonlinearly by a mass function. Its learning approach is supported by an analysis from the lattice theory and genetic algorithms. grSOM copes with ambiguity and can process fuzzy and interval input data. Computational results (62) recommend grSOM as a support tool in decision making and performance gain in classification tasks. Although grSOM copes with ambiguity and can process fuzzy and interval input data, it needs the entire data set to be available *a priori*. Learning is not incremental, and model structure and parameters are not adaptive to test data.

Fuzzy granular neural networks (FGNN) (157) consider numeric-linguistic data fusion, missing value prediction, and granular knowledge discovering in heterogeneous data sets. The FGNN packs relatively detailed granular data into coarser and more intelligible granules. Since FGNN requires offline learning and uses a gradient descent method to back propagate errors and to adapt parameters, it is not suitable to handle nonstationary data available gradually.

Granular reflex fuzzy min-max neural networks (GrRFMN) (108) learn from and classify interval granular online data. The network structure emulates the reflex mechanism of the human brain and deals with class overlapping using compensation neurons. The GrRFMN training algorithm gives a way to compute datum-model membership which leads to better network performance. Simulations with real data sets (108) assert the effectiveness of the approach.

The fuzzy min-max neural network (GFMM) (46) summarized in Section 3.2.2 is another example of a closely related granular neurofuzzy approach. The eGNN

approach distinguishes from the granular neural network approaches described above because it is oriented to fuzzy interval data streams, self-adapts its structure continuously, and does not need to store past data.

6.2.2 Data Stream Oriented Neural Networks

The continuous increase in availability of large amounts of data has motivated development of algorithms to process online data streams (11) (66). Evolving the structure of models using new information in nonstationary data streams is a challenging issue. Most of the traditional data mining and machine learning and statistical algorithms usually assume a form of stationarity and demand multiple passes over data sets. Thus, they do not meet the requirements needed for online learning. The design of evolving models is concerned with gradual model construction aiming at inducing new knowledge without catastrophic forgetting, and refining current knowledge keeping the system in operation.

Unsupervised evolving neural networks (65) can perform classification from unlabeled data streams. An example in this category is the Evolving Self-Organizing Map (eSOM), which uses standard principles of self-organization in an incremental basis. eSOM allows prototype neurons to evolve in the input space to simultaneously acquire and keep a topological representation. The neighborhoods of evolved neurons are not predefined, and differently from non-evolving SOM, they are determined online according to distances between neurons (65). The eSOM learning algorithm is fast since it dispenses neighborhood ranking search, but it is unable to cope with fuzzy intervals. eSOM is not proper for function approximation.

Evolving fuzzy neural networks (EFuNN) (63) adapt their structure and parameters through incremental, hybrid supervised/unsupervised online learning. EFuNN is able to model streams of data, new features, and new classes. Neurons and connections are created during learning. Fuzzy or non-fuzzy rules can be extracted from the network at any time. EFuNN has shown to be efficient in pattern recognition. The dynamic evolving neural-fuzzy inference system (DENFIS) (64) is another type of fuzzy rule-based system for adaptive online learning. Like EFuNN, DENFIS evolves using a hybrid incremental algorithm to fit new input

data. Fuzzy rules may be created, updated or deleted before or during system operation. It has been shown that DENFIS can learn complex temporal sequences and outperform similar approaches. Both, EFuNN and DENFIS, cannot process fuzzy intervals nor provide granular output.

The eGNN approach is different from the evolving neural network approaches described above primarily because of its ability to process fuzzy granular data streams. Moreover, eGNN is able to simultaneously provide single-valued and granular function approximation or classification.

6.3 Fuzzy Aggregation Neuron Model

This section introduces fuzzy aggregation neurons which are pertinent when processing data through successive layers of evolving granular neural networks.

Aggregation neurons are artificial neuron models based on aggregation operators (see Section 2.5). Evolving granular neural networks may use different types of aggregation neurons to perform information fusion (26). In general, there are no specific guidelines to choose a particular aggregation operator to construct a fuzzy neuron. The choice depends on the application environment and domain knowledge. Although the choice usually conforms to simplicity, transparency, and flexibility requirements, occasionally, it may conform to the system performance using the available data.

Let $\widetilde{x} = (\widetilde{x}_1, ..., \widetilde{x}_n)$ be a vector of membership degrees of a sample $x = (x_1, ..., x_n)$ in the fuzzy sets G_j of $G = (G_1, ..., G_n)$. Let $w = (w_1, ..., w_n)$ be a weighting vector such that

$$w_j \in [0, 1],\ j = 1, ..., n. \tag{6.1}$$

Fuzzy aggregation neurons employ product T-norm to perform synaptic processing and an aggregation operator $\mathbb{C}$ to fuse the individual results of synaptic processing in the neuron body. The output of a fuzzy aggregation neuron is:

$$o = \mathbb{C}(\widetilde{x}_1 w_1, ..., \widetilde{x}_n w_n). \tag{6.2}$$

An aggregation neuron produces a diversity of nonlinear mappings between neuron inputs and output depending on the choice of weights w, triangular norms T and S, and parameters e (neutral element of uninorms, see Section 2.5.3) and v (v-factor of compensatory aggregations, see Section 2.5.5). The structure of a fuzzy aggregation neuron is shown in Fig. 6.1. Examples of outputs generated by fuzzy aggregation neurons using uninorm U with $T = \min$, $S = \max$, $e = 0.3$, $v = 0$; and T-S aggregation L with $T = \min$, $S = \max$, $e = 0$, $v = 0.3$ are illustrated in Figs. 6.2(a) and 6.2(b), respectively.

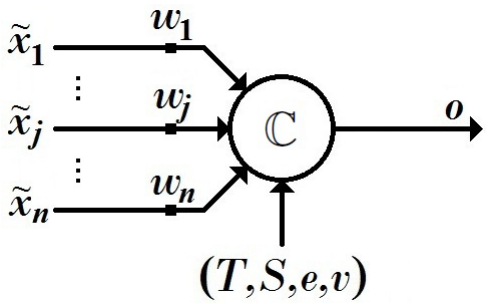
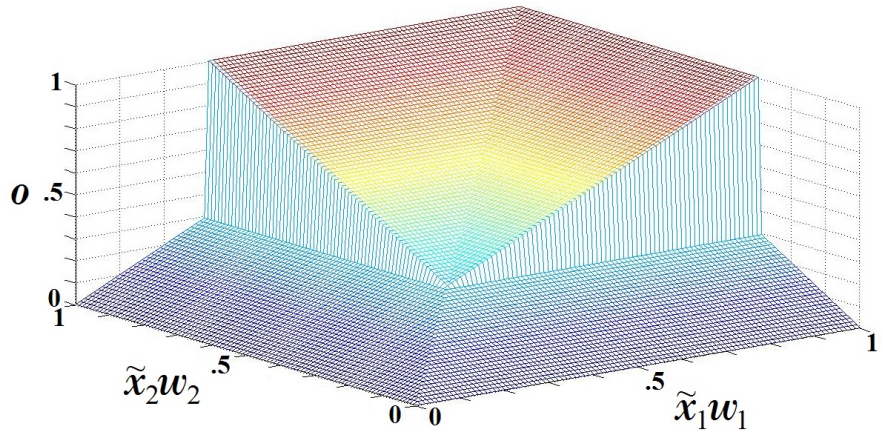


Figure 6.1: Fuzzy aggregation neuron model

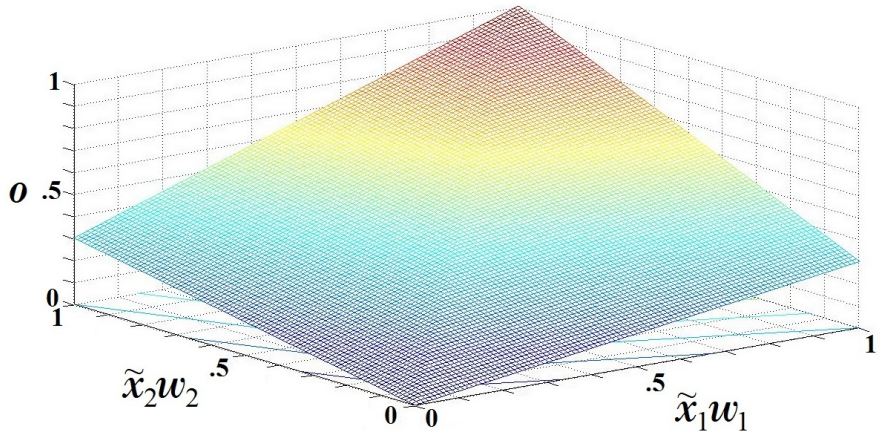
6.4 Structure and Processing

Let $x = (x_1, ..., x_n)$ be an input vector and y its corresponding output. Assume that the data stream $(x, y)^{[h]}$, $h = 1, ...$, are samples derived from a time-varying, nonstationary function f. Inputs x_j and output y are symmetric fuzzy data.

eGNN has a four-layer structure as shown in Fig. 6.3. The input layer delivers samples $x^{[h]}$, one at a time, to the network. The granular layer consists of a collection of fuzzy sets G_j^i, $j = 1, ..., n$; $i = 1, ..., c$, stratified from the input data. Fuzzy sets G_j^i, $i = 1, ..., c$, form a fuzzy partition of the j-th input domain, X_j. Similarly, fuzzy sets Γ^i, $i = 1, ..., c$, assemble a fuzzy partition of the output domain



(a) Uninorm $U_{min,max}$, $e = 0.3$, $v = 0$



(b) T-S operator $L_{min,max}$, $e = 0$, $v = 0.3$

Figure 6.2: Examples of input/output functions of fuzzy aggregation neurons

Y. A granule $\gamma^i = G_1^i \times ... \times G_n^i \times \Gamma^i$ is a fuzzy relation in $X_1 \times ... \times X_n \times Y$. Thus, granule γ^i has membership function $\gamma^i(x, y) = min\{G_1^i(x_1), ..., G_n^i(x_n), \Gamma^i(y)\}$ in $X_1 \times ... \times X_n \times Y$. Granule γ^i is denoted by $\gamma^i = (G^i, \Gamma^i)$, with $G^i = (G_1^i, ..., G_n^i)$, for short. Moreover, the granule γ^i comes with an associated local function p^i. In general, functions p^i can be of different type and are not required to be linear. This study employs non-fuzzy, real-valued affine functions:

$$p^i(\hat{x}_1, ..., \hat{x}_n) = \hat{y}^i = a_0^i + \sum_{j=1}^{n} a_j^i \hat{x}_j, \tag{6.3}$$

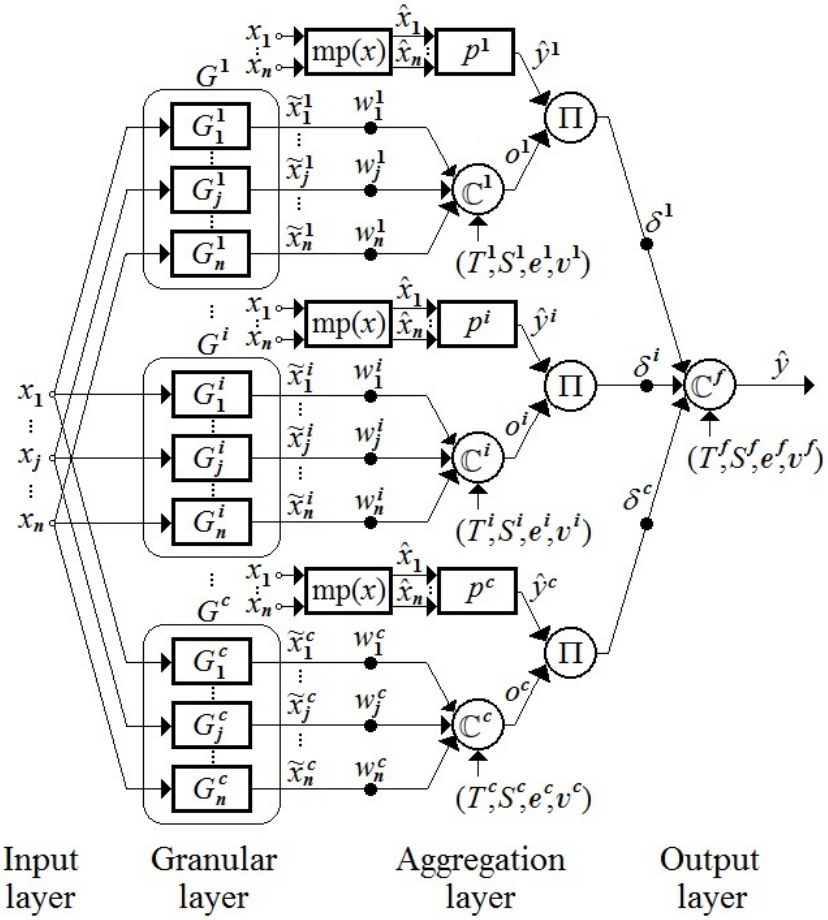


Figure 6.3: Single-valued approximation provided from input data processing

for simplicity. Parameters a_0^i and a_j^i are real values; $\hat{x}_j$ is the midpoint of $x_j = (\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j)$, which is obtained from:

$$mp(x_j) = \hat{x}_j = \frac{\underline{x}_j + \overline{x}_j}{2}. \tag{6.4}$$

Similarity degrees $\widetilde{x}^i = (\widetilde{x}_1^i, ..., \widetilde{x}_n^i)$ arise as result of matching between an input $x = (x_1, ..., x_n)$ and fuzzy sets of $G^i = (G_1^i, ..., G_n^i)$, see Section 2.4.3. The aggregation layer comprises fuzzy aggregation neurons $\mathbb{C}^i$, $i = 1, ..., c$, whose role is to combine information from the different inputs. A fuzzy aggregation neu-

ron $\mathbb{C}^i$ combines weighted similarity degrees $(\widetilde{x}_1^i w_1^i, ..., \widetilde{x}_n^i w_n^i)$ into a single value o^i. The output layer aggregates weighted values $(o^1\hat{y}^1\delta^1, ..., o^c\hat{y}^c\delta^c)$ using a fuzzy aggregation neuron $\mathbb{C}^f$ to produce the single network output $\hat{y}^{[h]}$. Formally,

$$\hat{y} = \mathbb{C}^f(o^1\hat{y}^1\delta^1, ..., o^c\hat{y}^c\delta^c). \tag{6.5}$$

An m-output eGNN requires a vector of local functions $(p_1^i, ..., p_m^i)$, output layer neurons $(\mathbb{C}_1^f, ..., \mathbb{C}_m^f)$, and outputs $(\hat{y}_1, ..., \hat{y}_m)$. The network output $\hat{y}$, obtained as illustrated in Fig. 6.3, is single-valued approximation of f independent whether the input data are real numbers, fuzzy numbers or fuzzy intervals.

Granular approximation of a function f at step H is given by a set of granules γ^i, $i = 1, ..., c$, such that:

$$(x, y)^{[h]} \subseteq \bigcup_{i=1}^{c} \gamma^i, \ h = 1, ..., H. \tag{6.6}$$

Granular approximation is formed by granulating input data $x^{[h]}$ into fuzzy sets of G^i, as in Fig. 6.3, and output data $y^{[h]}$ into fuzzy sets Γ^i, as in Fig. 6.4. Note in Fig. 6.4 that the granular approximation is the convex hull of output fuzzy sets Γ^{i^*}, where i^* are indices of the active granules, that is, those for which $o^i > 0$. This guarantees that the single approximation $\hat{y}^{[h]}$ is included in the granule.

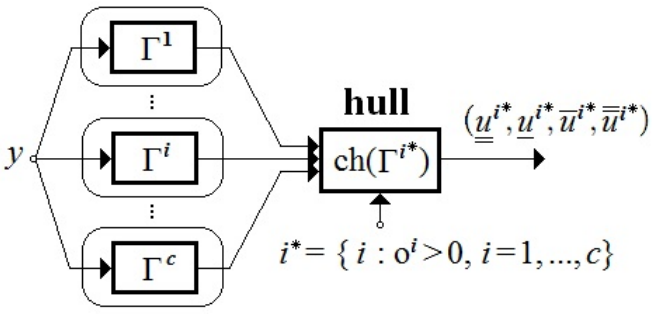


Figure 6.4: Granular approximation formed by input and output data granulation

The convex hull of trapezoidal fuzzy sets $\Gamma^1, ..., \Gamma^i, ..., \Gamma^c$, with $\Gamma^i = (\underline{\underline{u}}^i, \underline{u}^i, \overline{u}^i, \overline{\overline{u}}^i)$, is a trapezoidal fuzzy set $ch(\Gamma^1, ..., \Gamma^c)$ such that:

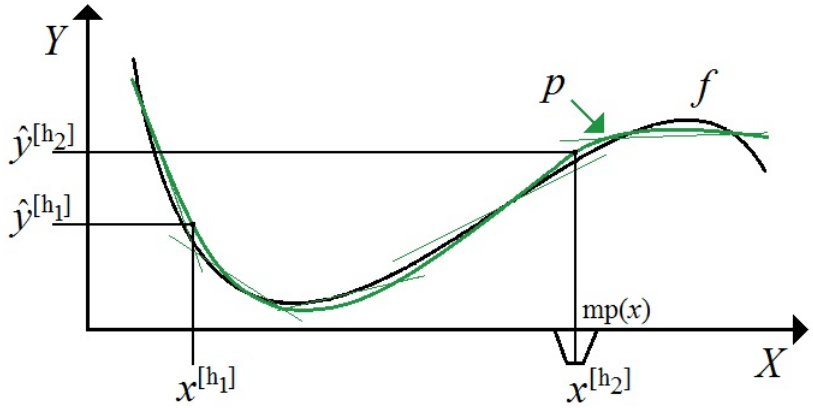
$$ch(\Gamma^1, ..., \Gamma^c) = (min(\underline{\underline{u}}^1, ..., \underline{\underline{u}}^c), min(\underline{u}^1, ..., \underline{u}^c), \\ max(\overline{u}^1, ..., \overline{u}^c), max(\overline{\overline{u}}^1, ..., \overline{\overline{u}}^c)). \tag{6.7}$$

Particularly in Fig. 6.4, the trapezoid $(\underline{\underline{u}}^{i^*}, \underline{u}^{i^*}, \overline{u}^{i^*}, \overline{\overline{u}}^{i^*})$ that results from $ch(\Gamma^{i^*})$, where $i^* = \{i : o^i > 0, i = 1, ..., c\}$, is a granular approximation of output data y. It is worth noting that granular approximation at instant h does not depend on the availability of $y^{[h]}$ because o^i is obtained from $x^{[h]}$ (see Fig. 6.3). Only the collection of output fuzzy sets Γ^i is required.

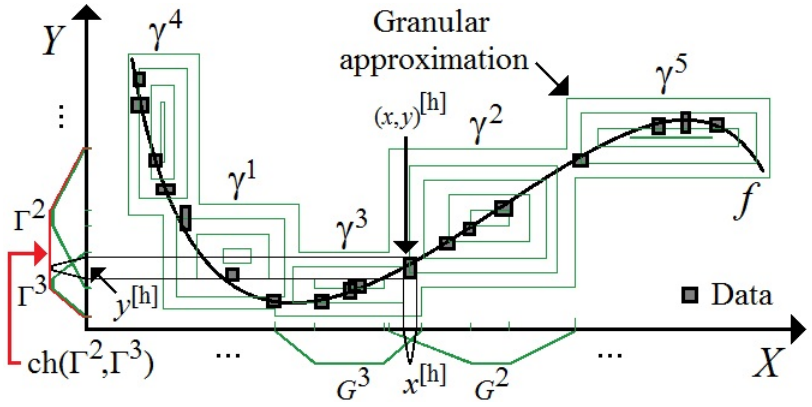
Figure 6.5 shows an example of single-valued and granular approximation, p and $\bigcup_{i=1}^{c} \gamma^i$, of a function f. In Fig. 6.5(a), a singular input $x^{[h_1]}$ and a granular input $x^{[h_2]}$ produce singular outputs $\hat{y}^{[h_1]}$ and $\hat{y}^{[h_2]}$ by using p. In Fig. 6.5(b), the granular input $x^{[h]}$ activates the fuzzy sets of G^2 and G^3. Therefore, granular output is obtained from the convex hull operation $ch(\Gamma^2, \Gamma^3)$. It follows that $y^{[h]} \subset ch(\Gamma^2, \Gamma^3)$. If $y^{[h]} \not\subset ch(\Gamma^2, \Gamma^3)$, then either fuzzy set Γ^2 or Γ^3 is adjusted to enclose $y^{[h]}$. Granule adaptation is addressed in the next section.

Information processing in intermediate layers of eGNN is single-valued to speed up calculations. Thus, input and output fuzzy sets can be viewed as decoders and encoders of granular data. eGNN comes with an incremental learning algorithm to adapt its structure and parameters over time. The algorithm will be detailed in Section 6.5.

eGNN evolves functional and linguistic fuzzy models. While functional fuzzy systems are more precise, linguistic fuzzy systems are more interpretable. Accuracy and interpretability require tradeoffs and one usually excels over the other. eGNN joins functional and linguistic systems into a single framework. Under the assumption of specific weights and neurons, fuzzy rules extracted from eGNN can be of the type:



(a) eGNN single-valued approximation p of function f



(b) eGNN granular approximation $\bigcup_{i=1}^{5} \gamma^i$ of function f

Figure 6.5: eGNN single-valued (a) and granular (b) approximation of a function

$$R^i\text{: IF } (x_1 \text{ is } G_1^i) \text{ AND ... AND } (x_n \text{ is } G_n^i)$$
$$\text{THEN } \underbrace{(y \text{ is } \Gamma^i)}_{\text{linguistic}} \text{ AND } \underbrace{\hat{y} = p^i(x_1, ..., x_n)}_{\text{functional}}$$

This type of eGNN combines Mamdani and Takagi-Sugeno fuzzy models.

6.5 Learning in eGNN

This section details the eGNN learning algorithm. Differently from the usual top-down granular approaches used, because the data domain is unknown beforehand,

the eGNN learning approach is mostly bottom-up.

Developing the fuzzy rules encoded in the network structure and approximating nonstationary functions from granular data streams are the key concerns of the learning approach. The eGNN learning employs a sample-per-sample testing-before-training method on a recursive basis. This method portrays a truly online data stream scenario. We assume that no granules, neurons and connections exist before training starts. The algorithm builds the network structure in plug-and-play mode. Single pass over data enables eGNN to address the issues of unbounded data sets and scalability of computationally hard problems.

We assume trapezoidal membership functions $G_j^i = (\underline{\underline{g}}_j^i, \underline{g}_j^i, \overline{g}_j^i, \overline{\overline{g}}_j^i)$ and input data $x_j = (\underline{\underline{x_j}}, \underline{x_j}, \overline{x_j}, \overline{\overline{x_j}})$. Similarly, $\Gamma^i = (\underline{\underline{u}}^i, \underline{u}^i, \overline{u}^i, \overline{\overline{u}}^i)$ and output data $y = (\underline{\underline{y}}, \underline{y}, \overline{y}, \overline{\overline{y}})$ are trapezoids. Each rule antecedent $G^i = (G_1^i, ..., G_n^i)$ has a correspondent consequent Γ^i. With $\gamma^i = (G^i, \Gamma^i)$, eGNN looks at examples (x, y) at a coarser granule size.

6.5.1 Adapting the Granularity

Balancing parametric and structural adaptation is a key to capture gradual and abrupt changes of nonstationary systems online. The neural pattern recognition literature refers to this issue as the stability-plasticity dilemma (29). Structural plasticity in eGNN means creating new granules and rules to memorize new concepts. This avoids the rules learned to be exposed to catastrophic forgetting. Structural stability preserves the eGNN structure, but allows adaptation of the existing granules and rules to smooth and slow changes. Parametric refinement partially retains the information. The procedure suggested below is a way to parsimoniously reconcile plasticity and stability in eGNN.

The maximum width fuzzy sets G_j^i are allowed to expand is denoted by ρ, that is, $wdt(G_j^i) \leq \rho$, $j = 1, ..., n$; $i = 1, ..., c$. The value of ρ affects the granularity, accuracy and transparency of the models. Similar as in Chapter 5, the expansion region of a fuzzy set G_j^i is

$$E_j^i = [mp(G_j^i) - \frac{\rho}{2}, mp(G_j^i) + \frac{\rho}{2}]. \tag{6.8}$$

It follows that

$$wdt(G_j^i) \;\leq\; wdt(E_j^i)\ \forall j, i. \tag{6.9}$$

ρ plays a pivotal role in mediating plasticity and stability of eGNN structures. Expressions similar to (6.8) and (6.9) can be derived for fuzzy sets Γ^i. Expansion regions help to derive criteria for deciding whether or not granular data should be considered enclosed by the current granular model.

In practice, $\rho \in [0, 1]$ determines the need to create or adapt rules. In the most general case, the neural network starts learning with an empty rule base and devoid of knowledge about data properties. Therefore, in these cases it is interesting to initialize ρ with an intermediate value to allow structural stability and plasticity equally. We use $\rho^{[0]} = 0.5$ as default initial value.

A simple and fast procedure to evolve ρ is as follows. Let r be the number of rules created after h_r steps. If the number of rules grows faster than a rate η, that is $r > \eta$, then ρ is increased:

$$\rho(\text{new}) \;=\; \left(1 + \frac{r}{h_r}\right) \rho(\text{old}). \tag{6.10}$$

The idea is to reject large rule bases because they increase model complexity and may not help generalization. Equation (6.10) punishes ρ and acts against outbursts of growth.

Otherwise, if the number of rules grows at a rate smaller than η, that is $r \leq \eta$, then ρ is decreased,

$$\rho(\text{new}) \;=\; \left(1 - \frac{(\eta - r)}{h_r}\right) \rho(\text{old}). \tag{6.11}$$

If $\rho = 1$, then eGNN is structurally stable, but unable to capture abrupt changes. Conversely, if $\rho = 0$, then eGNN overfits the data causing excessive complexity and irreproducible optimistic results. Life-long adaptability is reached choosing intermediate values for ρ as depicted in Fig. 6.6.

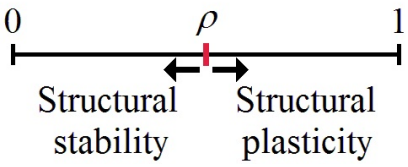


Figure 6.6: Stability-plasticity tradeoff and the role ρ in eGNN systems

Reducing the maximum width allowed for granules may require shrinking larger granules to fit them to new data. In this case, the support of a fuzzy set G_j^i is narrowed as follows:

$$\text{If } mp(G_j^i) - \frac{\rho(\text{new})}{2} > \underline{\underline{g}}_j^i \text{ then } \underline{\underline{g}}_j^i(\text{new}) = mp(G_j^i) - \frac{\rho(\text{new})}{2}$$
$$\text{If } mp(G_j^i) + \frac{\rho(\text{new})}{2} < \overline{\overline{g}}_j^i \text{ then } \overline{\overline{g}}_j^i(\text{new}) = mp(G_j^i) + \frac{\rho(\text{new})}{2}$$

Cores $[\underline{\underline{g}}_j^i, \overline{g}_j^i]$ and supports $[\underline{\underline{u}}^i, \overline{\overline{u}}^i]$ and cores $[\underline{u}^i, \overline{u}^i]$ of fuzzy sets Γ^i are handled similarly. Time-varying granularity is useful to avoid guesses on how fast and how often the data stream changes. The accuracy-interpretability tradeoff is an important issue in neurofuzzy computing (113).

6.5.2 Calculating Similarity Degree

As input data and granules are trapezoidal fuzzy objects, a potential similarity measure to determine how much they match is:

$$\widetilde{x}_j^i = \begin{cases} 1 - \left(\frac{|\underline{\underline{g}}_j^i - \underline{\underline{x}}_j| + |\underline{g}_j^i - \underline{x}_j| + |\overline{g}_j^i - \overline{x}_j| + |\overline{\overline{g}}_j^i - \overline{\overline{x}}_j|}{4(max(\overline{\overline{g}}_j^i, \overline{\overline{x}}_j) - min(\underline{\underline{g}}_j^i, \underline{\underline{x}}_j))} \right) & \text{if } x_j \cap G_j^i \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{6.12}$$

This measure returns $\widetilde{x}_j^i$ equal to 1 for superposed trapezoids and decreases linearly as any numerator term increases. The numerator terms are Hamming distances between pairs of parameters of two different trapezoids. The denominator scales the result to lie in the range from zero to one. Non-overlapped trapezoids

are considered dissimilar, yielding $\widetilde{x}_j^i$ equal to 0.

Reference (46) introduces a similarity measure between an interval and a fuzzy set. The measure is based on the minimum T-norm and decreases monotonically as the distance between the interval and the fuzzy set increases. Reference (108) pointed out that this similarity measure gives low values to significantly overlapped objects. In extreme cases, the similarity measure proposed in (46) returns zero for a fuzzy set contained in an interval. To avoid this situation, (108) proposes the average of the overlapping area between the interval and fuzzy set as similarity measure.

The similarity measure in equation (6.12) extends the measures in (46) and (108) to trapezoidal fuzzy sets. It overcomes problems which may arise due to some T-norms when considering the boundaries of intervals, as in (46), but is faster than the measure in (108) because it does not need to compute the area of arbitrary polygons.

6.5.3 Creating Granules

The incremental procedure to create granules runs whenever the support of at least one entry of an input vector $(x_1, ..., x_n)$ is not enclosed by expansion regions $(E_1^i, ..., E_n^i)$, $i = 1, ..., c$. In this case, fuzzy sets G^i cannot be expanded beyond the limit ρ to fit the sample. Analogously, if $supp(y)$ is not enclosed by E^i for at least one Γ^i, then the sample should be enclosed by a new granule.

A new granule γ^{c+1} is assembled from fuzzy sets G_j^{c+1} and Γ^{c+1} whose parameters match the sample:

$$(\underline{\underline{g}}_j^{c+1}, \underline{g}_j^{c+1}, \overline{g}_j^{c+1}, \overline{\overline{g}}_j^{c+1}) = (\underline{\underline{x}}_j, \underline{x}_j, \overline{x}_j, \overline{\overline{x}}_j), \tag{6.13}$$

$$(\underline{\underline{u}}^{c+1}, \underline{u}^{c+1}, \overline{u}^{c+1}, \overline{\overline{u}}^{c+1}) = (\underline{\underline{y}},\ \underline{y},\ \overline{y},\ \overline{\overline{y}}). \tag{6.14}$$

Coefficients of the real-valued local function p^{c+1} are set to

$$a_0^{c+1} = mp(y),\ a_j^{c+1} = 0,\ j \neq 0. \tag{6.15}$$

6.5.4 Adapting Granules

Adaptation of granules means to expand or contract the support and the core of fuzzy sets G_j^i and Γ^i to enclose new data, and to refine the coefficients of the local functions p^i.

Granule γ^i can be adapted whenever a sample (x, y) falls within its expansion region, that is,

$$supp(x_j) \subset E_j^i,\ j = 1, ..., n \text{ and } supp(y) \subset E^i. \tag{6.16}$$

This means that either the sample is enclosed by granule γ^i or it is close enough so that the granule can be expanded to enclose it. In situations in which two or more granules are qualified to enclose the data, adapting only one of the granules is enough to guarantee data inclusion. In particular, we may chose γ^i such that $i = arg\ max(o^1, ..., o^c)$. In other words, γ^i is the granule with the highest activation level for a given sample.

Adaptation proceeds depending on where the input datum x_j is regarding fuzzy set G_j^i:

$$\begin{array}{lll}
\text{If } \underline{\underline{x}}_j \in [mp(G_j^i) - \frac{\rho}{2}, \underline{\underline{g}}_j^i] & \text{then} & \underline{\underline{g}}_j^i(\text{new}) = \underline{\underline{x}}_j \\
\text{If } \underline{x}_j \in [mp(G_j^i) - \frac{\rho}{2}, \underline{g}_j^i] & \text{then} & \underline{g}_j^i(\text{new}) = \underline{x}_j \\
\text{If } \underline{x}_j \in [\underline{g}_j^i, mp(G_j^i)] & \text{then} & \underline{g}_j^i(\text{new}) = \underline{x}_j \\
\text{If } \underline{x}_j \in [mp(G_j^i), mp(G_j^i) + \frac{\rho}{2}] & \text{then} & \underline{g}_j^i(\text{new}) = mp(G_j^i) \\
\text{If } \overline{x}_j \in [mp(G_j^i) - \frac{\rho}{2}, mp(G_j^i)] & \text{then} & \overline{g}_j^i(\text{new}) = mp(G_j^i) \\
\text{If } \overline{x}_j \in [mp(G_j^i), \overline{g}_j^i] & \text{then} & \overline{g}_j^i(\text{new}) = \overline{x}_j \\
\text{If } \overline{x}_j \in [\overline{g}_j^i, mp(G_j^i) + \frac{\rho}{2}] & \text{then} & \overline{g}_j^i(\text{new}) = \overline{x}_j \\
\text{If } \overline{\overline{x}}_j \in [\overline{\overline{g}}_j^i, mp(G_j^i) + \frac{\rho}{2}] & \text{then} & \overline{\overline{g}}_j^i(\text{new}) = \overline{\overline{x}}_j
\end{array}$$

The first and last rules imply support expansion, the second and seventh rules core expansion, and the remaining cases core contraction. Notice that these adaptation procedures are similar to those of Section 5.4.4 related to the FBeM approach (refer to Fig. 5.2 for examples).

Operations on core parameters, $\underline{g}_j^i$ and $\overline{g}_j^i$, require adjustment of the midpoint of the respective fuzzy sets:

$$mp(G_j^i)(\text{new}) = \frac{\underline{g}_j^i(\text{new}) + \overline{g}_j^i(\text{new})}{2}. \tag{6.17}$$

As a result, support contraction may happen in two occasions:

$$\begin{aligned} &\text{If } mp(G_j^i)(\text{new}) - \tfrac{\rho}{2} > \underline{\underline{g}}_j^i \text{ then } \underline{\underline{g}}_j^i(\text{new}) = mp(G_j^i)(\text{new}) - \tfrac{\rho}{2} \\ &\text{If } mp(G_j^i)(\text{new}) + \tfrac{\rho}{2} < \overline{\overline{g}}_j^i \text{ then } \overline{\overline{g}}_j^i(\text{new}) = mp(G_j^i)(\text{new}) + \tfrac{\rho}{2} \end{aligned}$$

The adaptation of consequent fuzzy sets Γ^i is done similarly using output data y. Coefficients a_j^i of the local functions p^i are updated using the recursive least squares algorithm as detailed in Appendix B.

6.5.5 Incremental Weighting

Aggregation layer weights $w_j^i \in [0, 1]$ embody the importance of the membership degree of the j-th attribute in fuzzy set G_j^i to the neural network output. If $w_j^i = 1$, then the output is not affected. A relatively lower value of w_j^i discounts the impact of the respective attribute. If $w_j^i = 0$, then the attribute is ignored. The procedure described below assigns lower weight values to less helpful attributes.

Whenever a new granule γ^{c+1} is created, the learning procedure assigns $w_j^{c+1} = 1$, $j = 1, ..., n$. If it is known *a priori* that the input variables have different importance, then values for w_j^{c+1} can be chosen differently to reflect domain knowledge.

Taking into account the similarity measure (6.12) and the approximation error (B.4), weights w_j^i corresponding to the most active granule γ^i, where $i = arg\ max(o^1, ..., o^c)$, are recursively updated using:

$$w_j^i(\text{new}) = w_j^i(\text{old}) - \widetilde{x}_j^i o^i |\epsilon|. \tag{6.18}$$

The idea here is that the single-valued approximation $\hat{y}$ is more affected by more active granules and attributes. Equation (6.18) ascribes to the j-th attribute of G^i a proportion of the approximation error.

Incremental weighting looks for relevant subsets of input variables. The procedure (6.18) suggested is particularly simple and fast to compute. More elaborate approaches for incremental weighting are addressed in (98).

6.5.6 Pruning

Pruning granules simplifies the neural network structure and keeps it flexible to track dynamic behavior. We opt to prune the most inactive granules because retaining a small number of highly active granules favors compactness and speed.

Output layer weights $\delta^i \in [0, 1]$ help pruning by encoding the amount of data assigned to granule γ^i. Learning starts with $\delta^i = 1$. During the next steps, δ^i is reduced whenever γ^i is not activated within h_r steps as follows:

$$\delta^i(\text{new}) = \zeta\delta^i(\text{old}), \tag{6.19}$$

where $\zeta \in [0, 1]$. Otherwise, if γ^i is activated at least once within h_r steps, then δ^i is increased:

$$\delta^i(\text{new}) = \delta^i(\text{old}) + \zeta(1 - \delta^i(\text{old})). \tag{6.20}$$

If the value of δ^i is less than a threshold ϑ, then granule γ^i, its respective neuron A^i and connections are pruned since they do not affect system accuracy significantly. If the application requires memorization of rare events or cyclical behavior is envisioned, then it may be the case to set $\vartheta = 0$ and let $\delta^i \to 0^+$. In this case, the granule is kept in the network structure.

6.5.7 Combining Granules

Relationships between granules may be strong enough to justify forming a larger granule that inherits the information of the lower level granules. A metric to measure the distance between trapezoidal objects, say granules γ^{i_1} and γ^{i_2}, is:

$$D(\gamma^{i_1},\gamma^{i_2}) = \frac{1}{4(n+1)}(\sum_{j=1}^{n}(|\underline{\underline{g}}_j^{i_1} - \underline{\underline{g}}_j^{i_2}| + |\underline{g}_j^{i_1} - \underline{g}_j^{i_2}| + |\overline{g}_j^{i_1} - \overline{g}_j^{i_2}| + \\ + |\overline{\overline{g}}_j^{i_1} - \overline{\overline{g}}_j^{i_2}|) + |\underline{\underline{u}}^{i_1} - \underline{\underline{u}}^{i_2}| + |\underline{u}^{i_1} - \underline{u}^{i_2}| + |\overline{u}^{i_1} - \overline{u}^{i_2}| + |\overline{\overline{u}}^{i_1} - \overline{\overline{u}}^{i_2}|). \quad (6.21)$$

D is a distance measure since it satisfies

$D(\gamma^{i_1},\gamma^{i_2}) \geq 0$
$D(\gamma^{i_1},\gamma^{i_2}) = 0$ if and only if $\gamma^{i_1} = \gamma^{i_2}$
$D(\gamma^{i_1},\gamma^{i_2}) = D(\gamma^{i_2},\gamma^{i_1})$
$D(\gamma^{i_1},\gamma^{i_3}) \leq D(\gamma^{i_1},\gamma^{i_2}) + D(\gamma^{i_2},\gamma^{i_3})$

for any γ^{i_1}, γ^{i_2} and γ^{i_3}. Distance D is fast to compute and is more accurate than both distance between midpoints of trapezoids and distance between closest points of trapezoids. Change in any of the parameters of the underlying trapezoids is reflected in the value of distance D.

Granules are combined after h_r steps considering the lowest value of $D(\gamma^{i_1},\gamma^{i_2})$, $i_1, i_2 = 1, ..., c$, $i_1 \neq i_2$, and a decision criterion. For instance, the decision criterion may consider if the new granule obeys the maximum width allowed ρ.

A new granule γ^i, coarsening of γ^{i_1} and γ^{i_2}, is formed by trapezoidal membership functions G_j^i as follows:

$$G_j^i = ch(G_j^{i_1}, G_j^{i_2}),\ j = 1, ..., n. \quad (6.22)$$

Γ^i is obtained similarly. The new granule γ^i encloses the support and core of the granules combined.

The coefficients of the new local function p^i are found as:

$$a_j^i = \frac{1}{2}(a_j^{i_1} + a_j^{i_2}),\ j = 0, ..., n. \tag{6.23}$$

Combining granules avoids redundancy by eliminating similar rules of the rule base. Reference (96) has emphasized the importance of having a compact rule base in evolving fuzzy systems.

6.5.8 Learning Algorithm

The learning procedure to evolve granular neural networks can be summarized by the following algorithm:

BEGIN
Select a type of neuron for the aggregation and output layers;
Set parameters ρ, h_r, η, ζ, ϑ, $c = 0$;
Read $(x, y)^{[h]}$, $h = 1$;
Create granule γ^{c+1}, neurons $\mathbb{C}^{c+1}$, $\mathbb{C}^f$, and respective connections;
For $h = 2, ...$ do
  Read $(x, y)^{[h]}$;
  Input $x^{[h]}$ to the network;
  Compute compatibility degrees $(o^1, ..., o^c)$;
  Aggregate values using $\mathbb{C}^f$ to get single-valued approximation $\hat{y}^{[h]}$;
  Compute the convex hull of $\Gamma^{i^*}, i^* = \{i, o^i > 0\}$;
  Find granular approximation $(\underline{\underline{u}}^{i^*}, \underline{u}^{i^*}, \overline{u}^{i^*}, \overline{\overline{u}}^{i^*})$;
  Compute output error $\epsilon^{[h]} = mp(y^{[h]}) - \hat{y}^{[h]}$;
  If $x^{[h]}$ is not within expansion regions $E^i \forall i$
    Create granule γ^{c+1}, neuron $\mathbb{C}^{c+1}$ and connections;
  Else
    Update the most active granule γ^i, $i = arg\ max(o^1, ..., o^c)$;
    Update local function parameters a_j^i using RLS;
    Update connection weights $w_j^i \forall j, i$;
  If $h = \alpha h_r$, $\alpha = 1, 2, ...$
    Combine granules when feasible;
    Update model granularity ρ;
    Adapt connection weights $\delta^i \forall i$;
    Prune inactive granules and respective connections;
END

6.6 Summary

This chapter has introduced a fuzzy data stream modeling framework based on an evolving fuzzy granular neural network approach. The eGNN framework processes fuzzy data streams using fuzzy granular models, fuzzy aggregation neurons, and an online incremental learning algorithm. Its neurofuzzy structure encodes a set of fuzzy rules and a fuzzy inference system that establishes a tradeoff between precision and interpretability combining functional and linguistic fuzzy models. The eGNN provides single-valued approximation as well as granular approximation of functions.

Chapter 7

Application Examples

The application examples addressed in this section consider numeric, interval, and fuzzy data streams to demonstrate the usefulness of evolving granular approaches. We look forward to a low error rate, concise constructs, a high processing speed, and meaningful understandable rules in semi-supervised classification, function approximation, time series prediction, and control problems.

7.1 Introduction

The experimental work described in this chapter is based on data sets that have already been collected for some purpose. Information extraction and knowledge discovery are therefore based on simulations of singular and granular data streams in online environment. All experiments require evolving granular systems to deal with data they have never seen before and that demand a prompt response before being used for model adaptation. The following assumptions hold true:

- online approaches start learning from scratch, unless otherwise stated;
- previous data is neither stored nor retrieved (space constraint);
- the data streams - no matter single-valued or granular - dictate the granularity of the underlying models;
- no missing values are found in the original data sets;

- the data sampling frequency is constant for the different scenarios;
- the per-sample latency of the algorithms is not larger than the time interval between samples (time constraint).

Ideally, online modeling methods, such as IBeM, FBeM and eGNN, should retain all previous relevant knowledge and rely on the newest input data to perform classification, prediction, approximation or control.

7.2 Semi-Supervised Classification

Semi-supervised learning methods use both labeled and unlabeled data to build pattern classification systems. Mixtures of labeled and unlabeled data are easily found in practice (47) (102) (111) (158). Often, the acquisition of labeled data requires human experts to manually classify training instances. Manual classification can be greatly influenced by subjectivity as well as not be feasible, as in when we handle large data sets in online environment. There are situations in which instances are labeled and apparently ask for fully supervised learning methods and standard procedures of classifier design. However, the labeling process may have been unreliable so that our confidence in the labels already assigned is relatively low (114). In these cases we resort to semi-supervised learning and accept only a fraction of instances that we deem to have been labeled correctly.

Let an input-output pair (x, y) be related through $y = f(x)$. We seek an approximation to f that allows us to predict the value of y given x. In classification problems, y is a class label, a value in a set $\{C_1, ..., C_m\} \in \mathbb{N}^m$, and relation f specifies class boundaries. In the more general, semi-supervised case, C_k may or may not be known when x arrives. Classification of data streams involves pairs $(x, C)^{[h]}$ of time-sequenced data indexed by h. Nonstationarity requires evolving granular classifiers to identify time-varying relations $f^{[h]}$.

The experiments described next aim to demonstrate the ability of evolving granular methods in classifying unbalanced single-valued partially-supervised streaming data subject to gradual and abrupt concept changes.

7.2.1 Rotation of Twin Gaussians

Gradual change is evaluated taking into account a two-attribute classification problem where two partially overlapping Gaussians rotate anti-clockwise around a central point, as shown in Fig. 7.1. The Gaussians are initially centered at $(4, 4)$ and $(6, 6)$ with standard deviation fixed at 0.8. The kinematics of their movement around the point $(5, 5)$ is as follows:

$$\theta^{[h]} = \theta^{[h-1]} + \phi \tag{7.1}$$

$$x_1^{[h]} = 5 + \sqrt{2}\ cos(\theta^{[h]}) \tag{7.2}$$

$$x_2^{[h]} = 5 + \sqrt{2}\ sin(\theta^{[h]}) \tag{7.3}$$

The initial reference angle $\theta^{[0]}$ for Class 1 is 225^o and for Class 2 is 45^o. For $h = 1, ..., 200$, the rotating rate ϕ is kept equal to 0, meaning that no drift is present. The rotation starts at $h = 201$, and, for $h = 201, ..., 400$, $\phi = 0.45$. The final positions of the Gaussians are $(6, 4)$ and $(4, 6)$, respectively, for classes 1 and 2. The total rotation angle is 90^o. Samples from both classes arrive randomly and sequentially.

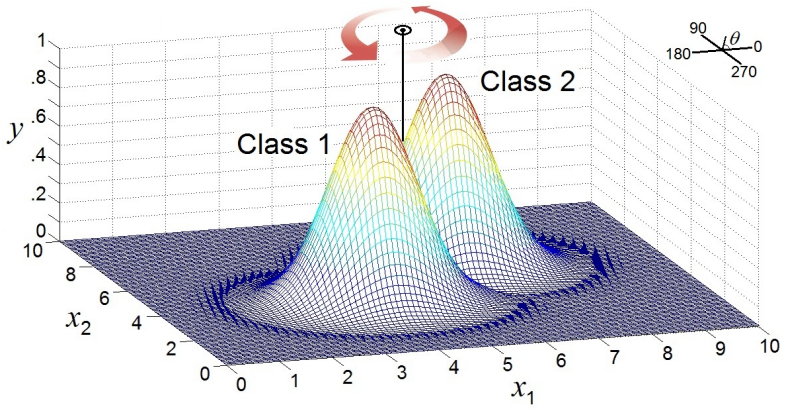


Figure 7.1: The rotating Gaussians problem

Assume classes 1 and 2 in Fig. 7.1 correspond to the positive and negative classes, respectively. Consider a confusion matrix consisting of two rows and two

columns representing the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) as in the receiver operating characteristic (ROC) method (45). The prediction accuracy of a classifier can be defined as

$$\text{Acc} \;=\; \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FP}+\text{TN}+\text{FN}} \;.\; 100\%. \tag{7.4}$$

This metric is usually employed for balanced data sets. The ROC method provides a convenient way to evaluate the quality of evolving classifiers in unknown nonstationary environment because it is insensitive to both changes in class distribution and proportion of samples per class (45).

The ROC space is defined over the TP_ratio and FP_ratio,

$$\text{TP\_ratio} \;=\; \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{7.5}$$

$$\text{FP\_ratio} \;=\; \frac{\text{FP}}{\text{FP}+\text{TN}}. \tag{7.6}$$

For each class, ROC applies threshold values across the interval [0,1] to outputs. Each cut-off threshold corresponds to a point (sensitivity/specificity pair) in the ROC space. The closer the ROC curve is to the upper left corner, the better the classification. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner.

We look for the decision boundary between the Gaussians using the newest input data. Starting IBeM parameters are $\rho = 0.35$, $h_r = 40$ and $\eta = 2$; FBeM starts with the same parameters as IBeM plus $1/\psi = h_r$; and eGNN employs $\mathbb{C}^i = T_{min}$, $\mathbb{C}^f = S_{max}$, $\rho = 0.42$, $h_r = 40$, $\eta = 2$ and $\zeta = \vartheta = 0.5$. These values have worked well for a range of classification problems. The functional consequents of IBeM, FBeM and eGNN can be neglected in classification tasks.

Aiming at emphasizing the importance of incremental learning in nonstationary data stream classification, we first compare evolving and non-evolving approaches for the rotating Gaussians problem. We consider widely known non-evolving methods, viz. a multi-layer perceptron (MLP) neural network trained offline via gradient descent (55), and a fuzzy C-means (FCM) clustering method (22). Table 7.1 summarizes the results averaged over 5 runs for each method.

Table 7.1: Rotating Gaussians: comparing evolving/non-evolving methods

Model	# Avg.Rules	Acc	CPU*
MLP	–	57.0	41.0
FCM	10.0	60.5	0.8
IBeM	2.8	87.4	0.9
FBeM	3.4	92.3	0.4
eGNN	4.3	92.1	0.3

* Average CPU time per sample in milliseconds

Table 7.1 shows that FBeM is the most accurate approach in this example, producing an Acc index slightly superior to that of eGNN. The performance of non-evolving methods degrades when the concept changes in online drifting scenarios because the structure and parameters of the underlying models are fixed. MLP and FCM could not track the rotation of the Gaussians leading to relatively worse results. We also consider for performance evaluation the average number of rules in the model structure over the learning steps, and CPU time in a dual-core 2.54GHz processor with 4GB of RAM. IBeM provided the most compact model with an average of 2.8 rules during the learning process, whereas eGNN was the fastest method being able to process each sample in 0.3 milliseconds.

Figure 7.2 shows the ROC curves produced by the evolving and non-evolving methods. The results obtained were essentially the same in the different runs. The diagonal line corresponds to random guessing, e.g., coin flipping.

Note in Fig. 7.2 that the area under the ROC curves of evolving granular approaches is larger than that referring to ROC curves of non-evolving approaches. The ROC analysis confirms that FBeM is slightly superior to eGNN and IBeM in this classification problem no matter if the Gaussian distribution is changed to any other distribution or if the dataset is unbalanced. The area above the FBeM ROC curve refers in part to 7.7% Acc of classification error and in part to the overlapping between granules with different assigned labels.

Figure 7.3 shows an example of eGNN decision boundary (the overall best of all experiments) for a 0.5 ROC cut-off threshold applied to outputs. At $h = 200$, eGNN has 5 granules embodying its structure, two associated to Class 1 and

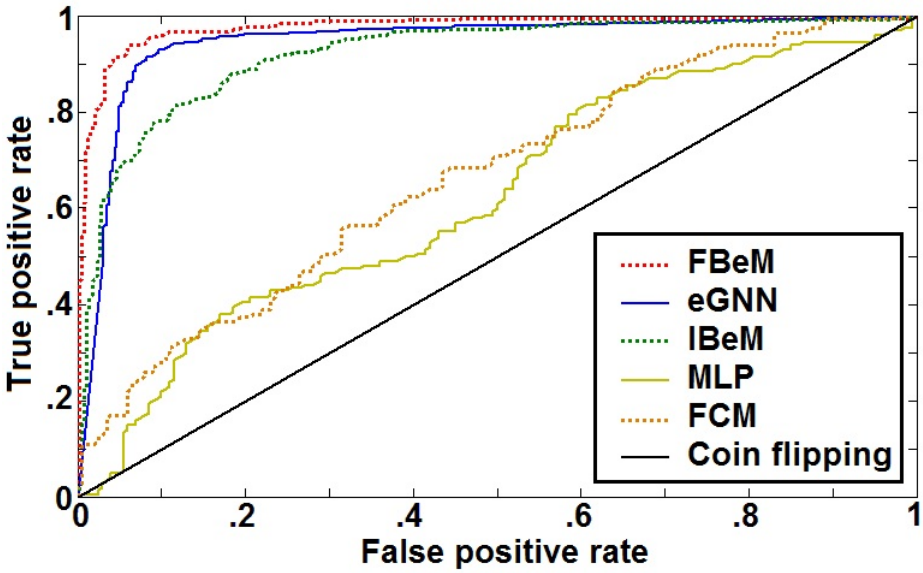


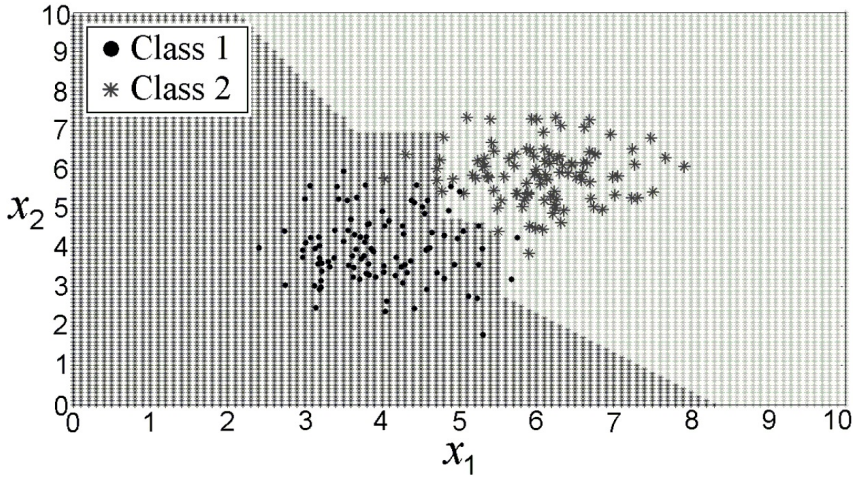
Figure 7.2: ROC curves of different methods for the rotating Gaussians

three to Class 2. It attained a 94.5% Acc classification rate. After rotation, that is, after $h = 400$, eGNN employs 5 granules in its structure, three for Class 1 and two for Class 2. It achieves a 97.5% Acc recognition performance.

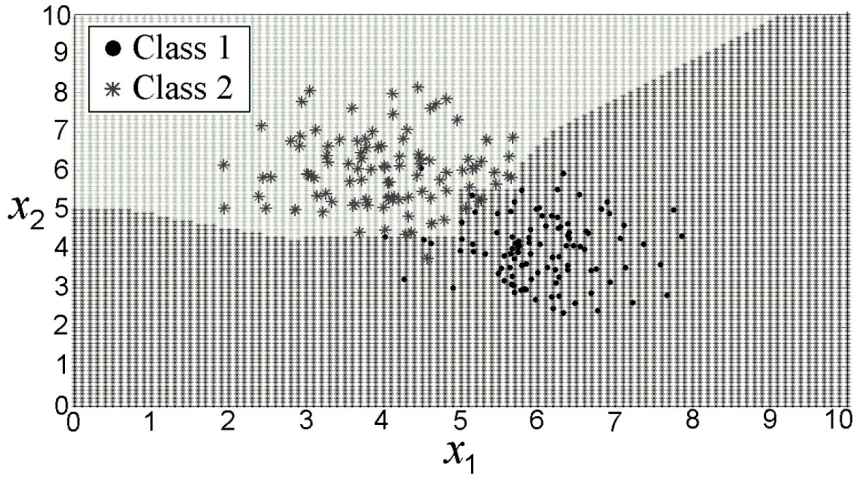
As an example, a highly active eGNN rule at $h = 400$ is:

R^4: IF $(x_1$ is $[3.1774, 4.0022, 4.605, 4.8683]$ AND
x_2 is $[4.9950, 5.1767, 5.4811, 6.9495])$
THEN $\hat{y}$ is Class 1

Linguistically, and assuming that the five partitions of the input variables are described by the adjectives: 'very low', 'low', 'medium', 'high' and 'very high', and, for example, a problem of classifying wines produced with grapes from different vineyards. Thus, rule R^4 can be read: if the 'concentration of flavanoids' (x_1) is 'medium' and 'color intensity' (x_2) is 'high', then the wine was produced by 'vineyard number 1' ($\hat{y}$).



(a) Snapshot at $h = 200$



(b) Snapshot at $h = 400$

Figure 7.3: eGNN decision boundary and last 200 data at particular time steps

7.2.2 New Class

A second experiment concerns an abrupt change: a new class in the data stream. We introduce a new Gaussian class centered at $(7, 3)$ with dispersion 0.8 at $h = 200$, as shown in Fig. 7.4. Evolving granular methods should learn the previously unknown class as soon as related information appears in the data stream.

Table 7.2 shows the results of evolving methods averaged over 5 independent runs. Non-evolving and even parametric adaptive methods are unable to discover

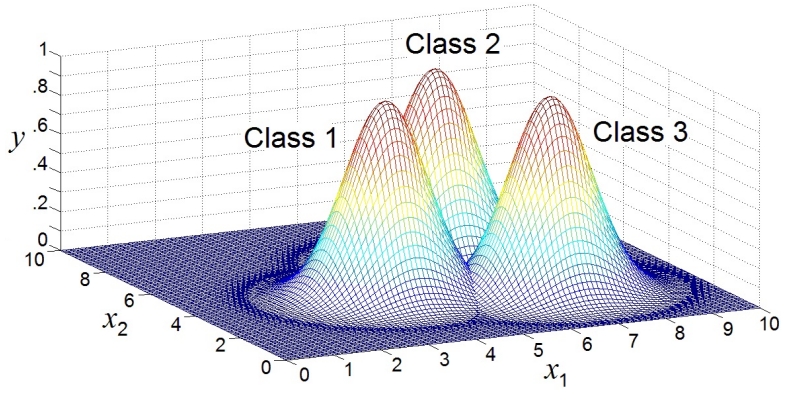


Figure 7.4: A third class appears at $h = 200$ and remains

new classes in data streams without redesigning and retraining the classifier from the beginning. Hence, they are inappropriate to this problem.

Table 7.2: New class problem: comparing evolving granular methods

Model	# Avg.Rules	Acc	CPU*
IBeM	3.1	83.8	0.7
FBeM	3.4	89.5	0.4
eGNN	4.5	88.1	0.3

* Average CPU time per sample in milliseconds

We note from Table 7.2 that FBeM is the most accurate method, giving a marginally better Acc index than eGNN. IBeM provided the most concise classifier; eGNN was the fastest method. Figure 7.5 illustrates the evolution of the Acc index, number of rules, and granularity taking into consideration FBeM. The results for the remaining methods are essentially the same.

We observe in Fig. 7.5 that the accuracy of the FBeM classifier is kept at a similar level after the shifting of the concept at $h = 200$. The Acc index reduced from 90.05% to 89.00%, which is quite acceptable. The robustness of the FBeM system to nonstationarities, as shown in the figure, is typical of evolving granular systems in view of their structural and parametric flexibility.

Figure 7.6 depicts the decision boundaries and the last 200 instances at $h = 200$ and $h = 400$ regarding eGNN. The neural network evolved a total of 6

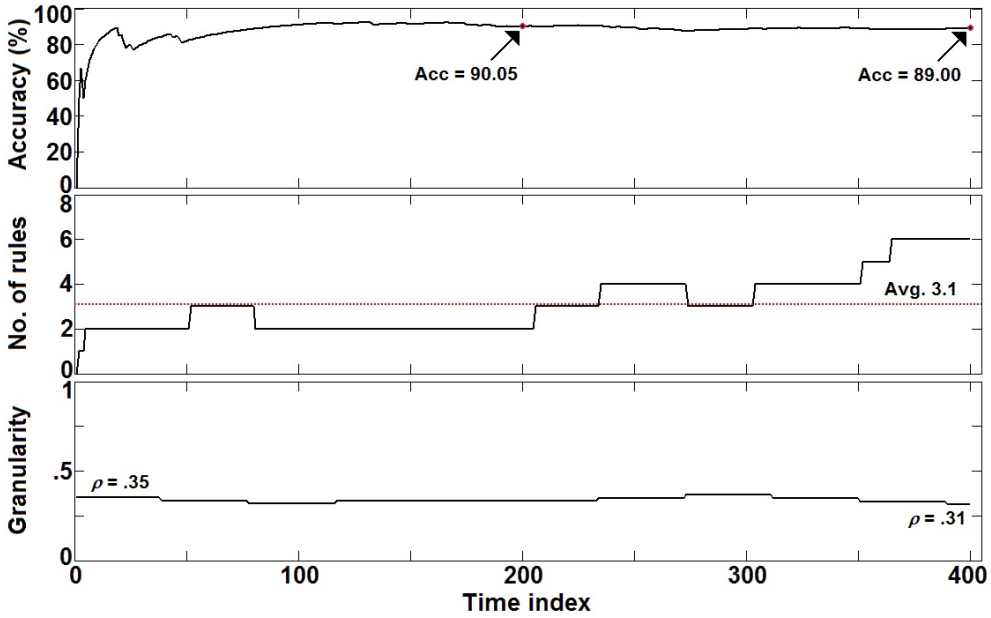


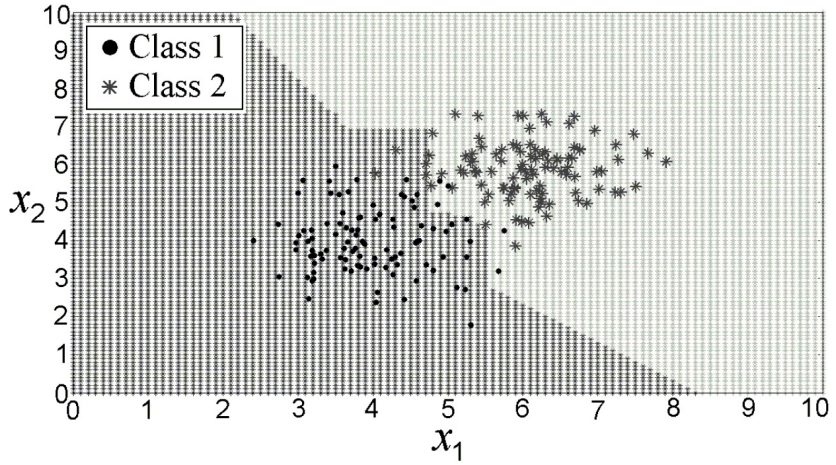
Figure 7.5: FBeM evolution of the Acc index, rule base and granularity for the new-class problem

granules during the first 200 steps, three associated with each of the two first classes. At this point, the eGNN Acc rate was 94.5%. Data about the third class started to arrive at $h = 200$ and at $h = 400$ eGNN developed 8 granules, three assigned to Class 1, two to Class 2, and three to Class 3. Assuming classes 2 and 3 as negative classes, eGNN reached a 92.5% Acc classification rate - the overall best accuracy of all experiments conducted.

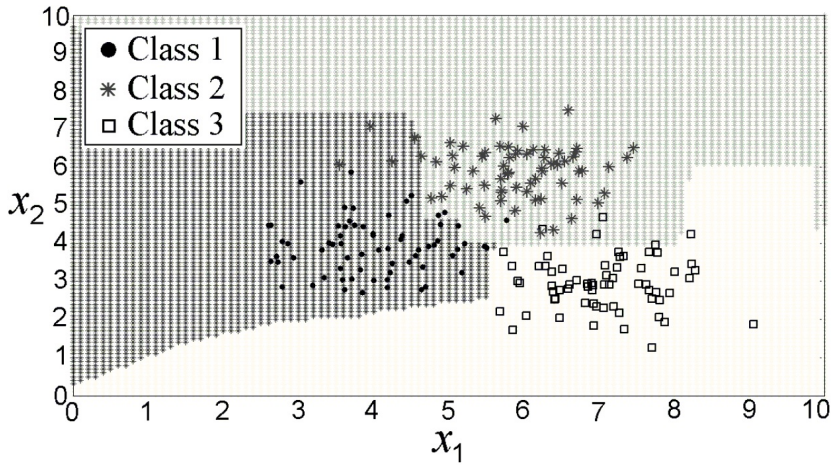
7.2.3 Combining Labeled and Unlabeled Data

We analyze the behavior of granular approaches in semi-supervised online classification. Partially supervised learning methods combine labeled and unlabeled data for training. Such mixtures are frequently found in practice (47) (102) (158).

Our approach for hybrid clustering and classification is: if an unlabeled sample causes the creation of a granule, then the class of the granule remains undefined until a new labeled sample falls within its bounds. The class label of the new sample tags the granule. Contrarily, if an unlabeled sample rests within the



(a) Snapshot at $h = 200$



(b) Snapshot at $h = 400$

Figure 7.6: eGNN decision boundaries for the 3-class problem

bounds of an existing granule whose label is known, it borrows the granule label.

We propose to change the proportion of unlabeled data from 0% to 100% in the twin (rotating and non-rotating) Gaussians problem, and in the new-arising-class problem such that all spectrum of possibilities of semi-supervised learning could be evaluated. Non-rotating and rotating Gaussians generate stationary and gradually-changing data streams, respectively; a new class represents a sudden abrupt shift.

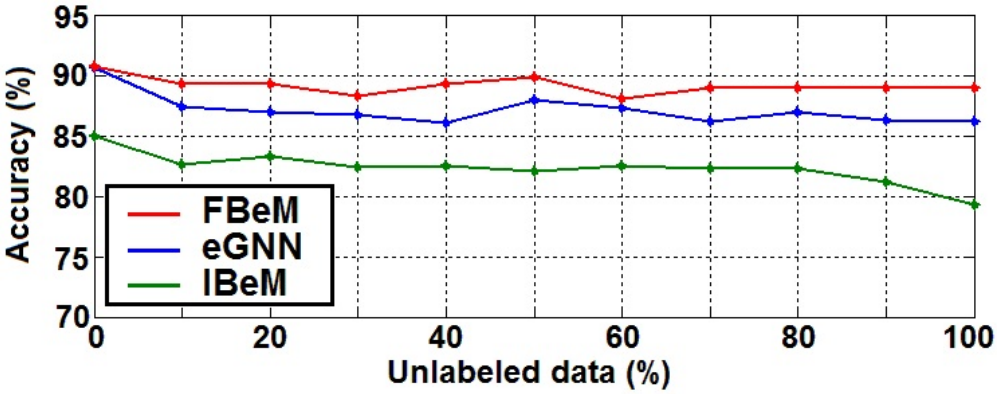
Granularities ρ were chosen differently in the range $[0.3, 0.45]$ to forbid the

number of granules to exceed 5 and emphasize the semantic aspect of the resulting construct. The remaining parameters are the same as considered in the previous sections. They are repeated here for convenience. IBeM and FBeM use $h_r = 1/\psi = 40$ and $\eta = 2$; eGNN employs $\mathbb{C}^i = T_{min}$, $\mathbb{C}^f = S_{max}$, $h_r = 40$, $\eta = 2$ and $\zeta = \vartheta = 0.5$. Figure 7.7 illustrates the performance of the evolving granular methods averaged over 5 runs for each condition.

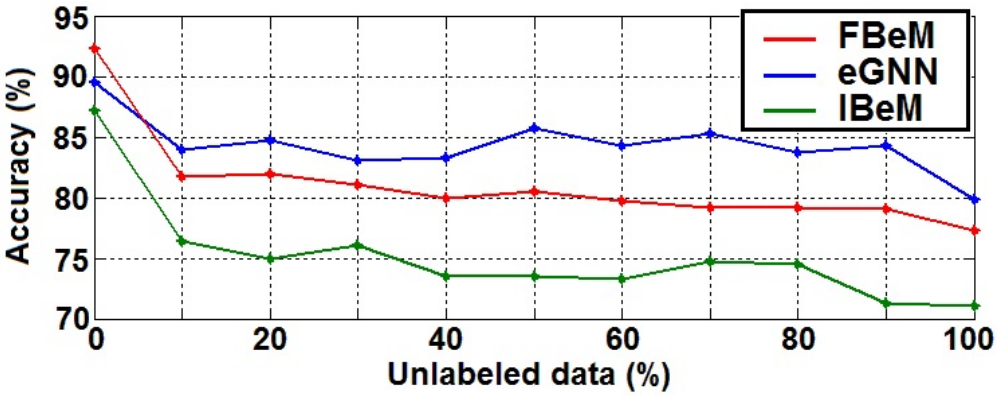
Figure 7.7 shows that evolving granular methods benefit from all information contained in the data stream, including that from unlabeled samples (input domain information) to perform classification. Conventional and evolving classifiers which operate on a supervised basis by simply discarding unlabeled data cannot deal with small fractions of labeled samples as in the situations in the right side of the graphs. Note that the left and right extremes of the plots indicate total supervision and no supervision, respectively. In both cases the final result is a partition of data into classes. By contrasting Fig. 7.7(a) with Figs. 7.7(b)-(c), all classifiers, IBeM, FBeM and eGNN, are not significantly affected by concept drift and shift. The generalization of sharp boundaries of IBeM to fuzzy boundaries of FBeM and eGNN is particularly decisive for the precision of the models in these classification applications. FBeM and eGNN interchanged as the most efficient approach. When the environment is stationary or processes are not overly complex, as in the non-rotating Gaussians problem, the high learning capability and structural plasticity of eGNN seem to be unnecessary. FBeM has shown to be superior in these cases. Conversely, as the complexity of the problem increases with nonstationarities and mixtures of labeled and unlabeled data, the eGNN modeling approach has demonstrated to be equivalent or better than FBeM.

7.3 Time Series Prediction

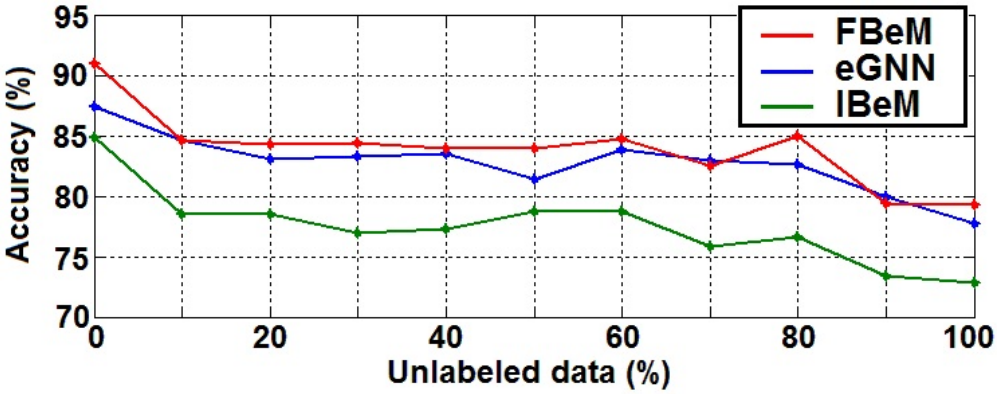
Observing past outcomes of a system to estimate its future behavior is the essence of forecasting and prediction (28) (51). When a complete mathematical model of a system can be developed and the corresponding initial conditions are known, prediction is an easy task. However, when no mathematical model is available or only partially known models are feasible, an alternative to forecasting is to build



(a) Non-rotating Gaussians



(b) Rotating Gaussians



(c) New class

Figure 7.7: Performance of evolving granular classifiers using different proportions of unlabeled data

models that consider current and past outcomes of the system, while neglecting any external inputs. This is a *look at what it does, not why* approach (39). Time series prediction is based on the idea that the series carry the potential information needed to predict their future behavior. Analyzing data produced by actual phenomena can give good insights into the phenomena themselves and knowledge about the laws underlying the data.

Forward prediction of a discrete time series can be defined as follows. Given a finite sequence $x^{[h]}$, $x^{[h-1]}$, ..., $x^{[h-M]}$, find the continuation $x^{[h+1]}$, $x^{[h+2]}$, ... This involves finding a scalar M and a function f, such that the value $x^{[h+1]}$ can be estimated by:

$$\hat{x}^{[h+1]} = f(x^{[h]}, x^{[h-1]}, ..., x^{[h-M]}). \tag{7.7}$$

This is equivalent to modeling the time series as

$$x^{[h+1]} = f(x^{[h]}, x^{[h-1]}, ..., x^{[h-M]}) + \psi^{[h+1]}, \tag{7.8}$$

with $\psi^{[h+1]}$ being a white noise process. If the statistics of the time series is non-Gaussian or the time series is the result of some nonlinear operation, the function f is nonlinear. f is the function we aim to model using evolving granular systems.

This section considers interval and fuzzy granular data streams derived from monthly mean, minimum, and maximum temperatures of weather time series of geographic regions with different climatic patterns. The aim is to predict monthly temperatures for all regions.

7.3.1 Weather Prediction

Weather predictions are useful for people to plan activities, protect property; and to assist decision making in many different sectors such as energy, transportation, aviation, agriculture, and inventory planning. Any system that is sensitive to the state of the atmosphere may benefit from weather forecasts.

Monthly temperature data carry a degree of uncertainty due to imprecision

of atmospheric measurements, instrument malfunction, equivocated transcripts, and different standards in acquiring and pre-processing the collected data. Usually temperature data are numerical, but the processes which originate and supply the data are imprecise. Temperature estimates in finer time granularities (days, weeks) are commonly demanded. Evolving granular approaches provide guaranteed granular predictions of the time series in these cases. The satisfaction of the granular prediction depends upon the prediction model compactness. Granular predictions together with single-valued predictions are important because they convey a value and a range of possible temperature values.

In the experiment we translate minimum, mean, and maximum monthly temperatures to triangular fuzzy numbers. Numerically-driven modeling approaches use mean monthly temperatures only; interval approaches consider minimum and maximum temperatures. The data were linearly scaled to the range $[0, 1]$. We use data from different weather stations as summarized in Table 7.3 (data available at http://eca.knmi.nl and http://cdiac.ornl.gov/epubs/ndp/ushcn/ushcn.html).

Table 7.3: Monthly temperature values

Station	# Samples	From	To	Std.Dev.
Bucharest	960	Jan 1930	Dec 2010	0.1795
Death Valley	1308	Jan 1901	Dec 2009	0.1835
Helsinki	1680	Jan 1871	Dec 2010	0.1842
Lisbon	1200	Jan 1910	Dec 2009	0.1556
Ottawa	1380	Jan 1895	Dec 2009	0.1790

As shown in Table 7.3, we consider five weather stations. In Death Valley (Furnace Creek), super-heated moving air masses are trapped into the valley by surrounding steep mountain ranges creating an extremely dry climate with high temperatures. Refer to (123) for a complete list of factors that produce high air temperatures in Death Valley. Conversely, Ottawa is one of the coldest capitals in the world. During the year, a wide range of temperatures can be observed, but the winters are very cold and snowy. Lisbon experiences more usual weather patterns. Summers are warm, sometimes hot, whereas winters are mild and moist. Helsinki and Bucharest are further weather stations considered for evaluation. Bucharest

has a continental climate owing to its distance from the open sea. Summers are generally hot while winters are quite cold. Helsinki combines characteristics of maritime and continental climates. The proximity of the Arctic Ocean and the North Atlantic creates cold weather, while the Gulf Stream conveys warm air.

During the computational experiments described subsequently, IBeM, FBeM and eGNN scan the data only once to build their structure and adapt parameters. This simulates online data stream processing. Testing and training are performed concomitantly on a per-sample basis. The performance of algorithms is evaluated using the root mean square error of singular predictions,

$$RMSE = \sqrt{\frac{1}{H}\sum_{h=1}^{H}(mp(y)^{[h]} - \hat{y}^{[h]})^2}, \tag{7.9}$$

the non-dimensional error index,

$$NDEI = \frac{RMSE}{std(mp(y)^{[h]}\forall h)}, \tag{7.10}$$

average number of rules in the model structure, and per-sample CPU time. The computer has a dual-core 2.54GHz processor with 4GB of RAM.

7.3.2 Performance Analysis

Different computational intelligence methods were chosen for performance assessment. They are: multilayer perceptron neural network (MLP) (55), evolving Takagi-Sugeno (eTS) (6), extended Takagi-Sugeno (xTS) (7), dynamic evolving neuro-fuzzy inference system (DENFIS) (64); and IBeM, FBeM and eGNN.

The task of the different methods is to provide one step ahead forecast of the monthly temperature $y^{[h+1]}$, using the last 12 observations, $x^{[h-11]}, ..., x^{[h]}$. The number of previous observations was chosen by trial and error to provide relatively accurate predictions. Online methods employ the sample-per-sample testing-before-training approach as follows. First, an estimation $\hat{y}^{[h+1]}$ is derived for a given input $(x^{[h-11]}, ..., x^{[h]})$. One time step after, the actual value $y^{[h+1]}$

becomes available and model adaptation is performed if necessary. In general, models should be robust to the trend and seasonal components of the time series, and not to the random noise component. Because the observed data contain random noise and irregular patterns, models that do not over fit them produce better generalizations and predictions of future values. Table 7.4 summarizes forecasting results for the Bucharest, Death Valley, Helsinki, Lisbon, and Ottawa monthly temperature data. IBeM starts with $\rho = 0.6$, $h_r = 84$, $\eta = 2$; FBeM uses $\rho = 0.7$, $h_r = 1/\psi = 48$, $\eta = 2$; and eGNN uses $\mathbb{C}^i = T_{min}$, $\mathbb{C}^f = M$, $\rho = 0.45$, $h_r = 84$, $\eta = 2$ and $\zeta = \vartheta = 0.5$.

Table 7.4 shows that eGNN gives the most precise forecasts in 3 of the 5 temperature data sets, seconded by eTS and FBeM with one each. The structures of the eGNN are, in average, the most parsimonious. Alternative evolving approaches such as DENFIS, eTS and xTS use numeric data - the mean temperature. In contrast, granular approaches such as eGNN, IBeM and FBeM take into account the mean and its neighbor data to bound forecasts. The trend component of the time series is taken into account in granular systems by procedures that gradually adapt granules and rules. The seasonal component is captured through different granules which represent different seasons and transitions between seasons. Since the content of a granule carries seasonal information, its corresponding rule tends to be activated in the specific months. IBeM and xTS are the fastest among the algorithms evaluated in this section.

We can also notice in Table 7.4 that the MLP neural network behaved well in all temperature time series. Our hypothesis is that the temperature time series obtained by the weather stations have not changed very much during the time period considered. In general, offline methods, such as the MLP, cannot deal with nonstationary functions, do not support one-pass training, and require higher CPU time and memory when compared with online methods. Moreover, the MLP neural network does not provide comprehensible models to support data description and interpretation.

As examples, the one-step singular and granular forecasts of FBeM for the Death Valley time series and of eGNN for the Helsinki time series are shown in Figs. 7.8 and 7.9. The additional plots in both figures show the granularity, error indices, and number of rules developed. Note that while the singular predic-

Table 7.4: Temperature forecasts

Station	Method	# Avg. Rules	$RMSE$	$NDEI$	CPU
Bucharest	DENFIS	5.00	0.0800	0.4457	4.7
	eGNN	3.80	**0.0594**	**0.3309**	1.6
	eTS	**3.00**	0.0598	0.3331	1.1
	FBeM	7.57	0.0603	0.3359	1.1
	IBeM	5.88	0.0643	0.3582	**1.0**
	MLP	–	0.0892	0.4969	35.5
	xTS	10.00	0.0643	0.3582	**1.0**
Death Valley	DENFIS	8.00	0.0600	0.3270	4.7
	eGNN	3.91	0.0498	0.2714	1.6
	eTS	**3.00**	**0.0491**	**0.2676**	**1.0**
	FBeM	8.00	0.0506	0.2757	1.1
	IBeM	8.79	0.0541	0.2948	**1.0**
	MLP	–	0.0584	0.3183	44.2
	xTS	10.00	0.0503	0.2741	1.1
Helsinki	DENFIS	24.00	0.0780	0.4235	5.7
	eGNN	**2.78**	0.0607	0.3295	1.6
	eTS	4.00	0.0634	0.3442	1.4
	FBeM	6.00	**0.0602**	**0.3268**	**1.1**
	IBeM	10.38	0.0764	0.4148	1.2
	MLP	–	0.0892	0.4843	35.5
	xTS	16.00	0.0651	0.3534	**1.1**
Lisbon	DENFIS	12.00	0.0880	0.5656	5.2
	eGNN	**2.77**	**0.0577**	**0.3708**	1.7
	eTS	4.00	0.0714	0.4589	2.3
	FBeM	5.63	0.0599	0.3850	1.2
	IBeM	3.59	0.0687	0.4415	**1.0**
	MLP	–	0.0955	0.6138	48.2
	xTS	11.00	0.0744	0.4781	**1.0**
Ottawa	DENFIS	7.00	0.0770	0.4302	4.9
	eGNN	3.88	**0.0575**	**0.3212**	1.5
	eTS	**3.00**	0.0604	0.3374	**1.0**
	FBeM	6.80	0.0609	0.3402	1.1
	IBeM	9.28	0.0734	0.4101	1.1
	MLP	–	0.0769	0.4296	41.3
	xTS	14.00	0.0631	0.3525	1.1

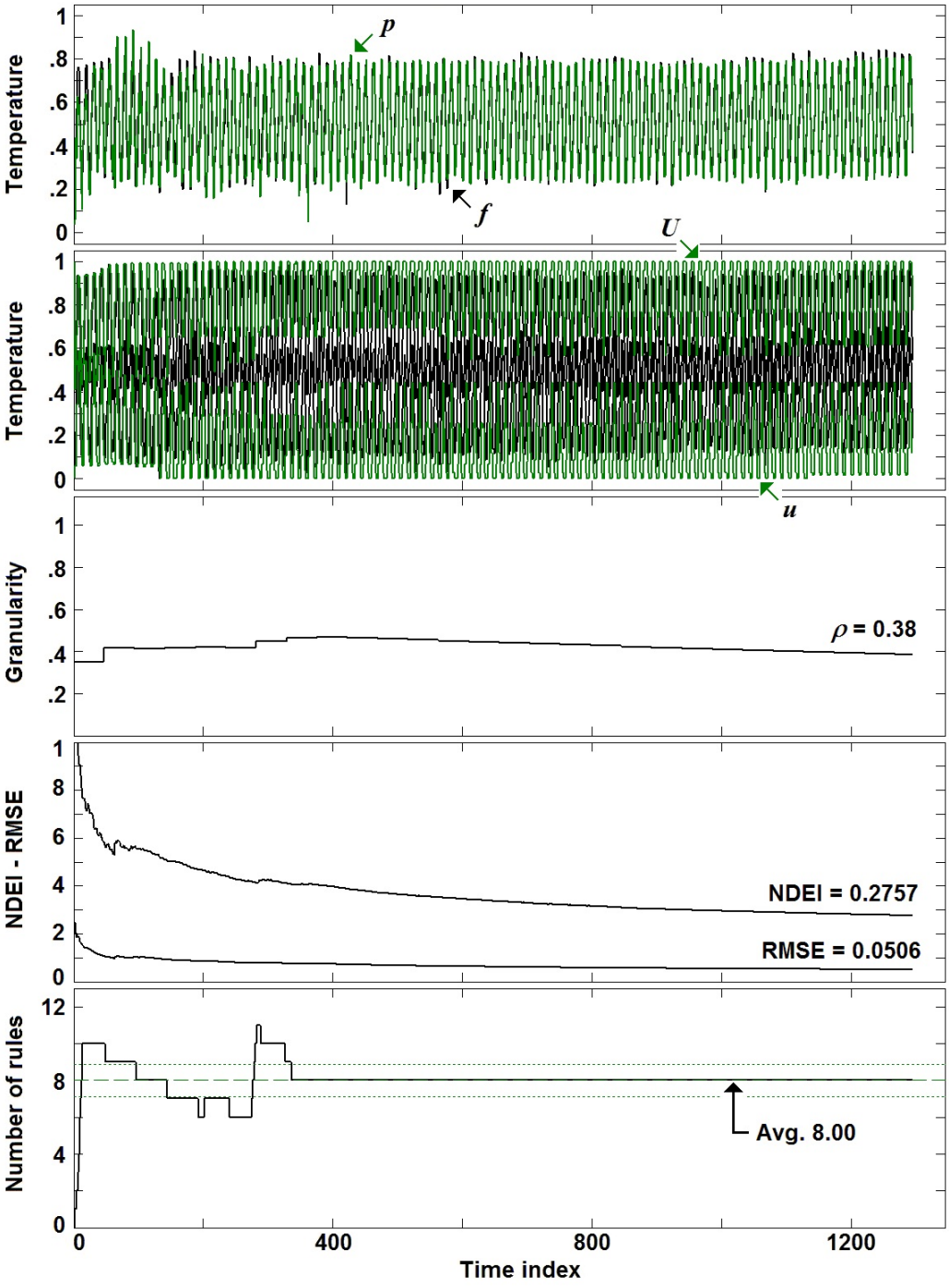


Figure 7.8: FBeM Death Valley temperature forecasts

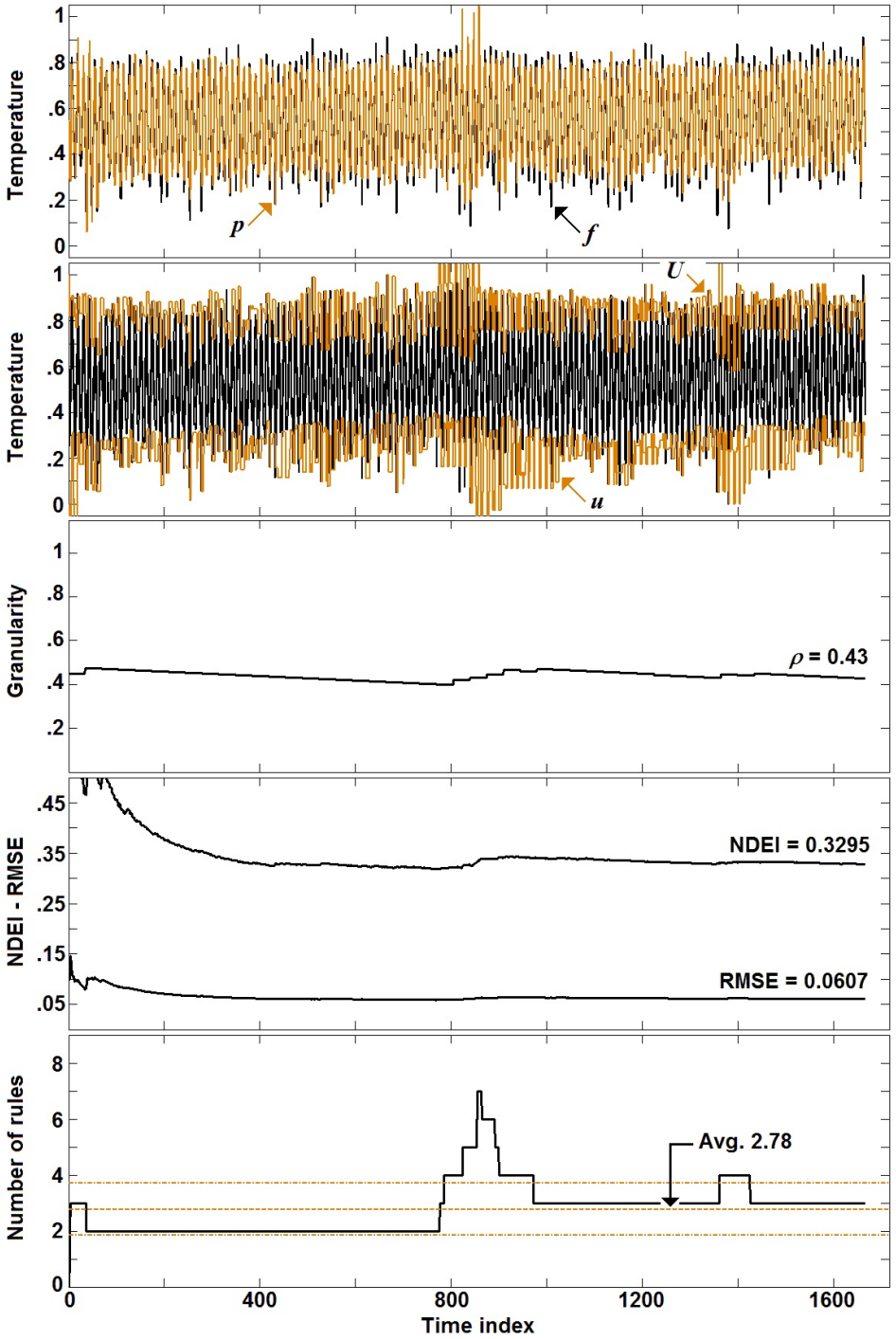
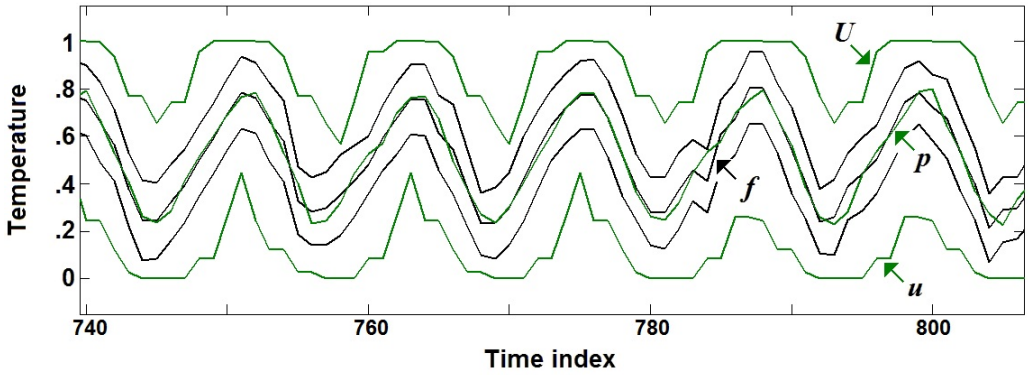


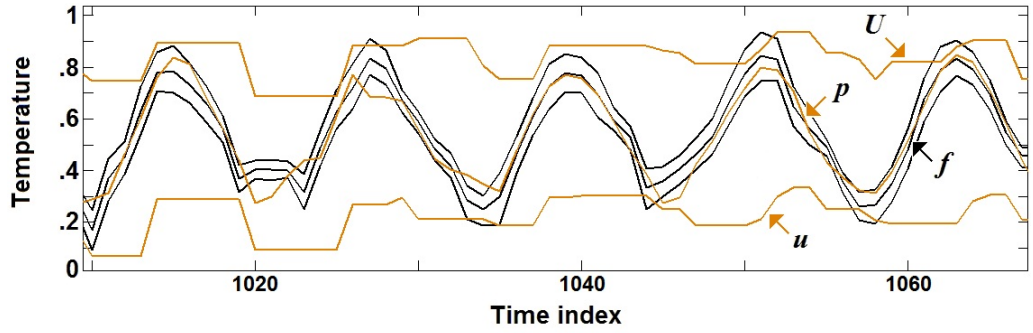
Figure 7.9: eGNN Helsinki temperature forecasts

tion p attempts to match the actual mean temperature value, the corresponding granular information $[u, U]$, formed by the lower and upper bounds of the consequent trapezoidal membership functions, intends to envelop previous data and the uncertainty of the unknown temperature function f.

Figure 7.10 enlarges the temperature predictions of the Figs. 7.8 and 7.9 for the time intervals [739, 807] and [1009, 1069], respectively. During these time intervals the respective granular models support 8 and 3 rules.



(a) Zoom of FBeM Death Valley forecast (Fig. 7.8) using 8 rules



(b) Zoom of eGNN Helsinki forecast (Fig. 7.9) using 3 rules

Figure 7.10: Comparing the narrowness of granular forecasts using rule bases of different sizes

We notice from Fig. 7.10 that relatively larger rule bases produce narrower ranges of values $[u, U]$ to bound predictions. Granular forecasts are determined based on past actual temperature values; they are particularly quite important since they usually come with a label and a linguistic description. FBeM and

eGNN are evolving approaches to handle fuzzy granular data streams, and to simultaneously provide singular and granular predictions.

Overall, the results in this section suggest that evolving granular systems benefit from data uncertainty and interval, fuzzy, and neurofuzzy granular frameworks to provide accurate and linguistic predictions of granular time series.

7.3.3 Time Complexity

This section examines how the performance of granular systems is affected by the number of input variables and rules. Here performance concerns temporal scalability and $RMSE$ to access processing time and prediction error, respectively.

For these purposes we first performed several independent experiments varying the number of input variables (lagged observations of temperature values). Initial parameters were chosen to give rule bases with about ten granular rules. This means that the size of the rule bases should not interfere in the temporal scalability analysis of evolving granular systems. We evaluate the processing time and prediction error when the number of input variables increases. Evaluation was performed in the context of temperature prediction. We assume FBeM and the Death Valley, Lisbon and Ottawa time series as the results for IBeM, eGNN and remaining time series are fundamentally the same. Figure 7.11 shows the processing time and $RSME$ for the time series chosen.

The bottom plot of Fig. 7.11 suggests that the time complexity of FBeM, and of evolving granular systems in general, is quasi-linear with the number of inputs. This is important since many computational intelligence and statistical algorithms behave polynomially or exponentially, which prohibit their use in handling massive data streams and modeling large-scale online processes. Evolving granular systems run linearly with respect to the number of samplings once their learning algorithms are one-pass and of incremental nature.

It is worth noting at the top of Fig. 7.11 that the weather time series require a small number of input variables, while the remainders tend to confuse the underlying predictor. The $RMSE$ indices for Death Valley, Ottawa, and Lisbon suggest local optima in the range between six to twelve input variables.

In the next experiment we fix the number of input variables to five, and

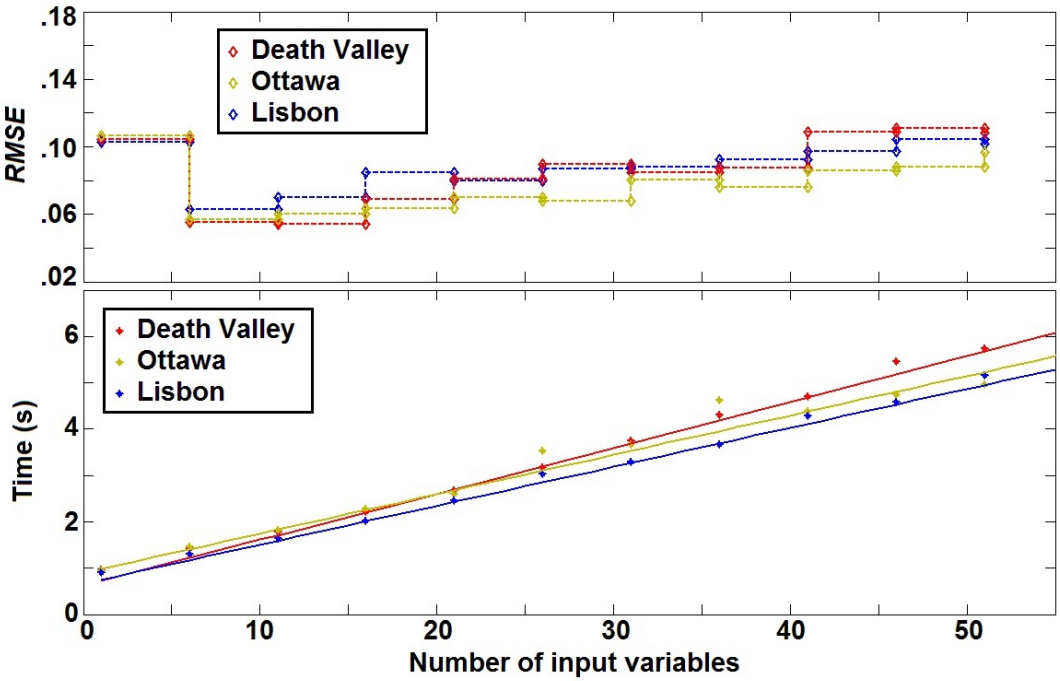


Figure 7.11: FBeM processing time and $RMSE$ using different amounts of input variables from temperature time series

run the FBeM algorithm with parameters that force it to generate an increasing number of rules. The goal here is to evaluate temporal scalability and $RMSE$ when the size of the rule base increases. Figure 7.12 shows the results obtained for the Death Valley, Ottawa, and Lisbon time series data.

The bottom plot of Fig. 7.12 shows that the processing time of FBeM grows exponentially with the number of rules. Although the algorithm deals linearly with the number of samples and input variables, granularity constraints within evolving granular framework is of utmost importance to keep the system operating online. Effective procedures to bound the rule base and protect evolving granular systems from outbursts of growth are: ($\boldsymbol{i}$) using the half-life value $1/\psi$ or the deletion threshold h_r. The total number of rules, c, is guaranteed to be less than or equal to $1/\psi$ anytime. For example, suppose $1/\psi = 6$ and that the rule base contains 7 rules. The last 6 samples can only activate 6 or less of the existing rules. Thus, at least one of the rules should be inactive for 7 time steps,

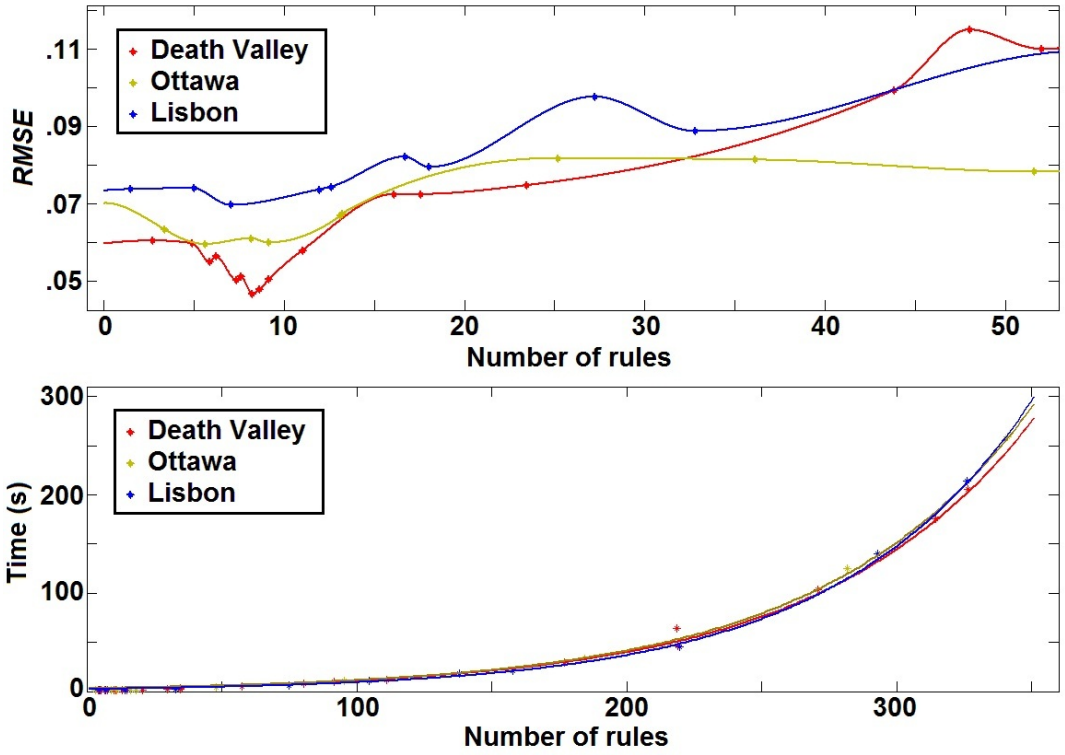


Figure 7.12: FBeM processing time and $RMSE$ for the Death Valley, Ottawa, and Lisbon time series considering different numbers of rules

which contradicts that $1/\psi = 6$; (***ii***) adapting the maximum width allowed for granules, ρ. This procedure develops only the necessary quantity of granules and rules. Notice that the points at the right of the plots of Fig. 7.12 can only be obtained by setting $1/\psi$ to a very large value, e.g. 10000, and turning the granularity adaptation procedure off.

The error curves at the top plot of Fig. 7.12 show that quite small and large rule bases decreases model accuracy. We employ piecewise cubic Hermite interpolation polynomials to fit the error data. Curiously, error values suggest more appropriate models with about 6 to 12 rules. This reinforces the hypothesis that seasonal trends are better modeled by single rules. Excessive granularity is detrimental because similar information is forcibly split into different granules and the underlying local models do not profit from the full information.

The average number of rules in FBeM depends on the choice of ρ and $1/\psi$. Reference (80) recommends $\rho^{[0]} = 0.5$ to balance structural stability and plasticity whenever we lack detailed knowledge of the modeling task and data properties. Monthly mean temperature prediction experiments suggest $\rho^{[0]}$ in the range from 0.5 to 0.8 to avoid rule overshoot after learning starts. This procedure helps to attain smoother structural development along the next time steps. Gradual adaptation of the granularity also alleviates initial guesses and guides the value of ρ according to the data stream. For monthly weather prediction, we suggest h_r values in the range between 48 and 84. The idea here is: if a trend does not appear again in the next four/eight years, then remove its corresponding rule.

7.3.4 Handling Abrupt Regime Changes

Long term climate changes cause average monthly temperatures to gradually drift over time yet abrupt shifts are hardly noticeable. The experiment addressed in this section show how evolving granular systems react when abrupt changes occur in nonstationary time series. We assume the FBeM method as the behavior of IBeM and eGNN are essentially equivalent.

For this purpose, we consider a hypothetical situation in which the time series of Death Valley, Ottawa, and Lisbon occur sequentially, forming a single time series. Two severe regime shifts are easily identified as the top plot of Fig. 7.13 illustrates. The bottom plot of Fig. 7.13 shows the fuzzy temperature predictions during the Ottawa-Lisbon shift (time interval between 2661 and 2740). In this experiment, FBeM should adapt the model to capture the new temperature profile and forget what is no longer relevant for the current environment. The initial parameters of FBeM were: $\rho = 0.6$, $h_r = 1/\psi = 48$ and $\eta = 2$. Figure 7.13 shows the $RMSE$, the number of rules, and the granular and singular predictions. Notice that the number of rules of the rule base peaks after the Death Valley-Ottawa and Ottawa-Lisbon transitions, but returns to the usual values afterwards. Similarly, the $RMSE$ increases slightly and decreases in the next steps after time series transitions. Online adaptability improves prediction accuracy after the transitions. Evolving granular systems are stable to abrupt changes in granular data streams, a challenge to a variety of machine learning algorithms.

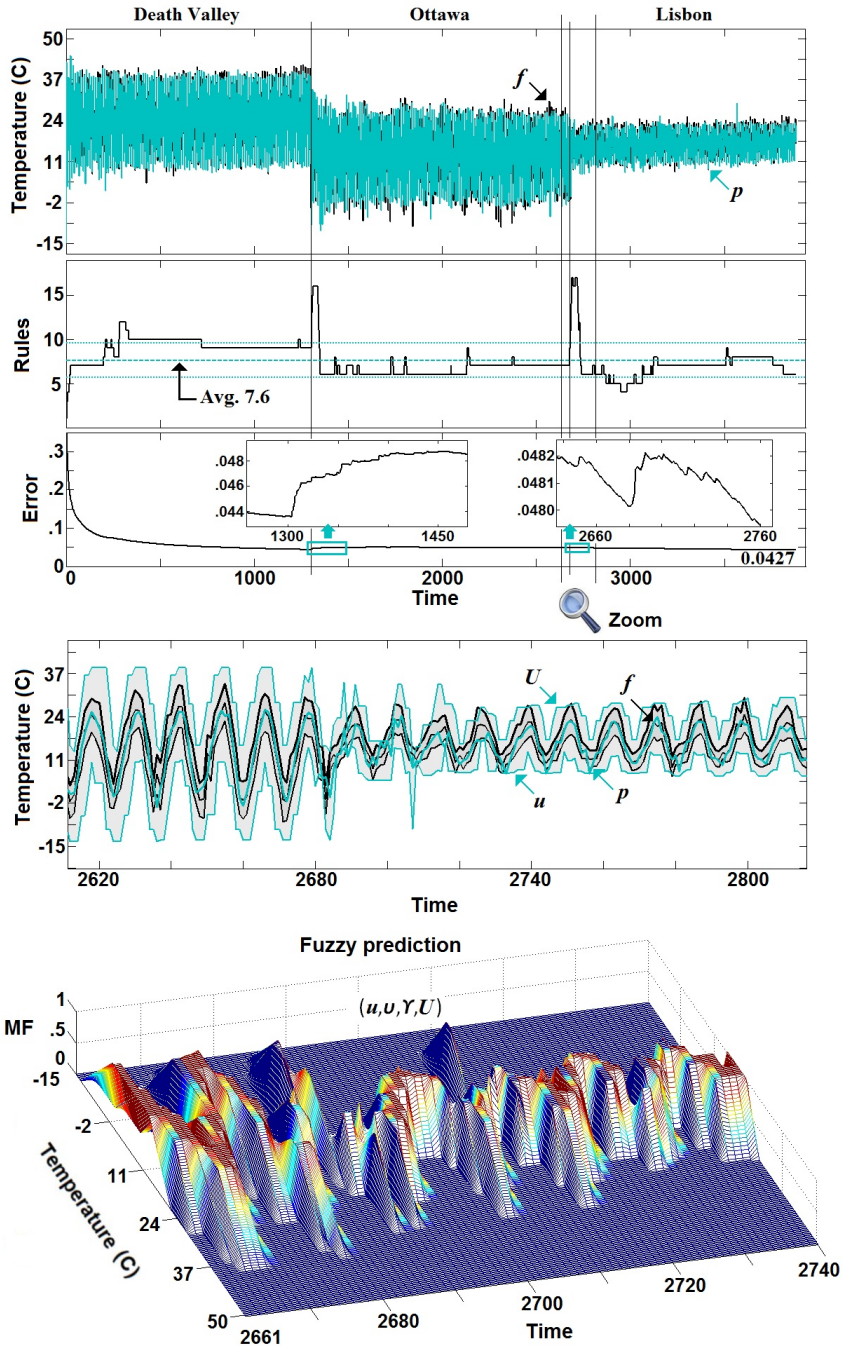


Figure 7.13: FBeM prediction of the Death Valley, Ottawa, and Lisbon temperature time series combined

7.4 Function Approximation

Function approximation consists in finding a function that matches in some extent a target function in a task-specific way. Here, the target functions are unknown, but perceived as streams of intervals or fuzzy intervals - with intervals and fuzzy intervals representing our intuitive notion of approximate data. Differently from time series prediction, in function approximation problems, external variables are available and the time span in which the data are obtained is unimportant.

The generic form of the function approximation problem is as follows: given a time-varying unknown function $f^{[h]}$, where $h = 1, ...$ is the time index; and a pair of observations $(x, y)^{[h]}$, $x \in X$ and $y \in Y$, find a finite collection of information granules $\gamma = \{\gamma^1, ..., \gamma^c\}$ and a time-varying real-valued map $p^{[h]} : X \to Y$ such that $\gamma^i \subseteq X \times Y$ and $p^{[h]}$ minimizes $(f^{[h]} - p^{[h]})^2$. The output $y^{[h]}$ is unknown when the input $x^{[h]}$ is available, but is known afterwards. Attributes x_j of an input vector $x = (x_1, ..., x_n)$ and the output y are trapezoidal fuzzy data.

Because every continuous function f can be approximated uniformly on a finite interval by continuous piecewise linear functions p (36), evolving granular systems are universal approximators (proof in Appendix A).

The following sections consider recent benchmark data sets in material and biomedical engineering to evaluate and illustrate the usefulness of the proposed evolving granular approaches in the function approximation task.

7.4.1 Concrete Compressive Strength

Compressive strength is the capacity of a material to withstand axially-directed pushing forces. When the limit of compressive strength is reached, materials are crushed. When building with concrete, it is important to know whether it can bear the compressive forces for safety sake (68).

Compressive tests measure how well concrete holds up to the compressive pressures around it. Test standards neglect uncertainties from different natures. For example, (***i***) the concrete cross sectional area changes in function of the compressive load applied. The material tends to spread laterally and hence increase the cross sectional area; (***ii***) compression tests clamp materials at the edges. Therefore, a variable frictional force (barreling phenomenon) arises which will oppose

the lateral spread. This results in a slightly inaccurate value of stress which is obtained from the experiment (68); (***iii***) standards for concrete structures do not consider weather conditions and permit specimen density to vary about 2%.

The Concrete Compressive Strength data set, available at the UCI Machine Learning Repository, consists of 1030 singular samples. We assume that the data are perceptions of the values of a variable. Thus, we consider 2% of imprecision in each input x_j and output y and represent it by symmetrical triangular fuzzy objects of the form $(.98x_j, x_j, x_j, 1.02x_j)$ and $(.98y, y, y, 1.02y)$, respectively. Interval methods consider intervals $(.98x_j, .98x_j, 1.02x_j, 1.02x_j)$ and $(.98y, .98y, 1.02y, 1.02y)$, naturally. Concrete ingredients and age of the mixture are the independent variables of the compression function. Ingredients include cement, blast furnace slag, fly ash, water, superplasticizer, and coarse and fine aggregate (147).

First, we perform a preliminary experiment to analyze different fuzzy aggregation neurons in eGNN. We consider 0.5 as default value of uninorm neutral elements e and T-S norm v-factors. Therefore, in this experiment fuzzy neurons use the T and S norms occurring in these constructs equally. Other fuzzy aggregation neurons are used with no restrictions.

Performance evaluation is based on the $RMSE$ and $NDEI$ indices computed similarly as in (7.9) and (7.10), respectively. We consider also the average number of rules, and CPU time using a dual-core 2.54GHz processor with 4GB of RAM. The original samples were shuffled and linearly scaled to the range $[0, 1]$. eGNN adopts $\rho = 0.45$, $h_r = 50$, $\eta = 2$ and $\zeta = \vartheta = 0.5$. Table 7.5 shows the best performance of each network setting (using different types of neurons in the aggregation and output layers) in 10 independent runs.

We notice in Table 7.5 that the accuracy of eGNN using product T-norm (T_{prod}) neurons in the aggregation layer and combined T-S neuron ($L_{min,max}$) in the output layer provides the most accurate results employing a relatively small number of rules. Although the averaging (M) output neuron configuration using T_{prod} aggregation produces very close results, the former construct which adopts T_{prod} and $L_{min,max}$ neurons is maintained for the next experiments.

Because outlier points can disturb the computation of the mean and standard deviation of a dataset, we consider replacing them with the mean of the values

Table 7.5: Concrete compressive strength prediction: evaluation of different types of eGNN neurons

Aggregation	Output	# Avg.Rules	$RMSE$	$NDEI$	CPU*
T_{min}	M	4.01	0.1268	0.6354	1.6
T_{prod}	M	**3.72**	0.1210	0.6064	1.8
T_L	M	5.26	0.1438	0.7209	1.8
$U_{min,max}$	M	4.53	0.1402	0.7029	**1.5**
$U_{prod,prob}$	M	3.86	0.1295	0.6488	1.7
T_{min}	$L_{min,max}$	4.01	0.1275	0.6389	1.8
T_{prod}	$L_{min,max}$	**3.72**	**0.1205**	**0.6040**	1.8
T_L	$L_{min,max}$	5.26	0.1417	0.7100	1.9
$U_{prod,prob}$	$L_{min,max}$	3.86	0.1276	0.6393	1.7

* Average CPU time per sample in milliseconds

available so far. This procedure avoids biased estimates caused by low levels of activation of granules. Put simply, if the concentration of concrete ingredients or the compressive strength exceeds the accumulated mean value plus 4 standard deviations, then it is replaced by the mean. Note that this procedure discards the uncommon value and handles the sample with a missing datum. Imputation methods for missing data (127) request certain constraints to be attended. For instance, if the number of outliers is large compared to the total number of samples then we run the risk to distort the covariance structure of the data and to bias covariances toward zero. The feasibility of the proposed method using 4 standard deviations was confirmed to the underlying data since that the quantity of outliers is smaller than 2%. A more elaborate mechanism to deal with outliers is, e.g., to use an arousal mechanism such as the one suggested in (128).

We compare evolving granular systems against alternative evolving methods. The approaches evaluated are: evolving Takagi-Sugeno (eTS) (6), extended Takagi-Sugeno (xTS) (7), dynamic evolving neural-fuzzy inference system (DENFIS) (64), evolving fuzzy linear regression tree (eFT) (83), evolving participatory learning (87), IBeM, FBeM, and eGNN. In this comparative experiment, we consider the dataset as originally provided instead of shuffling the data samples. Based upon a possible temporal correlation of the data, one lagged value of the compressive strength was considered as input to all of the methods. Therefore,

the number of input variables in this experiment totalizes 9. IBeM employs $\rho = 0.45$, $h_r = 50$ and $\eta = 2$; FBeM uses the same parameters as IBeM except for $h_r = 1/\psi = 40$; eGNN also adopts the same parameters as IBeM plus $\zeta = \vartheta = 0.5$. Table 7.6 shows the performance of each method.

Table 7.6: Concrete compressive strength prediction: evaluating different evolving methods

Model	# Avg.Rules	$RMSE$	$NDEI$	CPU
DENFIS	5.00	**0.1130**	**0.5670**	19.9
eFT	7.00	0.1380	0.6518	79.7
eGNN	3.72	0.1205	0.6040	1.8
ePL	6.00	0.1847	0.9259	24.4
eTS	7.00	0.1554	0.7343	**0.9**
FBeM	**3.48**	0.1398	0.7009	1.7
IBeM	4.43	0.1334	0.6683	1.2
xTS	8.00	0.1552	0.7333	**0.9**

We observe from Table 7.6 that DENFIS is the most accurate method to approximate the compressive strength function although it uses more rules and spends more time to process the data collection as opposed to evolving granular systems. From the accuracy/compactness point of view all, IBeM, FBeM and eGNN, have shown to be competitive. Particularly, eGNN reached a slightly higher error rate than DENFIS, 0.1205 against 0.1130, using an average of only 3.72 rules and a maximum of 10 rules. eTS and xTS were the fastest methods in this function approximation application. Non-granular methods give no prediction boundaries during processing steps.

Figure 7.14 shows an example of T_{prod} with $L_{min,max}$ eGNN approximation of the concrete compressive strength function and the evolution of the number of rules, error indices, and granularity. The bottom plot expands the approximation of the top plots in the range $[885, 987]$.

In Fig. 7.14, the single-valued approximation p together with the granular approximation $[u, U]$ gives a value of compressive strength and a range of values in the neighborhood of p induced by the input data. Moreover, the granular approximation may come with a label and a proper linguistic description; it

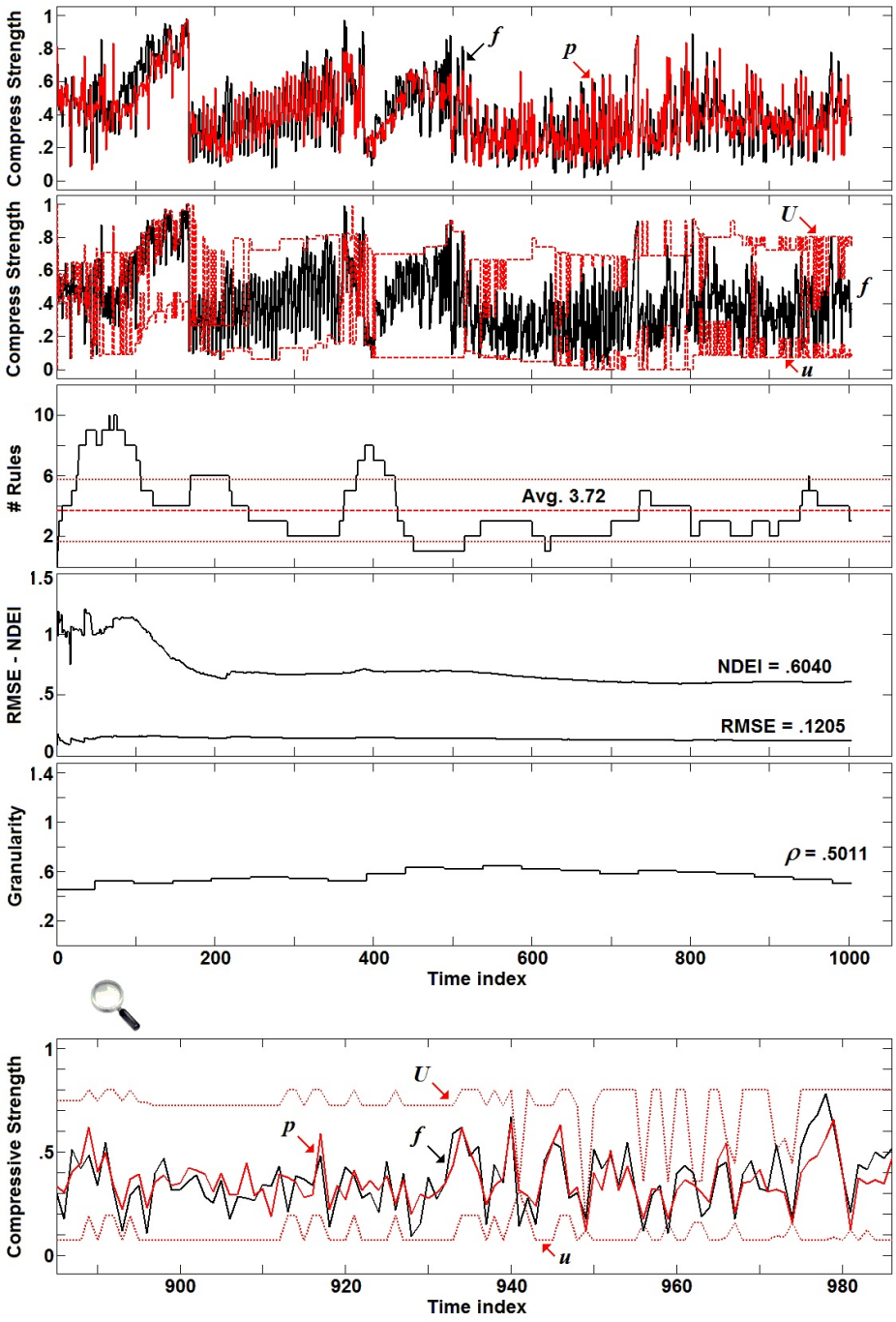


Figure 7.14: eGNN approximation of the concrete compressive strength function, and evolution of the rule base, error indices, and granularity

enhances model acceptability and the neighborhood can be made tighter if we accept a larger number of rules. The performance of eGNN profits from the combination of structural evolution and fuzzy granules. Results may recommend changing ingredients mix ratio and/or adding special hardeners to the concrete compound.

7.4.2 Parkinson's Telemonitoring

Parkinson's disease is one of the most common neurodegenerative disorders (133). Early diagnosis is a key to improve patients' quality of life and to prolong it. Frequent symptoms of Parkinson's disease include movement disorders and vocal impairment (dysphonia). Particularly, vocal degradation is one of the earliest indicators of the disease which patients consider a major barrier (57) (133).

The Parkinson's telemonitoring data set, accessible at the UCI Machine Learning Repository, consists of 5875 biomedical voice measurements from 42 patients with early-stage Parkinson's disease recruited to a six-month trial of a telemonitoring device for remote symptom progression accompaniment. Recordings were captured in typical home acoustic environments and transmitted over the Internet to a clinic. Inputs comprise 5 Jitter and 6 Shimmer measures related respectively to the frequency and amplitude of the speech signal, 2 measures of ratio of noise-to-tonal components, 2 measures associated to entropy, and a measure of detrended fluctuation, a total of 16 inputs. Uncertainty may arise from the processes of audio capturing and web transmission, steadiness of phonation and loudness, and environmental conditions, to name a few. The output considered in this study is the total score of the factor 'unified Parkinson's disease rating scale' (UPDRS), which reflects the presence and severity of symptoms. The larger the UPDRS, the more severe the patient disabilities.

We analyze different aggregation neurons in eGNN similar to the concrete compressive strength function approximation problem. New outlier data are replaced with the mean of the values available so far whenever the data surpass the range of plus or minus 3 standard deviations around the mean. We consider not shuffling data samples and taking advantage of 2 lagged UPDRS scores to benefit from temporal information. Table 7.7 summarizes the results.

Table 7.7: Parkinson's telemonitoring prediction: evaluation of different types of eGNN neurons

Aggregation	Output	# Rules	$RMSE$	$NDEI$	CPU*
T_{min}	M	9.10	**0.0667**	**0.3219**	**1.8**
T_{prod}	M	**8.75**	0.0679	0.3280	1.9
$U_{min,max}$	M	10.56	0.0755	0.3643	2.0
$U_{prod,prob}$	M	9.52	0.0754	0.3641	2.2
T_{min}	$L_{min,max}$	9.10	0.0668	0.3226	**1.8**
T_{prod}	$L_{min,max}$	**8.75**	0.0683	0.3296	2.1
$U_{min,max}$	$L_{min,max}$	10.56	0.0735	0.3550	2.1
$U_{prod,prob}$	$L_{min,max}$	9.52	0.0752	0.3630	2.3

* Average CPU time per sample in milliseconds

Table 7.7 shows that using different types of fuzzy neurons in the aggregation layer of eGNN affects prediction accuracy more than using different neurons in the output layer. The eGNN construct that combines minimum T-norm T_{min} and averaging M neurons in the aggregation and output layers, respectively, performs better than the remainder constructs according to the error indices. Therefore, this construct is maintained in the next experiments.

Studies in (90) (133) point that some dysphonia measures are highly correlated. Overall, highly correlated input variables tend to misguide the underlying granular evolving method. Variable selection has been considered to improve prediction accuracy, speed up the learning process, and provide easier-to-interpret models.

Input variables of IBeM, FBeM, and eGNN may be m variables of the speech function or we may select the n less correlated variables. We conducted offline ranking and progressive elimination of correlated variables. Leaving one out of two highly correlated variables allows assessing how well the results generalize to relatively independent data sets. Similar to (14) (133), the sequence of removed variables was chosen based on their maximum redundancy as calculated on partial autocorrelation analysis. The variables were: Shimmer:DDA, Jitter:DDP, Shimmer(dB), Shimmer:APQ5, Shimmer:APQ3, Jitter:RAP, Jitter:PPQ5, Jitter(Abs), Shimmer:APQ11, and HNR, in this order. Refer to the UCI Machine Learning Repository for a detailed description of the meaning and significance

of the variables. We ignore one variable at a time until a statistically significant degradation of the systems is noticed. Figure 7.15 shows the average results considering IBeM, FBeM, eGNN, and independent runs for each set of variables.

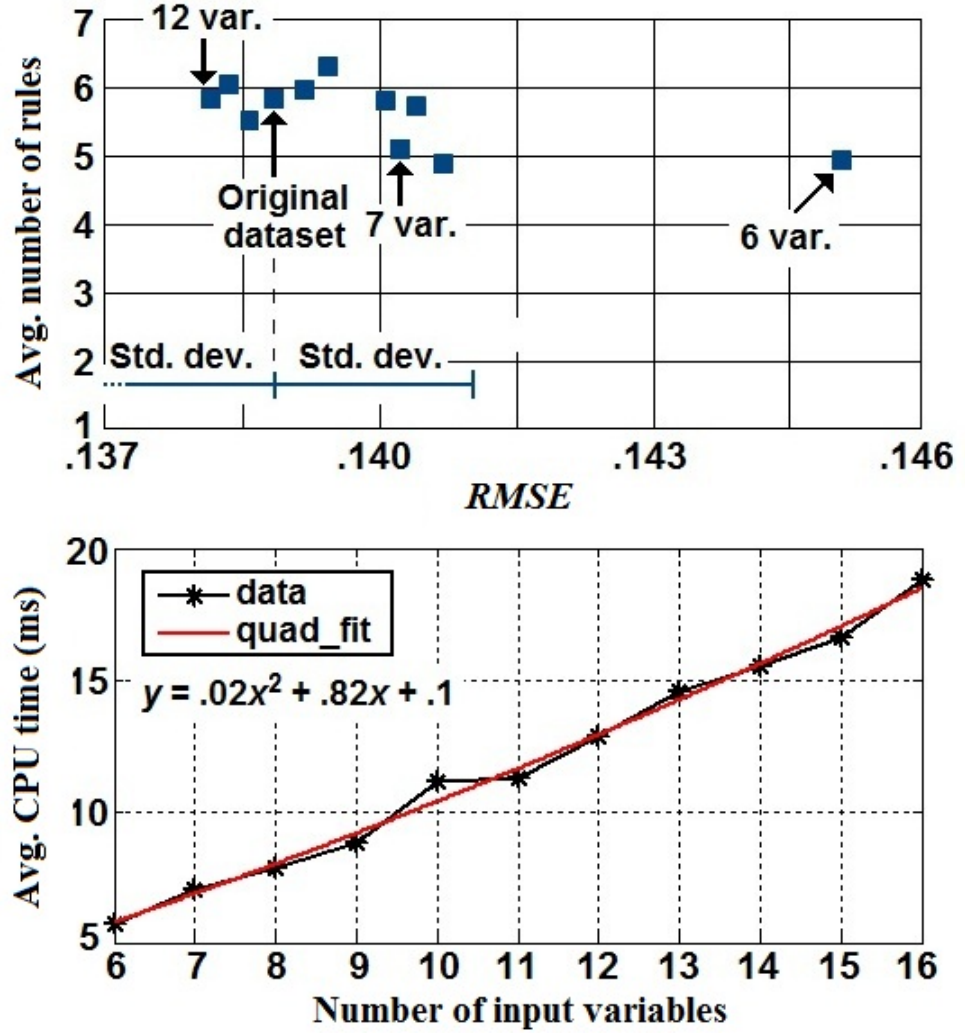


Figure 7.15: Evolving granular systems results on leave-one-variable-out approach to find less correlated subsets of input variables

The top plot of Fig. 7.15 shows that a substantial degradation of the $RMSE$ performance is verified only when the leave-one-out approach results in a 6-variable model (not counting the two lagged variables related to the previous UPDRS values). According to the principle of parsimony, which states that,

other things being equal, the simplest solution is the best, the models with the 7 less correlated variables are sufficient.

The bottom plot of Fig. 7.15 shows the average per-sample CPU time spent by evolving granular systems for different numbers of input variables. The data were fitted with a quadratic function whose small second-order coefficient suggests that time complexity is quasi-linear with the number of input variables. This is a major characteristic in view that many computational intelligence and statistical algorithms behave polynomially or exponentially and are unable to process massive data streams in large-scale online modeling. Moreover, evolving granular systems run in linear time with respect to the number of samplings because their learning algorithms are one-pass and incremental.

Comparison between granular and alternative methods is given in Table 7.8. The methods analyzed were: multi-layer perceptron (MLP) (55), least squares (LS) (54), iterative reweighed least squares (IRLS) (133), least absolute shrinkage and selection operator (LASSO) (132), classification and regression trees (CART) (54), extended Takagi-Sugeno (xTS) (7), and evolving Takagi-Sugeno (eTS) (6). In this experiment samples were shuffled so that their order does not matter to those methods which are online. Lagged values of the UPDRS score become superfluous since temporal information is lost when we mix the data sequence. The purpose of the experiment is to compare state-of-the-art function approximation methods only. IBeM uses the following parameters: $\rho = 0.5$, $h_r = 120$, $\eta = 2$; FBeM employs $\rho = 0.45$, $h_r = 120$, $\eta = 1/\psi = 2$; and eGNN employs $\rho = 0.4$, $h_r = 120$, $\eta = 2$ and $\zeta = \vartheta = 0.5$. IBeM was run processing input information translated to interval data. Conversely, FBeM and eGNN process symmetrical trapezoidal fuzzy data. More precisely, the original input and output data, x_j and y, were assumed to be $(.98x_j, .98x_j, 1.02x_j, 1.02x_j)$ and $(.98y, .98y, 1.02y, 1.02y)$ in the case of IBeM; and $(.98x_j, .99x_j, 1.01x_j, 1.02x_j)$ and $(.98y, .99y, 1.01y, 1.02y)$ in the case of FBeM and eGNN. Table 7.8 shows the results.

Table 7.8 shows that both granular evolving systems achieve satisfactory results taking into account the relation accuracy/compactness. Interestingly, FBeM outperforms the remaining methods not requiring a large number of rules. Based on its structure, learning algorithm and fuzzy granular framework, FBeM reaches a 0.1245 $RMSE$ rate using an average of 4.91 ± 1.68 rules, with a maximum of

Table 7.8: Parkinson's telemonitoring prediction: evaluating different methods

Model	# Rules	$RMSE$	$NDEI$
LASSO	–	0.3842	1.8402
LS	–	0.3820	1.8294
IRLS	–	0.3797	1.8186
CART	–	0.3588	1.7185
MLP	–	0.3559	1.7046
eTS*	7	0.1452	0.6954
xTS*	7	0.1443	0.6911
eGNN*	6	0.1394	0.6673
IBeM*	5	0.1358	0.6504
FBeM*	5	0.1245	0.5963

* Online methods

9 rules. Evolving granular methods have shown clear advantages over traditional statistical LASSO, LS, and IRLS methods because of its nonlinear nature.

Figure 7.16 depicts an example of FBeM approximation of the Parkinson's telemonitoring function and the evolution of the number of rules, error indices and granularity. The results in the figure stand for the best approximation attained in all experiments performed. It considers an FBeM model with 8 inputs, being 7 less correlated features (obtained as previously described using the leave-one-variable-out approach) and 1 lagged UPDRS value. Moreover, we discard samples that convey values out of the range of plus or minus 3 standard deviations around the current mean value of a variable.

The top plot of Fig. 7.16 shows that the FBeM single-valued approximation p provides quite accurate estimations when temporal information is intrinsic to the data stream. In these cases, the FBeM learning algorithm explores space information from the original features and, simultaneously, time information from past outputs. Note that the average number of rules during processing steps is only 4.16, which means that FBeM is not over fitting data to achieve the underlying error rate ($RMSE = 0.0509$), but generalizing the behavior of the actual function. The granular prediction $[u, U]$ (support of trapezoids) provides lower and upper bounds in which single-valued predictions must be within. Therefore, given a limited range of possible values, the chances to obtain more accurate single-valued predictions tend to be higher. The bottom plot of the figure ex-

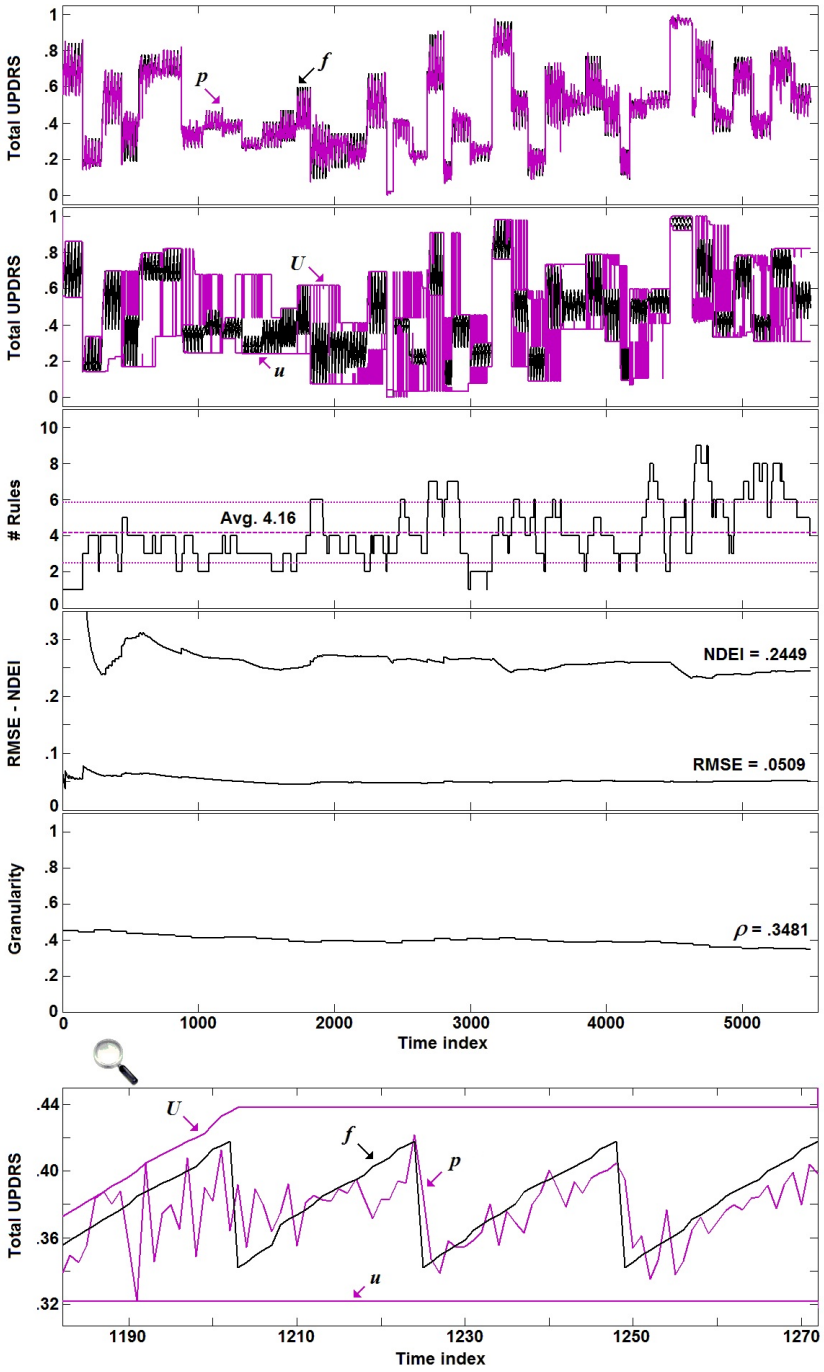


Figure 7.16: FBeM approximation of the Parkinson's telemonitoring function, and evolution of the rule base, error indices, and granularity

pands the top plot in the interval $[1181, 1273]$. If we accept larger rule bases by choosing higher values for the rate η and lower values of ρ, then FBeM takes advantage of a larger number of rules and may improve its accuracy because the granular approximation tends to become tighter around p. However, FBeM interpretability may reduce in this case as the number of linguistic terms and granules increases. The same phenomenon is observed in other granular systems.

Rules of particular interest can be displayed anytime. An example of a highly active FBeM rule at $h = 5460$ is:

$$
\begin{aligned}
R^i\text{: IF } & (x_1 \text{ is } [0.0381, 0.1081, 0.1154, 0.4283], \\
& x_2 \text{ is } [0.0756, 0.1469, 0.1619, 0.4582], \\
& x_3 \text{ is } [0.0169, 0.0556, 0.0667, 0.3899], \\
& x_4 \text{ is } [0.3259, 0.5531, 0.5753, 0.7571], \\
& x_5 \text{ is } [0.2813, 0.5241, 0.6081, 0.7232], \\
& x_6 \text{ is } [0.0692, 0.1783, 0.1890, 0.4656], \\
& x_7 \text{ is } [0.1272, 0.2724, 0.2887, 0.5295], \\
& y^{[-1]} \text{ is } [0.4283, 0.5202, 0.5300, 0.8147]) \\
\text{THEN } & \hat{y} \text{ is } [0.4327, 0.5103, 0.5202, 0.8177] \text{ AND} \\
\hat{y} = \; & 0.0367 + 0.0300\, x_1 - 0.0815\, x_2 - 0.0449\, x_3 + 0.0115\, x_4 + \\
& + 0.0225\, x_5 + 0.0396\, x_6 + 0.0532\, x_7 + 0.8812\, y^{[-1]}
\end{aligned}
$$

Here, x_1 stands for 'Jitter(%)'; x_2, 'Shimmer'; x_3, 'NHR'; x_4, 'HNR'; x_5, 'RPDE'; x_6, 'DFA'; x_7, 'PPE'; $y^{[-1]}$, 'last UPDRS'; and $\hat{y}$ is the predicted 'UPDRS'. Linguistically, and based on all existing rules at $h = 5460$, rule R^i can be read: if 'Jitter(%)' is 'very low', 'Shimmer' is 'low', 'NHR' is 'very low', 'HNR' is 'high', 'RPDE' is 'moderate', 'DFA' is 'low', 'PPE' is 'low', and 'last UPDRS' is 'high', then 'UPDRS' is 'high'. The outputs of granular modeling approaches help to monitor Parkinson's disease symptoms.

Evolving granular systems are capable to process online interval and fuzzy granular data as well as handle ranges of possible values to approximate functions. In addition, they give transparency and interpretability of the resulting construct.

7.5 Control

Process controllers aim at keeping the output of a specific process within a desired range. In the following, we shall show how to use IBeM, FBeM and eGNN as controllers of a feedback system.

7.5.1 Sensor-Based Robust Navigation

We consider an instance of autonomous robot navigation in an unknown environment with obstacle avoidance. From the point of view of control, the autonomous navigation problem consists in designing driving rules based on available sensor data. Evolving granular systems for sensor-based navigation play the role of reactive adaptive controllers that prevent the robot from colliding with obstacles. We assume that the pair of sensors available for obstacle detection is infrared sensors directed head-on and symmetrically as shown in Fig. 7.17. Measurements from

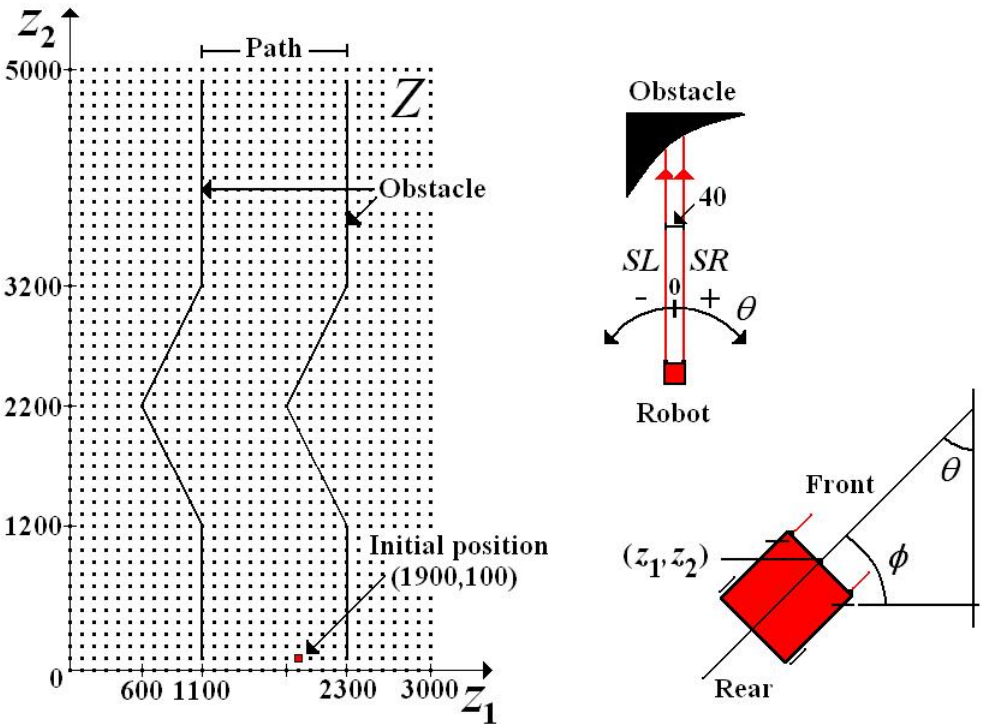


Figure 7.17: Environment for sensor-based navigation

sensors SL and SR give a linear approximation of the surface of an obstacle. The control variable is the wheel steering angle ϕ. Variable θ stands for the reference angle between the robot and the border of the track.

We assume the navigation environment is flat without slopes, but unknown. Coordinates z_1 and z_2 range between $[0, 3000]$ and $[0, 5000]$. Positive values of the steering angle ϕ represent clockwise rotation of the steering wheel, and negative values mean counterclockwise rotation. At every processing step, the controller yields a steering angle. Input sensor readings, SL and SR, are proportional to the distance between robot and obstacle and limited to 500. The perpendicular distance between infrared beams is 40. We want the robot to drive through the path without hitting the borderline.

Simple kinematical relations approximate the robot movement. For example, if the robot moves from position (z_1,z_2) to position (z'_1,z'_2) at step h with speed S, then:

$$\theta' = \theta + \phi \tag{7.11}$$

$$z'_1 = z_1 + S\ \sin(\theta') \tag{7.12}$$

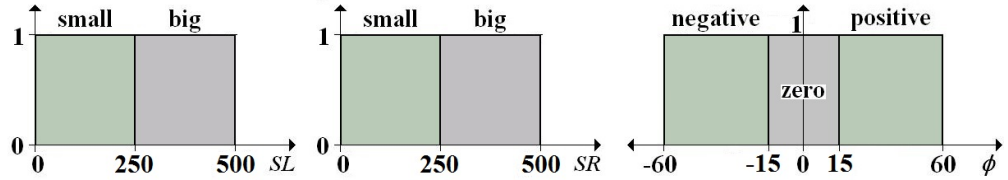
$$z'_2 = z_2 + S\ \cos(\theta') \tag{7.13}$$

Obstacle avoidance simulation models often ignore physical limitations and processing delays. Estimated paths are often unrealistic once the feasibility of the trajectory is not guaranteed. In addition, uncertainty in measurements may hinder the robot to follow trajectories precisely. Evolving granular systems deal with these constraints by keeping the robot between tolerance bounds $[u, U]$ around the more precise estimated path p.

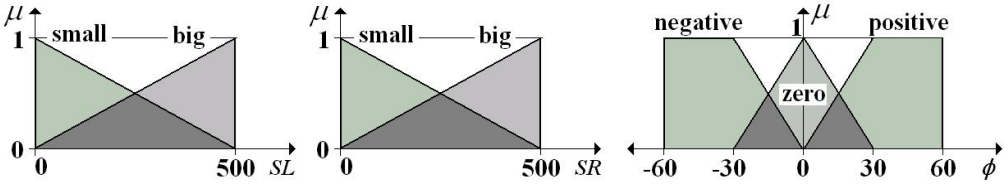
Experiments with different navigation speeds and noisy data were performed. Experts provided a few common-sense associations of how the state and control variables behave prior to learning and navigation. Three rules were considered:

R^1 : IF (SL is big) AND (SR is big) THEN (ϕ is zero) AND ($\phi = p^1(SL, SR)$)
R^2 : IF (SL is small)AND(SR is big)THEN(ϕ is positive)AND($\phi = p^2(SL, SR)$)
R^3 : IF (SL is big)AND(SR is small)THEN(ϕ is negative)AND($\phi = p^3(SL, SR)$)

The parameters of functions p^i are $a^1 = (0,-0.034,0.034)$, $a^2 = (5,0.04,0.1)$, and $a^3 = (-5,-0.1,-0.04)$. In the case of interval framework, sharp boundaries define the subsets 'big', 'small'; 'negative', 'zero', and 'positive' as shown in Fig. 7.18(a). In the cases of fuzzy and neurofuzzy frameworks, trapezoidal membership functions define the same subsets as illustrated in Fig. 7.18(b).



(a) Interval attributes



(b) Fuzzy attributes

Figure 7.18: Initial conditions for the autonomous navigation control problem

The robot is chosen to be initially placed at position $(1900, 100)$ with reference angle $\theta = 0^o$ in all experiments (see Fig. 7.17). Sensor data streams are singular and linearly scaled to the range [0,1]. The IBeM and FBeM controllers start with parameters $\rho = 0.5$, $h_r = 1/\psi = 10000$ and $\eta = 1$; eGNN adopts the same parameters as IBeM and FBeM plus $\mathbb{C}^i = T_L$, $\mathbb{C}^f = M$ and $\zeta = \vartheta = 0.5$. It is worth noting that although the support of the initial membership functions of Fig. 7.18 covers the whole variables domain, searching for more specific rules to fit never-before-seen stream data may contract granules and therefore activate structural adaptation of models.

Figure 7.19 shows different trajectories concerning the robot driving at speeds 5, 10, 20, 30 and 40, and using different granular controllers. We run each algorithm five times independently in this experiment. We notice in the figure that the robot responded faster to obstacle detection driving at lower speeds. Moreover, alignment (parallel to the obstacle) after left and right turns tended to be

more accurate at lower navigation speeds. Alignment yields smoother and shorter paths, which are intuitively preferable. Some classes of problems emphasize fast environment exploration though. Table 7.9 compares the performance of IBeM, FBeM and eGNN based on the results shown in Fig. 7.19.

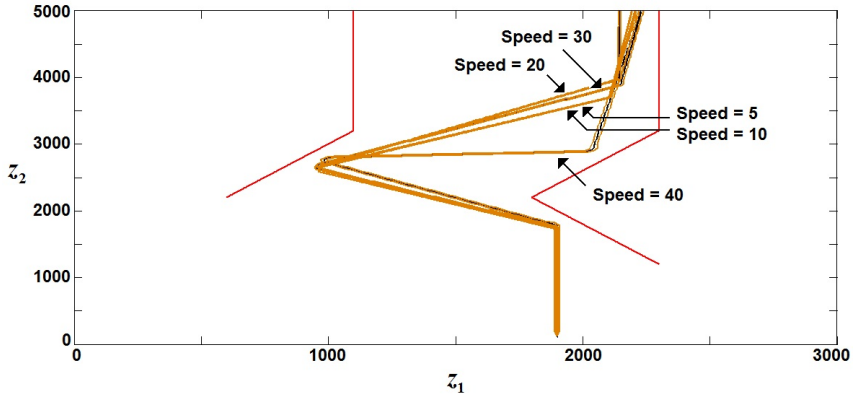
Table 7.9: Comparison of different evolving granular controllers

Model	Speed	# Steps to Goal	CPU	# Rules
IBeM	5	1164	5.57	3
	10	581	3.18	3
	20	289	1.95	3
	30	194	1.60	3
	40	156	**1.29**	4
FBeM	5	1094	4.70	3
	10	546	2.78	3
	20	290	1.91	4
	30	202	1.61	3
	40	157	1.51	4
eGNN	5	1086	5.71	3
	10	550	3.33	3
	20	279	2.15	3
	30	201	1.86	4
	40	**154**	1.70	4

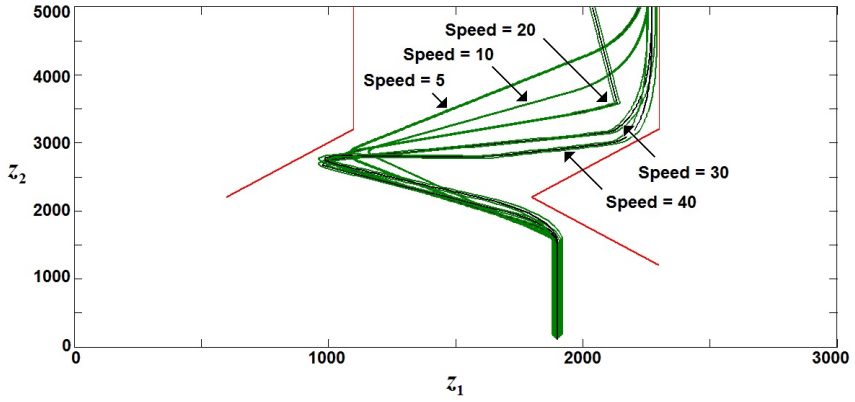
Table 7.9 shows that eGNN provided the shortest path in 3 of the 5 speed settings, seconded by IBeM and FBeM with one shortest path each. IBeM was able to process data faster than the remaining methods and therefore is the best controller in terms of time spent to achieve the goal.

Figure 7.20 evidences the numerical and granular outputs of an experiment with the FBeM controller. Numerical output p is provided by the functional part of the FBeM controller, and granular output $[u, U]$ is given by the bounds of the granular part of the FBeM consequent. Granular output is interpreted as a guaranteed safe path for navigation and maneuver.

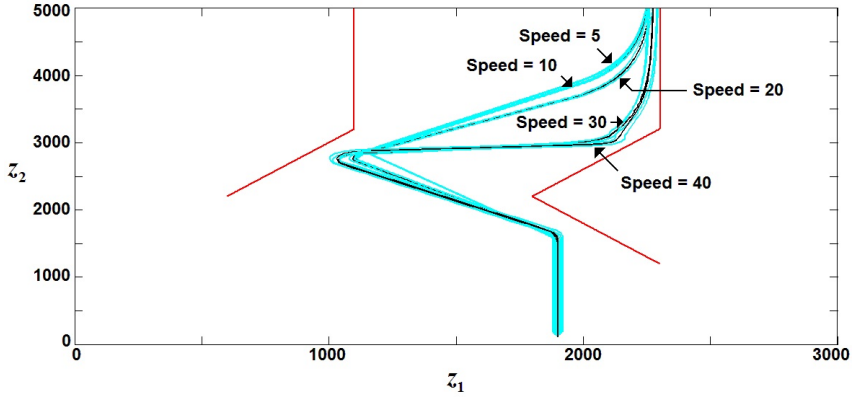
The simulation concerning speed 30 (Fig. 7.20) started with 3 rules and ended up with 4 rules. After contraction and drifting of initial membership functions of antecedents toward frequently requested regions around 500, when the robot approached the obstacle and the range of sensor readings got smaller quickly, a



(a) IBeM navigation



(b) FBeM navigation



(c) eGNN navigation

Figure 7.19: Granular controllers navigating at different speeds

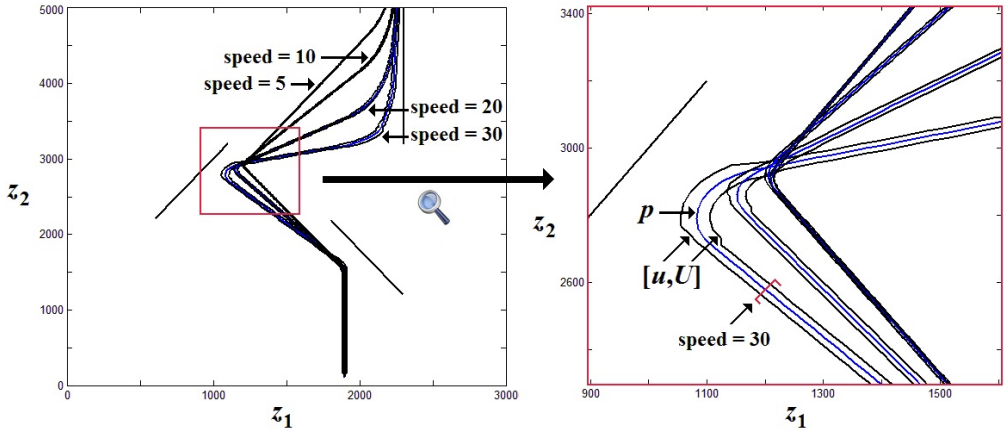


Figure 7.20: Detail of the FBeM navigation at different speeds

new rule:

$$R^4 : \text{IF } (SL \text{ is very small}) \text{ AND } (SR \text{ is small})$$
$$\text{THEN } (\phi \text{ is big positive}) \text{ AND } (\phi = p_1^4(SL, SR))$$

was created to help the robot turn right faster and avoid collision.

Experiment adding noise in the range $\vartheta = [-0.05, 0.05]$ to input data was conducted. Noise may swing the robot from one side to the other. In this experiment the robot speed remains fixed at 5 during the simulations. The initial parameters of IBeM, FBeM, and eGNN are the same as in the previous experiment. Figure 7.21 illustrates trajectories from independent simulations considering FBeM. Trajectories for IBeM and eGNN are similar. We notice that when obstacles are out of vision, the controller accepts input data as they are and lets the robot explore the environment freely. Otherwise, when obstacles are detected, the controller responds turning the robot left and right satisfactorily. Granular representation of data alleviates unlikely swing effect.

Naturally, the accuracy of the granular controllers for navigation can be improved using more sensors and considering speed as control variable. Evolving algorithms offer model-free estimation of the control system. Even if a mathematical model is available, evolving controllers may prove more robust, easier to adapt, and give additional linguistically interpretable granular information,

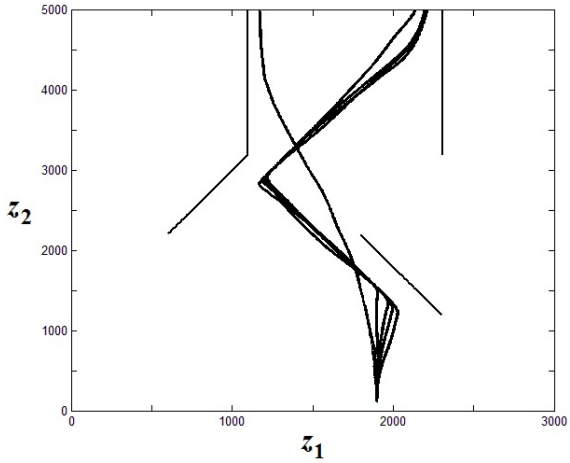


Figure 7.21: FBeM navigating with noisy input

which may help design, analysis and supervision. If experts can provide structured knowledge of the control system or if training data are unavailable, the evolving granular approach proceeds as an adaptive controller.

7.6 Summary

This chapter has shown that evolving granular modeling approaches can handle singular and granular data from distinct scenarios successfully. Moreover, they can outperform other offline, online adaptive and evolving approaches in terms of accuracy, conciseness, processing speed, and interpretability. IBeM, FBeM, and eGNN have provided meaningful understandable rules in semi-supervised classification, function approximation, time series prediction, and control problems.

Chapter 8

Conclusion

This chapter summarizes the major aspects of this work, reiterates the thesis contributions, and discusses potential future research.

8.1 Summary

This thesis has introduced a framework for evolving granular system modeling and a suite of methods for uncertain data processing. Evolving granular systems emphasize structural learning of rule-based models and their realization from online data streams. We consider uncertain data streams and transparent and linguistically appealing approaches that explore the data uncertainty. Evolving granular systems provide a way to construct models of real-world processes involving domain knowledge, experience, and empirical data.

The evolving granular framework is supported by notions of granular computing such as data granulation, granularity adaptation, and granular data processing. Development of the structure of models from scratch on a plug-and-play incremental basis and computation with multi-sized granules are fundamental characteristics of the framework. Instead of dealing with the problem as a whole, we gradually granulate it in simpler sub-parts. The premise is to discover more abstract granular knowledge from finer granular input-output data. Because uncertain data prevail in stream applications, excessive granularity (close to the singularity) becomes unnecessary and inefficient.

In spite of the fact that granular computing is a unified framework for granular

information processing, current literature is fairly dispersed and related ideas have been developed independently under different terminologies. Of particular concern to this work are numeric, interval, and fuzzy types of granular data; and interval, fuzzy, and neurofuzzy modeling frameworks. We have introduced three methods, viz., interval based evolving modeling (IBeM), fuzzy set based evolving modeling (FBeM), and evolving granular neural network (eGNN). IBeM uses interval data and interval preserving operations rooted in the theory of interval mathematics. FBeM deals with fuzzy data and produces results in fuzzy granular format. eGNN essentially encodes a set of fuzzy rules in its topology; therefore, neural processing conforms with a fuzzy inference system. Differently from FBeM, eGNN is equipped with fuzzy aggregation neurons, which provides it with a higher level of adaptability. Fuzzy sets and neurocomputing are complementary in terms of their strengths, thus motivating neurofuzzy granular computing.

We evaluate the methods on several nonstationary environments, namely, semi-supervised classification, weather time series prediction, function approximation in material and biomedical engineering, and control of an autonomous robot. Put broadly, spatial and temporal aspects of data stream processing were examined from a granular perspective. In general, the IBeM, FBeM, and eGNN methods were able to provide: (***i***) computational tractability and scalability with the number of samples and input variables; (***ii***) improved interpretability and transparency of models; (***iii***) reduced cost of data processing in relation to non-evolving methods; (***iv***) approximated results, bounds on the approximations, and description of actions. A user is presented with a range of values without pressure as to the commitment of choosing a specific numeric solution. A numeric value is also given as result of local linear functions associated to granules.

All methods have shown to be extremely general and able to outperform state-of-the-art evolving approaches. FBeM and eGNN alternate as the most accurate methods. A main disadvantage of IBeM is relatively lower accuracy. This is a consequence of parameter-free internal representation in opposition to the fuzzy membership functions of FBeM and eGNN. IBeM usually processes data faster than FBeM and eGNN for the same reason. An advantage of eGNN with respect to IBeM and FBeM is the ability of feature weighting: eGNN tends to be more robust against irrelevant input variables by changing the value of connection

weights over time. However, this characteristic was not of particular importance to the chosen applications.

All evolving granular modeling approaches provide a complete model description and offer different linguistic insights into the nature of granular data relationship. Results were at least as strong as the results obtained by recently proposed methods in the field of evolving intelligent systems.

8.2 Contributions

This thesis has proposed evolving granular systems, a rule-based modeling framework able to handle uncertain data streams from online information systems. Evolving granular systems suggest a paradigm shift in online data analysis supported by the fact that storage and offline processing of large amounts of data are quite often impractical or simply not cost effective. Accuracy, transparency, and interpretability are the keys in evolving granular systems.

Three practical approaches founded on principles from different theories were suggested to handle granular data streams. Interval based evolving modeling is an interval granular approach to enclose imprecise data streams revealed as tolerance intervals. Interval-based modeling comes with a recursive learning algorithm rooted in fundamentals of interval mathematics. Antecedent and consequent parts of interval rules are interval hyperboxes which are connected by an inclusion function. Fuzzy set based evolving modeling uses fuzzy granular models to deal with finer fuzzy data. For each fuzzy granule, there exists an associated fuzzy rule. The structure of the fuzzy rule base is gradually developed from an incremental learning algorithm suitable to process potentially unbounded fuzzy data streams. This approach renders linguistic models of systems and fuzzy granular approximation of functions. Evolving granular neural networks use fuzzy granules and fuzzy aggregation neurons for information fusion. The network can be translated into a knowledge base and a comprehensible rule-based inference system. Learning in evolving granular neural networks consists in building and adapting the network structure from fuzzy data streams. This means that the neural network captures new information from data streams, adapts itself to the

new scenario, and avoids redesigning and retraining.

A large set of experiments was performed to show the usefulness of the evolving granular approaches. The interval IBeM, fuzzy FBeM, and neurofuzzy eGNN approaches were evaluated in a variety of applications such as semi-supervised classification, time series prediction, function approximation, and control. The experiments emphasized the difficulty of currently existing machine learning and computational intelligence approaches to deal with nonstationary data streams. Comparative results have demonstrated the relevance of the proposed framework. Although important results have been achieved in this thesis, a lot of challenges still lie ahead.

8.3 Future Research

The result of this thesis provides potential insights for further research. Next, we briefly list some of the most immediate works that will improve our achievements.

In the near future, an important research topic is to elaborate on new clustering methods of interval and fuzzy interval data streams. Different closeness metrics for intervals and fuzzy intervals are likely to be rethought in a recursive way and therefore considered in unsupervised evolving granular modeling. Techniques for clustering categorical data streams are also worth addressing.

Another potential area of future work is the study of multi-dimensional granular models. Skewed, non-aligned, multi-dimensional granules allow dependencies among several input variables to be captured without necessarily committing to any directional association of the underlying variables. Multi-dimensional data representation preserves information about interactions between input variables through the use of dispersion matrices.

During this thesis, we did not go into chunk-based learning. Chunks of data are sets of sequential data samples buffered to be analyzed at once. Chunks are discarded soon after use. Although incremental data chunk driven learning algorithms usually require additional memory and processing time compared to incremental instance-based algorithms, we envision they may provide interesting insights in one-class classification problems and learning from imbalanced data

sets.

We have addressed data granulation in the time (sampling) and space (clustering) domains. We hope to extend this further to granulate the feature domain (feature selection). In this direction, uncertainty in data representation may be useful to help choosing granular features. For example, a feature with greater uncertainty may not be as important as one with smaller uncertainty. Data uncertainty works as a guideline to incremental granular feature selection.

The interval, fuzzy and neurofuzzy approaches discussed in this work are considered semi-active. In semi-active learning, not all available data samples are used for model adaptation. Implicitly, the approaches ignore indistinguishable samples as a result of temporal and spatial granulation. Conversely, active learning approaches deal with filtering mechanisms. Filtering mechanisms such as those based on the participatory learning paradigm may prevent evolving systems to be exposed to outliers, thus being an interesting issue to be explored.

This thesis did not cover the qualitative effects of data granulation which relates directly to computing with words. In computing with words, the objects of computation are words and propositions drawn from natural language. Often, we translate information expressed in words (soft information) into some tractable granular computing framework. When people want this information back from a fusion system, they want it retranslated in a way it can be used in the task they are really interested in. More precisely, the retranslation process consists in converting a formal mathematical representation into natural language statements that can be understood by human beings. We believe there is room to research on finding better criteria to retranslate granules into words based on external goals.

We envision as thesis topics of great importance in the near future hybrids between evolving rule-based systems and methods from the rough sets and support vector machines theories. In particular, rough set theory is an instance of granular computing framework. Imprecision in the rough set approach is expressed by a boundary region, and not by partial membership, as in fuzzy set theory. Two crisp sets - called lower and upper approximation - are associated to a rough set. The lower approximation of a set consists of all elements that surely belong to the set, whereas the upper approximation of the set constitutes of all elements that possibly belong to the set. The difference of the upper and the lower ap-

proximation is the boundary region. Any rough set, in contrast to a crisp set, has a non-empty boundary region. Evolving rough sets from uncertain data streams is still an approach to be explored. Support vector machines are characterized by the use of kernel mapping techniques. Although the support vector machine approach has been primarily applied to pattern recognition, many of its ideas carry over to the cases of function approximation. A support vector machine can be, for example, used in the consequent part of a granular rule, thus providing local nonlinear models for classification or regression of uncertain data.

Although granular computing extends real-valued computing to computing with intervals, fuzzy sets, etc., they remain deterministic in its very practical aspect because calculations remain based on real parameters which characterize granules. Computing with granules and words are recognized to be of great relevance to some matters in which the confusion of goals does not justify perfection of means. However, the current digital signal processing technology and the discrete/continuous dichotomy do not allow computing in a more abstract human-like level of thinking in its ultimate meaning. These are more philosophical issues worthy of future research.

On a more practical level, there is a need to reproduce our results on virtual or real-world environments such as in an online business or industrial setting. Although the data sets used in the experiments were recorded from actual applications, other software development issues must be addressed so that the proposed methods may effectively provide online decision support. System integration is needed to link together the different methods, computer networking, and software applications to act as a coordinate whole. Software testing is necessary to validate if the requirements of the applications were met and that the methods work as expected, producing similar results to those obtained in simulations. The extent to which the presented results can be reproduced as part of an integrated system still remains to be fully determined.

We expect research on evolving granular systems to grow further, and believe that it may have a valuable role in prediction and decision support systems.

Appendix A

Universal Approximation

In this appendix we provide a constructive proof that an evolving granular system can approximate any arbitrary continuous function with arbitrary accuracy in compact domains. The proof is based on a work of Davis (36). Here we emphasize multiple input single output models.

Let

$$p = \sum_{i=1}^{c} p^i \ , \tag{A.1}$$

be a piecewise linear continuous function where

$$p^i = \begin{cases} a_0^i + \sum_{j=1}^{n} a_j^i x_j & \text{if } x_j \in [l_j^i, L_j^i] \ \forall j \\ 0 & \text{otherwise} \end{cases}$$

The purpose of p is to approximate a continuous function $f : \mathbb{L} \rightarrow \mathbb{R}$ where $\mathbb{L} = [l_1^i, L_1^i] \times ... \times [l_j^i, L_j^i] \times ... \times [l_n^i, L_n^i]$; index i refers to an ordered set of non-overlapping granules, that is, $l_j^1 = min(x_j^{[1]}, ..., x_j^{[H]})$, $L_j^c = max(x_j^{[1]}, ..., x_j^{[H]})$, H is the total number of samples in a given instant of time. Lower and upper bounds

of intervals are such that $l_j^1 \leq L_j^1 = ... = l_j^i \leq L_j^i = ... = l_j^c \leq L_j^c$. In other words, our purpose is to approximate function f by piecewise linear functions.

In what follows we limit our discussion to real and continuous functions f that are defined on $\mathbb{L}$, i.e., the convex hull of the current set of granules $\gamma = (\gamma^1, ..., \gamma^c)$. Observations from the function f are designated as $x^{[A]}$, $x^{[B]}$, $x^{[C]}$, etc., instead of $x^{[h]}$ in order to stress independence from the order of data presentation.

Theorem 1: Let P be an enclosure of a family of functions $p(x)$ such that

$$p \in P \text{ implies } max(p^1, ..., p^c) \in P \text{ and } min(p^1, ..., p^c) \in P. \tag{A.2}$$

A continuous function f is uniformly approximable by members of p within P if and only if for any two points $x^{[A]}$ and $x^{[B]}$, and for any $e > 0$, there exists a p such that

$$|f(x^{[A]}) - p(x^{[A]})| < e \text{ and } |f(x^{[B]}) - p(x^{[B]})| < e. \tag{A.3}$$

Proof: If uniform approximation is possible, then given $e > 0$ we can find a $p(x^{[A]}) \in P$ such that

$$|f(x^{[A]}) - p(x^{[A]})| \quad < \quad e,$$

and so (A.3) follows trivially. Conversely, suppose that (A.3) holds. Select a fixed $x^{[B]} \in P$ and a fixed $e > 0$. Then, for any point $x^{[C]}$, we can find a function $p(x^{[A]}) = p(x^{[A]}; x^{[B]}, x^{[C]}, e)$ such that $|f(x^{[B]}) - p(x^{[B]})| < e$ and $|f(x^{[C]}) - p(x^{[C]})| < e$. In particular,

$$p(x^{[C]}) \quad < \quad f(x^{[C]}) + e.$$

By continuity of p and f, this inequality must persist in a certain neighborhood N of $x^{[C]}$. As $x^{[C]}$ runs over all the points of P, the corresponding neighborhoods must cover P. By the Heine-Borel theorem, which states that every closed interval in $\Re^n$ is compact, we can find a finite number of them, $N_1, ..., N_c$, that covers P. The corresponding functions $p(x^{[A]}; x^{[B]}, x^{[C_i]})$ satisfy

$$p(x^{[A]}; x^{[B]}, x^{[C_i]}) < f(x^{[A]}) + e,\ x^{[A]} \in N_i,\ i = 1, ..., c. \tag{A.4}$$

Define

$$p^-(x^{[A]}; x^{[B]}) = min\{p(x^{[A]}; x^{[B]}, x^{[C_1]}), ..., p(x^{[A]}; x^{[B]}, x^{[C_c]})\}. \tag{A.5}$$

By (A.2) iterated, $p^- \in P$ and by (A.4),

$$p^-(x^{[A]}, x^{[B]}) \ < \ f(x^{[A]}) + e,\ x^{[A]} \in P. \tag{A.6}$$

Once again, for each i we have

$$|f(x^{[B]}) - p(x^{[B]}; x^{[B]}, x^{[C_i]})| \ < \ e$$

so that

$$p(x^{[B]}; x^{[B]}, x^{[C_i]}) \ > \ f(x^{[B]}) - e.$$

It follows from (A.5) that

$$p^-(x^{[B]}; x^{[B]}) \;\; > \;\; f(x^{[B]}) - e. \tag{A.7}$$

By continuity, (A.7) must persist in a neighborhood O of $x^{[B]}$:

$$p^-(x^{[A]}; x^{[B]}) \;\; > \;\; f(x^{[A]}) - e.$$

Now, let $x^{[B]}$ run over P. These neighborhoods O cover P, and we may find a finite number of them, $O_1, O_2, ..., O_c$, corresponding to $x^{[B_1]}, ..., x^{[B_c]}$, that covers P. Since

$$p^-(x^{[A]}; x^{[B_i]}) > f(x^{[A]}) - e, \;\; x^{[A]} \in O_i, \;\; i = 1, 2, ..., c,$$

and since the O_i cover P, for every $x^{[A]} \in P$, the inequality

$$p^-(x^{[A]}; x^{[B_i]}) \;\; > \;\; f(x^{[A]}) - e$$

must hold for some i.

If we set

$$s(x^{[A]}) = max\{p^-(x^{[A]}, x^{[B_1]}), ..., p^-(x^{[A]}, x^{[B_c]})\}$$

then by what we have just said,

$$s(x^{[A]}) \;\; > \;\; f(x^{[A]}) - e, \;\; \forall x^{[A]} \in P. \tag{A.8}$$

On the other hand, by (A.6), $p^{-}(x^{[A]}; x^{[B]}) < f(x^{[A]}) + e$, $x^{[A]} \in P \; \forall \; x^{[B]}$. Hence,

$$s(x^{[A]}) \;\; < \;\; f(x^{[A]}) + e, \;\; x^{[A]} \in P. \tag{A.9}$$

Combining (A.9) with (A.8),

$$|f(x^{[A]}) - s(x^{[A]})| \;\; < \;\; e, \;\; x^{[A]} \in P.$$

Finally, by (A.2) iterated, $s(x^{[A]}) \in P$.

Being P a finite domain and p the set of all piecewise linear functions defined on P (A.1), it is easy to verify that p satisfies (A.2). Condition (A.3) can be satisfied with $e = 0$ by means of a linear function. These results theoretically guarantee that the desired approximation is always achievable **Q.E.D.** This is stated more precisely in the following corollary.

Corollary 1: Every continuous function can be approximated uniformly on a finite interval by continuous piecewise linear functions.

Let H be a sufficient large number of steam data $x^{[h]}, h = 1, ..., H$, so that there exists coverage for all granules in the problem space $\mathbb{L} = [l_1^i, L_1^i] \times ... \times [l_j^i, L_j^i] \times ... \times [l_n^i, L_n^i]$. Superscript $i = 1, ..., c$, refers to an ordered set of granules. Moreover, being H finite, the total number of granules in $\mathbb{L}$ is finite, with $1 \leq c \leq H$ depending on the granularity value, ρ.

Complete covarage of $\mathbb{L}$ after H time steps is guaranteed by creating granules that match every never-before-seen value and forbidding learning algorithms to delete granules. Then, for a model with c granules, the difference between a certain $p^i(x^{[h]})$ and $y^{[h]}$, a measure of f, provides the worst case approximation error $e^i = max(e^1, ..., e^c)$. Being all functions p^i, $i = 1, ..., c$, piecewise-linear and defined for all $x^{[h]}$ in $\mathbb{L}$, function $p = \sum_{i=1}^{c} p^i$ satisfies condition (A.3) for any $(x, y)^{[h]}$. Therefore, corollary 2 follows trivially from corollary 1.

Corollary 2: Evolving granular systems are universal approximators.

Appendix B

Recursive Least Squares Method

The recursive least squares (RLS) algorithm is used to adapt consequent function parameters a_j^i as follows.

Let $(x, y)^{[h]}$ be the sample available for training at step h. We adjust the coefficients a_j^i of p^i assuming that

$$y^{[h]} = a_0^i + \sum_{j=1}^{n} a_j^i x_j^{[h]}. \tag{B.1}$$

If x_j and y are intervals or symmetric trapezoids, then to adapt the coefficients a_j^i using the standard form of the RLS algorithm, we take advantage from the midpoint of the respective intervals or trapezoids. In the remainder of this appendix we assume $(x, y)^{[h]}$ are real numbers (midpoints of intervals or trapezoidal fuzzy data) for short. In case trapezoids are asymmetric, then an alternative is to use the center of the area.

In the matrix form, the equation (B.1) becomes

$$Y = X\Omega^i, \tag{B.2}$$

where $Y = [y^{[h]}]$, $X = [1 \quad x_1^{[h]} \quad ... \quad x_n^{[h]}]$, and $\Omega^i = [a_0^i \quad ... \quad a_n^i]^T$ is the vector of unknown parameters. To estimate the coefficients a_j^i we let

$$Y = X\Omega^i + E, \tag{B.3}$$

where

$$E = \epsilon^{[h]} = y^{[h]} - p(x^{[h]}) \tag{B.4}$$

is the approximation error. While in batch estimation the rows in Y, X and E increase with the number of available instances, in recursive mode only two rows are kept and we reformulate equations (B.2)-(B.4) as follows:

$$Y = \begin{bmatrix} y^{[h-1]} \\ y^{[h]} \end{bmatrix}, \; X = \begin{bmatrix} 1 & x_1^{[h-1]} & ... & x_n^{[h-1]} \\ 1 & x_1^{[h]} & ... & x_n^{[h]} \end{bmatrix}$$

$$\text{and } E = \begin{bmatrix} \epsilon^{[h-1]} \\ \epsilon^{[h]} \end{bmatrix}. \tag{B.5}$$

The rows in (B.5) refer to values before and just after adaptation. The RLS algorithm chooses Ω^i to minimize the functional

$$J(\Omega^i) = E^T E. \tag{B.6}$$

Ω^i is given by

$$\Omega^i = (X^T X)^{-1} X^T Y. \tag{B.7}$$

Assuming $P = (X^T X)^{-1}$ and using the matrix inversion lemma (148) we avoid inverting $X^T X$ using:

$$P(\text{new}) = P(\text{old}) \left[I - \frac{X X^T P(\text{old})}{1 + X^T P(\text{old}) X} \right], \tag{B.8}$$

where I is the identity matrix. In practice it is usual to choose large initial values for the entries of the main diagonal of P. We use $P^{[0]} = 10^3 I$ as the default value.

After simple mathematical transformations, the vector of parameters is rearranged recursively as follows:

$$\Omega^i(\text{new}) = \Omega^i(\text{old}) + P(\text{new}) X \left(Y - X^T \Omega^i(\text{old}) \right). \tag{B.9}$$

Detailed derivations of the RLS algorithm can be found in (13) and convergence proof in (61).

Bibliography

[1] Aggarwal, C. C.; Han, J.; Wang, J.; Yu, P. S. "A framework for on-demand classification of evolving data streams." IEEE Transactions on Knowledge and Data Engineering, Vol. 18, Issue 5, pp: 577-589, 2006. 32

[2] Aggarwal, C. C.; Yu, P. S. "A framework for clustering uncertain data streams." IEEE International Conference on Data Engineering, pp: 150-159, 2008. 44, 63

[3] Aggarwal, C. C.; Yu, P. S. (Eds.) Privacy-Preserving Data Mining: Models and Algorithms. Springer-Verlag (Series: Advances in Database Systems), Vol. 34, 513p. 2008. 39, 73

[4] Alonso, J. M.; Magdalena, L. "Special issue on interpretable fuzzy systems." Information Sciences, Vol. 181, pp: 4331-4339, 2011. 32

[5] Angelov, P. Evolving Rule-Based Models: A Tool for Design of Flexible Adaptive Systems. Springer-Verlag, Heidelberg, New York (Studies in Fuzziness and Soft Computing), 227p. 2002. 4, 33

[6] Angelov, P.; Filev, D. "An approach to online identification of Takagi-Sugeno fuzzy models." IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 34, Issue 1, pp: 484-498, 2004. 33, 40, 111, 124, 130

[7] Angelov, P.; Zhou, X. "Evolving fuzzy systems from data streams in real-time." IEEE Symposium on Evolving Fuzzy Systems, pp: 29-35, 2006. 111, 124, 130

[8] Angelov, P.; Zhou, X.; Filev, D.; Lughofer, E. "Architectures for evolving fuzzy rule-based classifiers." IEEE International Conference on Systems, Man and Cybernetics, pp: 2050-2055, 2007. 34

[9] Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Fuzzy Systems - Preface to the Special Section. IEEE Transactions on Fuzzy Systems, Vol. 6, Issue 6, pp: 1390-1392, 2008. 4, 32

[10] Angelov, P.; Zhou, X. "Evolving fuzzy-rule-based classifiers from data streams." IEEE Transactions on Fuzzy Systems, Vol. 16, Issue 6, pp: 1462-1475, 2008. 32, 34, 77

[11] Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Intelligent Systems: Methodology and Applications. Wiley-IEEE Press Series on Computational Intelligence, 444p. 2010. 4, 32, 33, 44, 79

[12] Ashokaraj, I.; Tsourdos, A.; Silson, P.; White, B. "Sensor based robot localisation and navigation: using interval analysis and extended Kalman filter." 5th Asian Control Conference, Vol. 2, pp: 1086-1093, 2004. 49

[13] Astrom, K. J.; Wittenmark, B. Adaptive Control. Prentice-Hall, Addison-Wesley, Boston, 2nd edition, 580p. 1994. 32, 155

[14] Ballini, R.; Mendonca, A.; Gomide, F. "Evolving fuzzy modeling of sovereign bonds." Journal of Financial Decision Making, Special Issue: The Fuzzy Logic in the Financial Uncertainty, Vol. 5, Issue 2, pp: 3-15, 2009. 128

[15] Bargiela, A.; Pedrycz, W. Granular Computing: An Introduction. Kluwer Academic Publishers - Boston, 1st edition, 452p. 2002. 2, 3, 11, 12, 44, 45

[16] Bargiela A.; Pedrycz, W. "Granulation of temporal data: a global view on time series." International Conference of the North American Fuzzy Information Processing Society, pp: 191-196, 2003. 40

[17] Bargiela, A.; Pedrycz, W. "Recursive information granulation: aggregation and interpretation issues." IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 33, Issue 1, pp: 96-112, 2003. 43, 44

[18] Bargiela, A.; Pedrycz, W. "Granular mappings." IEEE Transactions on System, Man, and Cybernetics - Part A, Vol. 35, Issue 2, pp: 292-297, 2005. 3, 11, 13

[19] Bargiela, A.; Pedrycz, W. "Toward a theory of granular computing for human-centered information processing." IEEE Transactions on Fuzzy Systems, Vol. 16, Issue 2, pp: 320-330, 2008. 11

[20] Beliakov, G.; Pradera, A.; Calvo, T. Aggregation Functions: A Guide for Practitioners. Springer-Verlag, Berlin, Heidelberg, 1st edition (Studies in Fuzziness and Soft Computing), 361p. 2007. 26

[21] Beringer, J.; Hullermeier, E. "Efficient instance-based learning on data streams." Intelligent Data Analysis, Vol. 11, Issue 6, pp: 627-650, 2007. 4

[22] Bezdek, J. Pattern Recognition with Fuzzy Objective Function Algoritms. Plenum Press, New York, 1981. 100

[23] Bifet, A.; Holmes, G.; Pfahringer, B.; Kranen, P.; Kremer, H.; Jansen, T.; Seidl, T. "MOA: Massive online analysis, a framework for stream classification and clustering." Journal of Machine Learning Research, Vol. 11, pp: 44-50, 2010. 37

[24] Bouchachia, A.; Gabrys, B.; Sahel, Z. "Overview of some incremental learning algorithms." IEEE International Conference on Fuzzy Systems, pp: 1-6, 2007. 32

[25] Bouchachia, A. "An evolving classification cascade with self-learning." Evolving Systems, Vol. 1, Issue 3, pp: 143-160, 2010. 32

[26] Bouchon-Meunier, B. (Ed.) Aggregation and Fusion of Imperfect Information. Physica-Verlag, Heidelberg, New York (Studies in Fuzziness and Soft Computing), 278p. 1998. 80

[27] Bouchon-Meunier, B.; Marsala, C.; Rifqi, M.; Yager, R. R. (Eds.) Uncertainty in Intelligent and Information Systems. World Scientific - Singapore, 536p. 2008. 2, 48

[28] Box, G. E. P.; Jenkins, G. M.; Reinsel, G. C. Time Series Analysis: Forecasting and Control. Wiley Series in Probability and Statistics, 4th edition, 746p. 2008. 107

[29] Carpenter, G. A.; Grossberg, S. "A massively parallel architecture for a self-organizing neural pattern recognition machine." Computer Vision, Graphics, and Image Processing, Vol. 37, pp: 54-115, 1987. 87

[30] Carpenter, G. A.; Grossberg, S.; Markuzon, N.; Reynolds, J. H.; Rosen, D. B. "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps." IEEE Transactions on Neural Networks, Vol. 3, Issue 5, pp: 698-713, 1992. 62

[31] Carvalho, F. A. T.; Souza, R. M. C. R.; Chavent, M.; Lechevallier, Y. "Adaptative Hausdorff distances and dynamic clustering of symbolic interval data." Pattern Recognition Letters, Vol. 27, Issue 3, pp: 167-179, 2006. 49

[32] Chen, S.; He, H. "Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach." Evolving Systems, Vol. 2, Issue 1, pp: 35-50, 2011. 32

[33] Cross, V. V.; Sudkamp, T. A. Similarity and compatibility in fuzzy set theory: assessment and applications. Physica-Verlag Heidelberg (Studies in Fuzziness and Soft Computing), 209p. 2002. 26, 70

[34] Da Deng; Kasabov, N. "ESOM: An algorithm to evolve self-organizing maps from online data streams." IEEE International Joint Conference on Neural Networks, Vol. 6, pp: 3-8, 2000. 34

[35] Darwin, C. R. The origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. John Murray - London, 6th edition, 1872. 31

[36] Davis, P. J. Interpolation and Approximation. Dover Publications, 393p. 1963. 122, 147

[37] Do, T.-N.; Poulet, F. "Kernel-based algorithms and visualization for interval data mining." In: Zighed, D. A.; Tsumoto, S.; Ras, Z. W., Mining Complex Data, SCI 165, Springer-Verlag, Berlin, Heidelberg, pp: 75-91, 2009. 49

[38] Domingos, P.; Hulten, G. "Mining high-speed data streams." International Conference on Knowledge Discovery and Data Mining, pp: 7180, 2000. 37

[39] Drossu, R.; Obradovic, Z. "Rapid design of neural networks for time series prediction." IEEE Computational Science & Engineering, Vol. 3, Issue 2, pp: 78-89, 1996. 109

[40] Dubois, D.; Kerre, E.; Mesiar, R.; Prade, H. "Fuzzy interval analysis." In: The Handbook of Fuzzy Sets, Vol. 1 - Fundamentals of Fuzzy Sets, Kluwer Academic - Bordrecht, pp: 483-581, 2000. 21

[41] Dubois D.; Prade, H. (Eds.) Fundamentals of Fuzzy Sets. Kluwer Academic Publishers, 1st edition, 653p. 2000. 5

[42] Dubois, D.; Prade, H. "On the use of aggregation operations in information fusion processes." Fuzzy Sets and Systems, Vol. 142, Issue 1, pp: 143-161, 2004. 25, 40

[43] Elwell, R.; Polikar, R. "Incremental learning of concept drift in nonstationary environments." IEEE Transactions on Neural Networks, Vol. 22, Issue 10, pp: 1517-1531, 2011. 37

[44] Engelbrecht, A. P. Computational Intelligence: An Introduction. Wiley - Chichester, England, 2nd edition, 597p. 2007. 4

[45] Fawcett, T. "An introduction to ROC analysis." Pattern Recognition Letters, Vol 27, pp: 861-874, 2006. 100

[46] Gabrys, B.; Bargiela, A. "General fuzzy min-max neural network for clustering and classification." IEEE Transactions on Neural Networks, Vol. 11, Issue 3, pp: 769-783, 2000. 32, 36, 49, 78, 90

[47] Gabrys, B.; Petrakieva, L. "Combining labelled and unlabelled data in the design of pattern classification systems." International Journal of Approximate Reasoning, Vol. 35, Issue 3, pp: 251-273, 2004. 98, 105

[48] Gama, J.; Medas, P. "Learning decision trees from dynamic data streams." Journal of Universal Computer Science, Vol. 11, Issue 8, pp: 1353-1366, 2005. 37

[49] Hahn, G. J.; Meeker, W. Q. Statistical Intervals: A Guide for Practitioners. Wiley, USA, 387p. 1991. 21

[50] Hall, D. L.; Llinas, J. "An introduction to multisensor data fusion." Proceedings of the IEEE, Vol. 85, Issue 1, pp: 6-23, 1997. 44

[51] Hamilton, J. D. Time Series Analysis. Princeton University Press, 1st edition, 799p. 1994. 4, 107

[52] Hamming, R. W. "Error detecting and error correcting codes." Bell System Technical Journal, Vol. 29, Issue 2, pp: 147160, 1950. 26, 69

[53] Hansen, E. R.; Walster, G. W. Global Optimization using Interval Analysis. 2nd edition, Marcel Dekker, New York - Basel, 489p. 2004. 14

[54] Hastie, T.; Tibshirani, R.; Friedman, J. The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer-Verlag, 2nd edition, 768p. 2009. 4, 33, 130

[55] Haykin, S. Neural Networks: A Comprehensive Foundation. Prentice Hall, 2nd edition, 823p. 1999. 6, 100, 111, 130

[56] Hickey, T.; Ju, Q.; van Emden, M. H. "Interval arithmetic: from principles to implementation." Journal of the ACM, Vol. 48, Issue 5, pp: 1038-1068, 2001. 14, 50

[57] Ho, A.; Iansek, R.; Marigliani, C.; Bradshaw, J.; Gates, S. "Speech impairment in a large sample of patients with Parkinson's disease." Behavioral Neurology, Vol. 11, pp: 131-137, 1998. 127

[58] Ho, W. L.; Tung, W. L.; Quek, C. "An evolving Mamdani-Takagi-Sugeno based neural-fuzzy inference system with improved interpretability-accuracy." IEEE International Conference on Fuzzy Systems, pp: 1-8, July, 2010. 62

[59] Iglesias, J. A.; Angelov, P.; Ledezma, A.; Sanchis, A. "Evolving classification of agents behaviors: a general approach." Evolving Systems, Vol. 1, Issue 3, pp: 161-171, 2010. 32

[60] Jaulin, L.; Keiffer, M.; Didrit, O.; Walter, E. Applied Interval Analysis. Springer-Verlag - London, 379p. 2001. 5, 14, 38

[61] Johnson, C. R. Lectures on Adaptive Parameter Estimation. Prentice-Hall - Upper Saddle River, USA, 185p. 1988. 155

[62] Kaburlasos, V. G.; Papadakis, S. E. "Granular self-organizing map (grSOM) for structure identification." Neural Networks, Vol. 19, Issue 5, pp: 623-643, 2006. 78

[63] Kasabov, N. "Evolving fuzzy neural networks for supervised / unsupervised online knowledge-based learning." IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 31, Issue 6, pp: 902-918, 2001. 34, 79

[64] Kasabov, N.; Song, Q. "DENFIS: Dynamic evolving neural-fuzzy inference system and its application." IEEE Transactions on Fuzzy Systems, Vol. 10, Issue 2, pp: 144-154, 2002. 34, 79, 111, 124

[65] Kasabov, N. Evolving Connectionist Systems: Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines. Springer-Verlag - London, 1st edition, 320p. 2003. 4, 33, 34, 79

[66] Kasabov, N. Evolving Connectionist Systems: The Knowledge Engineering Approach. Springer-Verlag - London, 2nd edition, 451p. 2007. 4, 32, 34, 40, 79

[67] Kaufmann, A.; Gupta, M. M. Introduction to Fuzzy Arithmetic: Theory and Applications. Van Nostrand Reinhold Company Inc., New York, 350p. 1985. 21

[68] Kausay, T.; Simon, T. K. "Acceptance of concrete compressive strength." Concrete Structures, Vol. 8, pp: 54-63, 2007. 122, 123

[69] Kearfott, R. B.; Kreinovich, V. Applications of Interval Computations. Kluwer Academic Publishers, 425p. 1996. 5, 14

[70] Klir, G. K.; Yuan, B. Fuzzy Sets and Fuzzy Logic: Theory and Applications. Prentice Hall, 1st edition, 592p. 1995. 21, 22

[71] Kosko, B. Neural Networks and Fuzzy Systems. A Dynamical Systems Approach to Machine Intelligence. Prentice-Hall, Englewood Cliffs, 449p. 1991. 44

[72] Kreinovich, V. "Interval computations as an important part of granular computing: an introduction." In: Pedrycz, W.; Skowron, A.; Kreinovich, V. (Eds.) Handbook of Granular Computing, pp: 1-31, 2008. 2

[73] Kuncheva L. I. Fuzzy Classifier Design. Springer-Verlag, Heidelberg, 321p. 2000. 44

[74] Last, M. "Online classification of nonstationary data streams." Intelligent Data Analysis, Vol. 6, Issue 2, pp: 129-147, 2002. 43

[75] Leite, D.; Costa, P.; Gomide, F. "Interval-based evolving modeling." IEEE Symposium Series on Computational Intelligence, pp: 1-8, 2009. 32, 40, 48

[76] Leite, D.; Costa, P.; Gomide, F. "Evolving granular classification neural networks." IEEE International Joint Conference on Neural Networks, pp: 1736-1743, 2009. 77

[77] Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural network for semi-supervised data stream classification." World Congress on Computational Intelligence - International Joint Conference on Neural Networks, pp: 1877-1884, 2010. 40, 77

[78] Leite, D.; Costa, P.; Gomide, F. "Granular approach for evolving system modeling." In: Hullermeier, E.; Kruse, R.; Hoffmann F. (Eds.) Lecture Notes in Artificial Intelligence, Vol. 6178, pp: 340-349, Springer, 2010. 40, 48

[79] Leite, D.; Gomide, F.; Ballini, R.; Costa, P. "Fuzzy granular evolving modeling for time series prediction." IEEE International Conference on Fuzzy Systems, pp: 2794-2801, 2011. 32, 40, 53, 67, 73

[80] Leite, D.; Gomide, F. "Evolving linguistic fuzzy models from data streams." In: Trillas, E.; Bonissone, P.; Magdalena, L.; Kacprycz, J. (Eds.) Combining Experimentation and Theory: A Hommage to Abe Mamdani (Studies in Fuzziness and Soft Computing), pp: 209-223, 2011. 40, 67, 120

[81] Leite, D.; Costa, P.; Gomide, F. "Interval approach for evolving granular system modeling." In: Mouchaweh, M. S.; Lughofer, E. (Eds.) Learning in Non-Stationary Environments: Methods and Applications, Springer-Verlag, pp: 271-301, 2012. 40, 48

[82] Leite, D.; Costa, P.; Gomide, F. "Evolving granular neural network for fuzzy time series forecasting." World Congress on Computational Intelligence - IEEE Joint Conference on Neural Networks, 8p. 2012. 77

[83] Lemos, A.; Caminhas, W.; Gomide, F. "Fuzzy evolving linear regression trees." Evolving Systems, Vol. 2, Issue 1, pp: 1-14, 2011. 32, 37, 124

[84] Lemos, A.; Caminhas, W.; Gomide, F. "Multivariable Gaussian evolving fuzzy modeling system." IEEE Transactions on Fuzzy Systems, Vol. 19, Issue 1, pp: 91-104, 2011. 32, 35

[85] Lemos, A.; Caminhas, W.; Gomide, F. "Evolving fuzzy linear regression trees with feature selection." IEEE Workshop on Evolving and Adaptive Intelligent Systems, pp: 31-38, 2011. 37

[86] Liggins, M. E.; Hall, D. L.; Llinas, J. (Eds.) Handbook of Multisensor Data Fusion: Theory and Practice. CRC Press, 2nd edition, 849p. 2008. 3

[87] Lima, E.; Gomide, F.; Ballini, R. "Participatory evolving fuzzy modeling." International Symposium on Evolving Fuzzy Systems, pp: 36-41, 2006. 35, 124

[88] Lin, T. Y. "Granular computing on binary relations." International Conference on Rough Sets and Current Trends in Computing, pp: 296-299, 2002. 78

[89] Lin, T. Y. "Neural networks, qualitative fuzzy logic and granular adaptive systems." World Congress of Computational Intelligence, pp: 566-571, 2002. 2, 11

[90] Little, M. A.; McSharry, P. E.; Hunter, E. J.; Spielman, J.; Ramig, L. O. "Suitability of dysphonia measurements for telemonitoring of Parkinson's disease." IEEE Transactions on Biomedical Engineering, Vol. 56, Issue 4, pp: 1015-1022, 2009. 128

[91] Little, R. J. A.; Rubin, D. B. Statistical Analysis with Missing Data. Wiley-Interscience, 2nd edition, 381p. 2002. 39

[92] Ljung, L. System Identification - Theory for the User. Prentice-Hall, Englewood Cliffs, NJ, 519p. 1988. 33

[93] Lodwick, W.; Jamison, K. D. "Special issue: interfaces between fuzzy set theory and interval analysis." Fuzzy Sets and Systems, Vol. 135, pp: 1-3, 2003. 21

[94] Lughofer, E. "FLEXFIS: A robust incremental learning approach for evolving Takagi-Sugeno fuzzy models." IEEE Transactions on Fuzzy Systems, Vol. 16 , Issue 6, pp: 1393-1410, 2008. 32, 36

[95] Lughofer, E.; Angelov, P. "Handling drifts and shifts in on-line data streams with evolving fuzzy systems." Applied Soft Computing, Vol. 11, Issue 2, pp: 2057-2068, 2011. 34, 44

[96] Lughofer, E.; Bouchot, J.-L.; Shaker, A. "On-line elimination of local redundancies in evolving fuzzy systems." Evolving Systems, Vol. 2, Issue 3, pp: 165-187, 2011. 73, 95

[97] Lughofer, E. Evolving Fuzzy Systems - Methodologies, Advanced Concepts and Applications. Springer-Verlag, Berlin Heidelberg, 460p. 2011. 4, 40, 59

[98] Lughofer, E. "On-line incremental feature weighting in evolving fuzzy classifiers." Fuzzy Sets and Systems, Vol. 163, Issue 1, pp: 1-23, 2011. 93

[99] Maimon, O. Z.; Rokach, L. The Data Mining and Knowledge Discovery Handbook. Springer - New York, USA, 1383p. 2005. 3

[100] Mendel, J. M. "Type-2 fuzzy sets and systems: an overview." IEEE Computational Intelligence Magazine, Vol. 2, Issue 2, pp: 20-29, 2007. 21

[101] Mitchell, T. M. Machine Learning. McGraw-Hill Science/Engineering/Math, 1st edition, 414p. 1997. 4

[102] Mitchell, T. M. "The role of unlabeled data in supervised learning." Proceedings of the Sixth International Colloquium on Cognitive Science, 8p. 1999. 98, 105

[103] Moore, R. E. Interval Analysis. Prentice Hall - Englewood Cliffs, NJ, 145p. 1966. 14, 21

[104] Moore, R. E. Methods and Applications of Interval Analysis. SIAM - Philadelphia, 190p. 1979. 5, 14, 20

[105] Moore, R. E.; Lodwick, W. "Interval analysis and fuzzy set theory." Fuzzy Sets and Systems, Vol. 135, Issue 1, pp: 5-9, 2003. 21

[106] Moore, R. E.; Kearfott, R. B.; Cloud, M. J. Introduction to Interval Analysis. SIAM - Philadelphia, 223p. 2009. 18, 21

[107] Muhlbaier, M.; Topalis, A.; Polikar, R. "Learn++.NC: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes." IEEE Transactions on Neural Networks, Vol. 20, Issue 1, pp: 152168, 2009. 37

[108] Nandedkar, A. V.; Biswas, P. K. "A granular reflex fuzzy min-max neural network for classification." IEEE Transactions on Neural Networks, Vol. 20, Issue 7, pp: 1117-1134, 2009. 49, 78, 90

[109] Neumaier, A. Interval Methods for Systems of Equations. Cambridge University Press, Cambridge, 272p. 1990. 14

[110] Ozawa, S.; Pang, S.; Kasabov, N. "Incremental learning of chunk data for online pattern classification systems." IEEE Transactions on Neural Networks, Vol. 19, Issue 6, pp: 1061-1074, 2008. 43

[111] Pedrycz, W.; Waletzky, J. "Fuzzy clustering with partial supervision." IEEE Transactions on Systems, Man and Cybernetics - Part B, Vol. 27, Issue 5, pp: 787795, 1997. 98

[112] Pedrycz, W.; Vukovich, W. "Granular neural networks." Neurocomputing, Vol. 36, pp: 205-224, 2001. 75, 76

[113] Pedrycz, W. "Heterogeneous fuzzy logic networks: fundamentals and development studies." IEEE Transactions on Neural Networks, Vol. 15, Issue 6, pp: 1466-1481, 2004. 89

[114] Pedrycz, W. Knowledge-based Clustering: From Data to Information Granules. Wiley, 1st edition, 336p. 2005. 98

[115] Pedrycz, W.; Kwak, K.-C. "The development of incremental models." IEEE Transactions on Fuzzy Systems, Vol. 15, Issue 3, pp: 507-518, 2007. 3

[116] Pedrycz, W.; Gomide, F. Fuzzy Systems Engineering: Toward Human-Centric Computing. Wiley - Hoboken, NJ, USA, 526p. 2007. 2, 3, 6, 12, 22, 25, 26

[117] Pedrycz, W. "Granular computing - the emerging paradigm." Journal of Uncertain Systems, Vol. 1, pp: 38-61, 2007. 11, 13

[118] Pedrycz, W.; Skowron, A.; Kreinovich, V. (Eds.) Handbook of Granular Computing. Wiley - Chichester, England, 1116p. 2008. 2, 11, 12

[119] Pedrycz, W. "Evolvable fuzzy systems: some insights and challenges." Evolving Systems, Vol. 1, Issue 2, pp: 73-82, 2010. 40

[120] Petkovic, M. S.; Petkovic, L, D. Complex Interval Arithmetic and Its Applications. Wiley - VCH, Germany, 280p. 1998. 21

[121] Polikar, R.; Udpa, L.; Udpa, S. S.; Honavar, V. "Learn++: An incremental learning algorithm for supervised neural networks." IEEE Transactions on Systems, Man, and Cybernetics - Part C, Vol. 31, Issue 4, pp: 497-508, 2001. 37

[122] Pouzols, F. M.; Lendasse, A. "Evolving fuzzy optimally pruned extreme learning machine for regression problems." Evolving Systems, Vol. 1, Issue 1, pp: 43-58, 2010. 32

[123] Roof, S.; Callagan, C. "The climate of Death Valley, California." Bulletin of the American Meteorological Sociecty, Vol. 84, pp: 1725-1739, 2003. 110

[124] Rubio, J. J. "SOFMLS: Online self-organizing fuzzy modified least-squares network." IEEE Transactions on Fuzzy Systems, Vol. 17, Issue 6, pp: 1296 - 1309, 2009. 32, 36

[125] Rubio, J. J. "Stability analysis for an online evolving neuro-fuzzy recurrent network." In: Angelov, P.; Filev, D.; Kasabov, N. (Eds.) Evolving Intelligent Systems: Methodology and Applications, Wiley - IEEE Press, pp: 173-199, 2010. 32

[126] Russell, S.; Norvig, P. Artificial Intelligence: A Modern Approach. Series in Artificial Intelligence, 3rd edition, 2009. 13

[127] Shafer, J. L. Analysis of Incomplete Multivariate Data. Chapman and Hall - London, 430p. 1997. 39, 124

[128] Silva, L.; Gomide, F.; Yager, R. "Participatory learning in fuzzy clustering." IEEE International Conference on Fuzzy Systems, pp: 857-861, 2005. 124

[129] Simpson, P. K. "Fuzzy min-max neural networks. Part I: classification." IEEE Transactions on Neural Networks, Vol. 3, Issue 5, pp: 776-786, 1992. 36

[130] Simpson, P. K. "Fuzzy min-max neural networks. Part II: clustering." IEEE Transactions on Fuzzy Systems, Vol. 1, Issue 1, pp: 32-45, 1993. 36

[131] Strother, W. "Continuous multi-valued functions." The Bulletin of Sao Paulo Mathematical Society, Vol. 10, pp: 87-120, 1958. 21

[132] Tibshirani, R. "Regression shrinkage and selection via the Lasso." Journal of the Royal Statistical Society - Series B (Methodological), Vol. 58, Issue 1, pp: 267-288, 1996. 130

[133] Tsanas, A.; Little, M. A.; McSharry, P. E.; Ramig, L. O. "Accurate telemonitoring of Parkinson's disease progression by noninvasive speech tests." IEEE Transactions on Biomedical Engineering, Vol. 57, Issue 4, pp: 884-893, 2010. 127, 128, 130

[134] Vachkov, G. "Spatial-temporal knowledge base for modeling and analysis of evolving systems." Evolving Systems, Vol. 2, Issue 2, pp: 131-143, 2011. 32

[135] Witten, I. H.; Frank, E.; Hall, M. A. Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, 3rd edition, 664p. 2011. 3, 33, 38

[136] Xiao, L.; Hung, E. "An efficient distance calculation method for uncertain objects." IEEE Symposium on Computational Intelligence and Data Mining, pp: 10-17, 2007. 72

[137] Yager, R. R. "A model of participatory learning." IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, Issue 5, pp: 1229-1234, 1990. 35

[138] Yager, R. R. "Learning from imprecise granular data using trapezoidal fuzzy set representations." In: Prade, H.; Subrahmanian, V. S. (Eds.) Lecture Notes in Computer Science, Springer - Berlim, Heidelberg, Vol. 4772, pp: 244-254, 2007. 21, 40, 65, 76

[139] Yager, R. R. "Measures of specificity over continuous spaces under similarity relations." Fuzzy Sets and Systems, Vol. 159, Issue 17, pp: 2193-2210, 2008. 44

[140] Yager, R. R. "Participatory learning with granular observations." IEEE Transactions on Fuzzy Systems, Vol. 17, Issue 1, pp: 1-13, 2009. 40, 65

[141] Yao, J. T. "A ten-year review of granular computing." IEEE International Conference on Granular Computing, pp: 734-739, 2007. 2, 11, 12

[142] Yao, Y. Y. "Perspectives of granular computing." IEEE International Conference on Granular Computing, pp: 85-90, 2005. 3, 78

[143] Yao, Y. Y. “The art of granular computing.” International Conference on Rough Sets and Emerging Intelligent Systems Paradigms, LNAI Vol. 4585, pp: 101-112, 2007. 41

[144] Yao, Y. Y. “Granular computing: past, present and future.” IEEE International Conference on Granular Computing, pp: 80-85, 2008. 2, 11

[145] Yao, Y. Y. “Interpreting concept learning in cognitive informatics and granular computing.” IEEE Transactions on Systems, Man, and Cybernetics - Part B, Vol. 39, Issue 4, pp: 855-866, 2009. 41

[146] Yao, Y. Y. “Human-inspired granular computing.” In: Yao, J. T. (Ed.) Novel Developments in Granular Computing: Applications for Advanced Human Reasoning and Soft Computing, 2010. 11, 41

[147] Yeh, I.-C. “Modeling of strength of high performance concrete using artificial neural networks.” Cement and Concrete Research, Vol. 28, Issue 12, pp: 1797-1808, 1998. 123

[148] Young, P. C. Recursive Estimation and Time-Series Analysis: An Introduction. Springer-Verlag - Berlin, 300p. 1984. 155

[149] Zadeh, L. “Fuzzy sets.” Information Control, Vol. 8, pp: 338-353, 1965. 21, 22

[150] Zadeh, L. “The concept of a linguistic variable and its application to approximate reasoning.” Information Science, Vol. 8, pp: 199-249, 1975. 21

[151] Zadeh, L. A. “Fuzzy sets and information granularity.” In: Gupta, M. M.; Ragade, R. K.; Yager, R. R. (Eds.) Advances in Fuzzy Set Theory and Applications, North Holland - Amsterdam, pp: 3-18, 1979. 2, 11

[152] Zadeh, L. A. “Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic.” Fuzzy Sets and Systems, Vol. 90, Issue 2, pp: 111-127, 1997. 3, 41

[153] Zadeh, L. A. “Toward a generalized theory of uncertainty (GTU) - an outline.” Information Sciences, Vol. 172, pp: 1-40, 2005. 3, 13, 48

[154] Zadeh, L. A. ”Generalized theory of uncertainty (GTU) - principal concepts and ideas.” Computational Statistics & Data Analysis, Vol. 51, pp: 15-46, 2006. 2, 3, 13, 40

[155] Zadeh, L. A. “Is there a need for fuzzy logic?” Information Sciences, Vol. 178, Issue 13, pp: 27512779, 2008. 5

[156] Zhang, L; Zhang B. "Fuzzy reasoning model under quotient space structure." Information Sciences, Vol. 173, pp: 353-364, 2005. 78

[157] Zhang, Y.-Q.; Fraser, M. D.; Gagliano, R. A.; Kandel, A. "Granular neural networks for numerical-linguistic data fusion and knowledge discovery." IEEE Transactions on Neural Networks, Vol. 11, Issue 3, pp: 658-667, 2000. 78

[158] Zhu, X.; Goldberg, A. B. Introduction to Semi-Supervised Learning. Morgan and Claypool Publishers (Synthesis Lectures on Artificial Intelligence and Machine Learning), 116p. 2009. 98, 105

[159] Zimmermann, H.-J. Fuzzy Set Theory and its Applications. Kluwer Academic Publishers, 4th edition, 544p. 2001. 5